UiT
THE ARCTIC
UNIVERSITY
OF NORWAY

Faculty of Science and Technology

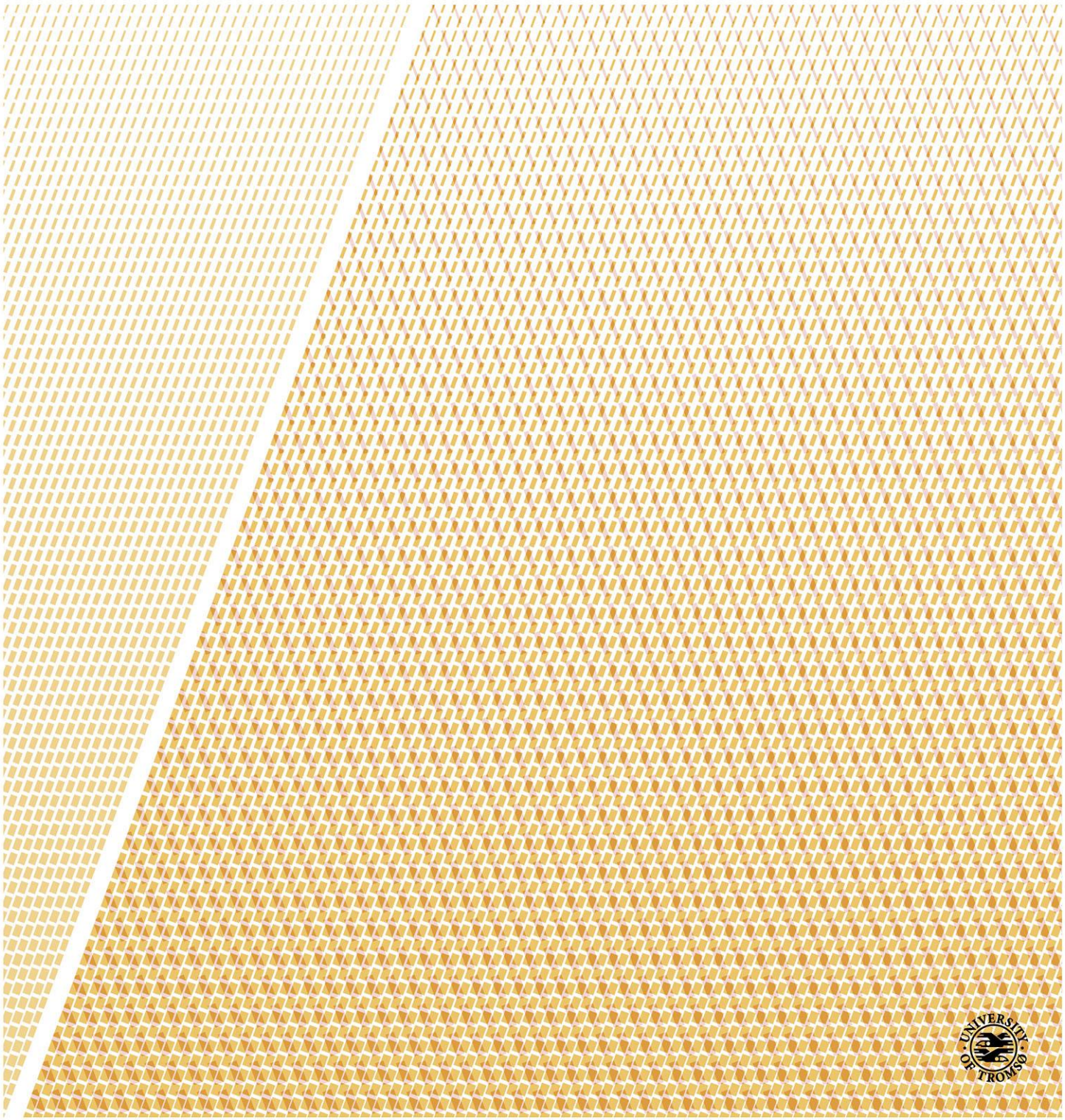
Department of Physics and Technology

Advancing Deep Learning for Automatic Autonomous Vision-based Power Line Inspection

—

Van Nhan Nguyen

A dissertation for the degree of Philosophiae Doctor – August 2019



Abstract

Electricity is fundamental to the ability to function of almost all modern-day societies. To maintain the reliability, availability, and sustainability of electricity supply, electric utilities are usually required to perform visual inspections on their electrical grids regularly. These inspections have been typically carried out using a combination of airborne surveys via low-flying helicopters and field surveys via foot patrol and tower climb. The primary purpose of these visual inspections is to plan for necessary repair or replacement works before any major damage that may lead to a power outage. These traditional inspection methods are not only slow and expensive but also potentially dangerous. In the past few years, numerous efforts have been made to automate these visual inspections. However, due to the high accuracy requirements of the task and its unique challenges, automatic vision-based inspection has not yet been widely adopted in this field.

In this dissertation, we exploit recent advances in Deep Learning (DL), especially deep Convolutional Neural Networks (CNNs), and Unmanned Aerial Vehicle (UAV) technologies for facilitating automatic autonomous vision-based power line inspection. We propose a novel automatic autonomous vision-based power line inspection concept that uses UAV inspection as the main inspection method, optical images as the primary data source, and deep learning as the backbone of data analysis.

Next, we conduct an extensive literature review on automatic vision-based power line inspection. Based on that, we identify the possibilities and six main challenges of DL vision-based UAV inspection: (i) the lack of training data; (ii) class imbalance; (iii) the detection of small power line components and defects; (iv) the detection of power lines in cluttered backgrounds; (v) the detection of previously unseen power line components and defects; and (vi) the lack of metrics for evaluating inspection performance.

We address the first three challenges by creating four medium-sized datasets for training component detection and classification models, by applying a series of effective data augmentation techniques to balance out the imbalanced classes, and by utilizing multi-stage component detection and classification based on Single Shot multibox Detector (SDD) and deep Residual Networks (ResNets) to detect small power line components and defects.

Then, we address the fourth challenge of DL vision-based UAV inspection, which is to detect power lines in cluttered backgrounds, by proposing LS-Net, a fast single-shot line-segment detector, for then to apply it to power line detection. The LS-Net is by design fully convolutional and consists of three modules: (i) a fully convolutional feature extractor; (ii) a classifier; and (iii) a line segment regressor. With a customized version of the VGG-16 network as the backbone, the proposed LS-Net outperforms the existing state-of-the-art DL-based power line detection approaches by a considerable margin and can detect power lines in near real-time.

Finally, we propose few-shot learning as a potential solution to the fifth challenge of DL vision-based UAV inspection, which is to detect previously unseen power line components and defects. To pave the way for addressing the challenge, we propose an innovative approach for advancing the state of the art of few-shot learning. Specifically, we propose a novel dissimilarity measure in terms of the Squared root of the Euclidean distance and the Norm distance (SEN) combined to address the existing issues of the traditional Euclidean distance in high dimensional spaces. We extend the powerful Prototypical Network (PN) by replacing the Euclidean distance by our proposed SEN dissimilarity measure, which we refer to as SEN PN. With minimal modifications, the SEN PN outperforms the original PN by a considerable margin and demonstrates good performance on the miniImageNet dataset with no additional parameters as well as almost no additional computational overhead. The sixth challenge, which is to address the lack of metrics for evaluating inspection performance, is left for future work.

The contribution of this dissertation is threefold: First, it proposes a novel automatic autonomous vision-based power line inspection concept that uses UAV inspection as the main inspection method, optical images as the primary data source, and deep learning as the backbone of data analysis. Second, it provides an overview of the possibilities and challenges of deep learning in automatic autonomous vision-based power line inspection. Third, it proposes approaches for addressing the identified challenges, for advancing deep learning, and for paving the way for realizing fully automatic autonomous vision-based power line inspection.

Acknowledgments

After an intensive period of three years, today is the day: writing this note of thanks is the finishing touch on my Ph.D. dissertation. It has been a period of intense learning for me, not only in the scientific arena but also on a personal level. Writing this dissertation has had a significant impact on me. I would like to reflect on the people who have supported me throughout this journey.

First, I would like to express my sincere gratitude to my supervisor Prof. Robert Jenssen for the continuous support of my Ph.D. study and related research, for his patience, motivation, and constant encouragement throughout this journey. I would also like to thank my co-supervisor Dr. Davide Roverso for the insightful discussions and advice, for his immense knowledge, motivation, and invaluable feedback. I could not have imagined having better advisors and mentors for my Ph.D. study.

Besides my supervisors, I would like to thank the rest of my dissertation committee for taking the time to read my dissertation and attending the defense.

My sincere thanks also go to Prof. Harald Holone and Knut H. H. Johansen - eSmart Systems' CEO, who provided me the opportunity to embark on this amazing journey and to be a part of eSmart Systems.

I thank everyone in the UiT Machine Learning Group, especially Michael Kampffmeyer, Sigurd Løkse, and Kristoffer Wickstrøm for their insightful comments and support.

Special thanks to everyone at eSmart Systems, especially to Henrik Bache for his constant encouragement and motivation; to Heidi Bjerke for her kindness and tremendous support; and to Quang Tran, Hoang Tran, and Manish Shrestha for the stimulating discussions and for all the fun we have had together in the last three years. I am grateful to everyone who has contributed to the Connected Drone project, especially to Tore Lie for always encouraging me to do the impossible; to Kathrin Sunde and Thomas Nergaard for the great support; and to the amazing Pluto team (Hieu Huynh, Mats Edvardsen, and Tuan Nguyen), the Backend team (Hans Gunnar Hansen), and the Frontend team for their fantastic work and dedication.

Above all, I would like to thank my wife Quynh for her love and the sacrifices she made to support me throughout this journey, for always believing in me and encouraging me to follow my dreams, and especially for keeping me sane over the past few intense months.

Last but definitely not least, I would like to thank my family in Vietnam: my parents, sisters, and brothers for their continuous support. Very special thanks to Kiem Phong & Vivi family for being my "second" family and for helping me survive the long, dark, cold winters in Norway.

Van Nhan Nguyen,
Halden, August 2019.

Contents

List of Figures

List of Tables

List of Abbreviations

AE	AutoEncoder
AI	Artificial Intelligence
ANN	Artificial Neural Network
APR	Average Precision Rate
ARR	Average Recall Rate
CIFAR	Canadian Institute For Advanced Research
CNN	Convolutional Neural Network
DCNN-WD	Dilated Convolution Neural Network for Wire Detection
DenseNet	Dense convolutional Network
DL	Deep Learning
DSSD	Deconvolutional Single Shot Detector
FL	Focal Loss
FPN	Feature Pyramid Network
GN	Group Normalization
GNN	Graph Neural Network
IoU	Intersection over Union
LRUA	Least Recently Used Access
LSTM	Long Short-Term Memory
MAML	Model-Agnostic Meta-Learning
MANN	Memory-Augmented Neural Network
mAP	mean Average Precision
MLP	MultiLayer Perceptron
MSE	Mean Square Error
NTM	Neural Turing Machine
PN	Prototypical Network
R-CNN	Regions with CNN features
R-FCN	Region-based Fully Convolutional Network
RBM	Restricted Boltzmann Machine
ReLU	Rectified Linear Unit
ResNet	Residual Network
RN	Relation Network
RNN	Recurrent Neural Network
RoI	Region of Interest
RPN	Region Proposal Network

SDD	Single Shot multibox Detector
SE	Squeeze-and-Excitation
SGD	Stochastic Gradient Descent
SNAIL	Simple Neural AttenIve Learner
SPP-net	Spatial Pyramid Pooling network
SVM	Support Vector Machine
UAV	Unmanned Aerial Vehicle
UID	Unique IDentifier
WRN	Wide Residual Network
WSL-CNN	Weakly Supervised Learning with CNN
YOLO	You Only Look Once

Chapter 1

Introduction

1.1 Background and Motivation

Modern-day societies are becoming increasingly dependent on electricity. This poses significant challenges in maintaining the reliability, availability, and sustainability of electricity supply. For example, the lack of incentives to invest in aged national power grid infrastructures, for example, in Europe and the US, is causing more and more power outages [47]. These power outages, both short and long-term, can have catastrophic effects on unprepared businesses as well as public services and cause substantial financial losses to producers, distributors, and consumers of electricity alike. To prevent power outages and to maintain secure and reliable electricity supply, electric utilities are typically required to perform visual inspections on their electrical grids regularly [31].

These inspections have been typically carried out using a combination of airborne surveys via low-flying helicopters and field surveys via foot patrol and tower climb. In field surveys, a team of usually two inspectors walks from pylon to pylon to visually inspect the power lines with the help of binoculars and sometimes with infrared and corona detection cameras. In airborne surveys, the inspection is typically conducted by a team of two: a pilot and a camera operator. The pilot flies the helicopter over the power lines while the camera operator takes pictures [31]. Many utilities and contractors take pictures only of potential defects and anomalies, while some others take pictures of the whole power grid including pictures of conductors, power line components (e.g., insulators, poles, and cross arms) and surrounding objects (e.g., vegetation). After the flight, the collected images are manually inspected one by one to identify potential defects. These traditional inspection methods are not only slow and expensive but also potentially dangerous since there is always a risk of contact with live lines and loss of life [46]. Although digital cameras can be utilized to separate the data acquisition from the data analysis, both processes have still been performed manually for decades.

In the past few years, numerous efforts have been made to automate visual power line inspections by, for example, employing automated helicopters, flying robots, and/or climbing robots; however, due to the high accuracy requirements of the task and its unique challenges, automatic vision-based inspection has not yet been widely adopted in this field.

Recently, breakthroughs in Deep Learning (DL), especially in deep Convolutional Neural Networks (CNNs), have revolutionized the field of computer vision and opened up new opportunities for automating the data analysis in automatic vision-based power line inspections. In addition, recent advances in battery and fuel cell technologies [62], sensors,

and Unmanned Aerial Vehicle (UAV) components [93] have significantly improved the feasibility of employing UAVs for automating the data acquisition in automatic vision-based power line inspections. Inspired by these achievements, in this dissertation, we explore the possibilities of combining UAVs and deep learning for facilitating fast, accurate, and safe automatic vision-based inspection. Specifically, we aim at realizing fully automatic autonomous vision-based power line inspection by employing UAVs for facilitating automatic data acquisition and by applying deep learning for automating the data analysis.

1.2 Research Statement and Method

As stated above, the primary goal of this dissertation is to facilitate automatic autonomous vision-based power line inspection using deep learning and UAVs. To achieve this goal, we first propose a novel automatic autonomous vision-based power line inspection concept that uses UAV inspection as the main inspection method, optical images as the primary data source, and deep learning as the backbone of data analysis. Next, we study the use of deep learning in power line inspection to have a general overview and a good understanding of the possibilities and challenges of deep learning in automatic autonomous vision-based power line inspection. Then, we propose approaches for addressing the identified challenges, for advancing deep learning, and for paving the way for realizing fully automatic autonomous vision-based power line inspection.

1.2.1 Research Questions

The research presented in this dissertation is guided by the following research questions:

RQ How can deep learning be employed to realize automatic autonomous vision-based power line inspection with UAVs?

- **RQ1.1** What are the possibilities and challenges of deep learning in vision-based UAV inspection?
- **RQ1.2** How and to what extent can the challenges of deep learning in vision-based UAV inspection be addressed?

1.2.2 Method

To answer the research questions, we first conduct an extensive literature review on automatic vision-based power line inspection in Paper I; we further identify the possibilities and six main challenges of DL vision-based UAV inspection, which are:

1. The lack of training data.
2. Class imbalance.
3. The detection of small power line components and defects.
4. The detection of power lines in cluttered backgrounds.
5. The detection of previously unseen power line components and defects.
6. The lack of metrics for evaluating inspection performance.

In Paper I, we answer the first secondary research question – "RQ1.1 What are the possibilities and challenges of deep learning in vision-based UAV inspection?"– and propose potential next steps to answer the remaining research questions and to implement the proposed concept.

In Paper II, Paper III, and Paper IV, we answer the second secondary research question – "RQ1.2 How and to what extent can the challenges of deep learning in vision-based UAV inspection be addressed?" – by proposing approaches for addressing the identified challenges. Specifically, in Paper II, we address the first three challenges by creating four medium-sized datasets for training component detection and classification models, by applying a series of effective data augmentation techniques to balance out the imbalanced classes, and by proposing a multi-stage component detection and classification approach based on Single Shot multibox Detector (SDD) [41] and deep Residual Networks (ResNets) [24] to detect small power line components and defects.

In Paper III, we address the fourth challenge of DL vision-based UAV inspection, which is to detect power lines in cluttered backgrounds. We propose LS-Net, a fast single-shot line-segment detector, for then to apply it to power line detection. The LS-Net is by design fully convolutional and consists of three modules: (i) a fully convolutional feature extractor; (ii) a classifier; and (iii) a line segment regressor. With a customized version of the VGG-16 network [66] as the backbone, the proposed LS-Net outperforms the existing state-of-the-art DL-based power line detection approaches by a considerable margin and can detect power lines in near real-time.

In Paper IV, we propose few-shot learning as a potential solution to the fifth challenge of DL vision-based UAV inspection, which is to detect previously unseen power line components and defects. To pave the way for addressing the challenge, we propose an innovative approach for advancing the state of the art of few-shot learning. Specifically, we propose a novel dissimilarity measure in terms of the Squared root of the Euclidean distance and the Norm distance (SEN) combined to address the existing issues of the traditional Euclidean distance in high dimensional spaces. We extend the powerful Prototypical Network (PN) by replacing the Euclidean distance by our proposed SEN dissimilarity measure, which we refer to as SEN PN. With minimal modifications, the SEN PN outperforms the original PN by a considerable margin and demonstrates good performance on the miniImageNet dataset with no additional parameters as well as almost no additional computational overhead. The sixth challenge, which is to address the lack of metrics for evaluating inspection performance, is left for future work.

Finally, we discuss the possibilities and challenges of deep learning in automatic autonomous vision-based power line inspection and evaluate the proposed approaches to answer the primary research question of this dissertation – "RQ1 How can deep learning be employed to realize automatic autonomous vision-based power line inspection?" – in Chapter 4 and Chapter 5.

1.3 Structure of the Dissertation

The remainder of the dissertation is structured as follows: Chapter 2 provides relevant background knowledge covering deep learning, image classification, object detection, and few-shot learning. In Chapter 3, the special case that forms the basis for the work on which this dissertation is built is presented. Chapter 4 summarizes research findings from each of the included papers. In Chapter 5 and Chapter 6, we discuss the contributions of

the work and conclude the dissertation with a summary and suggestions for further work. Following that, the papers included in the dissertation can be found as appendices.

Chapter 2

Theory and Related Work

This chapter serves five main purposes. Firstly, it presents a brief introduction to deep learning in general with special attention paid to cover MultiLayer Perceptrons (MLPs) and gradient-based learning. Secondly, it gives a brief description of the fundamentals of Convolutional Neural Networks (CNNs). Thirdly, it highlights recent state-of-the-art CNN-based image classification methods. Fourthly, it reviews recent state-of-the-art CNN-based object detection frameworks. Finally, it introduces the few-shot learning problem and summarizes recent relevant methods.

According to the father of Artificial Intelligence (AI), John McCarthy, AI is "the science and engineering of making intelligent machines, especially intelligent computer programs" [44]. In the early days of AI, many projects have sought to tackle and solve AI problems by attempting to hard-code knowledge about the world in formal languages. This approach is known as the knowledge base approach to AI. It has been shown that this traditional AI approach is very good at solving problems that are intellectually difficult for human beings but can be easily described by a list of formal, mathematical rules, such as playing chess. However, it performs poorly on tasks that are relatively straightforward for human beings but can not be described formally, for example, recognizing spoken words or animals in images. This is known as *the true challenge to artificial intelligence*. One of the main limitations of this traditional AI approach is that it requires formal rules with enough complexity to accurately describe the world [18].

2.1 Deep Learning

The existing problems with the knowledge base approach call for a new AI system that is capable of acquiring its own knowledge from raw data. This approach is known as machine learning. With the ability to extract patterns from raw data, machine learning has had many successful applications, such as email classification [3] and breast cancer diagnosis [94]. However, simple machine learning algorithms require hand-designed features that are typically very labor-intensive to create. In some cases, it is relatively straightforward to know what features should be extracted. For example, both words and phrases can be used as features for the email classification task. For many tasks, however, it is very difficult to identify the right set of features to extract. For instance, in the case of animal classification, we are supposed to build a classifier that takes an image as input and outputs an animal class name (e.g., dog, cat, or horse). Obviously, pixel values are not a useful feature since it is not easy to describe exactly what a cat looks like in terms of pixel values. We know

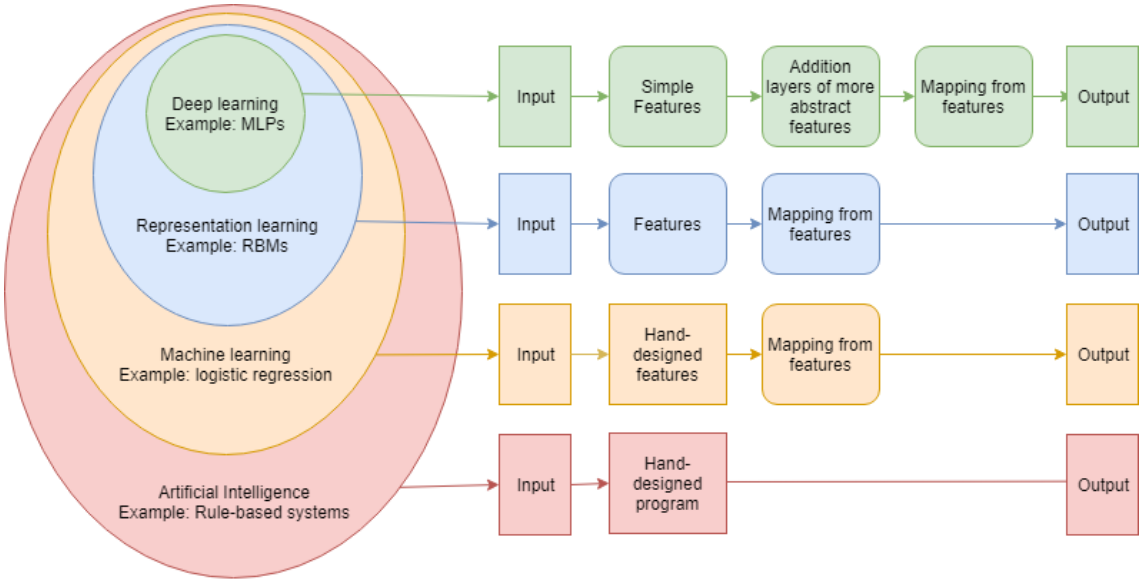


Figure 2.1: A diagram showing the relationship and difference between different AI disciplines. Rounded boxes indicate components that are able to learn from data, and square boxes indicate hand-designed components.

that a cat has four legs, two eyes, two ears, one tail, and hair, so we might like to use the presence of those body parts as features. Unfortunately, those body parts are not a useful set of features since other animals such as dogs and horses also have them. In addition, it is difficult to describe exactly what those body parts look like in terms of pixel values. One possible solution to this problem is to make machine learning algorithms less dependent on feature engineering by automatically discovering representations of data that make it easier to extract useful information. This approach is known as representation learning [4]. Learned representations have two major advantages over hand-designed representations. First, machine learning algorithms usually perform better on learned representations. The second advantage is that AI systems that use learned representations as input can rapidly adapt to new tasks by quickly discovering new good sets of features [18]. Some examples of feature learning algorithms are AutoEncoders (AEs) and Restricted Boltzmann Machines (RBMs).

Deep learning is a representation learning method that is capable of learning representations of data with multiple levels of abstraction. Deep learning enables computers to learn complex concepts, such as cars, by defining them in terms of multiple layers of simpler concepts. For example, a car can be defined in terms of car parts (e.g., wheels, doors, mirrors, hoods, headlights, and windows), which are in turn can be defined in terms of corners and contours. Corners and contours can then be defined in terms of simpler concepts, for example, edges. Finally, edges can be defined in terms of pixels. With that ability, deep learning methods have been advancing the state-of-the-art of many applications such as image recognition, machine translation, and speech recognition.

2.1.1 Multilayer Perceptrons

Multilayer perceptrons, which are also known as deep feedforward networks, are the quintessential deep learning models. MLPs are Artificial Neural Networks (ANNs) that consist of at least three layers of nodes, in which the first layer is called the input layer; the last layer is called the output layer, and the remaining layers are called hidden layers (see Figure 2.2). Layers in MLPs are made of perceptrons, which are also known as artificial neurons or units. The perceptrons, which are denoted as circles in Figure 2.2, are the basic computational unit of MLPs and ANNs in general.

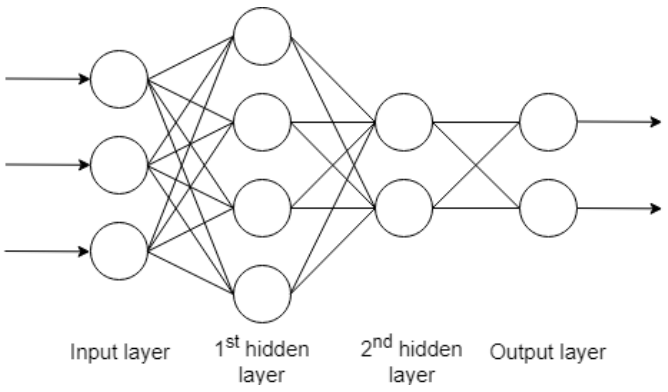


Figure 2.2: Illustration of a simple MLP with two hidden layers.

A perceptron (or a neuron) receives a real-valued vector $\mathbf{x} = (x_1, x_1, \ldots, x_n)$ as input and outputs a value y (see Figure 2.3). The output value y is computed as follows: First,

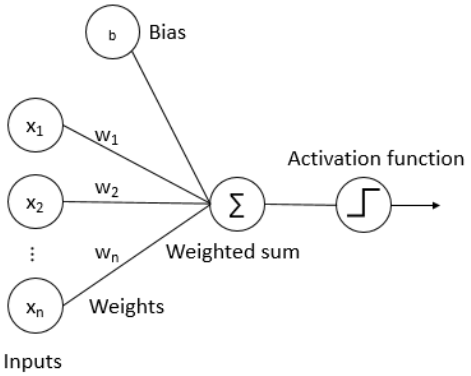


Figure 2.3: Illustration of a perceptron.

a pre-activation $a(\mathbf{x})$ is calculated by applying an affine transformation on the input using the following equation:

$$a(\mathbf{x}) = \sum_i w_i x_i + b, \tag{2.1}$$

where $\mathbf{w} = (w_1, w_2, \ldots, w_n)$ are the connection weights and b is the neural bias. Then, the output value y is computed by applying a transformation on top of the pre-activation

as follows:

$$y = g(a(\mathbf{x})) = g\left(\sum_i w_i x_i + b\right), \tag{2.2}$$

where $g(\cdot)$ is called activation function.

In standard MLPs, perceptrons can be employed either as output or hidden units. In the next two sections, we will review different types of output and hidden units with special attention paid to highlight their use in practice.

Output units

Output units are perceptrons that compose output layers. An output unit employs a three-step procedure to produce its output: First, it accepts a vector $\mathbf{x}$ as input. Next, it computes pre-activation via an affine transformation $z = a(\mathbf{x})$. Finally, it applies an activation function $g(\cdot)$ on top of the computed pre-activation to produce output $\hat{y} = g(z)$. The main difference between output units and normal perceptrons is that the choice of the activation function $g(\cdot)$ for output units depends heavily on the task that the network has to solve. Three types of output units that have been widely used in practice are:

- **Linear units.** Linear units are the simplest kind of output units, which are based only on an affine transformation with no nonlinearity. This type of output unit is usually used for tasks that require Gaussian output distributions. The activation function $g(\cdot)$ of linear units is just an identity mapping function:

$$g(z) = z. \tag{2.3}$$

- **Sigmoid units.** Sigmoid units are the units of choice for tasks in which the output value $\hat{y}$ is binary, for example, binary classification. In general, sigmoid units can be used for tasks that require Bernoulli output distributions. In other words, sigmoid units can be used for tasks in which the model needs to predict only $\hat{y} = P(y = 1|\mathbf{x})$. The activation function $g(\cdot)$ of sigmoid units is the logistic sigmoid function $\sigma(\cdot)$ and is defined as

$$g(z) = \sigma(z) = \frac{1}{1 + \exp(-z)}. \tag{2.4}$$

- **Softmax units.** Softmax units are typically employed for tasks that require Multinoulli output distributions. Softmax units can be interpreted as generalizations of sigmoid units for multiclass classification tasks, which typically require to produce an output vector $\hat{\mathbf{y}} = (\hat{y_1}, \hat{y_2}, \ldots, \hat{y_n})$, where $\hat{y_i} = P(y = i|\mathbf{x})$. The activation function $g(\cdot)$ of softmax units is defined as

$$g(\mathbf{z})_i = softmax(\mathbf{z})_i = \frac{\exp(z_i)}{\sum_j \exp(z_j)}. \tag{2.5}$$

The linear, sigmoid, and softmax units are the three major types of output units that have been widely used in practice; however, in theory, neural networks can employ any kind of output units.

Hidden units

Hidden units are perceptrons that form hidden layers. Hidden units work in the same way as output units. Specifically, a hidden unit applies the same three-step procedure employed by output units to produce its output. Some of the most well-known hidden units are:

- **Rectified Linear Units.** The activation function $g(\cdot)$ of Rectified Linear Units (ReLUs) is quite similar to that of linear units. The only difference is that ReLUs' activation function $g(\cdot)$ outputs zero across half its domain:

$$g(z) = ReLU(z) = \max\{0, z\}. \tag{2.6}$$

  There are many generalizations of ReLUs that have been widely used in practice, for examples, Leaky ReLUs [42], Parametric ReLUs [22], and ELUs [8].

- **Sigmoid and Tanh Units.** Before ReLUs, most of neural network models used sigmoid or tanh units as their default hidden units. The only difference between the two units is their activation function $g(\cdot)$. Sigmoid units use the logistic sigmoid activation function, while tanh units use the hyperbolic tangent activation function, which is defined as

$$g(z) = \tanh(z) = \frac{\exp(z) - \exp(-z)}{\exp(z) + \exp(-z)}. \tag{2.7}$$

In theory, any kind of perceptron that can be used as an output unit can also be used as a hidden unit and vice versa. However, in practice, most of the existing neural network models use ReLUs as their default hidden units. The use of sigmoid and tanh units as hidden units are now discouraged because of their widespread saturation, which can make gradient-based learning very difficult.

Summary

MLPs are composed of only fully-connected layers, which connect every neuron in one layer to every neuron in another layer. With this architecture, MLPs have proven to be a powerful computational tool for many problems in pattern recognition, function approximation, and data analysis; however, they have several major drawbacks, especially when it comes to processing high-dimensional data such as images [78]. The first is that the amount of weights rapidly becomes unmanageable for large images. The second drawback of MLPs is that they disregard spatial information and thus are typically not invariant to small translations as well as local distortions in the input. To address the existing problems of MLPs and to improve the performance of deep learning models in processing high-dimensional data, CNNs were proposed [36]. In the next sections, we briefly introduce CNNs and highlight some of their most common applications in computer vision, including image classification and object detection.

2.1.2 Gradient-Based Learning

Before introducing CNNs and reviewing their applications, we begin by detailing approaches that are typically used for training deep learning algorithms. In general, most

of the existing deep learning algorithms can be built based on three main elements: (i) a model family; (ii) a cost function; and (iii) an optimization procedure. Since MLPs and deep neural networks in general are the backbone of deep learning, in this section, we focus on reviewing costs functions and optimization procedures for training deep neural networks. These cost functions and the optimization procedures, however, can be easily generalized for training other types of deep learning models.

Since neural networks are nonlinear models, their cost functions can not be optimized in closed-form as linear models; instead, it is required to use an iterative numerical optimization procedure that aims at driving the cost function to a very low value [18].

Cost Functions

One of the most common tasks in deep learning is to learn a conditional probability distribution $P(y|\mathbf{x})$ that can be used for predicting y from $\mathbf{x}$. To solve this task, neural networks with parameters $\boldsymbol{\theta}$ that define a distribution $P(y|\mathbf{x};\boldsymbol{\theta})$ are typically employed. In most cases, these models are trained using the principle of maximum likelihood. This means that the models use the cross-entropy between the training data and their predictions as the cost function, which is simply the negative log-likelihood given by

$$J(\boldsymbol{\theta}) = -\mathbb{E}_{\mathbf{x},y\sim\hat{p}_{data}} \log p_{model}(y|\mathbf{x};\boldsymbol{\theta}). \tag{2.8}$$

When the cross-entropy between the data distribution and the model distribution is used as the cost function, the form of this function will be determined by the type of output units that the model uses. Some examples of cost functions that have been used for training neural networks are:

- **Quadratic cost function**. Quadratic cost function (also known as mean squared error or sum squared error) is one of the most widely used cost functions for training models that have linear output units. The quadratic cost function is defined as

$$J(\boldsymbol{\theta}) = \frac{1}{2}\sum_i (\hat{y}_i - y_i)^2. \tag{2.9}$$

- **Binary cross-entropy cost function**. Binary cross-entropy cost function (also known as logistic cost function or sigmoid cross-entropy cost function) is usually used for training models that have sigmoid output units. The binary cross-entropy cost function is given by

$$J(\boldsymbol{\theta}) = -y \log \hat{y} - (1-y)\log(1-\hat{y}). \tag{2.10}$$

- **Categorical cross-entropy cost function**. Categorical cross-entropy cost function (also known multinomial logistic cost function or softmax cost function) is usually used for training models that have softmax output units. The categorical cross-entropy cost function is given by

$$J(\boldsymbol{\theta}) = -\sum_i y_i \log(\hat{y}_i). \tag{2.11}$$

Back-Propagation

To train a neural network using an iterative numerical optimization algorithm, we need to compute the gradient of the cost function with respect to the network's weights, $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$. Back-propagation algorithm (often simply called backprop) is one of the most common methods for computing the gradient for training neural networks by using the chain rule of calculus [59]. The computed gradient is then used by the optimization algorithm to minimize the cost function. Specifically, to train a neural network, the optimization algorithm typically repeats the following three-step cycle:

- **Step 1: Forward propagation.** An input vector $\mathbf{x}$, which is stochastically drawn from training data, is propagated forward through the network, layer by layer, until it reaches the output layer, to compute a scalar cost $J(\boldsymbol{\theta})$.
- **Step 2: Back propagation.** The scalar cost $J(\boldsymbol{\theta})$ is then propagated backward through the network, layer by layer, until it reaches the first hidden layer, to compute the gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$.
- **Step 3: Weight update.** The network's weights $\boldsymbol{\theta}$ are updated based on the computed gradient as follows:

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}), \tag{2.12}$$

  where η is the learning rate.

To clarify the above definition of backprop and the three-step training cycle, let us consider a simple example in which the standard stochastic gradient descent algorithm and the quadratic cost function are employed for training a simple neural network composed of only one hidden layer that is made of only one artificial neuron. In this case, the hidden layer receives a vector $\mathbf{x}$ as input and outputs a vector $\mathbf{h} = f_1(\mathbf{x}, \boldsymbol{\theta}_1)$, where $\boldsymbol{\theta}_1$ is the hidden layer's weights. The output layer receives the output vector $\mathbf{h}$ from the hidden layer as input and produces an output value $\hat{y} = f_2(\mathbf{h}, \boldsymbol{\theta}_2)$, where $\boldsymbol{\theta}_2$ is the output layer's weights. With that assumption, the three-step training cycle works as follows:

- **Step 1: Forward propagation.** An input vector $\mathbf{x}$, which is stochastically drawn from training data, is propagated forward through the network, layer by layer, until it reaches the output layer, to compute a scalar cost $J(\boldsymbol{\theta})$:

$$\mathbf{h} = f_1(\mathbf{x}, \boldsymbol{\theta}_1), \tag{2.13}$$

$$\hat{y} = f_2(\mathbf{h}, \boldsymbol{\theta}_2), \tag{2.14}$$

$$J(\boldsymbol{\theta}) = \frac{1}{2}(\hat{y} - y)^2. \tag{2.15}$$

- **Step 2: Back propagation.** The scalar cost $J(\boldsymbol{\theta})$ is then propagated backward through the network, layer by layer, until it reaches the first hidden layer, to compute the gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \hat{y}} = \hat{y} - y, \tag{2.16}$$

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_2} = \frac{\partial J(\boldsymbol{\theta})}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \boldsymbol{\theta}_2}, \tag{2.17}$$

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_1} = \frac{\partial J(\boldsymbol{\theta})}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \mathbf{h}} \frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}_1}, \tag{2.18}$$

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = (\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_1}, \frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_2}). \tag{2.19}$$

- **Step 3: Weight update.** The network's weights $\boldsymbol{\theta}$ are updated based on the computed gradient as follows:

$$\boldsymbol{\theta}_1 = \boldsymbol{\theta}_1 - \eta \frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_1}, \tag{2.20}$$

$$\boldsymbol{\theta}_2 = \boldsymbol{\theta}_2 - \eta \frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_2}. \tag{2.21}$$

  where η is the learning rate.

The back-propagation algorithm and the three-step training cycle presented here can be easily generalized to train neural networks with an arbitrary number of layers and an arbitrary number of units per layer.

Challenges of Gradient-based Learning

Training deep neural networks with gradient-based learning is a challenging task. Some of the most common problems that usually occur during training deep neural networks in practice are:

- **Overfitting.** Overfitting is a problem that happens when the model works very well on the training set, but performs poorly on the test set. In other words, overfitting occurs when the model has memorized the training examples, but it has not learned to generalize to new examples. There are two main approaches for preventing overfitting: The first is to reduce the number of dimensions of the parameter space, for example, by reducing the size of the network or by employing weight sharing [48]. The second approach is to reduce the effective size of each dimension [52]. Regularization is one of the most commonly used techniques for reducing the effective size of each parameter dimension.

- **Vanishing and exploding gradients**. Vanishing and exploding gradients are difficulties found in training deep neural networks with gradient-based learning. The vanishing gradient problem arises when the gradient gets vanishingly small that the learning either becomes very slow or stops working. This problem happens because the training algorithm does not know which direction the parameters should move to improve the cost function. The exploding gradient problem, on the other hand, occurs when the gradient signal explodes, making the learning unstable [49].

Apart from these major problems, training deep neural network models with gradient-based learning also faces some minor challenges: First, training deep models with gradient-based learning typically requires careful hyperparameter tuning, which is a very time-consuming task. Second, large amounts of training data are usually required for training

deep neural networks. Finally, deep neural network models trained with gradient-based learning are typically uninterpretable. This means that the models can work well in practice; however, we usually do not know for sure why they work that way and how to improve them effectively.

Regularization

According to [18], regularization is "any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error". Some of the most widely used regularization strategies in practice are:

- **Dataset augmentation.** The easiest way to prevent neural networks from overfitting is to train them with more training data; however, it is not always possible to collect more training data in practice. Two possible solutions to this problem are synthetic data and data augmentation. To train neural networks, for example, for image classification, synthetic images generated from 3D models can be employed to increase the training set size; however, creating high-quality synthetic data is usually very time-consuming and expensive. Data augmentation, on the other hand, can be applied to generate more training data at a minimal cost. Some examples of data augmentation techniques that have been extensively used in practice are translation, rotation, random cropping, flipping, and color jittering [50].

- **Early stopping.** Early stopping is probably the most commonly used regularization technique in training neural networks. The basic idea of early stopping is to stop the training early to avoid overfitting [52]. Early stopping is applied in training neural networks as follows: Before training, the data is split into two sets: a training set and a validation set. The network is trained on the training set and tested on the validation set after every i iterations. During training, the error of the network on the validation set is monitored, and the training is stopped whenever the error has not improved for some amount of time t.

- **$\mathbf{L_2}$ Parameter Regularization.** L_2 parameter regularization, also known as weight decay, is a regularization strategy that aims at limiting the capacity of neural networks by adding a parameter norm penalty $\Omega(\boldsymbol{\theta})$ to the cost function $J(\boldsymbol{\theta})$. The regularized cost function $\tilde{J}(\boldsymbol{\theta})$ is defined as

$$\tilde{J}(\boldsymbol{\theta}) = J(\boldsymbol{\theta}) + \alpha\Omega(\boldsymbol{\theta}). \tag{2.22}$$

  L_2 is one of the simplest and most commonly used kinds of parameter norm penalty. This strategy penalizes the square value of the network's weights by adding a regularization term $\Omega(\boldsymbol{\theta}) = \frac{1}{2}\|\mathbf{w}\|_2^2$ to the cost function in order to drive all the weights $\mathbf{w}$ (except the bias b) to the origin (to smaller values).

- **$\mathbf{L_1}$ Parameter Regularization.** L_1 regularization is very similar to L_2 regularization. The main difference between these techniques is that L_1 regularization penalizes the absolute value instead of the square value of the network's weights as in L_2 regularization. In L_1 regularization, a regularization term $\Omega(\boldsymbol{\theta}) = \|\mathbf{w}\|_1 = \sum_i |w_i|$ is added to the cost function. This can drive some weights to exact zero while allowing other weights to be small, and thus introducing sparsity into the model.

- **Dropout.** Dropout is a simple but powerful regularization strategy proposed by Srivastava et al. [69]. Dropout works as follows: During training, some units in the network along with all of their incoming and outcoming connections are randomly removed with probability p. Dropout can be applied for both input layers and hidden layers. In practice, it is recommended to use $p = 0.5$ for hidden layers and $p = 0.2$ for output layers. When it is applied, dropout tends to reduce the co-adaptation between units. Consequently, it makes the model more robust. According to [18], training a neural network with dropout is equivalent to the ensemble of 2^n sub-networks that can be formed by removing some units along with all of their incoming and outcoming connections from the network.

- **Batch Normalization.** Batch normalization is a technique for accelerating neural network training by reducing internal covariate shift proposed by Sergey Ioffe et al. [30]. According to the authors, batch normalization can be used as a regularizer. In practice, it has been shown that batch normalization can eliminate the need for dropout and allow the use of much higher learning rates. In addition, batch normalization can also reduce the need for "careful weight initialization". The main idea of batch normalization is to normalize pre-activations of each neural network layer by its mean and variance over a mini-batch and then apply (optionally) a scale γ to it as well as an offset β. The batch normalizer is defined as

$$y_i = \frac{\gamma(x_i - \mu)}{\sqrt{\sigma^2 + \epsilon}} + \beta, \tag{2.23}$$

  where μ is the mini-batch mean, σ^2 is the mini-batch variance, and ϵ is a small constant added to the mini-batch variance for numerical stability. Batch normalization can be applied on pre-activation or activation; however, it is recommended to use batch normalization on pre-activation. In other words, batch normalization should be applied immediately before the nonlinearity transformation.

- **Group Normalization.** Although the normalization along the batch dimension allows batch normalization to reduce internal covariate shift and accelerate the training of deep neural networks, it causes many distinct drawbacks. For example, for batch normalization to work properly, it is required to have a sufficiently large batch size, which is typically not possible with training very deep neural networks due to GPU memory limitations [86]. With the aim of eliminating the dependence on batch sizes and avoiding batch statistics computation, Wu et al. proposed Group Normalization (GN) as a simple alternative to batch normalization [86]. The key innovation of GN is that it divides channels into groups and normalizes the features within each group.

Regularization is a very important method for dealing with the central problem in deep learning: overfitting. In the past few years, developing more effective regularization strategies has been a very active research area in the deep learning community.

Optimization Algorithms

Most of the common optimization algorithms that have been used for training deep neural networks typically repeat the following three-step cycle:

- **Step 1:** Propagate forward m examples stochastically sampled from a training set of size n through the network to compute a scalar cost $J(\boldsymbol{\theta})$.

- **Step 2:** Propagate backward the scalar cost $J(\boldsymbol{\theta})$ through the network to compute the gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$.
- **Step 3:** Update the network's weights $\boldsymbol{\theta}$ using the following weight update rule:

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}), \tag{2.24}$$

  where η is a learning rate.

By choosing different values of m and/or adding extra terms to the weight update formula and/or applying different learning rate adaptation techniques, most of the common optimization algorithms that have been used for training deep neural networks can be derived from the three-step cycle recipe presented above.

When $m = n$ (step 1 propagates the whole dataset at once), the three-step cycle presented above becomes the gradient descent algorithm (also known as batch gradient descent). When $m = 1$ (step 1 propagates only a single example at once), the three-step cycle becomes the Stochastic Gradient Descent (SGD) algorithm, and when $1 < m < n$, the three-step cycle becomes the mini-batch gradient descent algorithm.

By adding a velocity term, which is the gradient from the previous iteration, to the weight update formula, the SGD algorithm becomes the SGD with momentum algorithm. The learning rule of the SGD with momentum algorithm is defined as follows:

$$\text{Compute velocity update: } \mathbf{v} = \alpha \mathbf{v} - \eta \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

$$\text{Update weight: } \boldsymbol{\theta} = \boldsymbol{\theta} + \mathbf{v},$$

where η is the learning rate and $\alpha \in [0, 1)$ is the momentum hyperparameter.

By employing different learning rate adaptation strategies, the three-step cycle can derive some of the most commonly used optimization algorithms for training deep neural networks including AdaGrad, RMSProp, and Adam. According to [18], AdaGrad scales the learning rate η inversely proportional to the square root of the sum of all of historical squared gradients. The learning rule of AdaGrad is defined as follows:

$$\text{Accumulate squared gradient: } \mathbf{r} = \mathbf{r} + \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \cdot \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

$$\text{Update weight: } \boldsymbol{\theta} = \boldsymbol{\theta} - \frac{\eta}{\delta + \sqrt{\mathbf{r}}} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

where δ is a small constant added for numerical stability.

RMSProp is quite similar to AdaGrad; the main difference between the two algorithms is that RMSProp changes gradient accumulation into an exponentially weighted moving average. The learning rule of RMSProp is defined as follows:

$$\text{Accumulate squared gradient: } \mathbf{r} = \rho \mathbf{r} + (1 - \rho) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \cdot \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

$$\text{Update weight: } \boldsymbol{\theta} = \boldsymbol{\theta} - \frac{\eta}{\sqrt{\delta + \mathbf{r}}} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

where ρ is a hyperparameter which controls the length scale of the moving average and δ is a small constant added for numerical stability.

Adam can be seen as "a variant on the combination of RMSProp and momentum with a few important distinctions" [18]. The learning rule of Adam is defined as follows:

$$\text{Update biased first moment estimate: } \mathbf{s} = \rho_1 \mathbf{s} + (1 - \rho_1) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

$$\text{Update biased second moment estimate: } \mathbf{r} = \rho_2 \mathbf{r} + (1 - \rho_2) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \cdot \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}),$$

$$\text{Correct bias in first moment: } \hat{\mathbf{s}} = \frac{\mathbf{s}}{1 - \rho_1^t},$$

$$\text{Correct bias in second moment: } \hat{\mathbf{r}} = \frac{\mathbf{r}}{1 - \rho_2^t},$$

$$\text{Update weight: } \boldsymbol{\theta} = \boldsymbol{\theta} - \eta \frac{\hat{\mathbf{s}}}{\sqrt{\hat{\mathbf{r}}} + \delta},$$

where $\rho_1 \in [0, 1)$ and $\rho_2 \in [0, 1)$ are exponential decay rates for moment estimates, t is the time step, and δ is a small constant added for numerical stability.

2.2 Convolutional Neural Networks

In deep learning, Convolutional Neural Network (CNN) is a special class of deep neural network designed to take advantage of the 2D structure of visual imagery (or other 2D input such as a speech signal). In addition, CNNs be can generalized to work with other types of data that have a known grid-like topology such as time-series data. The four key ideas behind the success of CNNs in processing image data are local connections, shared weights, pooling, and the use of many layers [35].

Unlike MLPs, which are composed of only fully-connected layers, CNNs employ three main types of layers: convolutional layers, pooling layers, and fully-connected layers.

2.2.1 Convolutional Layer

Convolutional layers are the fundamental component of CNNs which leverage the three main ideas that make CNNs powerful: local connectivity, parameter sharing, and equivariant representations [18]. Specifically, a convolutional layer accepts a volume I of size $[W_I, H_I, D_I]$ as input and outputs a volume O of size $[W_O, H_O, D_O]$. The convolutional layer is composed of several convolution kernels K (often called filters). Each neuron in the output volume looks at a rectangular region in the input volume. The rectangular region is referred to as the neuron's receptive field in the previous layer, and the size of the region is often called the filter size [19]. The filters are slid across the input volume I with stride S to compute dot products to produce activation maps:

$$O_K(i, j) = \sum_m \sum_n I(m, n) K(i - m, j - n). \tag{2.25}$$

In practice, many deep learning libraries implement an alternative function called the cross-correlation:

$$O_K(i, j) = \sum_m \sum_n I(i + m, j + n) K(m, n). \tag{2.26}$$

To maintain the spatial dimensions, the input volume is often padded with zeros (see Figure 2.4).

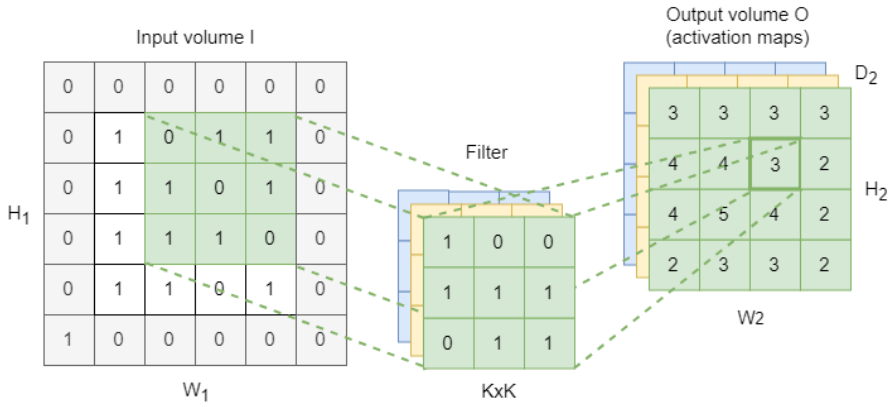


Figure 2.4: Illustration of a convolutional layer with three 3×3 filters, stride $S = 1$, and zero padding size $P = 1$.

2.2.2 Pooling Layer

According to [18], a pooling layer "replaces the output of the net at a certain location with a summary statistic of the nearby outputs". Pooling layers in CNNs serve two primary purposes: The first is to introduce invariance to small translations in the input. The second purpose is to reduce the number of parameters and the amount of computation in the network by progressively reducing the spatial dimension of the input volume. There are many pooling functions that can be used in pooling layers such as max-pooling, average-pooling, and L_2-pooling. However, in practice, it is recommended to use the max-pooling. The max-pooling function takes a rectangular region of size $K \times K$ as input and outputs the maximum value of the elements in the region. The function is slid across the input volume I with stride S to compute activation maps (see Figure 2.5).

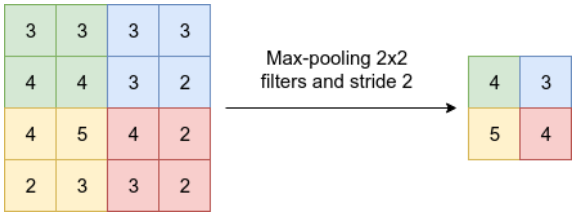


Figure 2.5: Illustration of a pooling layer with max-pooling 2×2 filters and stride $S = 2$.

Recently, it has been shown that max-pooling layers can simply be replaced by convolutional layers with increased stride without loss in accuracy [68].

2.2.3 Fully-connected Layer

Fully-connected layers in CNNs are very similar to fully-connected layers in MLPs; they are composed of neurons that are connected to all activations in the previous layer. In CNNs, fully-connected layers are typically responsible for high-level reasoning. Specifically, fully-connected layers are usually added to the end of CNNs to generate global semantic information [19] and to perform classification based on the features extracted by the previous layers.

2.2.4 CNN Architecture

Many well-known deep CNNs, such as AlexNet [34] and VGGNet [66], are formed by simply stacking up many convolutional layers, pooling layers, and fully-connected layers. In those deep CNNs, the information flowing through the network passes through many stages of multiplication; therefore, the gradients are needed to be back-propagated though many stages during training. This typically causes the gradients to either vanish or explode. The exploding gradient problem can be addressed easily by, for example, applying gradient clipping. The vanishing gradients, on the other hand, are quite hard to overcome. When the gradients vanish, the learning either becomes very slow or stops working. This issue is historically known as one of the main challenges of training very deep CNNs. An example of the vanishing gradient problem's cause is the use of saturated activation functions such as the hyperbolic tangent or the logistic sigmoid [88]. In modern CNNs, it is recommended to use non-saturated activation functions, which typically suffer less from the vanishing gradient problem, such as the ReLU, as alternatives to the hyperbolic tangent or logistic sigmoid [17].

In practice, other layers such as dropout [69], batch normalization [30], and group normalization [86] are often added to CNNs to improve performance and avoid overfitting. For more details on the underlying concept of CNNs and their existing challenges, we refer the interested reader to [35], [18], and [19].

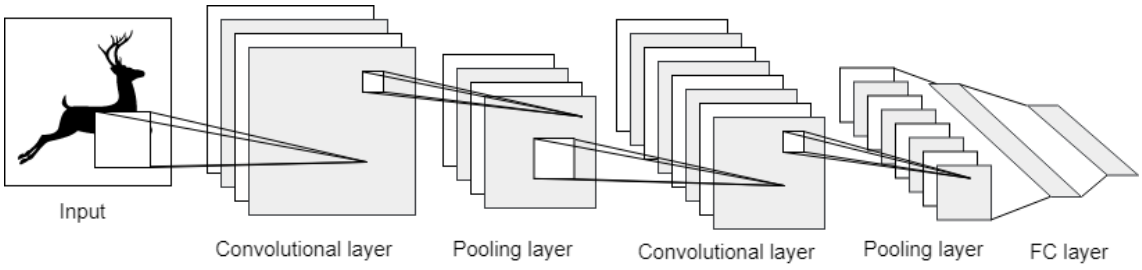


Figure 2.6: Illustration of a simple CNN for image classification.

In the past few years, CNNs have been advancing the state of the art of many computer vision tasks such as image classification and object detection. In the next two sections, we review some of the most well-known CNN architectures for image classification and object detection.

2.3 Image Classification

One of the most common applications of CNNs in computer vision is arguably image classification, which aims at recognizing the category of the dominant object in an image. Since the success of Krizhevsky et al. [34] with an 8-layer CNN (5 convolutional layers + 3 fully-connected layers) called AlexNet in the 2012 ImageNet challenge, CNNs have become a commodity in the computer vision field. In the last few years, many attempts have been made to improve the original AlexNet architecture by, for example, utilizing a smaller receptive window size and by increasing the depth of the network.

One of the most recognized such attempts is the VGGNet [66], which is a CNN architecture that secured the first place in the localization task and the second place in the

classification task in the 2014 ImageNet challenge. The key innovation of the VGGNet is the combination of small filters (3×3 filters) and deep networks (16-19 layers). The authors argued that a stack of three 3×3 convolutional layers has the same effective receptive field as one 7×7 convolutional layer, but is deeper, has more non-linearities, and has fewer parameters.

With the increasing complexity of image classification problems, higher performance CNNs are typically required. The most straightforward way of improving the performance of CNNs is to increase their size by, for example, increasing their depth and width. Deep CNNs constructed simply by stacking up many layers are computationally expensive and very difficult to train due to the notorious problem of vanishing/exploding gradients. Wide shallow CNNs typically suffer less from vanishing/exploding gradients; however, they are very computationally expensive. With the aim of increasing the performance of CNNs while keeping the computational budget constant, Szegedy et al. [75] proposed a novel deep CNN architecture codenamed Inception. Inception modules employ two main ideas: The first is employing filters of multiple sizes (1×1, 3×3, and 5×5) that operate at the same level (see the green boxes in Figure 2.7). The second idea is judiciously applying dimension reductions and projections to reduce computational requirements (see the purple boxes in Figure 2.7). These ideas enable a considerable performance gain at a modest increase in computational requirements compared to shallower and less wide CNNs. The original Inception architecture (Inception-v1) was further improved in [76] by adopting batch normalization (Inception-v2) and later by employing additional factorization ideas (Inception-v3).

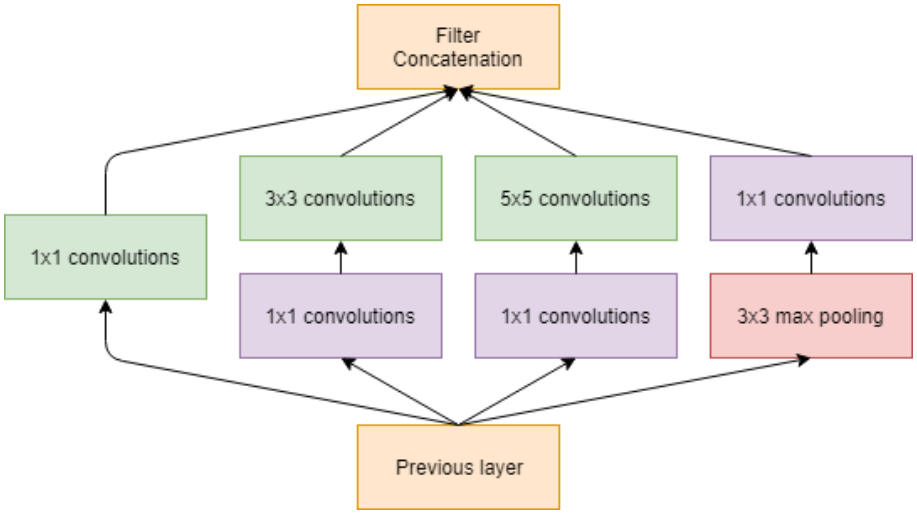


Figure 2.7: Illustration of an inception module with dimension reductions.

To further improve the performance of CNNs and to ease the training of very deep networks, Residual Networks (ResNets) were proposed [24]. The ResNets add "shortcut" connections (residual connections) to standard CNN layers to allow the gradient signal to travel back directly from later layers to early layers (See Figure 2.8). The "shortcut" connections allowed the authors of the ResNets to successfully train very deep CNNs with 50, 101, and even 152 layers.

To take advantage of both the Inception architecture and the residual connections, Szegedy et al. [73] proposed to replace the filter concatenation stage of the Inception

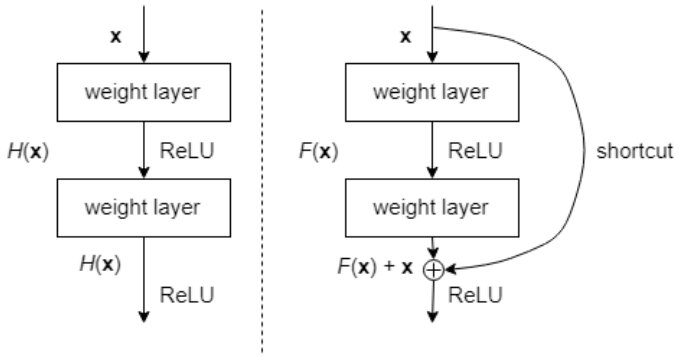


Figure 2.8: A comparison between standard CNNs (left) and ResNets with shortcut connections (right). $H(\mathbf{x})$ is the underlying mapping. $F(\mathbf{x}) = H(\mathbf{x}) - \mathbf{x}$ is the residual mapping adopted by ResNets.

architecture by residual connections (Inception-ResNet). The authors further improved the Inception-v3 by employing more inception modules and by making the architecture more uniform and simplified (Inception-v4) [73].

Although ResNets had a great success winning the ImageNet and COCO 2015 competitions as well as achieving the state-of-the-art performance in several benchmarks, it has many weaknesses. The first is the long training time. The second weakness is the diminishing feature reuse [28], which is also known as loss in information flow [70]. To address these issues, many improvements have been proposed. One example is stochastic depth [28], which randomly drops a subset of layers during training. The dropped layers are bypassed with the identity function. This simple modification allows better information and gradient flow, which results in a substantial reduction in training time and a considerable increase in accuracy. Stochastic depth allows the authors to successfully train ResNets with more than 1200 layers. Another example are Wide Residual Networks (WRNs) [91], which are adapted from ResNets by decreasing the depth and increasing the width of the networks. The authors demonstrated that a 16-layer WRN significantly outperforms 1000-layer ResNets on CIFAR (Canadian Institute For Advanced Research) datasets [33] and that a 50-layer WRN outperforms 152-layer ResNets on ImageNet. In addition, the authors showed that WRNs are several times faster to train compared to ResNets.

With the aim of improving the quality of representations produced by CNNs, Hu et al. [26] proposed a new architecture unit called Squeeze-and-Excitation (SE) block to model the interdependencies between CNN feature channels. The proposed SE blocks allow CNNs to perform feature recalibration, which enables the use of global information selectively by emphasizing informative features and suppressing less useful ones. Specifically, in each SE block, a global understanding of each channel is obtained by squeezing the feature maps into a $1 \times 1 \times C$ vector. The vector is used by an excitation operation to scale the channels to emphasize informative features and suppress less useful ones (see Figure 2.9). With a slight additional computational cost, SE blocks bring considerable improvements in performance to the existing state-of-the-art CNNs such as ResNets and Inception Nets.

Inspired by the success of "shortcut" connections in CNNs, Huang et al. proposed a novel network architecture called Dense convolutional Network (DenseNet) [27]. The core idea of DenseNets is the use of multiple densely connected blocks in which all layers

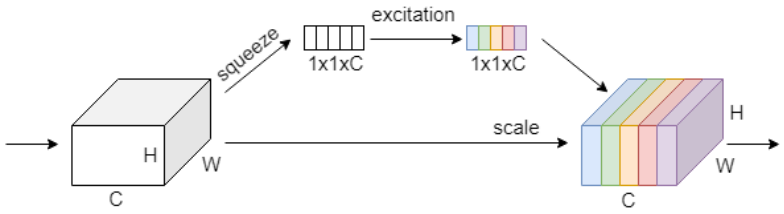


Figure 2.9: Illustration of a SE block.

(with matching feature-map sizes) are directly connected with each other. Specifically, a layer in a dense block uses feature maps of all preceding layers in the bock as inputs, and its own feature maps are used as inputs into all subsequent layers in the block. This allows for maximum information flow between layers in the network. DenseNets achieve the state-of-the-art performance while requiring substantially fewer parameters and less computation compared to other networks that support "shortcut" connections such as ResNets, ResNets with stochastic depth, and WRNs.

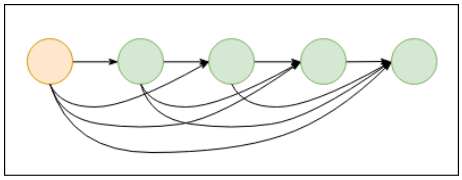


Figure 2.10: Illustration of a dense block. A layer in a dense block uses feature maps of all preceding layers in the bock as inputs, and its own feature maps are used as inputs into all subsequent layers in the block.

Despite the success of neural networks in many tasks such as image classification, speech recognition, and machine translation, these models are still hard to design and typically require lots of human effort in tuning. With the aim of generating good neural network architectures automatically, Barret Zoph and Quoc V. Le proposed Neural Architecture Search, a gradient-based method for finding good architectures based on Recurrent Neural Networks (RNNs) and reinforcement learning. The main idea is to use a RNN controller to generate neural network architectures (child networks) and use reinforcement learning to train the controller based on the accuracy of the child models on the validation set. By using a RNN as the controller, the proposed method is capable of searching in variable-length architecture space and rivals the best human-invented architectures, such as ResNets, ResNets with stochastic depth, WRNs, and DenseNets in terms of test set accuracy.

It can be clearly seen from this review that the general trend to achieve higher accuracy has been to make deeper and more complicated networks. This typically results in longer training time and significantly higher computational cost. With the aim of building very small, low latency models for mobile and embedded vision applications, Howard et al. [25] proposed an efficient network architecture called MobileNet based on depthwise separable convolutions. To make the network smaller and faster as well as to reduce the computational cost, the authors further proposed two hyper-parameters: a width multiplier and a resolution multiplier. The former is responsible for thinning the network uniformly at each layer while the latter is applied to the input image to reduce the internal representation

of every layer.

2.3.1 Summary

In this section, we have briefly introduced image classification, which is one of the most common applications of CNNs in computer vision, and have reviewed the state-of-the-art CNN-based image classification models, such as Inception architectures, ResNets, and Densets. In the next section, we introduce another common application of CNNs in computer vision: object detection.

2.4 Object Detection

Inspired by the success of CNNs in image classification, many researchers have proposed to use CNNs to solve the more challenging task of object detection. The main goal of generic object detection is to localize and classify existing objects in images. This is typically achieved by labeling each object with a bounding box that shows where it is, with the label of the class that the object belongs to, and with a confidence score that reflects its confidence of existence.

In the past few years, many CNN-based object detectors have been proposed. These detectors can be mainly categorized into two types: one-stage detectors and two-stage detectors. Details of these detectors are as follows.

2.4.1 Two-stage Object Detectors

Two-stage object detectors detect objects via two main steps: region proposal and region classification. One of the most pioneering two-stage object detectors is the selective search work [79]. For the region proposal step, a data-driven region proposal method called selective search, which aims at generating a small set of high-quality class-independent object locations, is proposed. Selective search is inspired by bottom-up segmentation and exhaustive search. Specifically, a data-driven grouping-based strategy is employed together with three diversification strategies including (i) utilizing a variety of color spaces; (ii) employing different similarity measures; and (iii) varying starting regions for improving the search's robustness. For the region classification step, a Support Vector Machine (SVM) with histogram intersection kernel is employed for classifying the proposals into foreground classes/background.

R-CNN (Regions with CNN features) [16] improves the selective search approach by replacing the second-stage classifier by a CNN that extracts a fixed-length feature vector from each region and a set of class-specific linear SVMs that score the feature vector and predict the presence of each object class in the candidate region. In addition, a linear regression model is employed for tightening the bounding boxes (see Figure 2.11). These improvements allow R-CNN to outperform the selective search approach significantly; however, R-CNN still has many notable drawbacks. The first is the complicated multi-stage training pipeline, which is employed for separately training the three trainable modules: the CNN feature extractor, the SVMs, and the bounding-box regressors. The second drawback is the large space requirements for storing features extracted from each object proposal in each image. The third is the long training time. The fourth drawback is slow detection speed due to the lack of shared computation. Specifically, R-CNN performs a separate CNN forward pass for each object proposal.

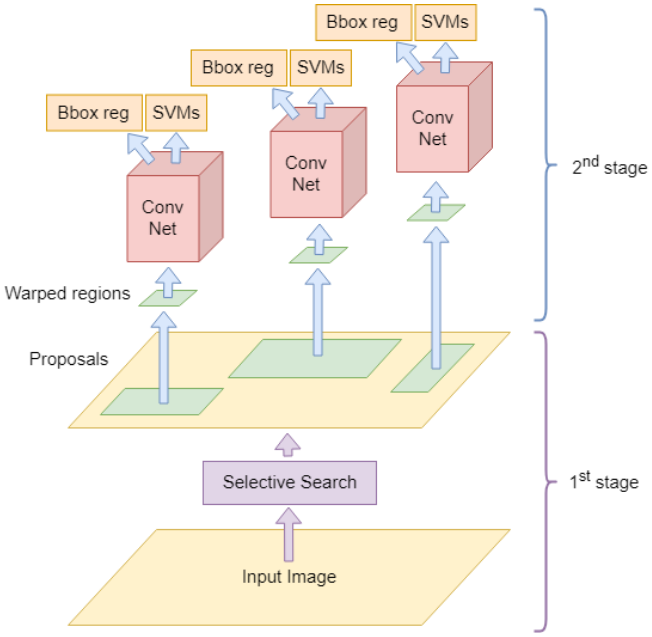


Figure 2.11: Illustration of R-CNN. In the first stage, the selective search is employed for extracting around 2000 bottom-up region proposals from the input image. In the second stage, the proposals are fed into a large CNN for extracting useful features, which are then used by a set of class-specific SVMs and bounding-box regressors for classifying each region and tightening the bounding boxes, respectively.

To speed up R-CNN, Spatial Pyramid Pooling networks (SPP-nets) [23] were proposed. SPP-nets employ computation sharing by first computing convolutional feature maps for the entire input image only once (possibly at multiple scales). Then, the spatial pyramid pooling is applied on each proposal candidate on the shared feature maps to generate a fixed-length representation. Finally, a set of class-specific linear SVMs are utilized for classifying the proposal candidates, and bounding-box regressors are applied for tightening the bounding boxes. By addressing the fourth drawback of R-CNN, specifically employing computation sharing via shared convolutional feature maps, SPP-nets can run at orders of magnitude faster than R-CNN. However, the other three drawbacks, including the complicated multi-stage training pipeline, the long training time, and the large space requirements for storing features, still remain unsolved.

To fix the disadvantages of R-CNN and SPP-nets and to improve their speed and accuracy, Fast R-CNN was proposed [15]. Fast R-CNN extends the computation sharing idea of the SPP-nets and employs a multi-task loss for facilitating singe-stage training. Specifically, shared convolutional feature maps for the entire input image are first extracted using a CNN. Then, a Region of Interest (RoI) pooling layer is employed for extracting a fixed-length feature vector from the shared feature maps for each proposal candidate. Finally, a sequence of fully-connected layers and two sibling output layers are employed for processing each feature vector for simultaneously producing softmax probabilities and per-class bounding-box regression offsets (see Figure 2.12). This architecture allows Fast R-CNN to be trained end-to-end with a multi-task loss, which is a combination of a log loss for training the classifier and a smooth L1 loss for training the bounding-box regressor.

Although Fast R-CNN is much better than R-CNN and SPP-nets both in terms of

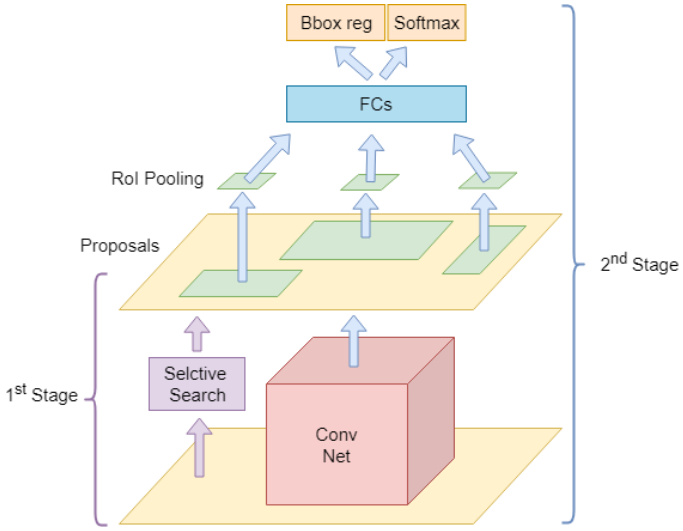


Figure 2.12: Illustration of Fast R-CNN. Instead of performing a separate CNN forward pass for each object proposal as in R-CNN, Fast R-CNN adopts the computation sharing idea proposed in SPP-nets and computes shared feature maps for the entire input image only once. Besides, Fast R-CNN employs a multi-task loss for facilitating singe-stage training.

performance and speed, its detection speed at test time is still very slow due to the use of the slow selective search in generating proposals. To further improve the detection speed and accuracy, a Region Proposal Network (RPN), which can run on shared feature maps generated by detection networks, was proposed [58] (see Figure 2.13). This enables nearly cost-free region proposals and significantly improves the detection speed of Fast R-CNN. Based on this, a single, unified network for object detection called Faster R-CNN was proposed [58]. Faster R-CNN consists of two main modules: a RPN that proposes regions and a Fast R-CNN detector that classifies the regions and refines their bounding boxes. Faster R-CNN is very accurate; however, it is quite slow due to the use of a costly per-region sub-network hundreds of times per image [9].

To address the existing issues of region-based detectors such as Fast R-CNN [15] and Faster R-CNN [24] and to further improve their speed and accuracy, Region-based Fully Convolutional Network (R-FCN) was proposed [9]. Instead of applying a costly per-region sub-network hundreds of times, R-FCN adopts a fully convolutional architecture with almost all computations shared across the entire image. To address the dilemma between translation-invariance in image classification and translation-variance in object detection, R-FCN proposes novel position-sensitive score maps that allow fully convolutional networks to effectively and efficiently perform both classification and detection in a single evaluation. With those novel improvements, R-FCN can run at 2.5-20 times faster and achieves higher accuracy than the Faster R-CNN counterpart.

To explicitly address multi-scale problems of CNNs, especially in object detection, Lin et al. [38] proposed Feature Pyramid Network (FPN) architecture. FPNs can create a feature pyramid that has strong semantics at all scales based on the pyramidal shape of a CNN's feature hierarchy. Specifically, FPNs employ a top-down pathway and lateral connections to combine low-resolution, semantically strong features with high-resolution, semantically weak features. To improve the performance of Faster R-CNN, FPNs were employed both in RPN for proposal generation and in Fast R-CNN for object detection.

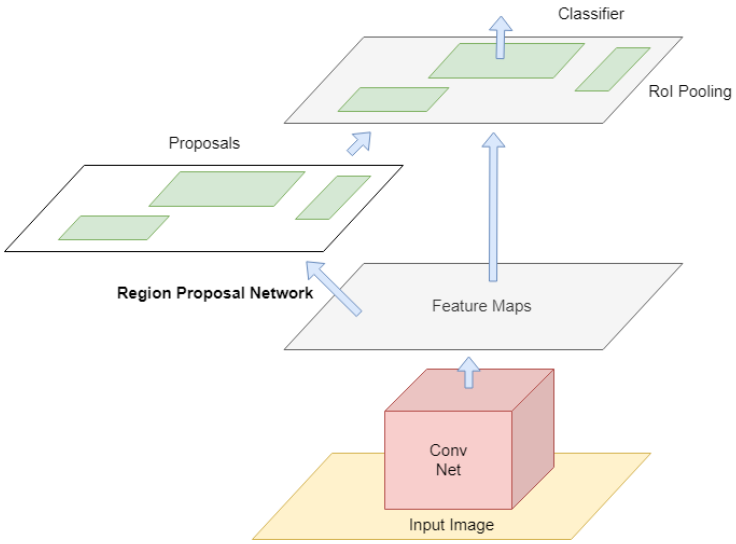


Figure 2.13: Illustration of Faster R-CNN. To speed up Fast R-CNN, Faster R-CNN replaces the slow selective search by a RPN, which can run on shared feature maps generated by detection networks.

With a marginal extra cost, FPNs significantly improve the performance of Faster R-CNN, especially in detecting small objects.

Mask R-CNN [21] extends Faster R-CNN for simultaneously detecting objects in an image and generating a high-quality segmentation mask for each instance. Specifically, Mask R-CNN adds an additional branch to the detection network of Faster R-CNN to output a binary mask for each RoI. To fix the misalignment between the network inputs and outputs of Faster R-CNN, a simple, quantization-free layer, called RoIAlign, was proposed and employed for preserving exact spatial locations. To train the network from end-to-end, a multi-task loss, which is a combination of a classification loss, a bounding-box loss, and a mask loss, was proposed. The classification loss and the bounding-box loss are identical as those employed in Faster R-CNN, while the mask loss is the average binary cross-entropy loss applied on K binary masks, one for each of the K classes.

2.4.2 One-stage Object Detectors

Two-stage object detectors are accurate; however, they are typically slow due to their complex multi-stage pipelines [54]. With the aim of facilitating real-time object detection, many single-shot object detectors, which take only one single shot to detect multiple objects within the image, have been proposed.

The OverFeat detector [64] is one of the first successful modern CNN-based one-stage object detectors. OverFeat is built based on three main ideas: First, a CNN is employed at multiple locations in the image, in a sliding window fashion, and over multiple scales to detect objects of different sizes and in different positions within the image. The second idea is to train the network not only for classifying each window but also for predicting the location and the size of the bounding box containing the object relative to the window. The third idea is to accumulate bounding boxes instead of suppressing them in order to increase detection confidence. By employing the three ideas, OverFeat can efficiently perform sliding window detection. However, OverFeat is still a disjoint system. In addition, since

OverFeat cannot reason about global context, it requires significant post-processing to produce coherent detections.

With the aim of addressing the existing issues of region proposal-based detectors that use complex pipelines and sliding window techniques that can not reason about global context, You Only Look Once (YOLO) was proposed [54]. YOLO is a real-time object detection framework that directly predicts bounding boxes and class probabilities with a single network in a single evaluation. To achieve this, YOLO unifies region proposal and region classification into a single neural network and, according to the authors, "frames object detection as a regression problem to spatially separated bounding boxes and associated class probabilities". Specifically, YOLO divides the input image into a $S \times S$ grid. Each grid cell predicts B bounding boxes and C conditional class probabilities, $P(Class_i|Object)$. Each bounding box consists of 5 predictions: x, y, w, h, and confidence c that represents the Intersection over Union (IoU) between the predicted box and any ground truth box. The (x, y) coordinates represent the center of the predicted box relative to the bounds of the grid cell; w and h are the width and height of the predicted box relative to the whole image, respectively. Thus, the output of YOLO is a $S \times S \times (B * 5 + C)$ tensor (see Figure 2.14). YOLO architecture is inspired by the GoogleLeNet model for

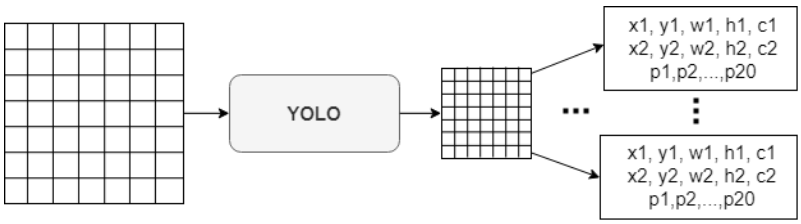


Figure 2.14: Illustration of YOLO. YOLO divides the input image into a $S \times S$ grid. Each grid cell predicts B bounding boxes each consists of 5 predictions (x, y, w, h, c) and C conditional class probabilities $(p_1, p_2, \ldots, p_C)$. In this illustration, we use $S = 7$, $B = 2$, and $C = 20$.

image classification [75]. Specifically, YOLO has 24 convolutional layers, followed by two fully-connected layers (see Figure 2.15). With a unified architecture, YOLO is extremely fast; it processes images in real-time. However, YOLO is not state-of-the-art in terms of accuracy. YOLO has two major drawbacks: First, it makes a considerable number of localization errors. Second, YOLO has relatively low recall compared to region proposal-based methods, such as Fast R-CNN.

To address the two main drawbacks of YOLO, which are high localization errors and low recall, YOLOv2 was proposed [55]. YOLOv2 enhances YOLO by combining a variety of existing ideas with some novel concepts. First, YOLOv2 adopts batch normalization on all the convolutional layers to regularize the model and improve its convergence. Next, a higher resolution classifier is employed. Next, instead of using fully-connected layers to predict the coordinates of bounding boxes directly, YOLOv2 utilizes convolutional layers to predict locations of anchor boxes. Then, to find good priors on anchor box dimensions, k-means clustering is employed. Further, direct location prediction is employed, and fine-grained features are adopted by adding a passthrough layer. Finally, multi-scale training is utilized in order to make the model robust to input images of different sizes, and a light-weight base model is employed for making predictions faster. In addition, a joint training algorithm that can train object detectors on both detection and classification data was proposed. Using this method, the authors trained YOLO9000, a real-time object detector

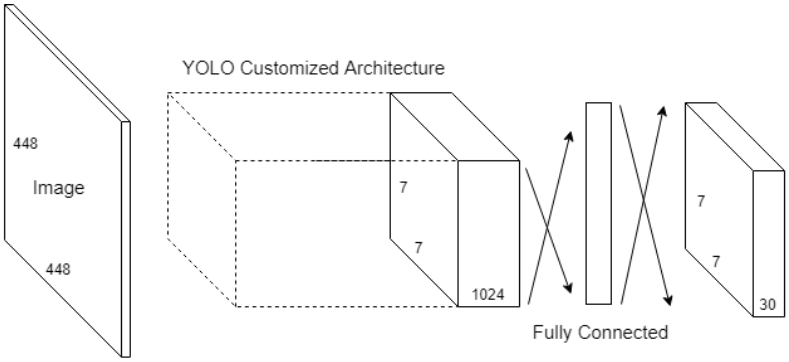


Figure 2.15: Illustration YOLO architecture. YOLO architecture is inspired by the GoogleLeNet model for image classification [75]. Specifically, YOLO has 24 convolutional layers, followed by two fully-connected layers.

that can detect over 9000 different object categories.

Recently, YOLOv3 was proposed by making some major design changes on YOLOv2 [56]. Specifically, YOLOv3 employs logistic regression to predict an objectness score for each bounding box instead of the sum of squared error loss. For class prediction, the softmax layer is replaced by multiple independent logistic classifiers. YOLOv3 adds several convolutional layers after the base feature extractor and predicts boxes at three different scales among these layers. A new base model called Darknet-53 is employed. Darknet-53 is similar to the original darknet architecture [53], but has residual blocks added.

Single Shot multibox Detector (SDD) improves YOLO by adding a series of modifications: (i) predictions are performed at multiple feature maps from the later stages of a network to enable detection at multiple scales; (ii) small convolutional filters are utilized to predict object classes and offsets in bounding box locations; and (iii) separate predictors (filters) are employed for predicting objects at different aspect ratios [41]. Specifically, SDD employs the VGG-16 model [66] pre-trained on the ImageNet dataset as its base network for extracting useful features. Instead of operating on a single scale feature map like YOLO [54] and Overfeat [64], SDD adds extra convolutional layers (feature layers), which decrease in size progressively, to the end of the base network to allow predictions of detections at multiple scales. Unlike YOLO, which uses a fully-connected layer for producing predictions, SDD attaches a set of small convolutional filters to each added feature layer (or optionally to an existing layer from the base network) and employs them for predicting object classes and offsets in bounding box locations (see Figure 2.17). SDD applies default boxes, which are similar to the anchor boxes used in Faster R-CNN [58], to several feature maps of different resolutions. These modifications allow SDD to effectively detect objects at multiple scales and aspect ratios, and make SDD both faster and more accurate than the YOLO counterpart.

SDD was enhanced in [14] by adding large-scale context to the detection network. To achieve this, SDD is first combined with Residual-101 network [24]. The SDD+Residual-101 detector is then augmented with deconvolution layers to introduce large-scale context. The resulting detector is called Deconvolutional Single Shot Detector (DSSD). By introducing large-scale context to the detection network and employing a better base network, DSSD can address the weakness of small object detection in SDD and improve its performance. Specifically, DSSD is able to match the speed of other detectors while surpassing

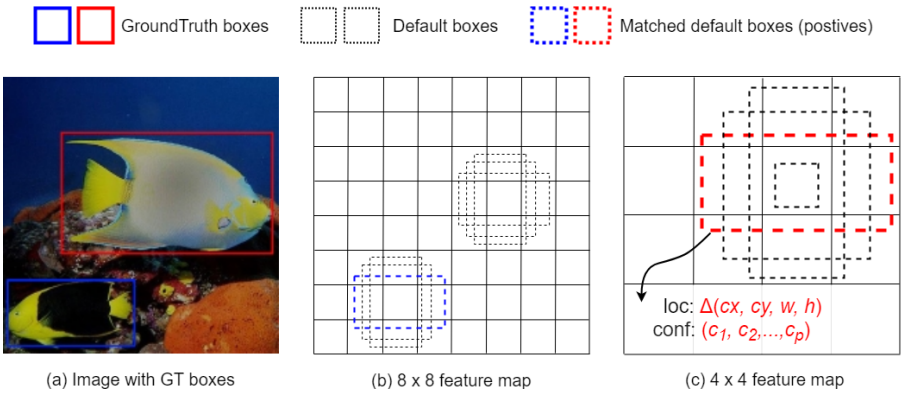


Figure 2.16: Illustration of SDD. SDD employs feature maps with different scales (e.g., 8×8 in (b) and 4×4 in (c)) and default boxes of different aspect ratios to enable detection at multiple scales and aspect ratios. For each default box, SDD predicts both shape offsets $\Delta(cx, cy, w, h)$ and the confidences for all object categories $(c_1, c_2, \ldots, c_p)$. During training, matched default boxes are treated as positives, and the rest of the default boxes are treated as negatives.

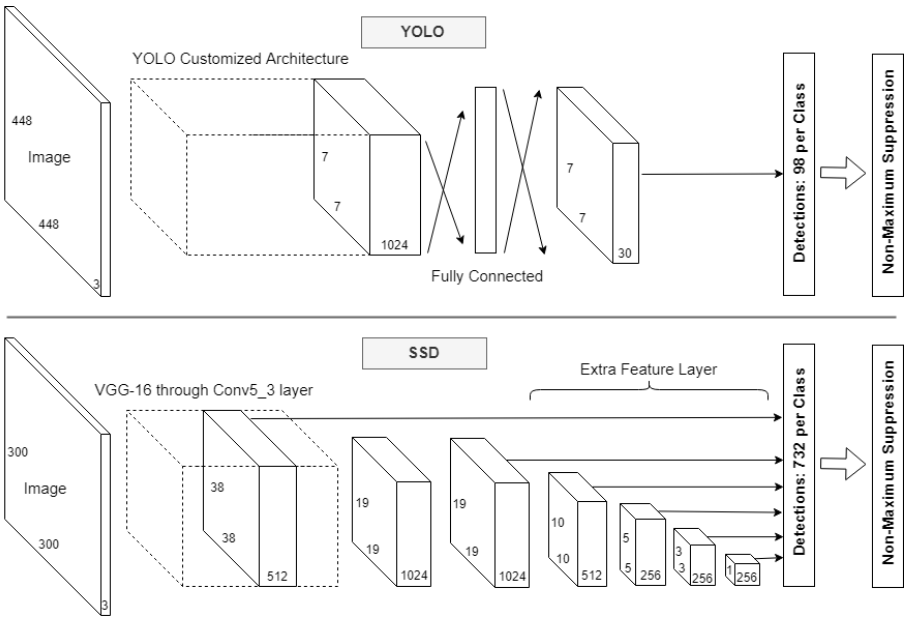


Figure 2.17: A comparison between YOLO and SDD. SDD adds extra convolutional layers (feature layers), which decrease in size progressively, to the end of the base network to allow predictions of detections at multiple scales.

the accuracy of both the previous SDD framework and other state-of-the-art object detectors such as Faster R-CNN and R-FCN.

Lin et at. [39] identified class imbalance during training as the main obstacle preventing one-stage detectors (e.g., SDD and YOLO) from achieving the state-of-the-art accuracy and proposed to address that by introducing a novel loss function named Focal Loss (FL). FL dynamically scales the standard cross-entropy loss with a scaling factor that decays to zero as confidence in the correct class increases. By doing that, FL automatically reduces the weights of easy examples during training and allows the model to focus on hard examples. To demonstrate the effectiveness of the proposed FL, a simple one-stage

object detector called RetinaNet was designed [39]. RetinaNet adopts some existing ideas and concepts from previous dense detectors. Specifically, RetinaNet adopts the FPN [38] as the backbone network and employs the concept of translation-invariant anchor boxes introduced by RPN [58]. When trained with FL, RetinaNet achieves similar speed as previous one-stage detectors while outperforming the accuracy of the state-of-the-art two-stage detectors.

2.4.3 Summary

In this section, we have briefly introduced object detection and have highlighted some of the most well-known two-stage object detectors (e.g., Faster R-CNN, R-FCN, and FPN) and one-stage object detectors, for example, YOLO, SDD, and RetinaNet. Because of simple architecture, one-state detectors are typically very fast; however, their accuracy usually trails that of two-stage methods. Recently, class imbalance was identified as the primary obstacle preventing one-stage detectors from surpassing top-performing, two-stage methods. To address this problem, focal loss was proposed. With the help of the focal loss, one-stage detectors such as RetinaNet is able to match the speed of one-stage detectors while surpassing the accuracy of two-stage detectors.

2.5 Few-shot Learning

The availability of large datasets (such as ImageNet [10] and Microsoft COCO [40]), advances in GPU-accelerated computing, and streamlined designs of deep neural networks, have enabled deep learning methods to achieve great success in a variety of AI-related tasks. This is especially the case in computer vision, such as image classification, object detection, and image segmentation. However, most of these successes are based on conventional supervised end-to-end learning approaches, which typically require lots of labeled data to train and are prone to overfitting when only a small amount of training data is available. In addition, these approaches are typically not able to generalize well to changing tasks. To avoid overfitting and to improve the generalization ability of conventional deep learning models, many researchers have relied on regularization (e.g., batch normalization [30] and dropout [69]) and data augmentation. These approaches work well on medium-sized (or sometimes even small-sized) datasets. However, they typically fail in extreme cases where only one or a few examples per class are available.

Humans are, on the other hand, capable of learning new concepts quickly from only one or a few examples, i.e., one-shot or few-shot learning, by effectively utilizing prior knowledge and experience. For example, a child who has learned what a horse looks like can rapidly transfer their knowledge to learn what a zebra looks like from just one or a few example images.

Inspired by humans' ability to learn new concepts quickly, there has been a recent resurgence of interest in designing specialized deep learning models for one-shot and few-shot learning tasks. In this section, we focus mainly on few-shot learning. One of the most common few-shot learning tasks is few-shot classification in which the goal is to adapt a classifier to previously unseen classes from just a handful of labeled examples per class.

In the past few years, many few-shot classification approaches have been proposed. These approaches can be roughly categorized as (i) learning to fine-tune approaches; (ii)

sequence-based approaches; (iii) generative modeling-based approaches; (vi) distance metric learning-based approaches; (v) deep distance metric learning-based approaches; and (vi) semi-supervised approaches. The literature on few-shot learning is vast; we present in this section a short summary of well-known approaches and works most relevant to ours. We refer the reader to [84] and [80] for more detailed reviews on few-shot learning.

2.5.1 Task Description

Before reviewing different few-shot classification approaches. We begin by detailing the few-shot learning task. In the traditional machine learning setting, we are typically given a dataset D. This dataset is usually split into two parts: D_{train} and D_{test}. The former is often used for training the parameters $\boldsymbol{\theta}$ of the model, while the latter is typically used for evaluating its generalization. In general few-shot learning, we are dealing with meta-datasets D_{meta} containing multiple regular datasets D [60]. Each dataset $D \in D_{meta}$ has a split of D_{train} and D_{test}; however, they are usually much smaller than that of regular datasets used in the traditional machine learning setting. Let $C = \{1, \ldots, K\}$ be the set of all classes available in D_{meta}. The set C is usually split into two disjoint sets: C_{train} containing training classes and C_{test} containing unseen classes for testing, i.e., $C_{train} \cap C_{test} = \emptyset$. The meta-dataset D_{meta} is often split into two parts: The first is a meta training set $D_{meta-train} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, where $\mathbf{x}_i$ is the feature vector of the i^{th} example, $y_i \in C_{train}$ is its corresponding label, and N is the number of training examples. The second part is a meta testing set $D_{meta-test}$. In a standard M-way K-shot classification task, the meta testing set $D_{meta-test}$ consists of a support set and a query set. The support set $S = \{(\mathbf{x}_j, y_j)\}_{j=1}^{N_S}$ contains K examples from each of the M classes from C_{test}, i.e., the number of support examples are $N_S = M \times K$ and $y_j \in C_{test}$. The query set contains N_Q unlabeled examples $Q = \{(\mathbf{x}_j)\}_{j=N_S+1}^{N_S+N_Q}$. The support set is employed by the model for learning the new task, while the query set is utilized by the model for evaluating its performance.

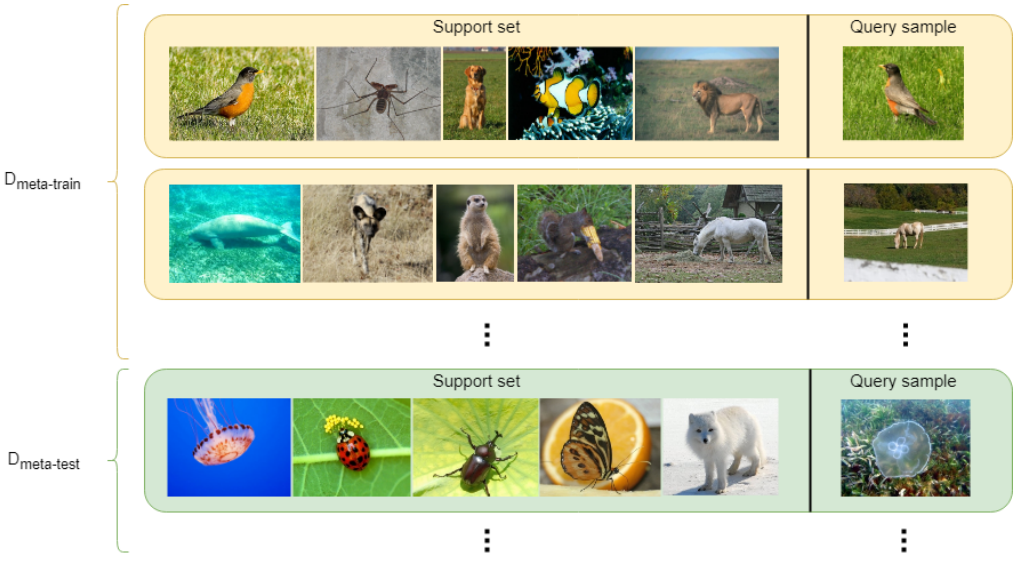


Figure 2.18: Illustration of a meta training set $D_{meta-traing}$ and a meta testing set $D_{meta-test}$ for the 5-way, 1-shot classification task. In this illustration, for each dataset, we have one example from each of five classes in the training set and one example for evaluation in the test set.

2.5.2 Learning to Fine-Tune Approaches

Finn et al. [13] propose a Model-Agnostic Meta-Learning (MAML) approach to learn a model's initial parameters such that it can be quickly adapted to a new task through only one or a few gradient update steps. In other words, MAML aims at learning a good internal representation that is broadly suitable for many tasks, and from there, good results can be achieved by simply fine-tuning the model slightly via one or a few weight update steps (see Figure 2.19). Specifically, MAML employs a base model, which is a neural network, $f_{\boldsymbol{\theta}}$ with parameters $\boldsymbol{\theta}$. Given a task T_i and its associated dataset $(D^{(i)}_{train}, D^{(i)}_{test})$, the base model parameters $\boldsymbol{\theta}$ are updated by one or a few gradient descent steps on the training set $D^{(i)}_{train}$ to produce $\boldsymbol{\theta}'_i$. The model parameters $\boldsymbol{\theta}$ are trained by optimizing for the performance of $f_{\boldsymbol{\theta}'_i}$ on the testing set $D^{(i)}_{test}$ with respect to $\boldsymbol{\theta}$ across tasks T_i sampled from the distribution over tasks $P(T)$. Since MAML is model-agnostic by design, it can handle any model representation that is amenable to gradient-based training and is applicable to a variety of different problems such as classification, regression, and reinforcement learning.

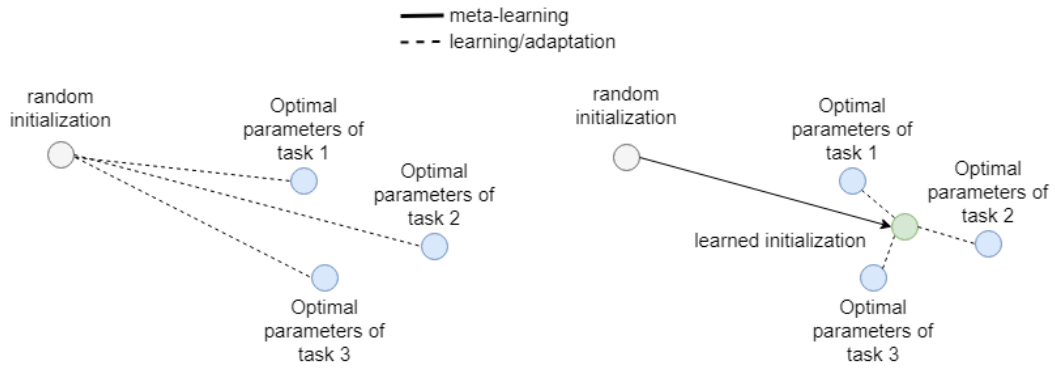


Figure 2.19: A comparison between traditional learning/adaptation (left) vs MAML (right). From a random initialization, traditional learning/adaptation algorithms need to make many weight updates to find optimal parameters for the target task. MAML, on the other hand, learns a good initialization that can significantly reduce the number of weight updates that learning/adaptation algorithms have to make.

Ravi and Larochelle [60] propose a Long Short-Term Memory (LSTM)-based meta-learner model not only to learn a good initialization for another learner (classifier) network that allows for quick training convergence but also to discover the exact optimization algorithm that can be employed for training the learner in the few-shot regime.

Although these approaches can handle many model representations, they both suffer from the need to fine-tune on the target problem, which makes them less appealing to few-shot learning.

2.5.3 Sequence Based Approaches

Santoro et al. [1] propose a method for few-shot classification based on Memory-Augmented Neural Networks (MANNs). The authors modify the Neural Turing Machines (NTMs) [2], which have the ability to quickly encode and retrieve new information using external memory, to excel at one-shot learning. The authors introduce a new method for accessing external memory, called the Least Recently Used Access (LRUA), which only uses

content-based location.

Mishra et al. [45] formalize meta-learning as a sequence-to-sequence problem and propose a novel class of model architectures, called the Simple Neural AttenIve Learner (SNAIL), to resolve the problem of existing approaches in quickly incorporating and referring to past experience. SNAIL employs a novel combination of temporal convolutions and soft attention, which enables the meta-learner to aggregate contextual information from past experience and allows it to pinpoint specific pieces of information, respectively.

While appealing, these methods typically require complex RNN architectures and complicated mechanisms for storing/retrieving all the historical information of relevance, both long-term and short-term, without forgetting [72].

2.5.4 Generative Modeling Based Approaches

Zhang et al. [92] argue that it may be easier to form a decision boundary between objects that look very different, for example, cats and cars, than between objects that look very similar, such as cats and dogs. Thus, it is difficult for conventional few-shot learning approaches to extract the correct features to separate similar classes (e.g., cats and dogs) if they are not in the training data. Based on this, the authors propose an adversarial training based framework called MetaGAN with the aim of providing additional signals to the classifiers and making the decision boundaries much sharper. MetaGAN casts the classifier in conventional few-shot learning approaches as a discriminator and employs an imperfect generator to provide fake data between the manifolds of different real data classes. Since the discriminator is forced to not only classify real classes but also to distinguish between real/fake classes, it has to extract stronger features that typically lead to much sharper decision boundaries between real classes.

Wang et al. [85] propose a novel approach to few-shot learning based on learning to hallucinate additional examples. The authors combine the "learning to learn" [77] and "learning to augment" [85] ideas by employing a hallucinator to produce additional training examples to allow the classification algorithm to learn a better classifier. The authors argue that the aim of the hallucinator should be to hallucinate examples that are useful for learning classifiers instead of diversity or realism and propose to train the classification algorithm and the hallucinator jointly.

2.5.5 Distance Metric Learning-Based Approaches

The basic idea of distance metric learning-based approaches is to learn a non-linear mapping of the input into an embedding space and define a metric distance which maps similar examples close and dissimilar ones distant in the embedding space, so that a query example can be easily classified by, for example, using nearest neighbor methods. The success of these methods relies heavily on the choice of the distance metric function. In the past few years, many fixed metric distance functions (e.g., the Euclidean distance [67] and the cosine distance [81]) and learnable deep metric distance functions, such as [72], have been applied to few-shot classification models. In this section, we review some of the most well-known distance metric learning-based approaches that employ fixed metric distance functions. Methods that utilize learnable deep metric distance functions are reviewed in the next section.

Snell et al. [67] propose a simple method called Prototypical Networks (PNs) for few-shot learning based on the assumption that there exists an embedding space in which

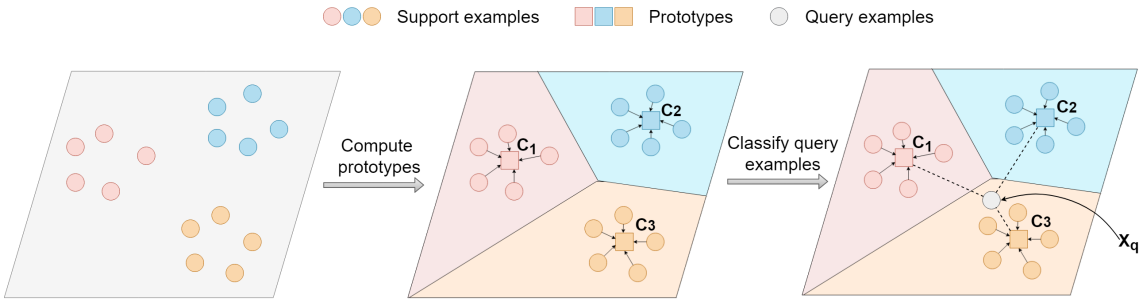


Figure 2.20: Illustration of prototypical networks. A prototype for each class in the embedding space is generated by taking the mean of the embeddings of its support examples. An embedded query point is classified by simply finding the nearest class prototype in the embedding space based on the squared Euclidean distance metric.

samples from each class cluster around a single prototype representation, which is simply the mean of the individual samples. Specifically, PNs learn a non-linear embedding function $f_{\boldsymbol{\phi}} : \mathbb{R}^D \rightarrow \mathbb{R}^E$ parameterized by $\boldsymbol{\phi}$ that maps a D-dimensional feature vector of an example $\mathbf{x}_i$ to an E-dimensional embedding $\mathbf{z}_i = f_{\boldsymbol{\phi}}(\mathbf{x}_i)$ [67]. In meta-testing, the embedding function $f_{\boldsymbol{\phi}}$ is employed for mapping examples in the support set $S = \{(\mathbf{x}_j, y_j)\}_{j=1}^{N_S}$ into the embedding space. An E-dimensional representation $\mathbf{c}_k$, or prototype, of each class is computed by taking the mean of the embedded support points belonging to the class:

$$\mathbf{c}_k = \frac{1}{|S_k|} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_{\boldsymbol{\phi}}(\mathbf{x}_i) = \frac{1}{|S_k|} \sum_{(\mathbf{x}_i, y_i) \in S_k} \mathbf{z}_i, \tag{2.27}$$

where S_k is the support set of class k. An embedded query point $\mathbf{x}_q$ is then classified by simply finding the nearest class prototype in the embedding space based on the squared Euclidean distance metric. To train PN, the episodic training strategy proposed in [81, 60] is adopted. Specifically, to train PN for the M-way, K-shot classification task, a training episode is formed from the meta training set D_{meta_train} as follows: K examples from each of M randomly selected classes from C_{train} are sampled to form a support set $S = \{S_1, \ldots, S_M\}$. A query set $Q = \{Q_1, \ldots, Q_M\}$ is formed by sampling from the rest of the M classes' samples. Next, for each class k, its support set $S_k \in S$ is used for computing a prototype using Equation 2.27. Then, a distribution over classes for each query point $\mathbf{x}_q \in Q$ based on a softmax over distances to the prototypes in the embedding space is produced:

$$p_{\boldsymbol{\phi}}(y = k | \mathbf{x}_q) = \frac{\exp(-d(f_{\boldsymbol{\phi}}(\mathbf{x}_q), \mathbf{c}_k))}{\sum_{k'} \exp(-d(f_{\boldsymbol{\phi}}(\mathbf{x}_q), \mathbf{c}_{k'}))}, \tag{2.28}$$

where $d = \mathbb{R}^E \times \mathbb{R}^E \rightarrow [0, +\infty)$ is a distance function. Based on that, the PN is trained by minimizing the negative log-probability of the true class k via SGD:

$$J(\boldsymbol{\phi}) = -\frac{1}{M} \sum_{k=1}^{M} \frac{1}{|Q_k|} \sum_{\mathbf{x}_q \in Q_k} \log p_{\boldsymbol{\phi}}(y = k | \mathbf{x}_q). \tag{2.29}$$

The training is repeated with new, randomly generated training episodes until a stopping criterion is met.

Algorithm 1 PN's training episode loss computation. $\mathcal{D}_{meta-train} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ is the meta-training set, where $\mathbf{x}_i$ is the feature vector of the i^{th} example, $y_i \in \{1, \ldots, K\}$ is its corresponding label, K is the number of classes in $\mathcal{D}_{meta-traing}$, and N is the number of training examples. $\mathcal{D}_k = \{(\mathbf{x}_j, y_j) \in \mathcal{D} \mid y_j = k\}$ is the meta-training set of class k. $N_C \leq K$, N_S, and N_Q are the number of classes per episode, the number of support examples per class, and the number of query examples per class, respectively. $\mathrm{RS}(S, N)$ is a function that returns a set of N elements chosen uniformly at random from set S, without replacement.

Require: Meta-training set $\mathcal{D}_{meta-train} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$.
Ensure: The loss $J(\boldsymbol{\phi})$ for a randomly generated training episode.

$V \leftarrow \mathrm{RS}(\{1, \ldots, K\}, N_C)$ ▷ Select class indices for an episode
for k in $\{1, \ldots, N_C\}$ **do**
  $S_k \leftarrow \mathrm{RS}(\mathcal{D}_{V_k}, N_S)$ ▷ Select support examples
  $Q_k \leftarrow \mathrm{RS}(\mathcal{D}_{V_k} \setminus S_k, N_Q)$ ▷ Select query examples
  $\mathbf{c}_k \leftarrow \frac{1}{N_S} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_{\boldsymbol{\phi}}(\mathbf{x}_i)$ ▷ Compute prototype from support examples
end for

$$J(\boldsymbol{\phi}) = \frac{1}{N_C} \sum_{k=1}^{N_C} \frac{1}{N_Q} \sum_{\mathbf{x}_q \in Q_k} \left[d(f_{\boldsymbol{\phi}}(\mathbf{x_q}), \mathbf{c}_k) + \log \sum_{k'} \exp(-d(f_{\boldsymbol{\phi}}(\mathbf{x_q}), \mathbf{c}_{k'})) \right]$$

Garcia and Bruna [61] argue that few-shot learning, which aims at propagating label information from labeled support examples towards unlabeled query images, can be formalized as a posterior inference over a graphical model determined by the images and labels in the support set and the query set. The authors cast posterior inference as message passing on graph neural networks and propose a graph-based model, which can be trained end-to-end, to solve the task. The authors further extend the algorithm for semi-supervised few-shot learning and active few-shot learning.

With the aim of improving the generalization capacity of metric-based methods for few-shot learning, Wang et al. propose to enforce a large margin between the class centers [90]. To do this, the authors propose to augment a large margin loss function to the standard softmax loss function for classification. The unnormalized triplet loss [63] is chosen to be the large margin distance function. The authors also provide experimental results with other existing large margin distance functions, including the normalized triplet loss, the normalized contrastive loss [20, 71], the normface loss [82], the cosface loss [83], and the arcface loss [11], and conclude that the unnormalized triplet loss is more robust than the above-mentioned loss functions. Experimental results show that the proposed approach slightly improves the performance of existing metric distance learning-based models such as Graph Neural Networks (GNNs) [61] and PNs [67].

2.5.6 Deep Distance Metric Learning-Based Approaches

To avoid the need of manually choosing the right distance metric (e.g., the Euclidean distance and the cosine distance), Sung et al. [72] propose a two-branch Relation Network (RN) which can learn both a deep embedding and a deep non-linear metric (similarity function) for comparing images in the embedding space (see Figure 2.21). Specifically, the RN consists of two modules: an embedding module $f_{\boldsymbol{\varphi}}$ and a relation module $g_{\boldsymbol{\phi}}$. The

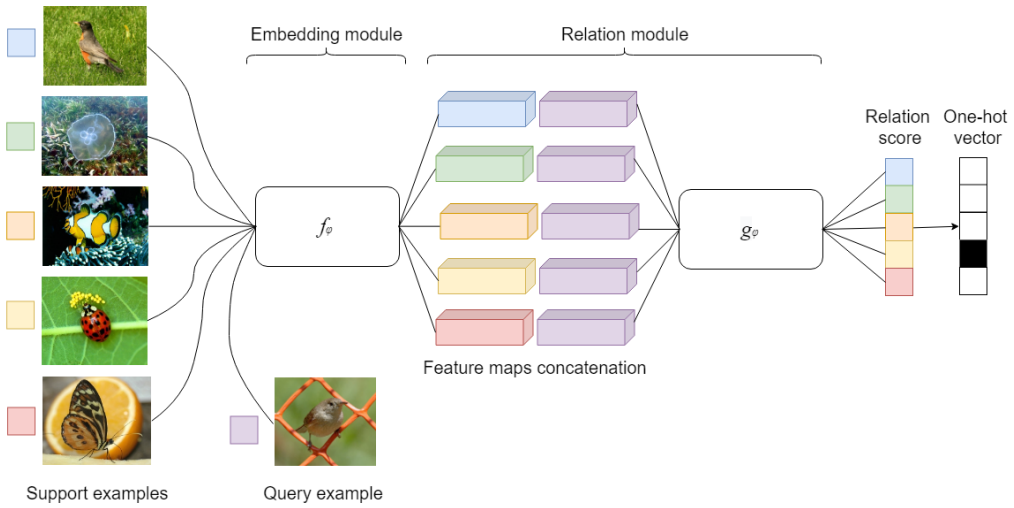


Figure 2.21: Illustration of Relation Network architecture for a 5-way, 1-shot classification task with one query example.

embedding module takes support examples $\mathbf{x}_i \in S$ and query examples $\mathbf{x}_j \in Q$ as inputs and produces feature maps $f_{\boldsymbol{\varphi}}(\mathbf{x}_i)$ and $f_{\boldsymbol{\varphi}}(\mathbf{x}_j)$, respectively. For the 1-shot classification task, the features maps $f_{\boldsymbol{\varphi}}(\mathbf{x}_i)$ and $f_{\boldsymbol{\varphi}}(\mathbf{x}_j)$ are concatenated in depth before being fed into the relation module $g_{\boldsymbol{\phi}}$ for producing a relation score $r_{i,j}$, which is a scalar in range of 0 to 1 representing the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$. For the K-shot classification task where $K > 1$, a class feature map for each class is produced by element-wise summing over the embedding module outputs of its support examples. The class feature map is then used in the exact same way as the feature map $f_{\boldsymbol{\varphi}}(\mathbf{x}_i)$ in the 1-shot classification case. To train RN, the authors propose to use the Mean Square Error (MSE). The main goal is to regress the predicted relation score $r_{i,j}$ to the ground truth relation score, which is 1 for matched pairs and 0 for mismatched pairs.

Although deep distance metric learning-based approaches can avoid the need for manually choosing the right distance metric, they are prone to overfitting and are more difficult to train compared to distance metric learning-based approaches due to the added parameters.

2.5.7 Semi-Supervised Approaches

To take advantage of both labeled and unlabeled data, Boney and Ilin [6] propose to extend prototypical networks to address the semi-supervised few-shot learning problem. Based on the observation that prototypical networks tend to produce clustered data representations, the authors cast the semi-supervised few-shot learning problem as a semi-supervised clustering problem and address it by applying guided hard K-means clustering in the embedding space found by prototypical networks at test time. The K-means clustering process is guided by the labeled examples, which are used for initializing the cluster means.

A similar approach was concurrently developed by Ren et al. [57]. However, the authors apply clustering both at testing and at training to refine the prototypes produced by prototypical networks. To keep the inference differentiable, soft K-means is applied instead of hard K-means. In addition, the authors consider a more challenging situation

where unlabeled examples can come from distractor classes and propose a soft-masking mechanism to learn to include or ignore entirely certain unlabeled examples in the prototype refining process.

2.5.8 Summary

In this section, we have briefly introduced few-shot learning and have reviewed some of the most well-known approaches including (i) learning to fine-tune approaches; (ii) sequence-based approaches; (iii) generative modeling-based approaches; (vi) distance metric learning-based approaches; (v) deep distance metric learning-based approaches; and (vi) semi-supervised approaches. Among these categories, distance metric learning-based approaches are typically preferred since they are simpler and more efficient than other few-shot learning approaches, which require complex inference mechanisms, complex RNN architectures, or fine-tuning the target problem.

Chapter 3

Case

This chapter describes the special case that forms the basis of the work on which this dissertation is built, which is to develop an automatic autonomous vision-based power line inspection system.

3.1 Case

With the aim of exploiting recent advances in deep learning, especially CNNs, and UAV technologies for facilitating automatic autonomous vision-based power line inspection, eSmart Systems[1] initiated a project, code-named Connected Drone, in 2016. The first phase of the project (Connected Drone 1) ran from 2016 to the end of 2018 and involved 12 Norwegian power grid companies (such as Hafslund and Ringeriks-Kraft), universities (NORUT/ASUF, NTNU AMOS, and UiO), technology partners (Telenor, Microsoft, Teleplan Globe, and Eker Design), and drone experts (IRIS Group and Robot Aviation). The second phase of the project (Connected Drone 2) is expected to run from the end of 2019 to at least until the end of 2022 and involves at present 22 Norwegian utilities.

The work represented by this dissertation is funded by the Research Council of Norway and eSmart Systems as an industrial Ph.D. project in collaboration with the UiT Machine Learning Group[2].

In this project, with the aim of taking advantage of recent advances in deep learning and UAV technologies to realize fast, accurate, and safe automatic vision-based power line inspection, we propose as a new potential solution an automatic autonomous vision-based power line inspection concept that uses UAV inspection as the main inspection method, optical images as the primary data source, and deep learning as the backbone of data analysis. The concept consists of two main modules: an automatic inspection module and an autonomous inspection module. The former module is responsible for automatically analyzing inspection images taken from UAVs using deep learning models for identifying power line components and defects, while the latter module is responsible for performing inspections autonomously with self-driving UAVs. Specifically, the concept works as follows: First, UAVs equipped with high-resolution cameras are flown along power lines and circled around power masts to take pictures of the masts, the conductors, power line components (e.g., insulators, poles, and cross arms), and surrounding objects (e.g., vegetation) from different angles. In automatic inspection systems, UAVs are flown by

[1] eSmart Systems: `https://www.esmartsystems.com/`

[2] UiT Machine Learning Group: `https://machine-learning.uit.no`

pilots; in fully automatic autonomous inspection systems, on the other hand, self-driving UAVs are employed for eliminating the need for pilots. For each power mast, around 20 (or more) images at 6048x4032 (or higher) resolution are collected. The images are uploaded to the Microsoft Azure cloud after the flight for inspection. Next, the collected images are analyzed by our mast detection and component detection models for detecting common power line components (e.g., insulators, poles, and cross arms). The detected components are then classified into more fine-grained power line component classes using our component classification models and used as inputs for identifying defects. Finally, images with potential defects will be assigned a higher priority in the inspection queue with the aim of reducing inspection time.

3.2 Implementation

The implementation of the concept is divided into two main phases:

Phase 1 Employ deep learning for enabling automatic analysis of power line inspection images and for facilitating automatic vision-based power line inspection.

Phase 2 Employ deep learning for facilitating self-driving UAVs and for realizing fully automatic autonomous vision-based power line inspection.

In this dissertation, we focus on implementing phase 1. We first identify six main challenges of DL vision-based UAV inspection: (i) the lack of training data; (ii) class imbalance; (iii) the detection of small power line components and defects; (iv) the detection of power lines in cluttered backgrounds; (v) the detection of previously unseen components and defects; and (vi) the lack of metrics for evaluating inspection performance. Next, we propose approaches for addressing the identified challenges. Based on that, we build a system that is capable of detecting a wide range of power line components (e.g., masts, insulators, poles, cross arms, top caps, transformers, chain shackles, cotter pins, pylon numbers, dampers, and weights) and defects, for example, missing top caps, cracks in poles and cross arms, woodpecker damage on poles, rot damage on cross arms, broken insulators, flashed insulators, contaminated insulators, missing cotter pins, loose cotter pins, and rusty cotter pins. The system is deployed in the Microsoft Azure cloud. With auto-scale functionality and access to GPU virtual machines, the system has demonstrated its ability to analyze over 180,000 images per hour.

To pave the way for self-driving UAVs and for realizing fully automatic autonomous vision-based power line inspection, we propose approaches to advance deep learning for power line detection and for few-shot learning. Power line detection is important not only for inspection but also for navigating self-driving UAVs when GPS information is not available. Few-shot learning allows deep learning models to adapt to new tasks and to recognize new classes without retraining, which can quickly increase the range of power line components and defects that our system can detect. The proposed approaches have, in our opinion, paved the way for realizing fully automatic autonomous vision-based power line inspection. The implementation of phase 2, however, is left for future work due to time constraints.

More details about the proposed approaches and the implementation of phase 1 can be found in Chapter 4 and the four included papers in Appendix A, Appendix B, Appendix C, and Appendix D.

Chapter 4

Research Findings

This chapter summarizes the research findings of the four included papers, which can be found in Appendix A, Appendix B, Appendix C, and Appendix D. The papers are presented together with the research questions that they answer. Paper I reviews the possibilities and challenges of deep learning in vision-based UAV inspection, while Paper II, Paper III, and Paper IV focus on addressing the challenges identified in Paper I.

4.1 Paper I

4.1.1 Research Question

RQ1.1 What are the possibilities and challenges of deep learning in vision-based UAV inspection?

4.1.2 Abstract

This paper, which can be found in Appendix A, aims at providing a comprehensive overview of the possibilities and challenges of deep learning in vision-based UAV inspection. To achieve this, we conduct an extensive literature review on automatic vision-based power line inspection. Specifically, we first provide a survey of the inspection methods available at present, including foot patrol, helicopter-assisted, automated helicopter-assisted, climbing robots, and UAV inspection, with an emphasis on their advantages and disadvantages. Next, we outline the four well-suited tasks for automatic vision-based inspection: (i) mapping and inspection of power line components; (ii) vegetation encroachment monitoring; (iii) icing detection and measurement; and (iv) disaster monitoring. Then, we give a brief summary of potential data sources for vision-based power line inspection, including synthetic aperture radar images, optical satellite images, optical aerial images, thermal images, ultraviolet images, airborne laser scanner data, land-based mobile mapping data, and UAV data, with special attention paid to point out their applicable tasks. Further, we conduct a comprehensive review of the current state of the art of automatic vision-based power line inspection and highlight the existing issues. Based on that, we propose as a new potential solution an automatic autonomous vision-based power line inspection concept that uses UAV inspection as the main inspection method, optical images as the primary data source, and deep learning as the backbone of data analysis.

To facilitate the implementation of the concept, we first examine the possibilities of deep learning in vision-based UAV navigation and UAV inspection. In UAV navigation, DL

vision-based navigation approaches (e.g., pole detection-based and power line detection-based) together with traditional navigation approaches, such as GPS way points-based, and UAV autopilots can facilitate self-driving UAVs. In UAV inspection, deep learning holds great promise for realizing automatic data acquisition systems. In these systems, the outputs from the mast detection and component detection models can be employed for guiding cameras to focus when taking pictures of power line components. This can potentially eliminate the need for camera operators and significantly increase the quality of data acquisition. Besides, deep learning offers enormous potential for automating power line inspection tasks, such as mapping and inspection of power line components, icing detection and measurement, vegetation encroachment monitoring, and disaster monitoring. For example, CNN-based object detectors (e.g., Faster R-CNN [58], SDD [41], YOLO [54], and R-FCN [9]) and CNN-based image classifiers, such as ResNets [24], Inception architectures [74, 75, 76], and DenseNets [27], can be combined for detecting, for classifying, and for mapping and inspection of power line components from inspection images.

We then identify six main challenges of DL vision-based UAV inspection: The first is the lack of training data. Large amounts of labeled data are typically required for training deep learning models; however, to the best of our knowledge, there are no publicly available datasets that are big enough for training such models in vision-based UAV inspection. The second challenge is class imbalance [32], which comes from the long-tailed distribution of power line component classes. In inspection images, a small number of power line component classes, such as wooden poles and insulators, appear very often, while most of the other power line component classes and defects, for instance, missing top caps and bird nests, appear rarely. This typically causes deep learning models to bias towards classes that have more examples and overlook classes that have fewer examples. The third challenge is the detection of small power line components and defects. Most of the state-of-the-art CNN-based object detectors such as YOLO [54], SDD [41], Faster R-CNN [58], and R-FCN [9] perform poorly on very small objects [29]. This is a major problem in mapping and inspection of power line components since many important components (e.g., insulators, cotter pins, and chain shackles) and defects on power line components, for instance, missing top caps, broken insulators, and missing cotter pins, are very small in inspection images taken from UAVs or helicopters. The fourth challenge is the detection of power lines in cluttered backgrounds. Power lines in inspection images taken from UAVs or helicopters are typically very thin, leading to a lack of rich features for their representation. In addition, separating power lines from backgrounds is very challenging in some cases due to color similarity, weather conditions, and lighting conditions. The fifth challenge is the detection of previously unseen power line components and defects. Most of the existing power grids utilize a wide range of power line components. Thus, the grids are vulnerable to a huge number of types of defects. In addition, defects on power line components, even very simple ones such as missing top caps and broken insulators, can occur in many different forms. This, together with the lack of training data, poses a major challenge for DL vision-based UAV inspection due to the limitation of deep learning models in detecting and classifying previously unseen classes. The last but not least challenge is the lack of metrics for evaluating inspection performance. Since there are no publicly available datasets for power line inspection that have a significant number of examples of defects, how to evaluate the performance of DL vision-based UAV inspection systems still remains an unsolved problem.

4.1.3 Contributions by the author

The idea was conceived by me and further developed in collaboration with the other co-authors. I conducted the literature review, identified the possibilities and challenges of DL vision-based UAV inspection, and proposed solutions to the identified challenges. I wrote the manuscript draft of the paper.

4.2 Paper II

4.2.1 Research Question

RQ1.2 How and to what extent can the challenges of deep learning in vision-based UAV inspection be addressed?

4.2.2 Abstract

In this paper, which can be found in Appendix B, we propose solutions to the first three challenges of DL vision-based UAV inspection: (i) the lack of training data; (ii) class imbalance; and (iii) the detection of small power line components and defects. Specifically, we first mitigate the lack of training data by creating four medium-sized datasets for training component detection and component classification models. The datasets consist of high-resolution inspection images collected using high-quality DSLR cameras (e.g., Nikon D810, Canon EOS 5D Mark III, and Nikon D3X) from helicopters and with multiple resolutions (e.g., 7360×4912, 6048×4032, and 5760×3840). To increase the diversity of the data, we combine images from multiple power grids in Norway, which are provided by Hafslund Nett and Troms Kraft.

Next, to address the class imbalance and to further alleviate the lack of training data, we propose a series of effective data augmentation techniques based on transformations in the data-space to generate more training data. The first is adding Gaussian-distributed additive noise, which is employed to account for noise that arises during image acquisition (e.g., sensor noise caused by poor illumination and/or high temperature, and/or transmission). The second technique involves applying Gaussian blur to account for possible out of focus or grainy images. The blurring process is performed by convolving the images with a two-dimensional Gaussian function. To simulate more image camera distances and viewing angles, zoom and rotation operators are employed. The zooming process is performed by randomly cropping the images and scaling the crops back to their original sizes. The rotation is performed by multiplying the images with a rotation matrix. The final technique involves flipping the images horizontally and vertically.

Further, to tackle the detection of small power line components and defects challenge, we propose a multi-stage component detection and classification pipeline, which consists of three stages: (i) mast detection; (ii) component detection; and (iii) component classification. The pipeline works as follows: First, input images are fed into the mast detector for detecting power masts. Next, the detected masts are cropped from the input images and used as inputs for the component detector to detect power line components including, for example, top caps, poles, cross arms, and insulators. Then, the detected components are mapped to the original input images for cropping. Finally, the cropped components are passed through their corresponding component classifiers for classification and defect

identification. With a multi-stage architecture, the pipeline can mimic the "zoom-in" operation when analyzing inspection images. This enables the detection of small power line components (e.g., cotter pins, chain shackles, and top caps) and small defects, for example, cracks on poles and cross arms, woodpecker damage on poles, and missing cotter pins.

To train a more robust component detector for the proposed pipeline and to further mitigate the lack of training data, we utilize the outputs from the mast detector to generate more training data. Specifically, when a mast is detected, we pad its predicted bounding box to be a square and crop the square from the original image to generate one additional training image. We then slightly shift the square in four directions (left, right, top, and bottom) to generate four more training images. In addition, we randomly flip the training images vertically during training. These data augmentation techniques allow us to generate a training set that is 12 times bigger than our original training set.

To facilitate the implementation of the pipeline, we first evaluate the state-of-the-art CNN-based object detectors including YOLO [54], SDD [41], Faster R-CNN [58], and R-FCN [9] in terms of speed and mean Average Precision (mAP). We then review the state-of-the-art CNN-based image classifiers such as ResNets [24], Inception architectures [74, 75, 76], and DenseNets [27]. Based on that, we select SDD as our main object detector and ResNets as our main image classifier for implementing the mast detector, the component detector, and the component classifiers.

In order to evaluate the effectiveness of our proposed multi-stage component detection and classification pipeline and our proposed data augmentation techniques, we conduct two tests. First, we compare against our proposed multi-stage component detection pipeline with data augmentation (MSCDP-Dataaug) and without data augmentation (MSCDP-Noaug), and a simple component detection model (SCDM) trained directly on original images. Second, we compare between the component classification models trained with and without augmented data generated by our proposed data augmentation techniques. The results show that the MSCDP-Dataaug method achieves the best results in terms of mAP with 81.3% and outperforms the other two methods (MSCDP-Noaug and SCDM) in 7/10 classes. In addition, the MSCDP-Dataaug method achieves higher average precision on small insulator classes and 1.2% higher in mAP compared to the SCDM method. This indicates that our proposed multi-stage component detection and classification pipeline, together with our proposed data augmentation techniques can address the detection of small power line components and defects challenge to some extent. This is due to the "zoom-in" operation enabled by our proposed multi-stage architecture. By using the outputs from the mast detector as inputs for the component detector, most of the irrelevant background is removed, and the relative sizes of the power line components, especially the small ones, such as insulators, top caps, and cotter pins, are significantly increased, resulting in richer features for the deep learning models to learn from. The results also reveal that our proposed data augmentation techniques, especially the use of mast crops in training, can overcome the lack of training data challenge and can significantly improve the performance of our proposed multi-stage component detection and classification pipeline.

4.2.3 Contributions by the author

The idea was conceived by me and further developed in collaboration with the other co-authors. The datasets were created by me. I designed the multi-stage component detection and classification pipeline. The implementation and experiments were carried out by me. I wrote the manuscript draft of the paper.

4.3 Paper III

4.3.1 Research Question

RQ1.2 How and to what extent can the challenges of deep learning in vision-based UAV inspection be addressed?

4.3.2 Abstract

This paper, which can be found in Appendix C, aims at addressing the fourth challenge of DL vision-based UAV inspection, which is to detect power lines in cluttered backgrounds. To achieve this, we first provide an overview of the power line detection methods available at present, including line-based methods, piece-wise line segment-based methods, auxiliaries assisted methods, and DL-based methods, with an emphasis on their advantages and disadvantages. Based on that, we propose LS-Net, a fast single-shot line-segment detector, for then to apply it to power line detection.

The LS-Net is by design fully convolutional and consists of three modules: (i) a fully convolutional feature extractor; (ii) a classifier; and (iii) a line segment regressor. The design of the LS-Net architecture is mainly inspired by the state-of-the-art single-shot object detectors such as SDD [41] and YOLO [54]. Specifically, the LS-Net divides the input image into a grid, and for each grid cell, it predicts coordinates and a confidence score for the longest line segment in the cell. The confidence score indicates the probability of the cell containing a line segment, and the coordinates are the normalized distances of the two endpoints of the line segment to the local x-axis and y-axis of the cell.

The traditional one grid approach has been proven to work well for single-shot object detectors such as SDD [41] and YOLO [54]; however, it faces two problems when applied to line segment detection: (i) discontinuities and gaps at cell borders and (ii) discontinuities and gaps at cell corners. In the one grid approach, due to regression errors, the detected line segments can be shorter than the ground truths. This can result in discontinuities and gaps in the detected lines at borders of adjacent cells that make regression errors. In addition, the one grid approach ignores short line segments, especially at cell corners, due to the lack of features. This can also lead to discontinuities and gaps in the detected lines.

To address the two above-mentioned problems, we propose to replace the one grid approach by a four-grid approach. Specifically, the four-grid LS-Net divides the input image into four overlapping grids: a $S_m \times S_m$ grid (main grid), a $S_m \times S_a$ grid (horizontal grid), a $S_a \times S_m$ grid (vertical grid), and a $S_a \times S_a$ grid (center grid), where $S_a = S_m - 1$. The main grid, which works in the same way as the grid used by SDD and YOLO for detecting objects, is employed for detecting line segments in grid cells. The horizontal and vertical grids are utilized for closing gaps at horizontal and vertical borders, respectively. The central grid is used for detecting short line segments at cell corners that were ignored by the main grid. All the detected line segments from the four grids are combined together to form a line segment map. Since the four-grid LS-Net utilizes three additional grids to detect short line segments ignored by the main grid and close gaps at horizontal and vertical borders, the discontinuities in the detected lines are significantly eliminated.

The LS-Net can be trained end-to-end by backpropagation and SGD via a weighted multi-task loss function. The proposed loss function is a combination of focal loss [39] for addressing the class imbalance in classification and wing loss [12] for restoring the balance between the influence of errors of different sizes in multiple points regression. Due to the

unavailability of large datasets with annotations of power lines for training the LS-Net, we render synthetic images of power lines using the physically-based rendering approach [51] and propose a series of effective data augmentation techniques, including, for example, adding Gaussian-distributed additive noise, Gaussian blur, color manipulations, elastic deformations, to generate more training data.

We evaluate the proposed LS-Net on the publicly available Ground Truth of Power Line Dataset (Infrared-IR and Visible Light-VL) [89], which is one of the most widely used power line datasets. With a customized version of the VGG-16 network [66] as the backbone, the LS-Net outperforms the existing state-of-the-art DL-based power line detection approaches by a considerable margin. In addition, since the LS-Net can be trained end-to-end and performs very well even when trained only on synthetic images, it can potentially be adapted for detecting other linear structures, such as railway tracks, unburied onshore pipelines, and roads from low- and mid-altitude aerial images.

4.3.3 Contributions by the author

The idea was conceived by me and further developed in collaboration with the other co-authors. I designed the LS-Net architecture and proposed the multi-task loss function. The implementation and experiments were carried out by me. I wrote the manuscript draft of the paper.

4.4 Paper IV

4.4.1 Research Question

RQ1.2 How and to what extent can the challenges of deep learning in vision-based UAV inspection be addressed?

4.4.2 Abstract

In this paper, which can be found in Appendix D, we propose few-shot learning as a potential solution to the fifth challenge of DL vision-based UAV inspection, which is to detect previously unseen power line components and defects. To pave the way for addressing the challenge, we first provide an overview of the few-shot learning approaches available at present, including (i) learning to fine-tune approaches; (ii) sequence-based approaches; (iii) generative modeling-based approaches; (vi) distance metric learning-based approaches; (v) deep distance metric learning-based approaches; and (vi) semi-supervised approaches.

Among these categories, distance metric learning-based approaches are typically preferred since they are simpler and more efficient than other few-shot learning approaches, which require complex inference mechanisms, complex RNN architectures, or fine-tuning the target problem. The basic idea of distance metric learning-based approaches is to learn a non-linear mapping of the input into an embedding space and to define a metric distance which maps similar examples close and dissimilar ones distant in the embedding space, so that a query example can be easily classified by, for example, using nearest neighbor methods. The success of these methods relies heavily on the choice of the distance metric function. In the past few years, many fixed metric distance functions (e.g., the Euclidean distance [67] and the cosine distance [81]) and learnable deep metric distance functions, such as RN [72], have been applied in few-shot learning.

We build on the distance metric learning line of work due to its simplicity and effectiveness. The (squared) Euclidean distance is the most common option when optimizing embedding distance metrics, for instance, via the recent Prototypical Network, for solving the few-shot learning task. However, the Euclidean distance is not optimal in high-dimensional embedding spaces since it typically suffers from the curse of dimensionality. L_2 normalization, which implicitly results in a hyperspherical embedding, can alleviate this problem to some extent. However, L_2 normalization leads to a non-convex loss formulation, which typically results in local minima. With the aim of performing soft feature normalization while preserving the convexity and the simplicity of the loss function, we propose a novel dissimilarity measure in terms of the Squared root of the Euclidean distance and the Norm distance (SEN) combined. The SEN addresses the existing issues of the Euclidean distance combined with L_2 normalization by forcing the data to gradually lie on a scaled unit hypersphere during training. This is done by encouraging the norm of samples to have the same value. We extend the powerful PN by replacing the Euclidean distance by our proposed SEN dissimilarity measure, which we refer to as SEN PN. With minimal modifications, the SEN PN outperforms the original PN by a considerable margin and demonstrates good performance on the miniImageNet dataset with no additional parameters as well as almost no additional computational overhead. We provide analyses showing that SEN indeed explicitly forces all embeddings to have the same norm during training which enables the SEN PN to generate a more robust embedding space. We experimentally show that the proposed SEN dissimilarity measure constantly outperforms the Euclidean distance in PN with different embedding sizes as well as with different embedding networks.

4.4.3 Contributions by the author

The idea was conceived by me and further developed in collaboration with the other co-authors. The implementation and experiments were carried out by me. I wrote the manuscript draft of the paper.

Chapter 5

Discussion

This chapter serves four main purposes: firstly, it revisits the research questions introduced in Chapter 1 and points to where the relevant findings and discussions can be found. Secondly, it examines the possibilities of DL vision-based UAV inspection. Thirdly, it revisits the six main challenges of DL vision-based UAV inspection and discusses how can the approaches proposed in this dissertation address the challenges and pave the way for fully automatic autonomous vision-based power line inspection. Finally, it addresses the primary research question of this dissertation by using the findings from the four included papers.

5.1 Research Questions

In this section, we present the research questions introduced in Chapter 1 together with brief summaries of where the relevant findings and discussions can be found.

RQ1.1 What are the possibilities and challenges of deep learning in vision-based UAV inspection?

The research question RQ1.1 is addressed primarily in Paper I, which can be found in Appendix A. This paper conducts an extensive literature review on automatic vision-based power line inspection. Based on that, the possibilities and six main challenges of deep learning in vision-based UAV inspection, including (i) the lack of training data; (ii) class imbalance; (iii) the detection of small power line components and defects; (iv) the detection of power lines in cluttered backgrounds; (v) the detection of previously unseen power line components and defects; and (vi) the lack of metrics for evaluating inspection performance, are identified and discussed. A summary of the findings of the paper can be found in Section 4.1.

RQ1.2 How and to what extent can the challenges of deep learning in vision-based UAV inspection be addressed?

The research question RQ1.2 is answered mainly in Paper II, Paper III, and Paper IV, which can be found in Appendix B, Appendix C, and Appendix D, respectively. Specifically, Paper II addresses the first three challenges by creating four medium-sized datasets for training power line component detection and classification models, by applying a series of effective data augmentation techniques to balance out the imbalanced classes, and by

proposing a multi-stage component detection and classification pipeline based on SDD [41] and ResNets [24] to detect small power line components and defects. Paper III addresses the fourth challenge by proposing LS-Net, a fast single-shot line-segment detector, for then to apply it to power line detection. Paper IV proposes few-shot learning as a potential solution to the fifth challenge. Specifically, to pave the way for addressing the challenge, Paper IV proposes a novel dissimilarity measure for distance metric learning-based few-shot learning, called SEN, to address the existing issues of the traditional Euclidean distance in high dimensional spaces. The sixth challenge, which is to address the lack of metrics for evaluating inspection performance, is left for future work. Summaries of the findings of Paper II, Paper III, and Paper IV can be found in Section 4.2, Section 4.3, and Section 4.4, respectively. Relevant background knowledge and related work can be found in Chapter 2.

RQ1 How can deep learning be employed to realize automatic autonomous vision-based power line inspection with UAVs?

We have revisited the two secondary research questions and have pointed to where the relevant findings and discussions can be found. In the following sections, we focus on answering the main research question by reexamining the possibilities and challenges of deep learning in vision-based UAV inspection and by discussing how can the approaches proposed in this dissertation address the challenges and facilitate the implementation of fully automatic autonomous vision-based power line inspection systems.

5.2 Possibilities of DL Vision-based UAV Inspection

Recent advances in deep learning have opened up new opportunities not only for facilitating automatic vision-based UAV inspection but also for enabling self-driving UAVs and for paving the way for realizing fully automatic autonomous vision-based power line inspection. Specifically, in UAV inspection, deep learning holds great promise for realizing automatic data acquisition systems. In these systems, the outputs from the mast detection and component detection models can be employed for guiding cameras to focus when taking pictures of power line components. This can potentially eliminate the need for camera operators and significantly increase the quality of data acquisition. Besides, deep learning offers enormous potential for automating power line inspection tasks, such as mapping and inspection of power line components, icing detection and measurement, vegetation encroachment monitoring, and disaster monitoring. For example, CNN-based object detectors (e.g., Faster R-CNN [58], SDD [41], YOLO [54], and R-FCN [9]) and CNN-based image classifiers, such as ResNets [24], Inception architectures [74, 75, 76], and DenseNets [27], can be combined for detecting, for classifying, and for mapping and inspection of power line components from inspection images. This approach is also useful in the case of disaster monitoring. For instance, a tree detection model can be employed together with a power line detection model for detecting trees lying across or against power lines after natural disasters, such as storms, hurricanes, and earthquakes. In the case of vegetation encroachment monitoring and icing detection and measurement, detecting power lines is one of the most challenging tasks. However, recent advances in deep learning, especially CNNs for edge and line detection [87, 5, 65], have opened up many possibilities for addressing this challenge. In UAV navigation, DL vision-based navigation approaches (e.g., pole detection-based and power line detection-based) together with traditional navigation

approaches, such as GPS way points-based, and UAV autopilots can facilitate self-driving UAVs. This can eliminate the need for pilots, which can significantly reduce the inspection costs, save the inspection time, and increase the flexibility of the inspection.

Although deep learning has opened up new opportunities for realizing fully automatic autonomous vision-based power line inspection, it comes with many challenges. In the next section, we revisit the six main challenges of deep learning in vision-based UAV inspection and discuss how can the approaches proposed in this dissertation address the challenges and pave the way for fully automatic autonomous vision-based power line inspection.

5.3 Challenges of DL in Vision-based UAV Inspection

The first and foremost challenge of deep learning in vision-based UAV inspection is the lack of training data. To training deep learning models, large amounts of labeled data are typically required; however, to the best of our knowledge, there are no publicly available datasets of inspection images with annotations that are big enough for training such models. To move forward, we create four medium-sized datasets for training component detection and component classification models. The details of the four datasets are shown in Table 5.1. The first dataset (DS1_Co), which is created for training component detection models, is annotated with bounding boxes, while the other three datasets (DS2_Tc, DS3_Po, and DS4_Cr) are annotated with class labels for training component classification models. More details on the datasets can be found in Paper II in Appendix B. These four datasets alleviate the lack of training data and allow us to start the journey of automating power line inspection with deep learning.

Dataset	Annotation type	#Images	Image size	Speed (images/hour)	Total time (hours)
DS1_Co	bounding box + label	28674	6048x4032	40	716.85
DS2_Tc	label	34002	256x256	1000	34
DS3_Po	label	26446	256x256	1000	26.5
DS4_Cr	label	34029	256x256	1000	34

Table 5.1: Properties of the DS1_Co, DS2_Tc, DS3_Po, and DS4_Cr datasets.

Based on the four datasets, we discover the second challenge of deep learning in vision-based UAV inspection: class imbalance. This challenge comes from the long-tailed distribution of power line component classes. Specifically, a small number of power line component classes, such as wooden poles and insulators, appear very often, while most of the other power line component classes and defects, for instance, missing top caps and bird nests, appear rarely in our datasets. This typically causes deep learning models to bias towards classes that have more examples and overlook classes that have fewer examples. To address this problem, we propose a series of effective data augmentation techniques, including adding Gaussian-distributed additive noise, Gaussian blur, zoom, rotation, and flipping, to generate more training data to balance out the imbalanced classes. To evaluate the effectiveness of our proposed data augmentation techniques, we compare between the ResNet50 model trained with (ResNet50_cvgj+Dataaug) and without (ResNet50_cvgj) augmented data for two tasks: (i) pole crop classification and (ii) cross arm crop classification. The accuracy of the models for the pole crop classification and the cross arm

crop classification tasks are shown in Table 5.2 and Table 5.3, respectively. In both tasks, the augmented data generated by our proposed data augmentation techniques improves the accuracy of the models significantly. Specifically, the ResNet50 model trained with augmented data outperforms the ResNet50 model trained with original images only in terms of wF_1 score by 8.7% and 2.0% on the pole crop classification and the cross arm crop classification tasks, respectively.

Method	wP	wR	wF_1 score
ResNet50_cvgj	88.00	88.67	88.31
ResNet50_cvgj+Dataaug	**96.93**	**97.09**	**97.01**

Table 5.2: Pole crop classifier test results on the DS3_Po dataset.

Method	wP	wR	wF_1 score
ResNet50_cvgj	72.02	83.54	75.96
ResNet50_cvgj+Dataaug	**74.00**	**84.10**	**77.96**

Table 5.3: Cross arm cop classifier test results on the DS4_Cr dataset.

The third challenge is the detection of small power line components and defects. UAV inspection images are usually taken from far away, and as a result, some power line components (e.g., insulators, cotter pins, chain shackles) and defects, such as broken insulators and missing cotter pins, can be very small. This is a major problem in mapping and inspection of power line components since most of the state-of-the-art CNN-based object detectors such as SDD [41], YOLO [54], R-FCN [9] and Faster R-CNN [58] perform poorly on very small objects [29]. To address this problem, we propose a multi-stage component detection and classification pipeline, which consists of three stages: (i) mast detection; (ii) component detection; and (iii) component classification. The models are connected as shown in Figure 5.1. With a multi-stage architecture, the pipeline can mimic

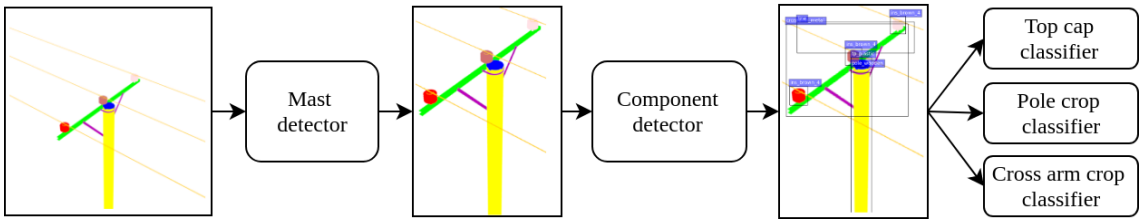


Figure 5.1: The general structure of our proposed multi-stage component detection and classification pipeline. The pipeline consists of five components: a mast detector, a component detector, a top cap classifier, a pole crop classifier, and a cross arm crop classifier.

the "zoom-in" operation when analyzing inspection images. This enables the detection of small power line components (e.g., insulators, cotter pins, chain shackles) and small defects, for example, broken insulators, missing cotter pins, cracks on poles and cross arms, and woodpecker damage on poles. The test results of our proposed multi-stage component detection and classification pipeline with data augmentation (MSCDP-Dataaug) and without data augmentation (MSCDP-Noaug) and a simple component detection model (SCDM) trained directly on original images are shown in Table 5.4. As can be seen from

Table 5.4, the MSCDP-Dataaug method achieves the best results in terms of mAP with 81.3% and outperforms the SCDM method in 7/10 classes, especially in small classes such as insg3, inso9, insw9, insw6, and inso. In addition, the MSCDP-Dataaug method achieves 1.2% mAP higher than the SCDM method. This indicates that our proposed multi-stage component detection and classification pipeline can significantly improve the accuracy of the model, especially in detecting small power line components and defects.

Method	mAP	pole	cross arm	top cap	insb4	insg2	insg3	inso9	insw9	insw6	inso
SCDM	80.1	86.3	**85.7**	85.1	**84.2**	**90.6**	89.0	55.2	78.4	75.5	71.5
MSCDP-Noaug	78.5	85.9	83.4	84.3	78.7	87.4	87.4	53.8	78.9	76.5	69.1
MSCDP-Dataaug	**81.3**	**88.0**	83.4	**86.3**	82.4	89.0	**89.9**	**59.5**	**81.5**	**78.8**	**73.8**

Table 5.4: SCDM, MSCDP-Dataaug, and MSCDP-Noaug detection results on our medium-sized DS1_Co dataset.

Having addressed the first three challenges, we build a basic automatic vision-based power line inspection system with two custom-built UAVs and five DL-based models. The system is capable of detecting a wide range of power line components (e.g., insulators, poles, cross arms, top caps, chain shackles, cotter pins, pylon numbers, dampers, and weights) and defects, including missing top caps, cracks in poles and cross arms, woodpecker damage on poles, rot damage on cross arms, broken insulators, flashed insulators, contaminated insulators, missing cotter pins, loose cotter pins, and rusty cotter pins.

To move forward, we extend the system to inspect not only power line components but also the power lines themselves. In addition, we aim at realizing vision-based UAV navigation for paving the way for fully automatic autonomous vision-based power line inspection. To achieve this, we address the fourth challenge of deep learning in vision-based UAV inspection, which is to detect power lines in cluttered backgrounds. Power lines in inspection images taken from UAVs or helicopters are typically very thin, leading to a lack of rich features for their representation. This makes power line detection in inspection images a very challenging task. To address this challenge, we propose LS-Net, a fast single-shot line-segment detector, for then to apply it to power line detection. The LS-Net is by design fully convolutional and consists of three modules: (i) a fully convolutional feature extractor; (ii) a classifier; and (iii) a line segment regressor. The modules are connected as shown in Figure 5.2. We compare against our proposed LS-Net and the state-of-the-art

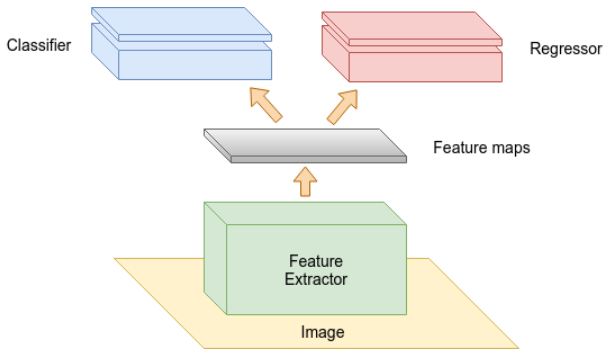


Figure 5.2: The LS-Net is a feed-forward fully convolutional neural network and consists of three modules: (i) a fully convolutional feature extractor; (ii) a classifier; and (iii) a line segment regressor.

DL-based power line detection methods including the Weakly Supervised Learning with

CNN (WSL-CNN) [37] approach and the Dilated Convolution Neural Network for Wire Detection (DCNN-WD) [43] approach and its improved versions on the publicly available Ground Truth of Power Line Dataset (Infrared-IR and Visible Light-VL) [89]. Since each line segment predicted by the LS-Net is represented by a pair of two end-points, we apply the 8-connected Bresenham algorithm [7] to form a close approximation to a straight line between the two end-points. To generate segmentation maps, we vary the width of the straight line, W_l, from 1 to 5 and select $W_l = 2$ and $W_l = 3$ since they result in the highest F_1 scores. We call these models LS-Net-W2 and LS-Net-W3, respectively. The test results are shown in Table 5.5. As can be seen from Table 5.5, both our proposed

	ARR	APR	F_1 Score
WSL-CNN [37]*	0.6256	-	-
DCNN-WD [43]	0.7192	0.4713	0.4835
DCNN-WD + GN	0.8292	0.4148	0.4882
DCNN-WD + FL	0.7514	0.4680	0.5079
DCNN-WD + GN + FL	0.7930	0.4690	0.5218
LS-Net-W2 ($W_l = 2$)	0.7972	**0.4874**	**0.5344**
LS-Net-W3 ($W_l = 3$)	**0.8525**	0.4483	**0.5256**

Table 5.5: Comparisons of the proposed LS-Net-W2 and LS-Net-W3 models and the state-of-the-art DL-based approaches for power line detection including the WSL-CNN approach and the DCNN-WD approach and its improved versions on the Ground Truth of Power line dataset (Infrared-IR and Visible Light-VL). *Results reported by [37].

LS-Net-W2 and LS-Net-W3 models achieve the state-of-the-art performance in terms of F_1 score. In addition, the LS-Net-W2 model surpasses all the existing state-of-the-art methods in terms of Average Precision Rate (APR), and the LS-Net-W3 model attains state-of-the-art Average Recall Rate (ARR) by considerable margins. This suggests that our proposed LS-Net can alleviate the detection of power lines in cluttered backgrounds challenge to some extent.

To realize fully automatic autonomous vision-based power line inspection, the system has to be able to detect a wide range of power line components and defects. This leads to the fifth challenge of deep learning in vision-based UAV inspection, which is to detect previously unseen power line components and defects. Most of the existing power grids utilize a wide range of power line components. Thus, the grids are vulnerable to a huge number of types of defects. In addition, defects on power line components, even very simple ones such as missing top caps and broken insulators, can occur in many different forms. This, together with the lack of training data, poses a major challenge for DL vision-based UAV inspection due to the limitation of deep learning models in detecting and classifying previously unseen classes. To mitigate this problem, we propose few-shot learning, which allows trained models to adapt to new classes with only a few examples per class, as a potential solution. Specifically, to pave the way for addressing the challenge, we propose a novel dissimilarity measure for distance metric learning-based few-shot learning, called SEN, to address the existing issues of the traditional Euclidean distance in high dimensional spaces. The SEN is a combination of the Euclidean distance and the norm distance. We extend the powerful PN by replacing the Euclidean distance by our proposed SEN dissimilarity measure, which we refer to as SEN PN. The test results of our proposed approach and the state-of-the-art few-shot learning approaches are shown in Table 5.6. As

can be seen from Table 5.6, the SEN PN outperforms the original PN by a considerable margin and demonstrates good performance on the miniImageNet dataset. This suggests that our proposed approach holds great promise for addressing the detection of previously unseen power line components and defects challenge.

Model	Accuracy
PN (original paper, 20-way training) [67]	**68.2%**
Large Margin GNN (5-way training) [90]	67.6%
Large Margin PN (5-way training) [90]	66.8%
SNAIL (5-way training) [45]	68.9%
RN (5-way training) [72]	65.3%
MetaGAN + RN (5-way training) [92]	68.6%
Semi-Supervised PN (5-way training) [6]	65.5%
Supervised ResNet PN (20-way training) [6]	69.6%
Semi-Supervised ResNet PN (20-way training) [6]	**70.9%**
PN (ours, 20-way training)	65.7%
PN (ours, 5-way training, baseline)	67.8%
SEN PN (ours, 20-way training)	67.9%
SEN PN (ours, 5-way training)	**69.8%**
ResNet PN (ours, 5-way training)	71.0%
SEN ResNet PN (ours, 5-way training)	**72.3%**

Table 5.6: Few-shot classification accuracies on MiniImagenet (5-way 5-shot testing).

5.4 Summary

In this chapter, we have revisited the possibilities and the first five challenges of deep learning in vision-based UAV inspection and have discussed how and to what extent can the approaches proposed in this dissertation address the challenges. The last challenge, which is to address the lack of metrics for evaluating inspection performance, is left for future work.

The work presented in this dissertation has in our opinion paved the way for fully automatic autonomous vision-based power line inspection by proposing approaches (i) for realizing automatic analysis of inspection images for mapping and inspection of power line components; (ii) for enabling power line detection in cluttered backgrounds for facilitating vision-based UAV navigation for self-driving UAVs, and (iii) for enhancing deep learning models capability in detecting previously unseen power line components and defects.

Even though the primary goal of this dissertation is to realize fully automatic autonomous vision-based power line inspection with deep learning and UAVs, the approaches proposed in this dissertation can also potentially be applied for automating other vision-based inspection methods such as foot patrol, helicopter-assisted, automated helicopter-assisted, and climbing robots. Specifically, our proposed multi-stage component detection and classification pipeline can potentially be adapted for analyzing images taken by foot patrol teams from the ground, images taken from helicopters, and even images taken by climbing robots for enabling automatic analysis of inspection images for mapping and inspection of power line components.

To automate visual power line inspections, we have proposed a novel automatic autonomous vision-based power line inspection concept that uses UAV inspection as the main inspection method, optical images as the primary data source, and deep learning as the backbone of data analysis. We have demonstrated that proposed concept has a promising role in the intelligent monitoring and inspection of power line components and as a valuable addition to smart grids. The proposed concept can potentially be further generalized to inspect a wider range of infrastructures such as railway tracks, pipelines, bridges, windmills, and solar farms. Specifically, for railway track inspection, our proposed LS-Net can be adapted for detecting railway tracks from inspection images taken by UAVs. These detections can be utilized both for navigating the UAVs and for guiding the cameras to focus when taking pictures of the railway tracks. This approach is also useful in the case of pipeline inspection. For bridge inspection, windmill inspection, and solar farm inspection, our proposed multi-stage detection and classification architecture can potentially be generalized for first detecting RoIs (bridges, windmills, and solar panels) from inspection images taken by UAVs. Then, a second-stage detection model can be employed for detecting relevant components from the detected RoIs. Finally, multiple component classification models can be utilized for classifying the detected components for identifying defects. For solar farm inspection, multiple data sources such as optical images and thermal images can potentially be combined for detecting both visible defects (e.g., broken solar panels) and defects that are invisible to the unaided human eye, for instance, hot spots.

Chapter 6

Conclusion and Further Work

6.1 Conclusion

In this dissertation, we present our work in employing deep learning for enabling automatic analysis of power line inspection images and for facilitating automatic autonomous vision-based power line inspection.

The primary research question of this dissertation – "RQ1 How can deep learning be employed to realize automatic autonomous vision-based power line inspection?" – is addressed mainly in Chapter 4 and Chapter 5. The two secondary research questions – "RQ1.1 What are the possibilities and challenges of deep learning in vision-based UAV inspection?" and "RQ1.2 How and to what extent can the challenges of deep learning in vision-based UAV inspection be addressed?" – are addressed primarily in Chapter 4, Chapter 5, and the four included papers, which can be found in Appendix A, Appendix B, Appendix C, and Appendix D.

To answer the research questions, we first conduct an extensive literature review on automatic vision-based power line inspection. Based on that, we identify the possibilities and six main challenges of DL vision-based UAV inspection: (i) the lack of training data; (ii) class imbalance; (iii) the detection of small power line components and defects; (iv) the detection of power lines in cluttered backgrounds; (v) the detection of previously unseen power line components and defects; and (vi) the lack of metrics for evaluating inspection performance. Next, we address the first three challenges by creating four medium-sized datasets for training component detection and classification models, by applying a series of effective data augmentation techniques to balance out the imbalanced classes, and by proposing a multi-stage component detection and classification approach based on SDD [41] and deep ResNets [24] to detect small power line components and defects. Then, we address the fourth challenge by proposing LS-Net, a fast single-shot line-segment detector, for then to apply it to power line detection. After that, we propose few-shot learning as a potential solution to the fifth challenge. Specifically, to pave the way for addressing the challenge, we propose a novel dissimilarity measure for distance metric learning-based few-shot learning, called SEN, to address the existing issues of the traditional Euclidean distance in high dimensional spaces. Finally, we discuss the possibilities and challenges of DL vision-based UAV inspection and outline how can the approaches proposed in this dissertation alleviate the challenges and facilitate the implementation of fully automatic autonomous vision-based power line inspection systems.

6.2 Further Work

In this dissertation, we have implemented phase 1, which is to employ deep learning for enabling automatic analysis of power line inspection images and for facilitating automatic vision-based power line inspection.

To move forward, we suggest five potential next steps for further improving phase 1 and for implementing phase 2, which is to employ deep learning for facilitating self-driving UAVs and for realizing fully automatic autonomous vision-based power line inspection: The first is to extend the multi-stage component detection and classification pipeline to detect defects in various sizes, forms, and severity levels. Our current multi-stage component detection and classification pipeline consists of three stages: (i) mast detection; (ii) component detection; and (iii) component classification. With this architecture, the pipeline can detect a wide range of power line components and defects from inspection images; however, it is not yet able to quantify the size and the severity of the detected defects. One possible solution is to adopt object instance segmentation [21] to identify the boundaries of the detected power line components and defects at the detailed pixel level and utilize this information to estimate the size and the severity of the detected defects by, for example, calculating the defect/component pixel ratio.

The second step is to apply multiple data sources fusion (e.g., optical images, thermal images, ultraviolet images, and depth images) for detecting a wider range of defects, for instance, vegetation encroachment and defects that are invisible to the unaided human eye, such as equipment bad connections, corona discharges, and hot spots on power line components.

The third step is to combine pole detection-based, power line detection-based, and GPS way points-based navigation approaches with UAV autopilots to facilitate self-driving UAVs and automatic data acquisition. The main objective of this step is to build self-driving UAVs that not only can fly themselves along power lines but also can detect power line components in real-time to guide cameras to focus when taking pictures to acquire high-quality inspection images.

The fourth step is to combine the automatic data acquisition module with online multiple-object tracking algorithms [95] to enable fully automatic asset management. Specifically, camera information such as GPS coordinates and timestamp can potentially be utilized together with the automatic data acquisition module to group images of the same mast. During data acquisition, online multiple-object tracking can potentially be employed to track power line components. Each tracked component will be assigned a Unique IDentifier (UID) and linked to its corresponding mast. The objectives of this step are threefold: The first is to allow utilities to quickly query all images of a certain asset via its UID. The second is to give utilities an overview of their power grids. The third is to enable change monitoring and to realize fully automatic asset management.

The final step is to define metrics for evaluating the performance of automatic autonomous vision-based power line inspection systems.

Bibliography

[1] Santoro Adam, Bartunov Sergey, Botvinick Matthew, Wierstra Daan, and P. Lillicrap Timothy. One-shot learning with memory-augmented neural networks. *CoRR*, abs/1605.06065, 2016.

[2] Graves Alex, Wayne Greg, and Danihelka Ivo. Neural turing machines. *CoRR*, abs/1410.5401, 2014.

[3] WA Awad and SM ELseuofi. Machine learning methods for spam e-mail classification. *International Journal of Computer Science & Information Technology (IJCSIT)*, 3(1):173–184, 2011.

[4] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[5] Gedas Bertasius, Jianbo Shi, and Lorenzo Torresani. Deepedge: A multi-scale bifurcated deep network for top-down contour detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4380–4389, 2015.

[6] Rinu Boney and Alexander Ilin. Semi-supervised few-shot learning with prototypical networks. *CoRR*, abs/1711.10856, 2017.

[7] Jack Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems Journal*, 4(1):25–30, 1965.

[8] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[9] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-FCN: object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 379–387, 2016.

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255, 2009.

[11] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019.

[12] Zhen-Hua Feng, Josef Kittler, Muhammad Awais, Patrik Huber, and Xiao-Jun Wu. Wing loss for robust facial landmark localisation with convolutional neural networks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2235–2245, 2018.

[13] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 1126–1135, 2017.

[14] Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C. Berg. DSSD : Deconvolutional single shot detector. *CoRR*, abs/1701.06659, 2017.

[15] Ross Girshick. Fast r-cnn. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448. IEEE, 2015.

[16] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 580–587, 2014.

[17] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, pages 315–323, 2011.

[18] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[19] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354–377, 2018.

[20] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), 17-22 June 2006, New York, NY, USA*, pages 1735–1742, 2006.

[21] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of The IEEE International Conference on Computer Vision (ICCV)*, pages 2961–2969, 2017.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034. IEEE Computer Society, 2015.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, 2015.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778, 2016.

[25] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[26] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7132–7141, 2018.

[27] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2261–2269, 2017.

[28] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q. Weinberger. Deep networks with stochastic depth. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, pages 646–661, 2016.

[29] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 3296–3297, 2017.

[30] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 448–456, 2015.

[31] Jaka Katrasnik, Franjo Pernus, and Bostjan Likar. A survey of mobile robots for distribution power line inspection. *IEEE Transactions on power delivery*, 25(1):485–493, 2009.

[32] Bartosz Krawczyk. Learning from imbalanced data: open challenges and future directions. *Progress in AI*, 5(4):221–232, 2016.

[33] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[34] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, 2017.

[35] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[36] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[37] Sang Jun Lee, Jong Pil Yun, Hyeyeon Choi, Wookyong Kwon, Gyogwon Koo, and Sang Woo Kim. Weakly supervised learning with convolutional neural networks for power line localization. In *2017 IEEE Symposium Series on Computational Intelligence, SSCI 2017, Honolulu, HI, USA, November 27 - Dec. 1, 2017*, pages 1–8, 2017.

[38] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 936–944, 2017.

[39] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2999–3007, 2017.

[40] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, pages 740–755, 2014.

[41] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part I*, pages 21–37, 2016.

[42] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*. Citeseer, 2013.

[43] Ratnesh Madaan, Daniel Maturana, and Sebastian Scherer. Wire detection using synthetic data and dilated convolutional networks for unmanned aerial vehicles. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017*, pages 3487–3494, 2017.

[44] John McCarthy. What is artificial intelligence? `http://www-formal.stanford.edu/jmc/whatisai/whatisai.html`, 2007.

[45] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[46] Mostafa Nayyerloo, XiaoQi Chen, Wenhui Wang, and J Geoffrey Chase. Cable-climbing robots for power transmission lines inspection. In *Mobile Robots-State of the Art in Land, Sea, Air, and Collaborative Missions*, pages 63–84. IntechOpen, 2009.

[47] Van Nhan Nguyen, Robert Jenssen, and Davide Roverso. Automatic autonomous vision-based power line inspection: A review of current status and the potential role of deep learning. *International Journal of Electrical Power & Energy Systems*, 99:107–120, 2018.

[48] Steven J Nowlan and Geoffrey E Hinton. Simplifying neural networks by soft weight-sharing. *Neural computation*, 4(4):473–493, 1992.

[49] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, pages 1310–1318, 2013.

[50] Luis Perez and Jason Wang. The effectiveness of data augmentation in image classification using deep learning. *CoRR*, abs/1712.04621, 2017.

[51] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016.

[52] Lutz Prechelt. Early stopping-but when? In *Neural Networks: Tricks of the trade*, pages 55–69. Springer, 1998.

[53] Joseph Redmon. Darknet: Open source neural networks in C. `http://pjreddie.com/darknet/`, 2013–2016.

[54] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 779–788, 2016.

[55] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6517–6525, 2017.

[56] Joseph Redmon and Ali Farhadi. YOLOv3: An incremental improvement. *CoRR*, abs/1804.02767, 2018.

[57] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. Meta-learning for semi-supervised few-shot classification. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[58] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2016.

[59] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[60] Ravi Sachin and Larochelle Hugo. Optimization as a model for few-shot learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[61] Victor Garcia Satorras and Joan Bruna Estrach. Few-shot learning with graph neural networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[62] Al Savvaris, Ye Xie, Konstantinos Malandrakis, Matias Lopez, and Antonios Tsourdos. Development of a fuel cell hybrid-powered unmanned aerial vehicle. In *24th Mediterranean Conference on Control and Automation, MED 2016, Athens, Greece, June 21-24, 2016*, pages 1242–1247, 2016.

[63] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 815–823, 2015.

[64] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[65] Wei Shen, Xinggang Wang, Yan Wang, Xiang Bai, and Zhijiang Zhang. Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3982–3991, 2015.

[66] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[67] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 4077–4087, 2017.

[68] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin A. Riedmiller. Striving for simplicity: The all convolutional net. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, 2015.

[69] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[70] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *CoRR*, abs/1505.00387, 2015.

[71] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 1988–1996, 2014.

[72] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 1199–1208, 2018.

[73] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 4278–4284, 2017.

[74] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 4278–4284, 2017.

[75] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1–9, 2015.

[76] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2818–2826, 2016.

[77] Sebastian Thrun. Lifelong learning algorithms. In *Learning to learn*, pages 181–209. Springer, 1998.

[78] Fok Hing Chi Tivive and Abdesselam Bouzerdoum. Application of siconnets to handwritten digit recognition. *International Journal of Computational Intelligence and Applications*, 6(01):45–59, 2006.

[79] Jasper R. R. Uijlings, Koen E. A. van de Sande, Theo Gevers, and Arnold W. M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

[80] Joaquin Vanschoren. Meta-learning: A survey. *CoRR*, abs/1810.03548, 2018.

[81] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 3630–3638, 2016.

[82] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L_2 hypersphere embedding for face verification. In *Proceedings of the 2017 ACM on Multimedia Conference, MM 2017, Mountain View, CA, USA, October 23-27, 2017*, pages 1041–1049, 2017.

[83] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 5265–5274, 2018.

[84] Yaqing Wang and Quanming Yao. Few-shot learning: A survey. *CoRR*, abs/1904.05046, 2019.

[85] Yu-Xiong Wang, Ross B. Girshick, Martial Hebert, and Bharath Hariharan. Low-shot learning from imaginary data. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7278–7286, 2018.

[86] Yuxin Wu and Kaiming He. Group normalization. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIII*, pages 3–19, 2018.

[87] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1395–1403, 2015.

[88] Bing Xu, Ruitong Huang, and Mu Li. Revise saturated activation functions. *CoRR*, abs/1602.05980, 2016.

[89] Ömer Emre Yetgin and Ömer Nezih Gerek. Ground truth of powerline dataset (infrared-ir and visible light-vl). *Mendeley Data, v8, http://dx. doi. org/10.17632/twxp8xccsw*, 8, 2017.

[90] Wang Yong, Wu Xiao-Ming, Li Qimai, Gu Jiatao, Xiang Wangmeng, Zhang Lei, and O. K. Li Victor. Large margin few-shot learning. *CoRR*, abs/1807.02872, 2018.

[91] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, September 19-22, 2016*, 2016.

[92] Ruixiang Zhang, Tong Che, Zoubin Ghahramani, Yoshua Bengio, and Yangqiu Song. Metagan: An adversarial approach to few-shot learning. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pages 2371–2380, 2018.

[93] Tao Zhang, Qing Li, Chang-shui Zhang, Hua-wei Liang, Ping Li, Tian-miao Wang, Shuo Li, Yun-long Zhu, and Cheng Wu. Current trends in the development of intelligent unmanned autonomous systems. *Frontiers of Information Technology & Electronic Engineering*, 18(1):68–85, 2017.

[94] Bichen Zheng, Sang Won Yoon, and Sarah S Lam. Breast cancer diagnosis based on feature extraction using a hybrid of k-means and support vector machine algorithms. *Expert Systems with Applications*, 41(4):1476–1482, 2014.

[95] Ji Zhu, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. Online multi-object tracking with dual matching attention networks. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part V*, pages 379–396, 2018.

Appendix A

Paper I

Electrical Power and Energy Systems 99 (2018) 107–120

Contents lists available at ScienceDirect

Electrical Power and Energy Systems

journal homepage: www.elsevier.com/locate/ijepes

Review

Automatic autonomous vision-based power line inspection: A review of current status and the potential role of deep learning

Van Nhan Nguyen[a,b,*], Robert Jenssen[a], Davide Roverso[b]

[a] UiT Machine Learning Group, Faculty of Science and Technology, Department of Physics and Technology, University of Tromsø, 9019 Tromsø, Norway
[b] Analytics Department, eSmart Systems, 1783 Halden, Norway

ARTICLE INFO

Keywords:
Power line inspection
Vision-based inspection
Deep learning
UAVs

ABSTRACT

To maintain the reliability, availability, and sustainability of electricity supply, electricity companies regularly perform visual inspections on their transmission and distribution networks. These inspections have been typically carried out using foot patrol and/or helicopter-assisted methods to plan for necessary repair or replacement works before any major damage, which may cause power outage. This solution is quite slow, expensive, and potentially dangerous. In recent years, numerous researches have been conducted to automate the visual inspections by using automated helicopters, flying robots, and/or climbing robots. However, due to the high accuracy requirements of the task and its unique challenges, automatic vision-based inspection has not been widely adopted. In this paper, with the aim of providing a good starting point for researchers who are interested in developing a fully automatic autonomous vision-based power line inspection system, we conduct an extensive literature review. First, we examine existing power line inspection methods with special attention paid to highlight their advantages and disadvantages. Next, we summarize well-suited tasks and review potential data sources for automatic vision-based inspection. Then, we survey existing automatic vision-based power line inspection systems. Based on that, we propose a new automatic autonomous vision-based power line inspection concept that uses Unmanned Aerial Vehicle (UAV) inspection as the main inspection method, optical images as the primary data source, and deep learning as the backbone of data analysis and inspection. Then, we present an overview of possibilities and challenges of deep vision (deep learning for computer vision) approaches for both UAV navigation and UAV inspection and discuss possible solutions to the challenges. Finally, we conclude the paper with an outlook for the future of this field and propose potential next steps for implementing the concept.

1. Introduction

The increasing dependence of modern-day societies on electricity poses corresponding challenges on the monitoring, inspection, and maintenance of the electric power grids to ensure an uninterrupted flow of electricity. Due to the lack of incentives to invest in aged national grid infrastructures, for example, in Europe and the US, power cuts are becoming more and more frequent [1]. While short-term power failures typically last only a few hours, long-term blackouts can last days or even weeks. Power outages, both short and long-term, can have catastrophic effects on unprepared businesses. For instance, blackouts can completely shut down production at companies and critical infrastructures such as telecommunication networks, financial services, water supplies, and hospitals [2]. Nowadays, most of the power grids are interconnected. Hence, a blackout in one region can trigger a domino effect that could result in supra-regional blackouts [3]. According to [1], a 30-min power cut in the US results in an average loss of US$15,709 for medium and large industrial clients, and nearly US $94,000 for an eight-hour interruption. Thus, power grids are required to be monitored, inspected, and maintained regularly to prevent faults which may cause power failures (Figs. 1 and 2).

The traditional methodologies for inspecting power networks typically include field surveys and airborne surveys, which have been unchanged for decades [4]. On a regular basis or in emergency situations, such as after storms, hurricanes, and earthquakes, teams of inspectors are sent out, traveling either on foot or by helicopters, to visually inspect the power lines with the help of binoculars and sometimes with InfraRed (IR) and corona detection cameras [5]. The main reason why visual inspection is popular is that it can cover a wide range of common faults on both power lines components (Fig. 2) and the power lines themselves (Fig. 1) in one inspection. However, this methodology is quite slow, expensive, potentially dangerous, and with detection rates limited by the visual observation skills of the inspectors. Although digital cameras can be used to separate the data collection from data

* Corresponding author at: UiT Machine Learning Group, Faculty of Science and Technology, Department of Physics and Technology, University of Tromsø, 9019 Tromsø, Norway.
E-mail addresses: nhan.v.nguyen@uit.no (V.N. Nguyen), robert.jenssen@uit.no (R. Jenssen), Davide.Roverso@esmartsystems.com (D. Roverso).

https://doi.org/10.1016/j.ijepes.2017.12.016
Received 19 September 2017; Received in revised form 20 November 2017; Accepted 16 December 2017

Fig. 1. Common faults on power lines (from left to right): trees growing too close to power lines, trees lying across power lines, icing on power lines. In warm countries, icing on power lines may not be a relevant fault; however, in cold countries, such as Norway, it is a very serious problem since thick icing accumulated on power lines can cause a great deal of damage to the lines.

Fig. 2. Common faults on power lines components (from left to right): broken poles, broken crossarms, missing toppads. In Norway, toppads are usually used for protecting wooden poles from rain; thus missing toppads are considered faults. In the US, however, they may not be considered as faults since toppads are not widely applied.

analysis, both processes have still been performed manually.

Over the past few years, many studies have been conducted to automate the visual inspection by using automated helicopters, flying robots, and/or climbing robots; however, there are very few published reviews of different approaches to the problem, including vision-based approaches. In this paper, with the aim of providing a comprehensive overview of the possibilities and challenges of automatic autonomous vision-based inspection of power lines, we conduct an extensive literature review. First, we examine existing power line inspection methods including foot patrol, helicopter-assisted, automated helicopter-assisted, climbing robots, and Unmanned Aerial Vehicles (UAV) inspection with special attention paid to highlight their advantages and disadvantages. Next, we summarize inspection tasks which are well suited for automatic autonomous vision-based inspection: mapping and inspection of power line components, vegetation encroachment monitoring, icing detection and measurement, and disaster monitoring. Then, we review potential data sources for vision-based inspection including Synthetic Aperture Radar (SAR) images, optical satellite images, optical aerial images, thermal images, ultraviolet images, Airborne Laser Scanner (ALS) data, land-based mobile mapping data, and UAV data and point out their applicable tasks. Further, we conduct a comprehensive review of current automatic vision-based power line inspection systems.

Based on that, we propose as a new potential solution an automatic autonomous vision-based power line inspection concept that uses UAV as the main inspection method, optical images as the primary data source, and Deep Learning (DL) as the backbone of data analysis and inspection.

To facilitate the implementation of the concept, we first discuss the potential role of deep learning in automatic autonomous vision-based power line inspection. We then highlight the possibilities and challenges of deep vision (deep learning for computer vision) approaches for both UAV navigation and UAV inspection. Finally, we propose possible solutions to the challenges and suggest potential next steps.

The remainder of the paper is structured as follows: Section 2 summarizes existing relevant literature reviews (related work) including inspection with vision-based approaches, before we present a brief introduction to power line inspection in general together with a summary of different inspection methods, inspection tasks, and data sources for visual inspection in Section 3. Next, in Section 4, we present a comprehensive review of existing vision-based approaches for both UAV navigation and UAV inspection. Finally, in Sections 5 and 6, we identify the remaining challenges of vision-based approaches in automatic inspection, discuss possible solutions to these challenges, propose potential next steps for implementing the automatic autonomous vision-based power line inspection concept, and conclude the paper with a summary and an outlook for the future of the field.

2. Related work

Although vision-based inspection is one of the most promising approaches for reducing or completely eliminating people, in both data collection and analysis, there are very few published reviews of vision-based approaches for power line inspections.

A review of recent techniques for vegetation encroachment monitoring was presented in [6]. According to the authors, current methods for vegetation monitoring, including visual field inspection, aerial video surveillance, aerial multispectral imaging, and LiDAR scanning are not viable due to their high cost, inaccuracy, and high time complexity. To increase the reliability of vegetation monitoring, satellite stereo and Wireless Multimedia Sensor Networks (WMSNs) were proposed to be the two future technologies. Based on that, the authors discussed the concept of utilizing multispectral satellite stereo images and WMSNs in detecting dangerous vegetation.

François et al. conducted a survey of computer vision applications in power line inspection in [7]. The surveyed applications include detection of power lines, inspection of power lines, detection and inspection of insulators, power line corridor maintenance, and pylon detection. According to the authors, computer vision appears to be one of the most important technologies for automatic vision-based power line inspection since both robots and UAVs need it not only for guidance, navigation, and control but also for inspection. Although this technology has been facilitating some applications, for example vegetation monitoring, it has not been widely used for other important applications such as inspection of defects on the cables or insulators.

Recent advantages in remote sensing sensors and data analysis approaches have opened new possibilities in automatic power line inspection. To facilitate the use of remote sensing techniques in power line inspection, a very comprehensive survey of the possibilities of modern remote sensing sensors for automatic power inspection was

conducted in [4]. The authors summarized the main applications, advantages, and challenges of different remote sensing data sources for power line inspection including SAR images, optical satellite and aerial images, thermal images, airborne laser scanner data, mobile mapping data, and UAV data. The authors pointed out that while mapping and analysis of network components is the main focus of most of the previous studies, vegetation monitoring has been received less attention. Since each remote sensing approach has its own specific advantages and disadvantages, the authors suggested that more attention should be given to the use of multiple data sources fusion in future research to benefit from various approaches in an optimal way.

3. Vision-based power line inspection

In this section, we first review existing power line inspection methods with special attention paid to highlighting their advantages and disadvantages. Next, we summarize power line inspection tasks that are well suited for automatic autonomous vision-based inspection. Then, we survey potential data sources for vision-based power line inspection and point out their applicable inspection tasks. Finally, we give a brief introduction to automatic vision-based power line inspection systems and their required components.

3.1. Power line inspection methods

3.1.1. Foot patrol inspection

Foot patrol inspection is typically conducted by a team of two inspectors traveling on foot to visually inspect the power lines with the help of binoculars and sometimes with infrared cameras and corona detection cameras [5]. The main disadvantage of this inspection method is that it is very slow, tedious, and labor-intensive. Another major drawback of this method is that it may not be possible in harsh terrains, in extreme weather conditions, and after extreme events, such as hurricanes, wind storms, and snow storms. However, despite its disadvantages, foot patrol inspection has still been widely applied because of its high detection rate [4].

3.1.2. Helicopter-assisted inspection

In helicopter-assisted inspection, a team of usually three (or two) people including a pilot, an inspector, and/or a camera operator is sent out traveling by helicopter to perform online inspection and acquire data for offline inspection. The inspector is responsible for identifying obvious faults, for example trees growing too close to power lines, trees lying across or against power lines, and collapsed poles. The camera operator films the conductors, pylons, power components (e.g., insulators, transformers, crossarms, toppads), and objects around the pylons and under the lines. After the flight, an offline inspection is typically conducted by a team of inspectors manually browsing through the collected videos and images to identify more complicated and smaller faults, such as broken insulators, missing toppads, and missing splints. Although this inspection method allows access to hard-to-reach locations and increases inspection speed, it comes with many disadvantages, such as high cost, low accuracy due to high speeds, the dependence on human visual observation skills, and always there is a risk of contacting live lines and loss of lifes [8].

3.1.3. Automated helicopter-assisted inspection

Automated helicopter-assisted inspection is quite similar to the helicopter-assisted inspection method. The main difference is that the former method utilizes vision-based approaches in both data collection and data analysis. Vision-based approaches, for example power mast detection, can be applied for guiding cameras to automatically film the conductors, pylons, power components, and objects around the pylons and under the power lines. The acquired videos and images can be later analyzed automatically by vision-based data analysis approaches. Although automated helicopter-assisted inspection can reduce the dependence on human visual observation skills and increases inspection speed, it has not been widely applied due to high inspection cost, safety issues, and challenges of applying vision-based approaches in power line inspection.

3.1.4. Climbing robots inspection

In climbing robots inspection, the inspection is conducted by a robot traveling on power lines. The robot is typically equipped with many sensors and cameras (e.g., visual cameras, thermal cameras) for navigating along power lines, crossing obstacles on the lines, and inspecting the lines and power components. The main advantage of this method is the inspection accuracy due to its proximity to the power lines. However, according to [9], the weight of the robots could damage the lines, and the robots may not be able to pass across various obstacles on the lines. Another major disadvantage of this inspection method is that it is relatively slow compared to other inspection methods such as automated helicopter-assisted inspection or UAV inspection.

3.1.5. UAV inspection

In UAV inspection, the inspection is conducted by UAVs equipped with multiple sensors and cameras for navigating along power lines, performing online inspection to detect obvious faults, and collecting data for later offline inspections. UAV inspection offers great possibilities for addressing most of the existing issues of other inspection methods such as high inspection cost, low speed, and safety. The operation costs of UAV inspection is relative low compared to foot patrol and (automated) helicopter-assisted inspections. In addition, UAVs can fly relatively close to power lines to take detailed images of conductors, pylons, and power components, which can significantly improve inspection accuracy.

However, fully automatic autonomous power line inspection systems using UAVs still come with great challenges. For example, In UAV navigation, detection and tracking of power lines are crucial tasks; however, existing vision-based detection and tracking methods are still not accurate enough for detecting and tracking the power lines since they are typically too small leading to a lack of rich features for their representation.

3.2. Power line inspection tasks

3.2.1. Mapping and inspection of power line components

Mapping and inspection of power line components, such as conductors, pylons, and power components, are among the most studied topics in the field of power line inspection. These tasks are typically conducted either online by, for example, a foot patrol team with the help of binoculars to manually observe and detect faults or offline by, for instance, analyzing images taken from UAVs and/or helicopters. Some common faults on power line components which can be identified by the two approaches are broken poles, broken crossarms, and missing toppads (Fig. 2). In addition to the above mentioned approaches, other data sources such as thermal images, airborne laser scanner, and land-based mobile mapping data have also been used as alternatives for mapping and inspection of power line components [4].

3.2.2. Vegetation encroachment monitoring

According to [6], one of the most frequent causes of flashovers in both transmission and distribution networks is encroachment to the power lines. In addition, maintaining safety distances between vegetation and the power lines along the corridor is a legal requirement in some countries. Thus, it is required that vegetation near power line corridors are cleared regularly to prevent power outages and damage to the power lines [10]. The main tasks of vegetation encroachment monitoring include detecting and classifying near zone vegetation, and estimating their height and their relative distance to the power lines [7].

3.2.3. Icing detection and measurement

In cold weather conditions, such as during snowstorm, hail, and freezing rain, thick icing accumulated on power lines can cause serious damage to the lines which may lead to power supply interruption [11]. One of the most important tasks in icing detection and measurement is estimating the icing thickness parameter, which is a crucial data source for energy companies to make decisions for icing accident prevention. The parameter is usually calculated by analyzing images collected from fixed monitoring terminals, which are typically mounted on power poles, for small areas. For large areas, foot patrol and helicopter-assisted inspections are usually conducted with the help of binoculars and/or telescopes to manually observe and estimate the icing thickness parameter [12].

3.2.4. Disaster monitoring

Natural disasters, such as storms, hurricanes, and earthquakes, can cause a great deal of damage to power lines that could result in outages or even completely shut down whole power grids. Thus, we are in an urgent need for a fast approach for damage assessments to quickly recover the power grids after natural disasters. However, according to our review, only a few research articles have focused on this topic.

3.3. Data sources for power line inspection

Inspired by the comprehensive review of different remote sensing data sources for automatic inspection conducted by Matikainen et al. in [4], in this section, we briefly summarize different relevant data sources for automatic autonomous vision-based power line inspection with special attention paid to highlighting their advantages and point out their potential applications.

3.3.1. Synthetic aperture radar images

Synthetic aperture radars are active imaging sensors, which are typically used for creating 2- or 3-dimensional images of objects, such as landscapes. SAR is also commonly used for change detection and 4-D mapping (space and time). SAR images are usually acquired by SARs mounted on various platforms such as small aircrafts and satellites [13]. Because SAR can provide high-resolution, day-and-night, and weather-independent images of large areas, it has been used for various applications in power line inspection such as vegetation mapping and monitoring [4].

3.3.2. Optical satellite images

Optical satellite images are collected by passive satellite sensors in the visible and Near-InfraRed (NIR) wavelengths. Although satellite imagery is very expensive and usually does not work well under cloudy and dark conditions, it offers many advantages for power inspection such as large-area coverage and multispectral data. According to [4], optical satellite images have been mainly used for vegetation monitoring in power line inspection.

3.3.3. Optical aerial images

Optical aerial images are usually collected by either a manned helicopter or a fixed wing aircraft, mainly in the visible and near-infrared wavelengths. With the flexibility in data acquisition and the ability to collect detailed images of conductors, pylons, power components (e.g., insulators, transformers, crossarms, toppads), and surrounding areas, optical aerial imaging has been used in many power line inspection tasks, for example vegetation monitoring, mapping of conductors and pylons, and monitoring faults in power line components [4].

3.3.4. Thermal images

Thermal images are formed by thermographic cameras, which are also known as infrared cameras or thermal imaging cameras. Thermographic cameras are based on infrared radiation to extend our vision beyond the short-wavelength red into the far infrared by making visible the light naturally emitted by warm objects [14]. With that ability, thermal images have been used for fault monitoring in power line components-based measurement and analysis of relative temperature differences [4].

3.3.5. Ultraviolet images

Ultraviolet (UV) images are typically formed using UV-sensitive cameras (e.g., Corona 350 II system) by recording images using light from the ultraviolet spectrum only. There are two main approaches for capturing ultraviolet images: reflected ultraviolet and ultraviolet induced fluorescence photography [15]. With the ability to visualize details which are invisible to the unaided human eye, images taken with ultraviolet light have been widely used for detecting corona discharges from high voltage electric power transmission lines [16].

3.3.6. Airborne laser scanner data

Airborne laser scanning is an active remote sensing technique that uses a laser beam as the sensing carrier [17]. This type of system is usually realised with the ability to obtain range images. This laser radar is also known as LADAR (LAser Detection And Ranging) and LIDAR (LIght Detection And Ranging). ALS data is a georeferenced point cloud of LIDAR measurements [18]. According to [4], ALS data is applicable for mapping of conductors, pylons, and individual trees near power lines since ALS can produce detailed 3D data directly without the dependence on external lightning conditions.

3.3.7. Land-based mobile mapping data

Land-based mobile mapping data is geospatial data collected by mapping sensors that are mounted on a mobile platform such as cars, All-Terrain Vehicles (ATV), boats, or on a backpack carried by a person. A mobile mapping system typically includes an image data collecting module (e.g., camera, laser scanner), navigation and positioning sensors (e.g., GPS), and an Inertial Measurement Unit (IMU) [19]. The two most common types of land-based mobile mapping data used in the power line inspection literature are images and point clouds. Because of the panoramic imaging geometry and the ability to produce very detailed 3D data and images, land-based mobile mapping data has been used for mapping conductors and pylons, and for inspecting power line components [4].

3.3.8. Unmanned aerial vehicle data

According to [4], the two most common types of data that have been acquired by UAVs in the power line inspection literature are optical images and laser scanning data. Because of the low operation costs, the high flexibility in data acquisition, and the ability to fly relatively close to power lines to take detailed pictures, UAV data has been used for both mapping and inspection of power lines components, and for detailed mapping of vegetation.

3.4. Automatic vision-based power line inspection

To build an automatic autonomous vision-based power line inspection system, we need three main components: an inspection method (e.g., UAV inspection, helicopter-assisted inspection), a primary data source (e.g., optical images, thermal images, SAR images), and a method for data analysis.

Based on the reviews presented above, we propose UAVs as a promising means for automatic autonomous vision-based inspection because of the following reasons: First, UAV inspection has significantly lower operation costs compared to other inspection methods, such as helicopter-assisted inspection and foot patrol inspection. Second, UAVs have the ability to fly relatively close to power lines to take detailed pictures, which is very useful for detecting small faults, for instance broken wires and missing splints. Third, with the ability to access hard-to-reach locations with high speeds, UAVs are considered as a highly promising solution for many inspection tasks, for example for

monitoring damage on power lines caused by natural disasters, or for estimating the height of near zone vegetation, for detecting dangerous surrounding objects, and for detecting and measuring icing thickness on power lines (Fig. 1). Finally, recent advances in battery and fuel cell technologies [20], sensors, and UAV components [21] have significantly improved the feasibility of UAV-based power line inspection.

We propose optical images collected by UAVs as a potential data source because (i) they are easy to collect, (ii) relatively easier to analyze than the other reviewed data sources while (iii) providing enough information for detecting a wide range of common faults on both power components (Fig. 2) and the power lines themselves (Fig. 1).

To have a closer look into different data analysis approaches for both navigation and inspection, we present a thorough literature review of current vision-based inspection systems with special attention paid to systems that are relevant to UAVs and optical images in the next section.

4. A review of current vision-based inspection systems

In this section, we highlight some of the most recently published vision-based approaches which are relevant for developing fully automatic autonomous vision-based power line inspection systems using UAVs. The review is divided in two parts. The first part summarizes vision-based approaches that are suitable for UAV navigation including power line detection-based and pole detection-based approaches. The second part is dedicated to vision-based approaches which have the potential for automating the four main UAV inspection tasks: icing detection and measurement, vegetation encroachment monitoring, mapping and inspection of power line components, and disaster monitoring.

4.1. Vision-based approaches for navigation

According to [9], there are three common approaches for UAV navigation in automatic power line inspection: GPS way points-based, pole detection-based, and power line detection-based. While the first approach has been widely applied for decades, the other two approaches have recently been facilitated by advances in visual recognition. In this section, we focus on reviewing recently published vision-based approaches for navigation in power line inspection with special attention paid to highlighting approaches which are relevant to UAVs.

4.1.1. Power line detection-based approaches

A knowledge-based technique specifically designed for power line detection in aerial images was proposed in [22] by combining bottom-up and top-down line detection approaches. First, a filter based on Pulse Coupled Neural Network (PCNN) was applied to remove background. Then, straight lines were detected using the Hough transform. Finally, spurious linear objects were eliminated using the K-means clustering approach. A similar three-step approach was also used in [23] for detecting power lines. First, linear objects in a clustered background were enhanced by a double-sided filter. Then, the Radon transform was used to detect straight lines. Finally, a parallel lines constraint was used to recognize power lines. Yang et al. proposed a three-step approach for detecting overhead power lines from UAV video images [24]. First, video images are binarized using an adaptive thresholding approach. Next, the Hough transform was utilized to detect lines-candidates in the binary images. Then, a Fuzzy C-Means (FCM) clustering algorithm, which uses the length and the slope of the detected lines as a feature vector, was applied to discriminate power lines from the detected line candidates. Finally, spurious lines were removed by using the properties of power lines.

In [25], an improved version of a Bayesian classifier was proposed for power line detection. The traditional Bayesian classifier was enhanced by utilizing heuristic knowledge obtained by applying the Hough transform to determine the prior and posterior probabilities. The Bayesian classifier was also used in [26] for detecting power lines in images collected by helicopters. To begin with, the Hough transform was used to improve the Bayesian classifier for classifying pixels. Then, small components (misclassified pixels) were removed from the classified images by performing connected component analysis.

A novel method named Circle Based Search (CBS) for power line detection based on the search of lines between two opposite points was proposed in [27]. First, Canny and Steerable filters were utilized to segment power line images taken from UAVs. Then, the CBS method, which was based on geometric relationships, together with a procedure for connecting contiguous segments were used for detecting power lines from the segmented images.

A sequential local-to-global power line detection algorithm for detecting power lines from optical images was proposed in [28]. First, line segments with symmetrical edges were detected using a Matched Filter (MF) and First-order Derivative Of Gaussian (FDOG) to create an edge map. Next, non-power-line candidates were filtered out by using a morphological filter. Finally, a graph-cut model based on graph theory was proposed to group the detected line segments into whole power lines.

To address the threshold selection problem in traditional edge detection approaches as well as the Hough transform, Zhou et al. proposed a method that can effectively tune the threshold in edge and line detection algorithms in [9]. A database was created for keeping optimal parameters for visited places, which can be retrieved later based on GPS coordinates. For unvisited places, an objective function and a power line model were used to predict the best parameters.

Zhang et al. proposed a method for detecting and tracking power lines in complex environments [29]. First, line segments were extracted using the Hough transform. Second, an improved K-means algorithm in Hough space was employed to cluster the extracted line segments to detect power lines. Finally, the detected power lines were tracked by a Kalman filter in Hough space.

Gerke et al. proposed a method for detecting and tracking power lines from cluttered backgrounds [30]. To begin with, a range filter was used to filter input images in four directions. Next, the images were converted to binary images, and morphological operations were then employed to remove unwanted objects. Then, the Canny edge detector and the Hough transform were used to detect power lines. Finally, the detected power lines were tracked with a gimbal system.

Jones et al. proposed a method for detecting and tracking power lines [31]. First, the contrast of input images was enhanced. Second, an edge map was formed by applying gradient computation and non-maximum suppression. Then, the Hough transform was computed followed by clustering points in the Hough transform to form the Aggregated Hough transform (AHT). Finally, tracking and acquisition were performed on the points in the AHT.

In summary, the presented approaches follow a general six-stage line detection/tracking process. First, input images are enhanced to remove noise. Second, backgrounds are removed using, for example, color-based background suppression and/or pulse coupled neural networks. Third, edge detectors, such as the Canny edge detector and steerable filters [27], are applied to generate edge maps. Fourth, straight lines from the generated edge maps are detected using, for instance, the Hough transform, the Radon transform, and/or the circle based search [27]. Fifth, clustering approaches (e.g., the K-means clustering and fuzzy C-mean clustering [24]) and/or power line constraints, such as parallel lines, are applied to eliminate spurious lines and detect power lines. Finally, the detected power lines are tracked using, for example, visual tracking methods based on fuzzy logic and Kalman filters. Although this general six-stage line detection/tracking system is simple to implement and has been widely used in power line inspection, it has many drawbacks, such as low speeds and inaccuracy. Thus, this approach is not well-suited for high-speed, fully autonomous vision-based navigation.

4.1.2. Pole detection-based approaches

An approach for identifying power pylons using UAVs and estimating the relative distance between the pylons and the UAVs was proposed in [32]. First, a Line Segment Detector (LSD) algorithm was adopted to detect lines segments. Second, line segments belonging to the background were removed by a color filter. Third, a triple matching strategy which uses Euclidean distance between the description histograms of the two segment candidates as metric and Left Right Checking (LRC) as criterion was proposed to match line segments across image sequences. Finally, an approach that integrates ego-motion of the UAV and the variation of the object size for estimating the depth of the pylons was proposed.

Golightly and Jones proposed an approach for detecting and matching corners for visual tracking in power line inspection [33]. To begin with, a corner map that contains clusters of corners was produced by a Cooper, Venkatesh, Kitchen (CVK) corner detector. Then, singleton corner points were removed by cluster aggregation. Finally, a basic corner matcher was proposed to track the detected corners in image sequences.

An approach for detecting and tracking electric towers was proposed in [34]. First, a two-class MultiLayer Perceptron (MLP) neural network was trained for tower-background classification based on HOG (Histogram of Oriented Gradients) features. Second, the network was applied as a sliding window detector. Third, a hierarchical tracking-by-registration tower tracker algorithm, which uses hierarchical multi-parametric and multi-resolution inverse compositional, was proposed to track the detected towers.

It has been shown in the last two sections that pole detection-based approaches for navigation have been received much less attention compared to power line detection-based approaches. This is partly due to its main drawback which is the lack of information for navigating between poles.

In conclusion, although many attempts have been made to improve vision-based navigation algorithms in the last decades, to the best of our knowledge, no high-speed, fully autonomous, vision-based navigation system for power line inspection has been successfully developed. Thus, we are in an urgent need for a novel vision-based navigation system which is not only fast but also accurate and capable of providing enough information for navigation between poles.

4.2. Vision-based approaches for inspection

As presented in Section 3.2, there are four main inspection tasks that are well-suited for automatic autonomous vision-based inspection using UAVs: mapping and inspection of power line components, vegetation monitoring, icing detection and measurement, and disaster monitoring. In this section, we review some of the most recently published vision-based research approaches which have the potential in automating the above mentioned tasks.

4.2.1. Mapping and inspection of power line components

Haibin et al. proposed an approach for detecting dampers in helicopter inspection of power transmission lines [35]. First, a threshold method was adopted to segment dampers from inspection images. Then, a modified version of the balloon force snake method and Hessian matrix were utilized to detect dampers from the segmented images.

A novel system utilizing Faster R-CNN (Region-based Convolutional Neural Networks) object detection framework and VGG16 model for detecting dead end body components (DEBC) was proposed in [36]. The authors proposed a series of simple image processing techniques for augmenting 146 input images to create 2437 training samples. The porposed data agumentation techniques include manually cropping, rotation, morphological operations (slight dilation and erosion), and “raster scan pattern” cropping. The system was tested on 111 aerial inspection photos, and achieved 83.7% accuracy and 91% precision. The detection accuracy and precision were increased to 97.8% and 99.1% by adding 270 additional training images and including a new insulator class.

A vision-based approach for broken spacer detection was proposed in [37]. To begin with, the Canny edge detector, the Hough transform, and scanning window approach were utilized to detect spacers as Region Of Interests (ROI). Second, image features were extracted using morphological operations. Finally, broken spacers were detected by connected domain analysis. Li et al. proposed an image processing system for conductor localization and spacer detection [38]. To begin with, conductors were localized by applying a template matching approach in a sliding window fashion. Then, the conductor localization results were used to crop and rotate images. Finally, Gabor filters were applied on the cropped and rotated images to detect spacers by looking for large clusters of pixels that respond strongly to the filtering process.

An approach based on Artificial Neural Networks (ANNs) for identifying poles and crossarms in images of power distribution lines was proposed in [39]. First, an ANN was trained to classify image pixels into four classes: poles, crossarms, vegetation, and others. Second, a threshold filter was used to remove unwanted areas. Finally, a template matching approach was applied to identify crossarms and poles. Sampedro et al. proposed an approach based on HOG features and MLP neural networks for detecting and classifying electric towers for power line inspection in [40]. First, a 2-class MLP was trained for background-foreground segmentation in a sliding-window fashion. Then, a multi-class MLP was trained for classifying within four different types of electric towers. A four-stage approach for detecting transmission towers from aerial images was proposed in [41]. First, filtering via optimized mean shift segmentation was utilized to reduce background clutter and simultaneously accentuate the foreground. Second, candidate granules were selected based on gradient density and cluster density based thresholding. Third, the selected granules were merged via shared boundary criterion. Finally, context information was used to discard false positives. An approach for detecting electricity pylons in aerial video sequences was proposed in [42]. First, straight line pixel extraction was performed by combing a two-dimensional separable Infinite Impulse Response (IIR) filter and non-maximal suppression. Next, straight lines were detected using the Hough transform. Finally, electricity pylon detection was performed based on the detected power lines.

Wang et al. proposed an approach for identifying insulators from aerial images in [43]. To begin with, a novel approach based on Otsu threshold and morphological operations was proposed for background suppression. Then, a Support Vector Machine (SVM) model was trained on features extracted by Gabor filters to classify insulators. In [44], a robust algorithm for detecting insulators in aerial images was proposed. First, an improved Harris corner selection strategy was used to detect local features. Second, a MultiScale and MultiFeature (MSMF) descriptor was proposed to represent the local features. Third, several Spatial Orders Features (SOFs) were found to improve the robustness of the algorithm. Finally, a coarse-to-fine matching strategy was utilized to determine the region of insulators. Zhao et al. proposed an approach based on lattice detection for detecting insulators in images of overhead transmission lines in [45]. First, corners detected by the KLT corner detection algorithm were used as low-level visual features. Mean shift clustering was then used to cluster the corners to find the repeating features that can represent the insulators. Next, insulators lattice modes were proposed based on a voting mechanism. Then, insulators were localized by performing lattice finding using a Markov Random Field (MRF) model together with the spacial context information. Finally, minimum bounding rectangles of the target image was extracted by analyzing the geometric characteristics of the insulators region. An approach for detecting and inspecting insulator were proposed in [46]. In the detection phase, a part-based model with Circular GLOH-Like (CGL) descriptor, which treats each insulator cap as one part of the model, was built. Next, a k-Nearest Neighbors (kNN) classifier with the descriptors of detected Difference of Gaussians (DoG) keypoints was

trained for distinguishing between insulators cap and clutter. Then, the bounding boxes for the insulators were determined from the classified keypoints. In the inspection phase, the detected insulators were first partitioned into caps. Then, an Elliptical GLOH-Like (EGL) descriptor was extracted from each cap. Finally, Local Outlier Factor (LOF) approach was utilized to identify faulty caps.

A method for detecting cables damaged by lightning strokes was proposed in [47]. To begin with, the statistical analysis of the brightness of the cable, which is based on the mean brightness of the cable and its standard deviation, was performed to detect arc marks. Then, shape information, which was obtained from the comparison between a real cable contour and an ideal cable contour, was utilized for detecting cut wires.

4.2.2. Vegetation encroachment monitoring

A novel method for monitoring vegetation encroachment of transmission lines using cameras integrated on each transmission pole was proposed in [48]. First, initial reference frames were acquired from the cameras. Next, image frames with inappropriate illumination (e.g., rainy images, foggy images) were filtered out. Then, the Hough transform was utilized to identify the horizontal and vertical lines of far away poles. Finally, motion tracking based on Laplacian kernel filtering and background-subtraction were applied to detect the encroached vegetation in the current scene images with respect to the reference images.

Larrauri et al. developed an automatic inspection system based on UAVs for buildings, trees, and vegetation encroachment monitoring and for detecting bad conductivity and hot spots on power lines in [10]. First, power lines were identified and recognized based on a regularized Hough transform. Second, the distance between the UAVs and the power lines was calculated. Third, the distance between the conductor lines and the ground was calculated based on inputs from laser altimeters. Fourth, trees, vegetation, and buildings were detected based on the identification of solid surfaces or group of pixels that have the same intensity value. Finally, stereoscopic vision methods using consecutive image frames were applied for calculating the distance from the conductor lines to trees, vegetation, and buildings.

4.2.3. Icing detection and measurement

Huang et al. proposed an approach for measuring icing thickness of transmission lines based on photogrammetry method in [12]. To begin with, images of power lines were obtained by a high resolution camera. Next, the distance and level angle of a target below the power lines were recorded by a laser rangefinder and an IMU. Then, icing thickness parameter was calculated based on the recorded distance and angel.

A new image-based algorithm for icing detection and icing thickness estimation was proposed in [11]. First, edge detection accuracy was improved by using an ice-prior-based scheme. Next, a novel method for calibrating a monitoring camera based on single-image-based scheme was proposed. Then, a new thickness estimation scheme was proposed by introducing calibration information and utilizing a ROI tracking scheme.

4.2.4. Disaster monitoring

Yan et al. proposed an approach for estimating the height of power transmission towers with the single TerraSAR-X imagery in [49]. According to the authors, the proposed approach opened many possibilities for monitoring the situation of power transmission towers in natural disaster conditions since it is based on images from SAR, which is capable of producing high resolution radar images in all-weather conditions.

A review of some of the possible applications of UAVs for routine and emergency power line inspection was presented in [50]. According to the authors, damage assessments after extreme events, such as hurricanes, wind storms, and snow storms, is the most immediate application for UAVs. With the ability to access hard-to-reach areas and fly at high speed, UAVs allow almost immediate assessment of power line damage after natural disasters.

To conclude, mapping and inspection of power line components is the main focus of most of the previous studies, whereas vegetation monitoring, icing detection and measuring, and disaster monitoring have been received much less attention. Many task-specific, traditional vision-based approaches have been proposed to detect and inspect power components (e.g., spaces and dampers) and power lines. A few deep learning approaches, which are based on ANNs, multi-class MLPs, and CNN-based object detectors such as Faster R-CNN, have been applied for detecting and inspecting poles, crossarms, and dead end body components. Although these approaches work to some extent, they still have many drawbacks. In the next sections, we analyze the drawbacks in detail, and based on that, we propose a new automatic autonomous power line inspection concept.

4.3. Limitations of current vision-based inspection systems

It can be clearly seen from the review that even though many attempts have been made to automate power line visual inspection, no fully automatic autonomous vision-based inspection system that is capable of detecting a wide range of faults has been successfully developed. Most of the previous studies presented in the review focused on proposing task-specific approaches for either inspection or navigation. Those task-specific approaches worked to some extent for tasks that they were designed for; however, they showed great limitations not only in terms of accuracy but also in terms of the ability to adapt to related tasks. Hence, a lot of efforts are required for developing a fully automatic autonomous vision-based inspection system since each subtask (e.g., detecting poles for navigation, detecting broken insulators, detecting missing toppads) requires its own carefully hand-designed solution.

Thus, we are in an urgent need for a new approach that is (i) more accurate, (ii) requires less effort in hand-designing solution, (iii) and generalizes well across related tasks. To address the existing limitations of current vision-based inspection systems, we propose deep learning as a potential data analysis approach for moving toward automatic autonomous vision-based inspection because of the following reasons: First, deep learning algorithms, especially Convolutional Neural Networks (CNNs, or ConvNets), have greatly improved the performance of visual recognition systems for many advanced applications such as self-driving cars, image search, and image understanding. Second, deep learning provides a general method for automatically learning features, which can dramatically reduce the effort in hand-designing solutions for every sub-task in power line inspection. Finally, deep learning approaches typically generalize well across related tasks [51]. The generalization ability of deep learning opens great possibilities for vision-based inspection since a model trained for one task with a very little effort in fine-tuning can be adapted for use in many other related tasks.

To facilitate the use of deep learning algorithms, particularly CNNs, in addressing the existing problems of current vision-based inspection systems, in the next section, we review the potential role of deep learning and CNNs as an advanced data analysis approach and as a potential solution to move forward automatic autonomous vision-based inspection.

5. The potential role of deep learning for automatic autonomous vision-based power line inspection

With the aim of utilizing recent advances in deep learning and UAV technologies for facilitating automatic autonomous UAV-based inspection of power line, eSmart Systems[1] has initiated a project, code named Connected Drone. The project is expected to run from 2016 to at least

[1] eSmart Systems: https://www.esmartsystems.com/.

until the end of 2018, and involves as of today 12 Norwegian power grid companies (such as Hafslund and Ringeriks-Kraft), universities (NORUT/ASUF, NTNU AMOS, and UiO), technology partners (Telenor, Microsoft, and Kongsberg), and drone experts (Bergen Robotics and Robot Aviation). The current work represented by this paper is funded by The Research Council of Norway (RCN) and eSmart Systems as an industrial PhD project in collaboration with the UiT Machine Learning Group.[2] In this project, we have developed a system for automatically mapping and inspecting power line components using CNN-based object detection and classification models (SSD [52] and ResNet [53]). The system is capable of mapping basic power components including poles, toppads, crossarms, and insulators (Fig. 3) and inspecting common faults on the components, for instance incorrectly mounted insulators, cracked poles, and missing toppads (Fig. 4). This line of work has in our opinion demonstrated a potential promising role of deep learning, especially CNNs, for automatic mapping and inspection of power line components, but has also revealed many challenges. From this starting point, we discuss in this section the potential role of deep learning and CNNs in automatic autonomous vision-based power line inspection using UAVs based on the reviews presented in Section 4 and our own work.

The remaining of the section is divided in three. First of all, we highlight the possibilities of DL vision-based navigation and DL vision-based inspection using UAVs. Secondly, we identify challenges of applying DL vision-based approaches in developing a fully automatic autonomous UAV-based inspection system. Finally, we suggest potential solutions to the challenges and propose approaches for facilitating the use of DL vision-based approaches in both UAV navigation and UAV inspection.

5.1. Possibilities of DL for vision-based UAV inspection

5.1.1. DL vision-based UAV navigation

DL vision-based navigation approaches (e.g., pole detection-based and power line detection-based) can be combined with traditional navigation approaches, such as GPS way points-based, and UAV autopilots to facilitate self-driving UAVs. This not only can dramatically reduce the risk of power line inspection, but it also can increase the speed of online inspections and data acquisition for offline inspections.

In addition, DL vision-based approaches hold great promise for facilitating the implementation of Automatic Data Acquisition (ADA) systems. In these systems, outputs from pole detectors and/or power line detectors are utilized for guiding cameras focus when taking pictures of power line components; as a result, the need of camera operators is eliminated and the quality of data acquisition is increased.

The combination of a self-driving UAV and an ADA system forms a basic automatic autonomous UAV-based power line inspection system. With a predefined flight plan, the UAV can automatically navigate along power lines to collect data for later offline inspections and perform online inspections.

5.1.2. DL vision-based UAV inspection

Advanced DL vision-based approaches and UAV technologies offer many possibilities for automating the four most common tasks in power line inspection: mapping and inspection of power line components, icing detection and measurement, vegetation encroachment monitoring, and disaster monitoring. For example, state-of-the-art object detectors powered by deep convolution neural networks (e.g., Faster R-CNN [54], SSD [52], YOLO [55], and R-FCN [56]) together with deep neural networks for image classification, such as ResNet [53], Inception-v4, and Inception-ResNet [57], can be used for detecting, classifying, and mapping power line components (Fig. 3). Next, deep learning-based semantic segmentation approaches (e.g., DPN [58] and Mask R-CNN [59]) and/or traditional background removal methods, for example color-based suppression [60] and pulse coupled neural filter [22], can be utilized for removing background from the detected components. Finally, inspections can be performed on the segmented images by, for example, performing texture analysis and/or vision-based anomaly detection to detect faults (Fig. 2). The above mentioned approaches are also useful in the case of disaster monitoring. For instance, a tree detection model can be used together with a power line detection model in detecting trees lying across or against power lines after natural disasters, such as storms, hurricanes, and earthquakes.

In the case of vegetation encroachment monitoring and icing detection and measurement, detecting power lines is one of the most challenging tasks. However, recent advances in deep vision have opened up many possibilities for addressing this challenge. For example, state-of-the-art deep CNN-based edge detectors, such as Holistically-Nested Edge Detection [61], and contour detectors, for instance DeepEdge [62] and DeepContour [63], can be applied to produce very high quality edge maps. The edge maps can then be used by traditional straight line detection methods (e.g., Hough transform [64]) to detect power lines.

5.2. Challenges of DL vision-based UAV inspection

5.2.1. DL vision-based UAV navigation

According to [9], there are three common approaches for UAV navigation in automatic power line inspection: pole detection-based, GPS way points-based, and power line detection-based. Unfortunately, none of the three methods offers a sufficient navigation accuracy for fully automatic autonomous UAV inspection.

Pole detection-based navigation detects electricity poles without providing specific methods for navigating along the power lines. In addition, when multiple poles are detected, it is very challenging to correctly identify the next target pole. In contrast, GPS way points-based navigation has no problem identifying the next target pole; nevertheless, it requires pre-specified exact locations of every electricity poles, which are typically not available for many existing power grids. Furthermore, GPS has a large error range that could result in low navigation accuracy. The power line detection-based navigation provides sufficient information for UAVs to navigate along the power lines; however, power lines are typically very thin and lack of rich features. Thus, detecting and tracking the lines are extremely challenging tasks. Another major challenge of the power line detection-based approach is “out-of-sight navigation”. During inspection, UAVs might be blown far away from power lines by, for example, strong wind. This could result in a deadlock since UAVs no longer “see” the power lines.

5.2.2. DL vision-based UAV inspection

Our work in automating UAV-based power line inspection has in our opinion demonstrated the potential role of deep learning for automatic mapping and inspection of power line, but has also revealed many challenges.

The first challenge is the lack of training data. Deep learning models for mapping and inspection of power line components typically require a huge amount of data for training. Unfortunately, to the best of our knowledge, there are no publicly available datasets that are big enough for training such models. The most straightforward solution is to create a training dataset from scratch by manually tagging images; however, it is a very slow, tedious, and expensive process. To move forward, we have created a medium-sized dataset by manually tagging 30,000 images with 54 classes (e.g., toppad_plastic, toppad_metal, pole, transformer). The average number of objects per image was 8; the average tagging speed (for a normal person) was 40 images per hour; thus, it required around 750 working hours to create the dataset.

The second challenge comes from the long-tailed distribution of component classes, which is also known as the class imbalance problem [65]. Datasets for training object detection, image classification, and

[2] UiT Machine Learning Group: http://site.uit.no/ml.

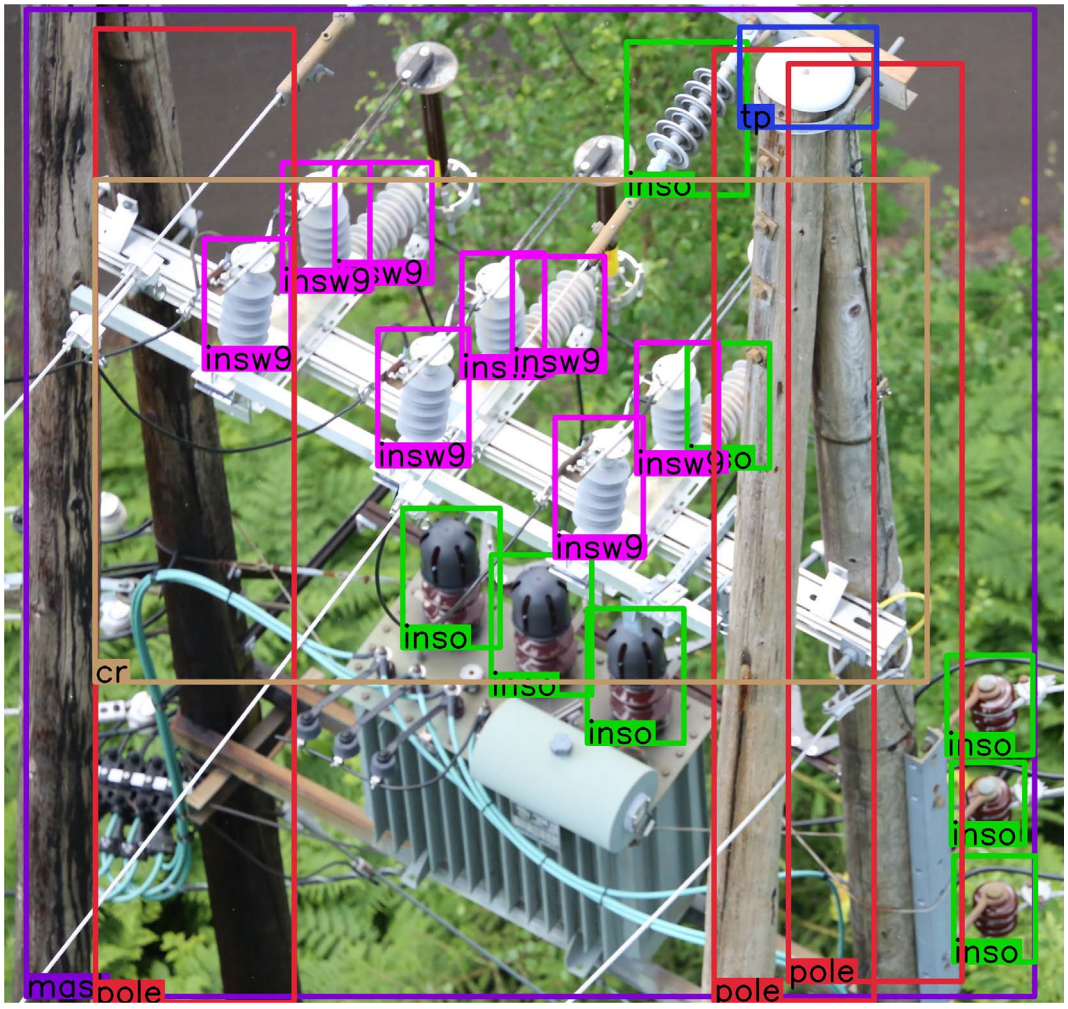


Fig. 3. An illustration of power line component mapping using our proposed power line inspection system. First, a power mast detection model is utilized to detect power masts from images acquired by UAVs. Then, the detected power masts are cropped from input images and passed through a power component detection model for mapping.

semantic segmentation models are typically required to have balanced classes. In other words, all classes in an ideal training dataset should have a similar number of examples [66]. Unfortunately, in power line images, a small number of component classes, such as Insulator-Brown-4 and Wooden-Pole, appear very often while most of the other classes, for instance Broken-Insulators and Toppad-Missing, appear more rarely. For example, in the 30,000-image dataset that we created, there are 43,275 examples of the Insulator-Brown-4 class while only 210 examples of the Toppad-Missing class are found (Fig. 5). The imbalance of object classes typically makes deep learning models biased towards classes that have more examples and overlook classes which have fewer examples. This is a huge problem in power line inspection since the classes which have very few examples are usually the faults that we need to identify, such as missing toppads, broken insulators, and cracked poles.

The third challenge is the detection of small power components and small faults. Most of the well-known CNN-based object detectors, such as SSD [52], YOLO [55], R-FCN [56], and Faster R-CNN [54], perform poorly on very small objects [67]. This is a major problem in mapping and inspection of power line components because many important power components (e.g., insulators) and faults on power lines, for instance missing toppads, broken wires, and broken insulators, are very small compared to other components (e.g., poles, crossarms) and surrounding objects, such as trees (Fig. 6).

The fourth challenge is the detection of unseen components and faults. Many existing power grids, especially in Norway, utilize a wide range of power components; thus, the grids are vulnerable to a huge number of types of faults. In addition, faults on power components and power lines, even the simple ones such as missing toppads, can occur in many different forms (Fig. 7). This, together with the lack of training data, pose a major challenge for DL vision-based inspection of power line due to the limitation of deep learning models in predicting previously unseen classes.

The fifth challenge is the detection of power lines in cluttered backgrounds. In both UAV navigation and UAV inspection (icing detection and measurement, vegetation encroachment monitoring), detecting power lines is one of the most crucial task. However, this is a very challenging task for two reasons: First, power lines in images taken from UAVs or helicopters are very thin leading to a lack of rich features for their representation. Second, because of color similarity, weather conditions, and lightning conditions, power lines are typically very difficult to be separated from backgrounds. For example. during the winter in cold countries, such as Norway, both power lines and the ground are usually covered in snow and/or ice making them indistinguishable, even for humans (Fig. 1).

The last but not least challenge is the lack of metrics for evaluating the performance of DL vision-based UAV inspection systems. Since there are no publicly available datasets for power line inspection that have a significant number of examples of faults, how to evaluate the performance of DL vision-based UAV inspection systems still remains an unsolved problem. Synthetic images of faults can be used to address the challenge to some degree; however, the performance of the system on

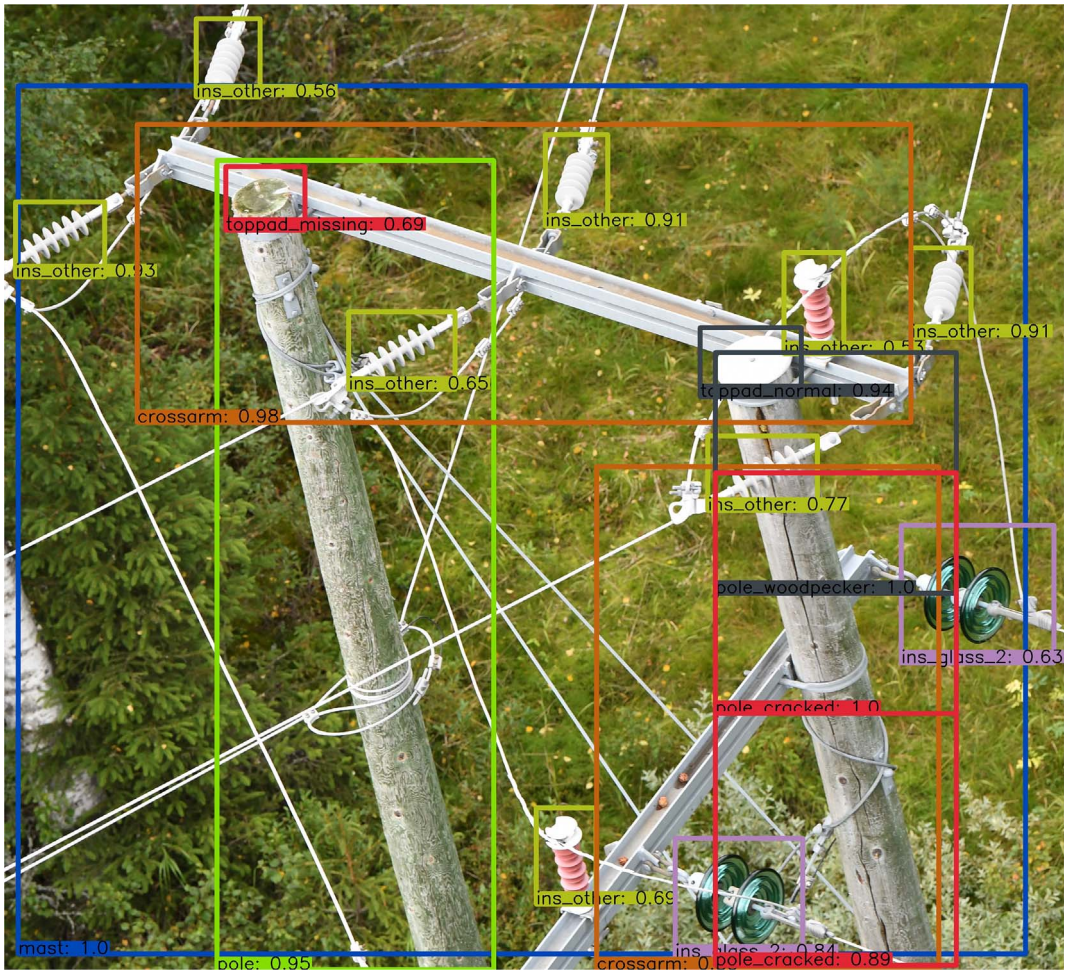


Fig. 4. An illustration of power line component inspection using our proposed power line inspection system. Results produced by power line component mapping models (Fig. 3) are passed through fault detection models to detect potential faults, such as missing toppads (toppad_missing), incorrectly mounted insulators (insb4_side), cracked poles (pole_cracked), and woodpecker attacks (pole_woodpecker).

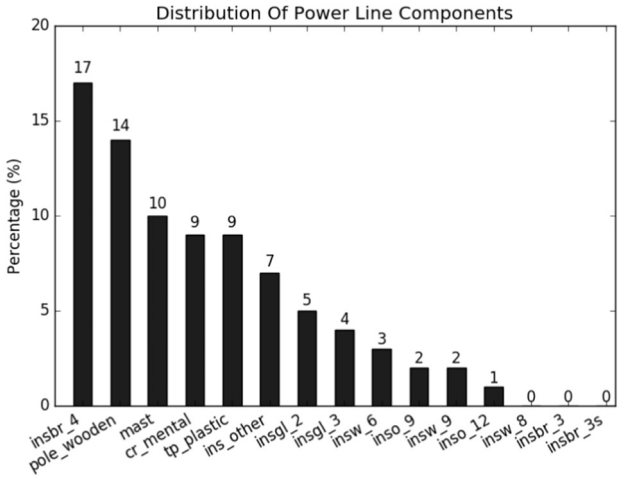


Fig. 5. Distribution of the 15 most common power line component classes in our 30,000-image dataset. While there are 43,275 (17%) examples of the Insulator-Brown-4 (insbr_4) class, only 210 (0.08%) examples of the Toppad-Missing (tp_missing) class are found.

synthetic images typically does not give an accurate estimation of the performance of the system on real images.

5.3. Possible solutions

5.3.1. DL vision-based UAV navigation

A potential solution for UAV navigation in automatic autonomous power line inspection is to combine the GPS way points-based, pole detection-based, and power line detection-based navigation approaches with an autopilot to build a hybrid navigation system. In this system, outputs from a pole detector can be used to together with GPS way points to accurately identify the next target poles. Based on that, an autopilot can be utilized to navigate the UAV to the identified poles by, for example, following the lines detected by a power line detector and/or tracker.

To address the “out-of-sight navigation” problem, wide angle cameras, such as 360-degree cameras, can be applied to ensure that UAVs can always “see” the power lines during navigation.

5.3.2. DL vision-based UAV inspection

There are four main approaches for mapping and inspection of power line components. The first approach is based on the comparison of power masts with their ideal models. This is a relatively simple approach; however, it is required that the ideal models must contain the perfect spatial configuration of the power masts, which is typically quite tedious, time-consuming, and expensive to create. The second

Fig. 6. Small faults on power lines components (from left to right): cracked insulator, missing splint, broken wire.

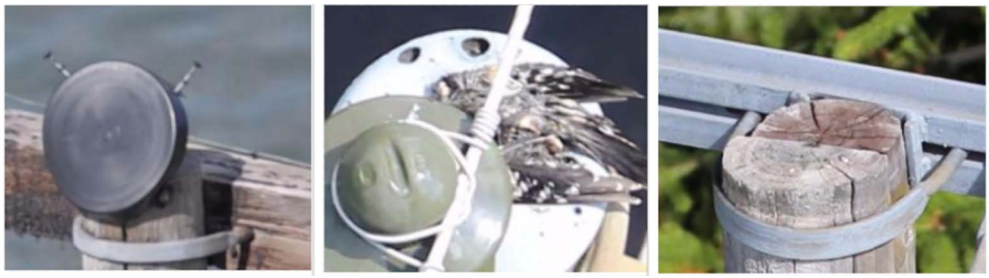


Fig. 7. Faults on toppads can be in many different forms (left to right): about to be missing, covered by unwanted objects, missing.

method is related to the detection of changes that appear after the last inspection. For example, the structure of a power mast, which is usually defined by its components and their relative position, can be estimated and compared with results from the previous inspection to detect changes, which may lead to power outages, for instance missing splints, missing toppads, and missing insulators. A similar approach can be applied for the detection of differences among a set of neighbouring pylons of the same type, usually in the same power line. Neighboring pylons of the same type in the same power line typically have the same architecture except for pylons at special locations; thus, faults on a power line can be detected by measuring the consistency of power mast structure on that line [68]. The final approach is related to the direct detection of faults from inspection images, usually based on deep learning models and/or traditional vision-based approaches, such as texture analysis and pattern matching. Each of the above-mentioned approaches has its own advantages and comes with unique challenges. Following are our proposed solutions to some of the major challenges.

One of the most straightforward approaches for dealing with the lack of training data problem is to manually create training data; however, this is a very slow, tedious, and expensive process. A potential way to speed up the process is to use pre-trained models and fine-tune them with a small amount of manually created training data to automatically create more data.

When only a small amount of training data is available, data augmentation techniques can be utilized to increase training performance. Some examples of simple data augmentation techniques that are useful for training deep learning models are flipping, cropping, and color jittering. Recent advances in image style transfer have opened up new possibilities in advanced data augmentation by, for example, transferring the time of day, weather, and season [69–71].

Another solution to the lack of training data problem is to use synthetic images. However, how to effectively combine synthetic images with real images in training deep learning models still a challenging question. To address this challenge to some extent, supervised domain adaptation [72] can be applied. In this approach, a model is first trained only on synthetic images and/or images from related tasks; it is then fine tuned for the target task that typically has very few training examples. In the case of no training examples available, unsupervised domain adaptation [73–76], which is capable of adapting models trained only on synthetic images and/or images from related tasks to use on the target task, is a potential solution.

The class imbalance problem can be tackled to some extent with synthetic images by, for example, generating more synthetic images for classes that have fewer training samples to balance out the imbalanced classes. Another alternative solution is to use the median frequency balancing approach in which classes with fewer training examples will be assigned higher weights during training [77–79].

To detect small components and faults, object detection, image classification, and semantic segmentation pipelines can be utilized. For example, a power mast detection model can be applied to locate masts in inspection images and crop them as ROIs. Then, detailed component and fault detection models can be employed to detect small components (e.g., insulators, toppads) and faults (missing toppads, cracked poles) from the cropped ROIs. The detected components can be further cropped and used as inputs for more detailed fault detection models to detect smaller faults, for example missing splints, broken wires, and cracked insulators (Fig. 6).

To detect unseen components and faults, one-shot learning [80–83], which allows a trained model to learn to detect new classes (components and faults) from only one or a few examples per class, is a very promising approach. An alternative solution is to first train a model with synthetic images of components and faults, then adapt it for detecting real components and faults using unsupervised domain adaptation [73–76]. Recently, advances in Generative Adversarial Networks (GANs) [84] have opened new possibilities for unsupervised anomaly detection. GANs can be employed for learning the data distribution that generates normal components. Then, different metrics, such as discrimination score and residual score, can be combined and used as "anomaly score" to perform anomaly detection [85]. The main advantage of this approach is that the training requires only images of normal components, which are relatively easy to collect.

In vegetation encroachment monitoring and icing detection and measurement, a power line detection pipeline can be utilized to address the thin line and the lack of rich feature challenges. First, outputs from edge detection algorithms, for instance the Canny edge detector [86], Matched filter [87], and Holistically-Nested Edge Detection [61], contour detectors, such as, DeepEdge [62] and DeepContour [63], and/or line detectors, for example the Hough transform [64] and the Radon transform [88], can be used together with prior knowledge of power lines properties (e.g., parallel lines) to locate ROIs in low resolution

images. Then, the identified ROIs are mapped to and cropped from higher resolution images. Finally, more advanced line detectors algorithms can be employed to detect power lines from the cropped ROIs in which power lines are typically bigger and have richer features than those in original images.

To separate power lines from clustered backgrounds, background removal approaches can be used as a pre-progressing step prior to the edge detection and line detection steps. Some examples of background removal techniques are color based suppression [60], pulse coupled neural filter [22], and deep learning-based semantic segmentation (e.g., DPN [58] and Mask R-CNN [59]). After the background is removed, clustering approaches (e.g., the K-means clustering [89] and fuzzy C-means clustering [90]) together with power line constrains (e.g., parallel lines) can be combined to eliminate spurious linear objects and detect power lines.

5.4. DL multimodal inspection

Optical image based fault detectors are capable of detecting a wide range of visual faults, for example missing toppads, pole cracks, and woodpecker attacks. However, they have numerous drawbacks, such as their sensitivity to illumination changes [91] and their incapability of detecting faults that are invisible to the unaided human eye (e.g., hot spots on power line components). To overcome the drawbacks, information of a visible camera can be fused with information provided by other sensors, such as thermal cameras and ultraviolet cameras, to perform multimodal inspection [92].

Recently, deep learning has been successfully used for learning from multimodal data sources for various vision tasks, for instance polar bear detection [93] and pedestrian detection [91,94,95]. This approach can be easily adapted to fuse images from multiple sensors, such as optical cameras, thermal cameras, and ultraviolet cameras, to improve the performance of power line inspection systems. The fusion can typically take place at three different levels of abstraction: pixel-level, feature-level, and decision-level fusion [91].

In power line inspection, images from different sensors are suitable for detecting a different set of faults. Optical images, for example, are well-suited for detecting visual faults, whereas thermal images and ultraviolet images are useful for detecting faults that are invisible to the unaided human eye, such as equipment bad connections and corona discharges. Thus, fusing images from multiple sensors can extend the range of faults that inspection systems can detect. In addition, images channels fusion can also improve inspection performance since images channels such as optical images and thermal images typically provide complementary visual information which are useful for deep learning models [94].

6. Conclusion

In this paper, we have presented a thorough literature review of automatic power line inspection research including vision-based approaches for both UAV navigation (power line detection-based and pole detection-based approaches) and UAV inspection (mapping and inspection of power line components, vegetation encroachment monitoring, icing detection and measurement, and disaster monitoring).

Further, we have summarized the possibilities of DL vision-based approaches and UAVs in developing a fully automatic autonomous power line inspection system.

Following a comprehensive review of current vision-based automatic power line inspection research approaches, we have identified existing challenges of both DL vision-based navigation and DL vision-based inspection and discussed solutions to these challenges.

Finally, with the aim of providing an initial starting point for researchers who are interested in developing a fully automatic autonomous vision-based power line inspection system using UAVs, we have proposed four potential next steps: (i) combine pole detection-based, GPS way points-based, and power line detection-based navigation approaches with autopilots to facilitate self-driving UAVs and automatic data acquisition, (ii) utilize multistage object detection, classification, and segmentation pipelines to detect faults in various sizes, forms, and conditions, (iii) employ contextual information (e.g., power mast structure) to further improve fault detection performance by, for example, eliminating invalid faults, and (iv) apply multiple data sources fusion (e.g., infrared cameras, ultraviolet cameras, and 3D cameras) for detecting complicated faults (e.g., cracked insulators, rotten poles, and corona discharges).

Acknowledgment

The authors would like to thank eSmart Systems (Dang Ha The Hien and Huyen Thi Phuong Vu) and UiT Machine Learning Group (Michael Kampffmeyer) for support in the work with this paper. This work was supported by the Research Council of Norway [RCN NÆRINGSPHD Grant No. 263894 (2016–2018) on Power Grid Image Analysis] and eSmart Systems.

References

[1] Bruch M, Münch V, Aichinger M, Kuhn M, Weymann M, Schmid G. Power blackout risks. In: CRO forum; 2011. p. 28.
[2] Castillo A. Risk analysis and management in power outage and restoration: a literature survey. Electr Power Syst Res 2014;107:9–15.
[3] Pradeep Y, Khaparde SA, Joshi RK. High level event ontology for multiarea power system. IEEE Trans Smart Grid 2012;3(1):193–202.
[4] Matikainen L, Lehtomäki M, Ahokas E, Hyyppä J, Karjalainen M, Jaakkola A, Kukko A, Heinonen T, et al. Remote sensing methods for power line corridor surveys. ISPRS J Photogramm Rem Sens 2016;119:10–31.
[5] Katrasnik J, Pernus F, Likar B. A survey of mobile robots for distribution power line inspection. IEEE Trans Power Deliv 2010;25(1):485–93.
[6] Ahmad J, Malik AS, Xia L, Ashikin N. Vegetation encroachment monitoring for transmission lines right-of-ways: a survey. Electr Power Syst Res 2013;95:339–52.
[7] Mirallès F, Pouliot N, Montambault S. State-of-the-art review of computer vision for the management of power transmission lines. International conference on applied robotics for the power industry (CARPI). IEEE; 2014. p. 1–6.
[8] Luque-Vega LF, Castillo-Toledo B, Loukianov A, Gonzalez-Jimenez LE. Power line inspection via an unmanned aerial system based on the quadrotor helicopter. MELECON 2014-2014 17th IEEE Mediterranean electrotechnical conference. IEEE; 2014. p. 393–7.
[9] Zhou G, Yuan J, Yen I-L, Bastani F. Robust real-time UAV based power line detection and tracking. IEEE international conference on image processing (ICIP). IEEE; 2016. p. 744–8.
[10] Larrauri JI, Sorrosal G, González M. Automatic system for overhead power line inspection using an unmanned aerial vehicle relief project. International conference on unmanned aircraft systems (ICUAS). IEEE; 2013. p. 244–52.
[11] Zhong Y-p, Zuo Q, Zhou Y, Zhang C. A new image-based algorithm for icing detection and icing thickness estimation for transmission lines. IEEE international conference on multimedia and expo workshops (ICMEW). IEEE; 2013. p. 1–6.
[12] Huang H, Ma X, Zhao L, Du H, Luo H, Mao X, et al. Transmission line icing measurement on photogrammetry method. 9th International symposium on multispectral image processing and pattern recognition (MIPPR2015). vol. 9815. International Society for Optics and Photonics; 2015. p. 98150Z-1–6.
[13] Moreira A, Prats-Iraola P, Younis M, Krieger G, Hajnsek I, Papathanassiou KP. A tutorial on synthetic aperture radar. IEEE Geosci Rem Sens Mag 2013;1(1):6–43.
[14] Lloyd JM. Thermal imaging systems. Springer Science & Business Media; 2013.
[15] Richards A, Partner OP. Reflected ultraviolet imaging for forensics applications. Online < http://www.uvcorder.com/pdf/Reflected_UV_Imaging_for_Forensics_V2.pdf > .
[16] Chen L, Lin L, Tian M, Bian X, Wang L, Guan Z. The ultraviolet detection of corona discharge in power transmission lines. Energy Power Eng 2013;5(04):1298.
[17] Wehr A, Lohr U. Airborne laser scanning an introduction and overview. ISPRS J Photogramm Rem Sens 1999;54(2):68–82.
[18] Beraldin J, Blais F, Lohr U. Laser scanning technology. Airborne Terr Laser Scan 2010:1–42.
[19] Tao CV, Li J. Advances in mobile mapping technology vol. 4. CRC Press; 2007.
[20] Savvaris A, Xie Y, Malandrakis K, Lopez M, Tsourdos A. Development of a fuel cell hybrid-powered unmanned aerial vehicle. 24th Mediterranean conference on control and automation (MED). IEEE; 2016. p. 1242–7.
[21] Zhang T, Li Q, Zhang C-S, Liang H-W, Li P, Wang T-M, et al. Current trends in the development of intelligent unmanned autonomous systems. Front Inform Technol Electron Eng 2017;18(1):68–85.
[22] Li Z, Liu Y, Hayward R, Zhang J, Cai J. Knowledge-based power line detection for UAV surveillance and inspection systems. 23rd International conference image and vision computing New Zealand. IEEE; 2008. p. 1–6.
[23] Zhu L, Cao W, Han J, Du Y. A double-side filter based power line recognition method for UAV vision system. IEEE international conference on robotics and

biomimetics (ROBIO). IEEE; 2013. p. 2655–60.
[24] Yang TW, Yin H, Ruan QQ, Da Han J, Qi JT, Yong Q, et al. Overhead power line detection from UAV video images. 19th International conference mechatronics and machine vision in practice (M2VIP). IEEE; 2012. p. 74–9.
[25] Du S, Van Wyk BJ, Tu C. Heuristic bayesian pixel classification for power line inspection. 3rd International congress on image and signal processing (CISP), vol. 2. IEEE; 2010. p. 960–3.
[26] Gaspar Z, Shengzhi Du B. Hough transform tuned bayesian classifier for overhead power line inspection. In: Proceedings of the 19th annual symposium of the pattern recognition association of South Africa; 2008. p. 137–40.
[27] Cerón A, Prieto F, et al. Power line detection using a circle based search with UAV images. International conference on unmanned aircraft systems (ICUAS). IEEE; 2014. p. 632–9.
[28] Song B, Li X. Power line detection from optical images. Neurocomputing 2014;129:350–61.
[29] Zhang J, Liu L, Wang B, Chen X, Wang Q, Zheng T. High speed automatic power line detection and tracking for a UAV-based inspection. International conference on industrial control and electronics engineering (ICICEE). IEEE; 2012. p. 266–9.
[30] Gerke M, Seibold P. Visual inspection of power lines by UAS. International conference and exposition on electrical and power engineering (EPE). IEEE; 2014. p. 1077–82.
[31] Jones D, Golightly I, Roberts J, Usher K. Modeling and control of a robotic power line inspection vehicle. Proceedings of the 2006 IEEE international conference on control applications. IEEE; 2006. p. 632–7.
[32] Araar O, Aouf N, Dietz JLV. Power pylon detection and monocular depth estimation from inspection UAVs. Ind Robot: Int J 2015;42(3):200–13.
[33] Golightly I, Jones D. Corner detection and matching for visual tracking during power line inspection. Image Vis Comput 2003;21(9):827–40.
[34] Martinez C, Sampedro C, Chauhan A, Campoy P. Towards autonomous detection and tracking of electric towers for aerial power line inspection. International conference on unmanned aircraft systems (ICUAS). IEEE; 2014. p. 284–95.
[35] Haibin W, Yanping X, Weimin F, Xiaoming S, Li J. Damper detection in helicopter inspection of power transmission line. Fourth international conference on instrumentation and measurement, computer, communication and control (IMCCC), 2014. IEEE; 2014. p. 628–32.
[36] Nordeng IE, Hasan A, Olsen D, Neubert J. DEBC detection with deep learning. Scandinavian conference on image analysis. Springer; 2017. p. 248–59.
[37] Song Y, Wang L, Jiang Y, Wang H, Jiang W, Wang C, et al. A vision-based method for the broken spacer detection. IEEE international conference on cyber technology in automation, control, and intelligent systems (CYBER). IEEE; 2015. p. 715–9.
[38] Li WH, Tajbakhsh A, Rathbone C, Vashishtha Y. Image processing to automate condition assessment of overhead line components. 1st International conference on applied robotics for the power industry (CARPI). IEEE; 2010. p. 1–6.
[39] Castellucci PB, Lucca LC, SantAnna M, Traballe G, Mustacio VH, da Silva JFR, et al. Pole and crossarm identification in distribution power line images. Robotics symposium and competition (LARS/LARC), 2013 Latin American. IEEE; 2013. p. 2–7.
[40] Sampedro C, Martinez C, Chauhan A, Campoy P. A supervised approach to electric tower detection and classification for power line inspection. International joint conference on neural networks (IJCNN). IEEE; 2014. p. 1970–7.
[41] Dutta T, Sharma H, Vellaiappan A, Balamuralidhar P. Image analysis-based automatic detection of transmission towers using aerial imagery. Iberian conference on pattern recognition and image analysis. Springer; 2015. p. 641–51.
[42] Tilawat J, Theera-Umpon N, Auephanwiriyakul S. Automatic detection of electricity pylons in aerial video sequences. International conference on electronics and information engineering (ICEIE), vol. 1. IEEE; 2010. p. V1-342–6.
[43] Wang X, Zhang Y. Insulator identification from aerial images using support vector machine with background suppression. International conference on unmanned aircraft systems (ICUAS). IEEE; 2016. p. 892–7.
[44] Liao S, An J. A robust insulator detection algorithm based on local features and spatial orders for aerial images. IEEE Geosci Rem Sens Lett 2015;12(5):963–7.
[45] Zhao J, Liu X, Sun J, Lei L. Detecting insulators in the image of overhead transmission lines. International conference on intelligent computing. Springer; 2012. p. 442–50.
[46] Oberweger M, Wendel A, Bischof H. Visual recognition and fault detection for power line insulators. In: 19th Proceedings of computer vision winter workshop (CVWW); 2014. p. 81–8.
[47] Ishino R, Tsutsumi F. Detection system of damaged cables using video obtained from an aerial inspection of transmission lines. Power engineering society general meeting. IEEE; 2004. p. 1857–62.
[48] Ahmad J, Malik AS, Abdullah MF, Kamel N, Xia L. A novel method for vegetation encroachment monitoring of transmission lines using a single 2d camera. Pattern Anal Appl 2015;18(2):419–40.
[49] Yan L, Wu W, Li T. Power transmission tower monitoring technology based on TerraSAR-X products. International symposium on lidar and radar mapping technologies. vol. 8286. International Society for Optics and Photonics; 2011. p. 82861E-1–7.
[50] Toth J, Gilpin-Jackson A. Smart view for a smart grid unmanned aerial vehicles for transmission lines. 1st International conference on applied robotics for the power industry (CARPI). IEEE; 2010. p. 1–6.
[51] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature 2015;521(7553):436–44.
[52] Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C-Y, et al. SSD: single shot multibox detector. European conference on computer vision. Springer; 2016. p. 21–37.
[53] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 770–8.
[54] Ren S, He K, Girshick R, Sun J. Faster r-cnn: towards real-time object detection with region proposal networks. In: Advances in neural information processing systems; 2015. p. 91–9.
[55] Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 779–88.
[56] Dai J, Li Y, He K, Sun J. R-FCN: object detection via region-based fully convolutional networks. In: Advances in neural information processing systems; 2016. p. 379–87.
[57] Szegedy C, Ioffe S, Vanhoucke V, Alemi A. Inception-v4, inception-resnet and the impact of residual connections on learning. Available from: arXiv:1602.07261.
[58] Liu Z, Li X, Luo P, Loy C-C, Tang X. Semantic image segmentation via deep parsing network. In: Proceedings of the IEEE international conference on computer vision; 2015. p. 1377–85.
[59] He K, Gkioxari G, Dollár P, Girshick R. Mask r-cnn. Available from: arXiv:1703.06870.
[60] Rajeev M, Adithya V, Hrishikesh S, Balamurali P. Detection of power-lines in complex natural surroundings. Comput Sci Inform Technol (CS & IT) 2013;3:101–8.
[61] Xie S, Tu Z. Holistically-nested edge detection. In: Proceedings of the IEEE international conference on computer vision; 2015. p. 1395–403.
[62] Bertasius G, Shi J, Torresani L. Deepedge: a multi-scale bifurcated deep network for top-down contour detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2015. p. 4380–9.
[63] Shen W, Wang X, Wang Y, Bai X, Zhang Z. Deepcontour: a deep convolutional feature learned by positive-sharing loss for contour detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2015. p. 3982–91.
[64] Duda RO, Hart PE. Use of the hough transformation to detect lines and curves in pictures. Commun ACM 1972;15(1):11–5.
[65] Krawczyk B. Learning from imbalanced data: open challenges and future directions. Prog Artif Intell 2016;5(4):221–32.
[66] Ouyang W, Wang X, Zhang C, Yang X. Factors in finetuning deep model for object detection with long-tail distribution. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 864–73.
[67] Huang J, Rathod V, Sun C, Zhu M, Korattikara A, Fathi A, et al. Speed/accuracy trade-offs for modern convolutional object detectors. Available from: arXiv:1611.10012.
[68] Mazurek P, Okarma K. Application of background estimation and removal techniques for the extraction of the power line components on the digital images for the automatic power line inspection systems. Pomiary Automatyka Kontrola 2008;54:698–9.
[69] Luan F, Paris S, Shechtman E, Bala K. Deep photo style transfer. Available from: arXiv:1703.07511.
[70] Gatys LA, Ecker AS, Bethge M. Image style transfer using convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 2414–23.
[71] Zhu J-Y, Park T, Isola P, Efros AA. Unpaired image-to-image translation using cycle-consistent adversarial networks. Available from: arXiv:1703.10593.
[72] Csurka G. Domain adaptation for visual applications: a comprehensive survey. Available from: arXiv:1702.05374.
[73] Ganin Y, Lempitsky V. Unsupervised domain adaptation by backpropagation. Available from: arXiv:1409.7495.
[74] Long M, Zhu H, Wang J, Jordan MI. Unsupervised domain adaptation with residual transfer networks. In: Advances in neural information processing systems; 2016. p. 136–44.
[75] Long M, Cao Y, Wang J, Jordan MI. Learning transferable features with deep adaptation networks. In: ICML; 2015. p. 97–105.
[76] Sener O, Song HO, Saxena A, Savarese S. Learning transferrable representations for unsupervised domain adaptation. In: Advances in neural information processing systems; 2016. p. 2110–8.
[77] Eigen D, Fergus R. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In: Proceedings of the IEEE international conference on computer vision; 2015. p. 2650–8.
[78] Badrinarayanan V, Kendall A, Cipolla R. SegNet: a deep convolutional encoder-decoder architecture for image segmentation. Available from: arXiv:1511.00561.
[79] Kampffmeyer M, Salberg A-B, Jenssen R. Semantic segmentation of small objects and modeling of uncertainty in urban remote sensing images using deep convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops; 2016. p. 1–9.
[80] Fei-Fei L, Fergus R, Perona P. One-shot learning of object categories. IEEE Trans Pattern Anal Mach Intell 2006;28(4):594–611.
[81] Santoro A, Bartunov S, Botvinick M, Wierstra D, Lillicrap T. One-shot learning with memory-augmented neural networks. Available from: arXiv:1605.06065.
[82] Vinyals O, Blundell C, Lillicrap T, Wierstra D, et al. Matching networks for one shot learning. In: Advances in neural information processing systems; 2016. p. 3630–8.
[83] Mehrotra A, Dukkipati A. Generative adversarial residual pairwise networks for one shot learning. Available from: arXiv:1703.08033.
[84] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al. Generative adversarial nets. In: Advances in neural information processing systems; 2014. p. 2672–80.
[85] Schlegl T, Seeböck P, Waldstein SM, Schmidt-Erfurth U, Langs G. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. International conference on information processing in medical imaging. Springer; 2017. p. 146–57.
[86] Canny J. A computational approach to edge detection. IEEE Trans Pattern Anal Mach Intell PAMI-8 1986(6):679–98. http://dx.doi.org/10.1109/TPAMI.1986.4767851.

[87] Turin G. An introduction to matched filters. IRE Trans Inform Theory 1960;6(3):311–29.
[88] Radon J. On the determination of functions from their integral values along certain manifolds. IEEE Trans Med Imag 1986;5(4):170–6.
[89] Lloyd S. Least squares quantization in PCM. IEEE Trans Inform Theory 1982;28(2):129–37.
[90] Dunn JC. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. Cybernetics 1973;3:32–57.
[91] Wagner J, Fischer V, Herman M, Behnke S. Multispectral pedestrian detection using deep fusion convolutional neural networks. In: 24th European symposium on artificial neural networks, computational intelligence and machine learning (ESANN); 2016. p. 509–14.
[92] Gade R, Moeslund TB. Thermal cameras and applications: a survey. Mach Vis Appl 2014;25(1):245–62.
[93] Sorensen S, Treible W, Hsu L, Wang X, Mahoney AR, Zitterbart DP, et al. Deep learning for polar bear detection. Scandinavian conference on image analysis. Springer; 2017. p. 457–67.
[94] Liu J, Zhang S, Wang S, Metaxas DN. Multispectral deep neural networks for pedestrian detection. Available from: arXiv:1611.02644.
[95] König D, Adam M, Jarvers C, Layher G, Neumann H, Teutsch M. Fully convolutional region proposal networks for multispectral person detection. IEEE conference on computer vision and pattern recognition workshops (CVPRW). IEEE; 2017. p. 243–50.

Appendix B

Paper II

IEEE Power and Energy Technology Systems Journal

Received 25 June 2018; revised 6 July 2018, 17 August 2018 and 9 October 2018; accepted 8 November 2018. Date of publication 11 January 2019; date of current version 25 March 2019.

Digital Object Identifier 10.1109/JPETS.2018.2881429

Intelligent Monitoring and Inspection of Power Line Components Powered by UAVs and Deep Learning

VAN NHAN NGUYEN[1,2], ROBERT JENSSEN[1], (Member, IEEE), AND DAVIDE ROVERSO[2]

[1]The UiT Machine Learning Group, UiT/The Arctic University of Norway, 9019 Tromsø, Norway
[2]Analytics Department, eSmart Systems, 1783 Halden, Norway

CORRESPONDING AUTHOR: V. N. NGUYEN (nhan.v.nguyen@uit.no)

This work was supported in part by the Research Council of Norway [RCN NÆRINGSPHD Grant 263894 (2016–2018) on Power Grid Image Analysis] and eSmart Systems.

ABSTRACT In this paper, we present a novel automatic autonomous vision-based power line inspection system that uses unmanned aerial vehicle inspection as the main inspection method, optical images as the primary data source, and deep learning as the backbone of the data analysis. To facilitate the implementation of the system, we address three main challenges of deep learning in vision-based power line inspection: (i) the lack of training data; (ii) class imbalance; and (iii) the detection of small components and faults. First, we create four medium-sized datasets for training component detection and classification models. Furthermore, we apply a series of effective data augmentation techniques to balance out the imbalanced classes. Finally, we propose the multi-stage component detection and classification based on the Single Shot Multibox detector and deep Residual Networks to detect small components and faults. The results show that the proposed system is fast and accurate in detecting common faults on power line components, including missing top caps, cracks in poles and cross arms, woodpecker damage on poles, and rot damage on cross arms. The field tests suggest that our system has a promising role in the intelligent monitoring and inspection of power line components and as a valuable addition to smart grids.

INDEX TERMS Intelligent monitoring, power line inspection, vision-based power line inspection, deep learning, UAVs, smart grids.

I. INTRODUCTION

TO PREVENT power outages, electric utilities regularly perform visual inspections on their power grids to plan for necessary repair or replacement work. These inspections have typically been carried out using traditional methods such as foot patrol and helicopter-assisted surveys, which are typically slow, expensive, and potentially dangerous. In recent years, many researchers have been seeking to develop fast and accurate methods for automatic autonomous power line inspection. Unfortunately, to the best of our knowledge, no fully automatic autonomous power line inspection systems have been developed.

The work presented in this paper aims to realize fast, accurate, and safe automatic autonomous power line inspection, and is part of an ongoing effort to exploit recent advances in Deep Learning (DL) and Unmanned Aerial Vehicle (UAV) technologies for this purpose. In our previous work [1], we conducted a review of the main power line inspection tasks, the existing power line inspection methods, and the potential data sources for power line inspection. Based on that, we proposed a novel automatic autonomous vision-based power line inspection concept that uses UAV inspection as the main inspection method, optical images as the primary data source, and deep learning as the backbone of the data analysis. To move forward, we identified six main challenges of DL vision-based UAV inspection: the lack of training data; class imbalance; the detection of small components and faults; the detection of previously unseen components and faults; the detection of power lines in cluttered backgrounds; and the lack of metrics for evaluating inspection performance.

In this paper, we take this concept further and address the first three challenges by creating four medium-sized datasets for training component detection and classification models, by applying a series of effective data augmentation techniques

to balance out the imbalanced classes, and by utilizing multi-stage component detection and classification based on Single Shot multibox Detector (SSD) [2] and deep Residual Networks (ResNets) [3] to detect small components and faults.

Having addressed the first three challenges, we build a basic automatic vision-based power line inspection system with two custom-built UAVs and five DL-based models for data analysis and inspection. The remaining three challenges are left for future work which involves facilitating self-driving UAVs with power line detection as well as advancing automatic inspection at large-scale with a wide range of faults to realize fully automatic autonomous power line inspection.

The work presented in this paper does, in our opinion, demonstrate the potential role and the importance of automatic autonomous power line inspection in the intelligent monitoring of power grids. High-speed UAVs equipped with sensors, cameras, and DL vision-based models for navigation and data analysis can automatically navigate along power lines to collect data for offline inspections and perform online inspections to quickly identify faults on both power line components and the power lines themselves.

The remainder of the paper is structured as follows: Section II presents background knowledge and relevant related work on UAV inspection, vision-based inspection, and DL-based classification and detection models, before we describe our proposed automatic autonomous vision-based power line inspection concept in Section III. Next, in section IV, we present in detail our proposed approaches. Then, in Section VI, we present experimental results and discuss the potential of our proposed system in the intelligent monitoring of power grids. Finally, in Section VII, we conclude the paper with a summary and an outlook for the future of the field.

II. BACKGROUND AND RELATED WORK

Power lines are traditionally inspected at regular intervals by foot patrol or by helicopter-assisted surveys. In these inspection methods, a team of inspectors is sent out traveling either on foot or by helicopter to collect data for offline inspections and for visual inspection of the power lines. To improve inspection speed, accuracy, and to reduce inspection costs, a considerable amount of research has been conducted in order to automate vision-based power line inspection.

A. UAV INSPECTION

In recent years, advances in battery and fuel cell technologies [4], sensors, and UAV components [5] have significantly improved the feasibility of UAV-based power line inspection. However, the current applications of UAVs in power line inspection are still facing many unsolved challenges.

Deng *et al.* [6] identified three main challenges that prevent UAV inspection from daily services. The first challenge is the application modality. A single UAV typically can only cope with a specific power line inspection task; thus, multi-UAVs are usually required to perform full inspections. The second challenge is the lack of communication and control systems, and the third challenge is the gap between data collection and data analysis. To address these challenges, the authors proposed a cooperative power line inspection paradigm using multi-platform UAVs: a fixed wing UAV for long-distance brief inspection, a multi-rotor UAV for short-distance thorough inspection, and a tethered muti-rotor UAV for communication relay. Field tests suggested that the proposed approach outperforms traditional inspection methods (e.g., foot patrol).

B. VISION-BASED INSPECTION

With recent advances in deep learning for computer vision, cameras, and sensors, vision based power line inspection is drawing increasing attention from the power industry. The main reason why vision-based inspection is popular is that it can cover a wide range of faults on a single inspection [1].

Reviews of different data sources for vision-based inspection and existing vision-based inspection systems can be found in [1] and [7]. Based on the reviews, the current authors proposed optical images collected by UAVs as a potential data source for vision-based inspection because (i) they are easy to collect; (ii) relatively easier to analyze than the other reviewed data sources, while; (iii) providing enough information for detecting a wide range of common faults on both power line components and the power lines themselves.

C. DL-BASED CLASSIFICATION AND DETECTION MODELS

In the past few years, Convolutional Neural Networks (CNNs) [8], which are a special kind of neural networks designed to take advantage of the 2D structure of image data, have been advancing the state of the art of many computer vision applications, such as image recognition and object detection. In this section, we briefly describe the underlying concept of CNNs and summarize some of the most well-known CNN architectures for image classification as well as CNN-based frameworks for object detection.

The four key ideas behind the successes of CNNs in processing image data are local connections, shared weights, pooling, and the use of many layers [9]. A CNN for image classification is typically composed of three types of layers: convolutional layers, pooling layers, and fully-connected layers.

Convolutional layers are the fundamental component of CNNs which leverages the three main ideas that make CNNs powerful: local connectivity, parameter sharing and equivariant representations [10]. A convolutional layer accepts a volume I of size $[W_I, H_I, D_I]$ as input and outputs a volume O of size $[W_O, H_O, D_O]$. The convolutional layer is composed of several convolution *kernels* K (often called *filters*). Each neuron in the output volume looks at a rectangular region in the input volume. The rectangular region is referred to as the neuron's *receptive field* in the previous layer, and the size of the region is often called the *filter size* [11]. The filters are slided across the input volume I with *stride* S to compute dot

products to produce *activation maps*:

$$O_K(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n). \tag{1}$$

In practice, many deep learning libraries implement an alternative function called the cross-correlation:

$$O_K(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n). \tag{2}$$

According to [10], pooling layers "replace the output of the net at certain locations with summaries statistic of the nearby outputs". There are many pooling functions that can be used in pooling layers, such as max pooling, average pooling, and $L2$-norm pooling. However, in practice, it is recommended to use the max pooling function, which takes a rectangular region P as input and outputs the maximum value of the elements in the region. The pooling function is slided across the input volume I with stride S to compute activation maps:

$$O_P(i,j) = \max_{m,n \in P(i,j)} I(m,n). \tag{3}$$

Pooling layers in CNNs serve two main purposes. First, pooling introduces invariance to small translations in the input. The second is that pooling reduces the amount of parameters and computation in the network by progressively reducing the spatial dimension of the input volume. However, it has been shown that max pooling can simply be replaced by a convolutional layer with increased stride [12].

Fully-connected layers, which are typically responsible for high-level reasoning in CNNs, are composed of neurons that are connected to all activations in the previous layer, as seen in regular neural networks.

Many well-known deep CNNs, such as AlexNet [8] and VGGNet [13], are formed by simply stacking up many convolutional layers, pooling layers, and fully-connected layers. In those deep CNNs, the information flowing through the network passes through many stages of multiplication; therefore, the gradients are needed to be back-propagated though many stages during training. This causes the gradients to either vanish or explode. The exploding gradients problem can be addressed easily by, for example, applying gradient clipping. The vanishing gradients, on the other hand, are quite hard to overcome. When the gradients vanish, the learning either becomes very slow or stops working. This issue is historically known as one of the main challenges of training very deep CNNs. An example of the vanishing gradient problem's cause is the use of saturated activation functions such as the hyperbolic tangent (*tanh*) or the logistic sigmoid [14]. In modern CNNs, it is recommended to use non-saturated activation functions, which typically suffer less from the vanishing gradient problem, such as the Rectified Linear Units (ReLU), as alternatives to the hyperbolic tangent or logistic sigmoid [15]. The ReLU is defined as

$$ReLU(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0. \end{cases} \tag{4}$$

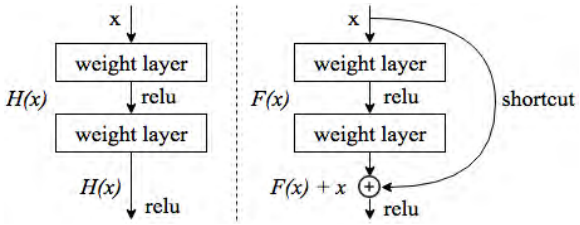


FIGURE 1. Standard CNNs (left) vs ResNets with shortcut connections (right). $H(x)$ is the underlying mapping. $F(x) = H(x) - x$ is the residual mapping adopted by ResNets.

For more details on the underlying concept of CNNs and their existing challenges, we refer the interested reader to [9], [10], and [11].

1) RESNET

With the increasing complexity of image classification problems, deeper CNNs are typically required. However, as mentioned above, deep CNNs constructed simply by stacking up many layers are very difficult to train due to the notorious problem of vanishing/exploding gradients. To ease the training of deep CNNs, Residual Networks (ResNets) were proposed [3]. ResNets add "shortcut" connections to the standard CNNs layers to allow the gradient signal to travel back directly from later layers to early layers (See Fig. 1). The "shortcut" connections allowed the authors of the ResNets to successfully train very deep CNNs with 50, 101, and even 152 layers.

2) FASTER R-CNN

Inspired by the successes of CNNs in image classification, Faster R-CNN (Region-based Convolutional Neural Network) was proposed to solve a more challenging task of object detection. Faster R-CNN is a single, unified network which performs object detection via two main steps: region proposal and region classification. First, a base network (e.g., ResNet [3]) is utilized to extract features from images. Next, the extracted features are fed into a Region Proposal Network (RPN) to find proposals. Then, a CNN-based classifier is applied on top of the extracted feature maps to classify the proposals and refine their bounding boxes, which are rectangles that enclose a single detected object. Finally, post-processing is used to refine the bounding boxes and eliminate duplicate detections. Faster R-CNN is very accurate; however, it is quite slow.

3) R-FCN

R-FCN (Region-based Fully Convolutional Network) is an accurate and efficient object detection framework proposed to address existing issues of region-based detectors such as Fast/Faster R-CNN [16]. Instead of applying costly per-region sub-network hundreds of times, R-FCN adopts a fully convolutional architecture with almost all computations shared across the entire image. To address the dilemma between translation-invariance in image classification and

translation-variance in object detection, R-FCN proposes novel position-sensitive score maps which allow fully convolutional networks to effectively and efficiently perform both classification and detection in a single evaluation. With those novel improvements, R-FCN can run at 2.5-20 times faster and achieve higher accuracy than the Faster R-CNN counterpart.

4) YOLO

YOLO (You Only Look Once) is a real-time object detection framework that directly predicts bounding boxes and class probabilities with a single network in a single evaluation [17]. To achieve this, YOLO unifies region proposal and region classification into a single neural network and, according to the authors, ''frames object detection as a regression problem to spatially separated bounding boxes and associated class probabilities''. YOLO divides the input image into an $S \times S$ grid. Each grid cell predicts B bounding boxes, confidence score for those boxes, and C conditional class probabilities. With a unified architecture, YOLO is extremely fast; it processes images in real-time. However, YOLO is not state-of-the-art in terms of accuracy.

5) SSD

SSD (Single Shot MultiBox Detector) improves YOLO by adding a series of modifications: (i) a small convolutional filter is utilized to predict object classes and offsets in bounding box locations; separate predictors (filters) are employed for predicting object at different aspect ratios; predictions are performed at multiple feature maps from the later stages of a network to enable detection at multiple scales [2]. These modifications make SSD both faster and more accurate than the YOLO counterpart.

III. THE PROPOSED AUTOMATIC AUTONOMOUS VISION-BASED POWER LINE INSPECTION CONCEPT

With the aim of utilizing recent advances in deep learning and UAV technologies to realize fast, accurate, and safe power line inspection, we propose a novel automatic autonomous vision-based power line inspection concept that uses UAV inspection as the main inspection method, optical images as the primary data source, and deep learning as the backbone of the data analysis.

A. CUSTOM-BUILT UAVS

We built two custom UAVs for two main inspection purposes (see Fig. 2). The first UAV is a full-scale UAV designed for large-scale inspections. The UAV is based on a Gryphon Dynamics XV-1400 frame; it is equipped with a Nvidia TX1 GPU and an Auvidea j140 carrier. Three cameras are mounted on the UAV: a Sony DSC-QX30U, a FLIR with USB frame grabber, and an USB FPV 2mpix camera. The UAV uses Kongsberg Seatex MBR-144 OEM for radio communication. The UAV is powered by four Tattu 22000mAh 22.2V 25C 6S1P Lipo Battery packs which allow it to fly up to 42 minutes. The UAV can fly at an average speed of 60km/h and can lift up to 40kg. The UAV is quite big and heavy; however, it is very stable when flying.

FIGURE 2. **Our custom-built full-scale UAV (left) for large-scale inspections and back pack UAV (right) for small-scale inspections.**

The second UAV is a backpack UAV built for small-scale inspections. The UAV is based on a 3DR Solo which is powered by a 5200 mAh 14.8V DC Lithium Polymer battery. It is customized by adding a custom gimbal from HDAir Studio and a Raspberry Pi computer for managing cameras. The UAV is equipped with a Sony DSC-QX30U camera. The UAV can fly up to 20 minutes at an average speed of 12km/h.

B. ACQUIRED OPTICAL IMAGES

Optical images are used as the main data source for inspection. Images are collected directly using cameras mounted on the UAVs. The UAVs are flown along power lines and circled around power masts to take pictures of the masts from different angles. For each power mast, around 20 images at 6048x4032 resolution are collected. The images are uploaded to the Microsoft Azure cloud after the flight for inspection.

C. DL-BASED DATA ANALYSIS AND INSPECTION

All images of power masts are analyzed by our component detection models to detect common power line components: poles, cross arms, top caps, and insulators. The detected components are then classified into more fine-grained power components classes using our component classification models and used as inputs for identifying faults. Images with potential faults will be assigned a higher priority in the inspection queue with the aim of reducing inspection time.

IV. DL VISION-BASED UAV INSPECTION

A. DATA ACQUISITION

Deep learning models for vision-based tasks typically require a huge amount of annotated data to train well. Unfortunately, to the best of our knowledge, there are no publicly available datasets that are big enough for satisfactory training of such models.

To move forward DL vision-based UAV inspection, we created four medium-sized datasets for training component detection and component classification models. The images in our datasets were collected using high quality DSLR cameras (e.g., Nikon D810, Canon EOS 5D Mark III, Nikon D3X) from helicopters and with multiple resolutions (e.g., 7360x4912, 6048x4032, 5760x3840). To increase the diversity of the data, we combined images from multiple power

TABLE 1. Properties of the DS1_Co, DS2_Tc, DS3_Po, and DS4_Cr datasets.

Dataset	Annotation type	#Images	Image size	Speed (images/hour)	Total time (hour)
DS1_Co	BB + label	28674	6048x4032	40	716.85
DS2_Tc	label	34002	256x256	1000	34
DS3_Po	label	26446	256x256	1000	26.5
DS4_Cr	label	34029	256x256	1000	34

grids in Norway, which were provided by Hafslund Nett and Troms Kraft.

The first dataset (DS1_Co), which is used for training component detection models, is annotated with bounding boxes (BB). The description of bounding boxes is (x_1, y_1, x_2, y_2), where (x_1, y_1) and (x_2, y_2) are the $(left, top)$ and $(right, bottom)$ locations of the bounding boxes. Each bounding box is associated with one of the 54 most common power line component classes that we selected. The selected component classes include three power pole classes (wooden poles, concrete poles, cracked poles), four cross arm classes (wooden cross arms, concrete cross arms, cracked cross arms, metal cross arms), three top cap classes (metal top caps, plastic top caps, missing top caps), a class for transformers, and 43 insulator classes including pin insulators, suspension insulators, and strain insulators.

The second dataset (DS2_Tc), which is used for training missing top cap detectors, is created by cropping top caps from images in the first dataset and annotating them with two classes, missing top caps and normal top caps. The third dataset (DS3_Po), which is used for training cracks in poles and woodpecker damage on poles detectors, is created by cropping poles from images in the first dataset, dividing the crops into overlapping squares, and annotating the squares with three classes, normal poles, cracked poles, and woodpecker-damaged poles.

The final dataset (DS4_Cr), which is used for training cracks on cross arms and rot damage on cross arms detectors, is created by rotating cross arm bounding boxes from images in the first dataset to remove background, cropping the rotated bounding boxes, dividing the crops into overlapping squares, and annotating the squares with four classes, cracked cross arms, rot-damaged cross arms, normal cross arms, and background. Properties and sample images of the four datasets are shown in Table 1 and Fig. 3.

As can be seen from Table 1, annotating images with labels only for training component classification models is quite fast; however, annotating images with both bounding boxes and labels for training component detection models is quite slow. The average annotating speed (in our experience) is 40 images/hour; thus, it requires around 717 hours to create our medium-sized DS1_Co dataset.

As discussed in our previous paper [1], these datasets come with many challenges. In the next sections, we describe the use of data augmentation techniques and our proposed multi-state component detection and classification approach to address the first three challenges of DL vision-based UAV inspection, including the lack of training data, class imbalance, and the detection of small components and faults.

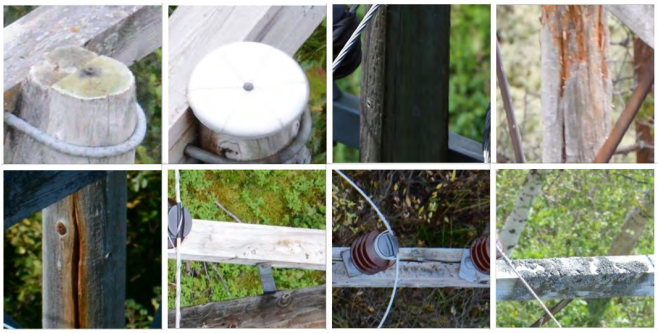


FIGURE 3. Sample images of the DS2_Tc, DS3_Po, and DS4_Cr datasets (from left to right, top to bottom): missing top cap, normal top cap, normal pole, woodpecker-damaged pole, cracked pole, normal cross arm, cracked cross arm, and rot-damaged cross arm.

B. DATA AUGMENTATION

Inspired by the successes of traditional data augmentation techniques in addressing the class imbalance and the lack of training data challenge [18], we propose a series of effective data augmentation techniques to generate more training data by applying transformations in the data-space.

To train a robust component detector for our pipeline, we combine original images with mast crops generated by the mast detector (see Fig. 5). When a mast is detected, its predicted bounding box is padded to be a square and cropped from the original image to generate one additional training image. The square is also slightly shifted in four directions (left, right, top, bottom) to generate four more training images. In addition, the training images are randomly flipped during training. These data augmentation techniques allow us to generate a training set that is 12 times bigger than our original training set.

To balance out the imbalanced classes and generate enough data to properly train our component classifiers, a series of effective data augmentation techniques are applied. All the data augmentation techniques are implemented using the scikit-image library [19].

The first technique involves adding Gaussian-distributed additive noise to account for noise that arises during image acquisition (e.g., sensor noise caused by poor illumination and/or high temperature, and/or transmission) [20]. The augmented image $f(i, j)$ is the sum of the true image $s(i, j)$ and the noise $n(i, j)$:

$$f(i, j) = s(i, j) + n(i, j). \tag{5}$$

The noise term, $n(i, j)$, follows a Gaussian random distribution:

$$p_G(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}}, \tag{6}$$

where z represents the grey level, μ is the mean value, and σ is the standard deviation.

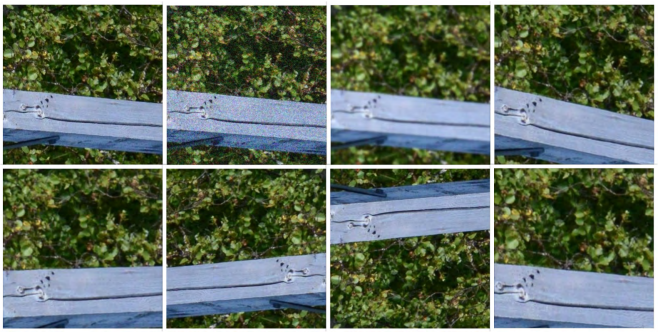


FIGURE 4. **Sample augmented images (from left to right, top to bottom): original image, image with Gaussian-distributed additive noise, Gaussian blurred image, left-rotated image, right-rotated image, horizontally flipped image, vertically flipped image, and center-cropped and zoomed in image.**

To account for possible out of focus, Gaussian blur is employed by convolving the image with a two-dimensional Gaussian function:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}, \tag{7}$$

where x and y are distances from the origin in the horizontal axis and the vertical axis respectively, and σ is the standard deviation [21].

To account for various camera distances and viewing angles, zoom and rotation operators are performed [22]. The zoom operator is applied by randomly cropping images and scaling them to their original size. The rotation operator is performed by using a rotation matrix R:

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix},$$

where θ is the rotation angle. The final technique involves flipping the images horizontally and vertically (see Fig. 4).

C. MULTI-STAGE COMPONENT DETECTION AND CLASSIFICATION

To tackle the detection of small power components and small faults challenge, we propose a multi-stage component detection and classification pipeline. The pipeline consists of five components connected as shown in Fig. 5. The pipeline works as follows: First, the mast detector detects power masts from input images. Then, the detected masts are cropped from the input images and used as inputs for the component detector to detect power components including top caps, poles, cross arms, and insulators. Finally, the detected top caps, poles, and cross arms are cropped from the input images and passed through their corresponding classifiers to identify faults. Algorithm 1 explains the workflow of the pipeline in detail. The pipeline allows us to mimic the ''zoom-in'' operation during inspection which enables the detection of small faults on power line components, such as cracks on poles and cross arms, woodpecker damage on poles, and rot damage on cross arms.

Algorithm 1 Multi-stage detection and classification

Input: Input image I, Mast detector's confidence threshold m_{thres}, Component detector's confidence threshold c_{thres}, Classification confidence threshold cls_{thres}, Mast detector $MD(I)$ that outputs bounding box coordinates (m_coords) and confidence scores (m_confs) of the detected masts, Component detector $CD(I)$ that outputs labels (c_labels), bounding box coordinates (c_coords), and confidence scores (c_confs) of the detected components, Classifier name list N, Classifier list indexed by name C (each classifier CLF in C takes an image as input and outputs a label (cls_label) and a confidence score cls_conf)

Output: A list of detected and classified components O (each item in O contains a label, bounding box coordinates, and a confidence score)

```
m_coords, m_confs ← MD(I)
m_index ← argmax_i(m_confs[i])
if m_confs[m_index] > m_thres then
    mast ← crop_image(I, m_coords[m_index])
else
    mast ← I
end if
c_labels, c_coords, c_confs ← CD(mast)
for c_index ← 1, size(c_labels) do
    c_label ← c_labels[c_index]
    c_coord ← c_coords[c_index]
    c_conf ← c_confs[c_index]
    if c_conf > c_thres then
        comp ← crop_image(mast, c_coord)
        if c_label ∈ N then
            CLF ← C[c_label]
            cls_label, cls_conf ← CLF.classify(comp)
        else
            cls_conf ← 0
            cls_label ← NONE
        end if
        if cls_conf > cls_thres then
            append [cls_label, c_coord, cls_conf] to O
        else
            append [c_label, c_coord, c_conf] to O
        end if
    end if
end for
```

To select an object detector for implementing the mast detector and the component detector, we evaluate four state-of-the-art object detectors, which are SSD [2], YOLO [17], Faster R-CNN [23], and R-FCN [16], in terms of speed and mean Average Precision (mAP) [24]:

$$mAR = \frac{1}{|classes|} \sum_{c \in classes} AP(c), \tag{8}$$

where $AP(c)$ is the average precision of class c which takes the mean precision at a set of eleven equally spaced recall

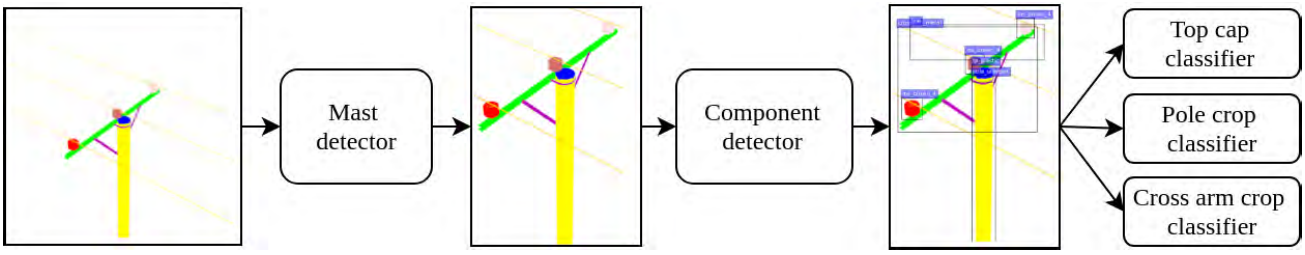


FIGURE 5. The general structure of our proposed multi-stage component detection and classification pipeline. The pipeline consists of five components: a mast detector, a component detector, a top cap classifier, a pole crop classifier, and a cross arm crop classifier.

levels [0, 0.1, ..., 1.0]:

$$\mathrm{AP(c)} = \frac{1}{11} \sum_{r \in (0, 0.1, ..., 1.0)} p_{interp}(r), \tag{9}$$

where $p_{interp}(r)$ is an interpolated precision that takes the maximum precision over all recalls that exceed r:

$$p_{interp}(r) = \max_{\tilde{r}:\tilde{r} \geq r} p(\tilde{r}), \tag{10}$$

where p and r are the precision and recall of class c:

$$p = \frac{TP(c)}{TP(c) + FP(c)}, \tag{11}$$

$$r = \frac{TP(c)}{TP(c) + FN(c)}, \tag{12}$$

where $TP(c)$, $FP(c)$, and $FN(c)$ are number of True Positives (TP), False Positives (FP), and False Negatives (FN) of class c respectively. A prediction is considered a true positive if its Intersection over Union (IoU) is greater than a predefined-threshold t (e.g., $t = 0.5$), otherwise it is considered a false positive. The IoU is defined as

$$IoU(G, P) = \frac{G \cap P}{G \cup P}, \tag{13}$$

where G and P are the ground-truth and the predicted bounding boxes respectively.

The performance of the four object detectors on the 21 most common insulator classes in our dataset is shown in Fig. 6 and Table 2. Although the SSD detector performs slightly worse than the R-FCN detector in terms of mAP, it is selected as our main object detector because of its speed, which is 3.47 times faster than that of the R-FCN detector.

TABLE 2. Performance of the SSD, YOLO, Faster R-CNN, and R-FCN detectors on our dataset.

Model	mAP	Speed (FPS)
SSD	**0.67**	**59**
Faster R-CNN	0.58	7
YOLO	0.61	45
R-FCN	0.68	17

In the last few years, many advanced CNN architectures have been proposed for image classification such as ResNet [3], Inception-v4 [25], and DenseNet [26]; however, ResNet was selected as our main classifier because it is easy to train, fast, and accurate.

V. EXPERIMENTS AND RESULTS

A. TRAINING

The component detectors and component classifiers are implemented using the Caffe [27] deep learning framework. To build the component detectors, we fine-tune the SSD512 model, which is pre-trained on the ILSVRC CLS-LOC dataset [24], using the Stochastic Gradient Descent optimizer (often shortened to SGD) with initial learning rate 0.001, 0.9 momentum, 0.0005 weight decay, and batch size 32 on four GPUs (3 Titan X Pascals and 1 GeForce GTX 1080 Ti).

The component classifiers are built by fine-tuning the ResNet50_cvgj [28] model, which is pre-trained on the ImageNet dataset [24], using the Adam optimizer [29] with initial learning rate 0.0001, 0.9 momentum1, 0.999 momentum2, 0.0001 weight decade, and batch size 16 on four GPUs (3 Titan X Pascals and 1 GeForce GTX 1080 Ti).

B. EXPERIMENTS

To evaluate the performance of our proposed multi-stage component detection pipeline and the effectiveness of our proposed data augmentation techniques, we conducted two experiments: a pipeline test and a data augmentation test. In the pipeline test, we compare against our proposed multi-stage component detection pipeline with data augmentation (MSCDP-Dataaug) and without data augmentation (MSCDP-Noaug) and a simple component detection model (SCDM) trained directly on original images. In the data augmentation test, we compare between the ResNet50_cvgj model trained with and without augmented data generated by our proposed data augmentation techniques for two tasks: pole crop classification and cross arm crop classification, respectively.

Since there are no publicly available datasets for power line inspection, both experiments are conducted on the DS1_Co, DS2_Tc, DS3_Po, and DS4_Cr datasets. The component detection models in all three methods in the first experiment are fine-tuned to detect ten component classes including poles, cross arms, top caps, and seven insulators classes on 80% of the images from the DS1_Co dataset. The remaining 20% of the images are used for evaluation. The models in the second experiment are trained on 80% of the data from

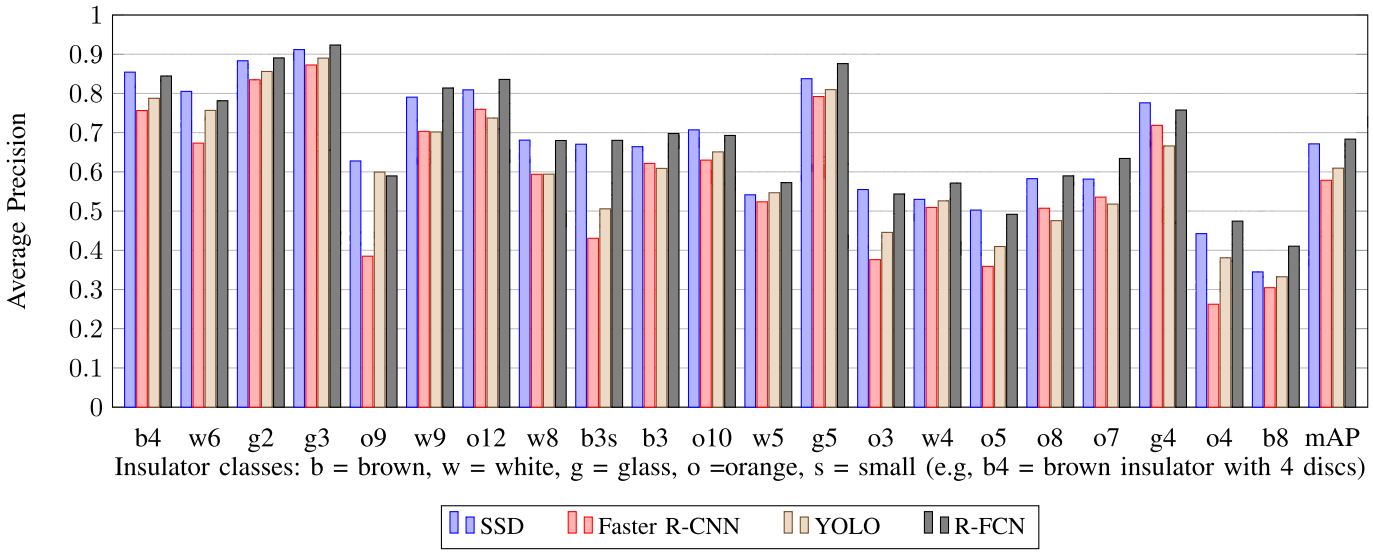


FIGURE 6. Mean Average Precision (mAP) of the SSD, YOLO, Faster R-CNN, and R-FCN detectors on the 21 most common insulator classes in our dataset.

the DS3_Po and DS4_Cr datasets. The remaining 20% of the data from the two datasets are used for testing.

The component detection models are evaluated in terms of *mAP* as defined in Equation 8. To account for class imbalance, the component classification models are evaluated in terms of weighted Precision (*wP*), weighted Recall (*wR*), and weighted F_1 score (wF_1):

$$wP = \frac{\sum_{c=1}^{|C|} sup(c) \cdot p(c)}{\sum_{c=1}^{|C|} sup(c)} \tag{14}$$

$$wR = \frac{\sum_{c=1}^{|C|} sup(c) \cdot r(c)}{\sum_{c=1}^{|C|} sup(c)} \tag{15}$$

$$wF_1 = \frac{\sum_{c=1}^{|C|} sup(c) \cdot F_1(c)}{\sum_{c=1}^{|C|} sup(c)} \tag{16}$$

where $sup(c)$, $p(c)$, $r(c)$, and $F_1(c)$ are the support, precision, recall, and F_1 score of class c respectively:

$$p(c) = \frac{TP(c)}{TP(c) + FP(c)} \tag{17}$$

$$r(c) = \frac{TP(c)}{TP(c) + FN(c)} \tag{18}$$

$$F_1(c) = 2 \cdot \frac{p(c) \cdot r(c)}{p(c) + r(c)} \tag{19}$$

where $TP(c)$, $FP(c)$, and $FN(c)$ are number of True Positives (TP), False Positives (FP), and False Negatives (FN) of class c respectively.

C. RESULTS

The detection results of the three methods in the pipeline test are shown in Table 3. Our proposed multi-stage component detection pipeline together with our proposed data augmentation techniques, i.e. the MSCDP-Dataaug method, achieves the best results in terms of *mAP* with 81.3% and outperforms the other two methods in 7/10 classes.

The test results of the pole crop classification and cross arm crop classification tasks in the data augmentation test are shown in Table 4 and Table 5 respectively. In both tasks, our proposed data augmentation techniques significantly improve *wP*, *wR* , and wF_1 score of the models. In particular, utilizing augmented data in training improves *wP*, *wR* , and wF_1 score by 8.93%, 8.42%, and 8.7% respectively on the pole crop classification task and by 1.98%, 0.56%, and 2% respectively on the cross arm crop classification task.

The top cap classifier is evaluated on a separate test set which consists of 681 missing top caps and 1103 normal top caps. The classifier achieves 0.987 weighted precision, 0.981 weighted recall, and 0.984 weighted F_1 score. Since the model achieves relatively high weighted precision, weighted recall, and weighted F_1 score, we skip data augmentation for this task.

VI. DISCUSSION

The testing results of the component detection models shown in Table 3 indicate that our proposed multi-stage component detection pipeline together with our proposed data augmentation techniques, i.e. the MSCDP-Dataaug method, can address the detection of small components and faults challenge. In particular, the MSCDP-Dataaug method achieves higher average precision on small insulator classes, such as insg3, inso9, insw9, insw6, and inso, compared to the simple component detection model, i.e. the SCDM method. In addition, the MSCDP-Dataaug method achieves 1.2% *mAP* higher than the SCDM method. This is due to the "zoom-in" operation enabled by our multi-stage component detection pipeline. By using outputs from the mast detector as inputs for the component detector, most of the irrelevant background is removed, and the relative sizes of the power components, especially the small ones, such as insulators and top caps, are significantly increased, resulting in richer features for the deep learning models to learn from.

TABLE 3. SCDM, MSCDP-Dataaug, and MSCDP-Noaug detection results on our medium-sized DS1_Co dataset.

Method	mAP	pole	cross arm	top cap	insb4	insg2	insg3	inso9	insw9	insw6	inso
SCDM	80.1	86.3	**85.7**	85.1	**84.2**	**90.6**	89.0	55.2	78.4	75.5	71.5
MSCDP-Noaug	78.5	85.9	83.4	84.3	78.7	87.4	87.4	53.8	78.9	76.5	69.1
MSCDP-Dataaug	**81.3**	**88.0**	83.4	**86.3**	82.4	89.0	**89.9**	**59.5**	**81.5**	**78.8**	**73.8**

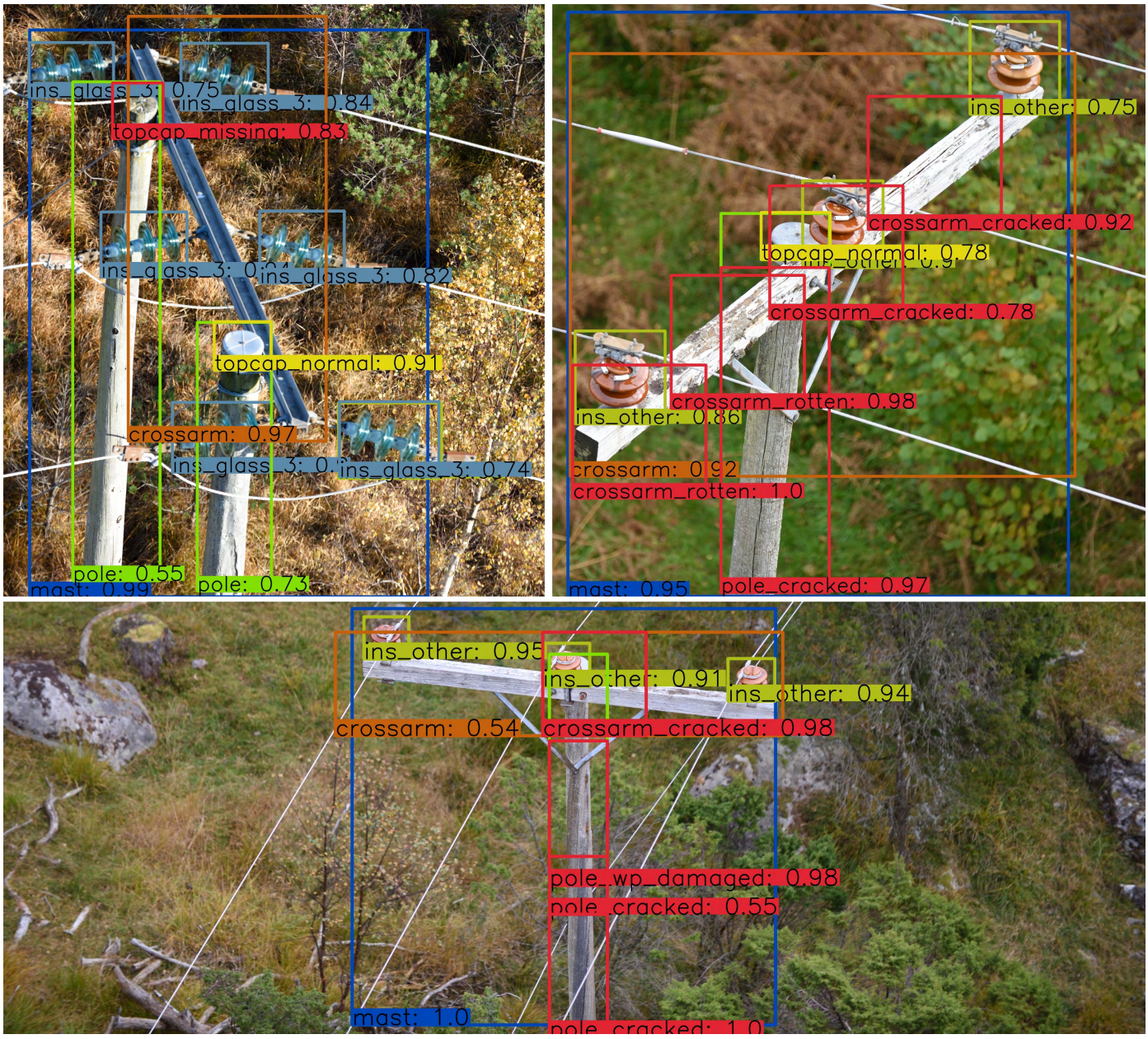


FIGURE 7. An illustration of power line component inspection using our proposed multi-stage component detection/classification pipeline. The pipeline is capable of detecting common power line components (e.g., insulators, poles, cross arms, top caps) and faults including missing top caps, cracks in poles and cross arms, woodpecker damage on poles, and rot damage on cross arms. In these three example images, our pipeline detected five faults (marked in red color): missing top cap (topcap_missing), rot damage on cross arm (crossarm_rotten), cracks in pole (pole_cracked), cracks in cross arm (crossarm_cracked), and woodpecker damage on poles (pole_wp_damaged).

The results shown in Table 3 also reveal that our proposed data augmentation techniques, especially the use of mast crops in training, can overcome the lack of training data challenge and significantly improve the performance of our proposed multi-stage component detection pipeline. In particular, our proposed pipeline without data augmentation, i.e. the MSCDP-Noaug method, performs worse than the SCDM method; the reason is that our proposed pipeline is

TABLE 4. Pole crop classifier test results on the DS3_Po dataset.

Method	wP	wR	wF_1 score
ResNet50_cvgj	88.00	88.67	88.31
ResNet50_cvgj+Dataaug	**96.93**	**97.09**	**97.01**

TABLE 5. Cross arm cop classifier results on the DS4_Cr dataset.

Method	wP	wR	wF_1 score
ResNet50_cvgj	72.02	83.54	75.96
ResNet50_cvgj+Dataaug	**74.00**	**84.10**	**77.96**

comprised of two models which typically require more training data compared to a single model in the SCDM method. However, our proposed pipeline with data augmentation, i.e. the MSCDP-Dataaug method, significantly outperforms the MSCDP-Noaug and the SCDM methods and improves *mAP* by 2.8% and 1.2% respectively. This suggests that data augmentation is crucial when addressing the lack of training data challenge for the component detection task.

The test results of the component classification models shown in Table 4 and Table 5 reveal that our proposed data augmentation techniques can address the class imbalance and the lack of training data challenge to some extent. In particular, using augmented data to balance out the imbalanced classes and increase the size of the training datasets improves wF_1 score in the pole crop classification and cross arm crop classification tasks by 8.7% and 2% respectively. In addition, models trained with augmented data in both tasks achieve higher *wP* and *wR* compared to models trained with original images only. Furthermore, as can be seen from Table 1, Table 4, and Table 5, wF_1 score of tasks with less training examples is improved more significantly when trained with augmented data. Specifically, while the cross arm crop classification task with 34029 training examples received 2% wF_1 score improvement when trained with augmented data, the pole crop classification task with only 26446 training examples, received much higher wF_1 score improvement of 8.7%. These results indicate that data augmentation is crucial when addressing the class imbalance and the lack of training data challenge for the component classification task.

The approaches proposed in this paper address the first three challenges of DL vision-based UAV inspection including the lack of training data, class imbalance, and the detection of small components and faults and facilitate the implementation of the automatic autonomous vision-base power line inspection system. The current version of our system is capable of detecting common power line components, such as poles, cross arms, top caps, insulators, and inspecting common faults on power line components including missing top caps, cracked poles, woodpecker-damaged poles, cracked cross arms, and rot-damaged cross arms (see Fig. 7).

When deployed in the Microsoft Azure cloud, with autoscale functionality and access to GPU VMs, our system has demonstrated its ability to analyze over 180,000 images per hour. This allows power utilities to inspect their power grids more often and at a lower cost than traditional inspection methods. Our system gives energy companies a fast and efficient tool to view the status of their infrastructure as well as export reports as a basis for their maintenance tasks. In addition, with the ability to access hard-to-reach areas and fly at high speed, our UAVs allow almost immediate assessment of power line damage after natural disasters for energy companies to plan for immediate repair or replacement work, which can greatly reduce the outage time and quickly reconnect the power grid.

Our system has, in our opinion, demonstrated its potential roles in the intelligent monitoring of power grids. Automatic autonomous UAVs equipped with sensors and cameras can automatically fly along power lines to perform online brief inspection to identify serious faults (e.g., collapsed poles, broken power lines, trees lying across and against power lines) and collect data for offline thorough inspection to detect potential faults that may lead to power outages such as broken insulators, missing top caps, cracked poles, woodpecker-damaged poles, cracked cross arms, and rot-damaged cross arms. These potential faults are very important information sources for electric utilities to make decisions for necessary repair or replacement work before any major damage that may cause power blackout.

VII. CONCLUSION AND FUTURE WORK

This paper presents a novel automatic autonomous vision-based power line inspection system that uses UAV inspection as the main inspection method, optical images as the primary data source, and deep learning as the backbone of the data analysis. To facilitate the implementation of the system, we address three main challenges of deep learning in vision-based power line inspection: (i) the lack of training data; (ii) class imbalance; and (iii) the detection of small power components and faults.

First, we create four medium-sized datasets for training component detection and classification models. Next, we propose a series of effective data augmentation techniques to generate more training data and balance out the imbalanced classes. Finally, we propose a multi-stage component detection and classification pipeline to detect small power components and faults.

The result indicates that the proposed approaches can address the three challenges and deliver significant improvement for detecting and classifying power line components. Compared with simple SSD detectors and ResNet50 classifiers, the proposed pipeline with data augmentation achieves 1.2% improvement in terms of *mAP* on the component detection task; using augmented data to balance out the imbalanced classes improves wF_1 score in the pole crop classification and cross arm crop classification tasks by 8.7% and 2% respectively. The proposed system can detect common faults on power line components: missing top caps, cracked poles, woodpecker-damaged poles, cracked cross arms, and rot-damaged cross arms at relatively high speed, over 18,000 images per hour.

With the aim of realizing a fully automatic autonomous vision-based power line inspection system, we propose two potential next steps: The first step is to combine the GPS way points-based, pole detection-based, and power line detection-based navigation approaches with UAV autopilots to facilitate self-driving UAVs. The second step involves upgrading the pipeline so that it can run directly on edge GPUs on UAVs to realize fully automatic autonomous online inspections.

ACKNOWLEDGMENT

The authors would like to thank eSmart Systems (Chi Hieu Huynh, Ngoc Hoang Tran, Dang Ha The Hien) and UiT Machine Learning Group (Michael Kampffmeyer) for support in the work with this paper.

REFERENCES

[1] V. N. Nguyen, R. Jenssen, and D. Roverso, "Automatic autonomous vision-based power line inspection: A review of current status and the potential role of deep learning," *Int. J. Elect. Power Energy Syst.*, vol. 99, pp. 107–120, Jul. 2018.

[2] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 21–37. [Online]. Available: https://link.springer.com/chapter/10.1007%2F978-3-319-46448-0_2

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[4] A. Savvaris, Y. Xie, K. Malandrakis, M. Lopez, and A. Tsourdos, "Development of a fuel cell hybrid-powered unmanned aerial vehicle," in *Proc. 24th Medit. Conf. Control Autom. (MED)*, Jun. 2016, pp. 1242–1247.

[5] T. Zhang *et al.*, "Current trends in the development of intelligent unmanned autonomous systems," *Frontiers Inf. Technol. Electron. Eng.*, vol. 18, no. 1, pp. 68–85, 2017.

[6] C. Deng, S. Wang, Z. Huang, Z. Tan, and J. Liu, "Unmanned aerial vehicles for power line inspection: A cooperative way in platforms and communications," *J. Commun.*, vol. 9, no. 9, pp. 687–692, 2014.

[7] L. Matikainen *et al.*, "Remote sensing methods for power line corridor surveys," *ISPRS J. Photogramn. Remote Sens.*, vol. 119, pp. 10–31, Sep. 2016.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[10] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[11] J. Gu *et al.*, "Recent advances in convolutional neural networks," *Pattern Recognit.*, vol. 77, pp. 354–377, May 2018.

[12] J. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," in *Proc. ICLR (Workshop Track)*, 2015, pp. 1–14.

[13] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556

[14] B. Xu, R. Huang, and M. Li. (2016). "Revise saturated activation functions." [Online]. Available: https://arxiv.org/abs/1602.05980

[15] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 315–323.

[16] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 379–387.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 779–788.

[18] S. C. Wong, A. Gatt, V. Stamatescu, and M. D. McDonnell, "Understanding data augmentation for classification: When to warp?" in *Proc. Int. Conf. Digit. Image Comput., Techn. Appl. (DICTA)*, 2016, pp. 1–6.

[19] S. van der Walt *et al.*, "Scikit-image: Image processing in Python," *PeerJ*, vol. 2, p. e453, Jun. 2014, doi: 10.7717/peerj.453.

[20] A. K. Boyat and B. K. Joshi, "A review paper: Noise models in digital image processing," *Signal Image Process.*, vol. 6, no. 2, p. 63, 2015.

[21] L. Shapiro and G. Stockman, *Computer Vision*, 1st ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2001, chs. 5.4, p. 154.

[22] G. Wolberg, *Digital Image Warping*, vol. 10662, Los Alamitos, CA, USA: IEEE Computer Society Press, 1990, chs. 3.3, pp. 47–49.

[23] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[24] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[25] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proc. AAAI*, vol. 4, 2017, p. 12.

[26] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 4700–4708.

[27] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 675–678.

[28] M. Simon, E. Rodner, and J. Denzler. (2016). "ImageNet pre-trained models with batch normalization." [Online]. Available: https://arxiv.org/abs/1612.01452

[29] D. P. Kingma and J. Ba. (2014). "Adam: A method for stochastic optimization." [Online]. Available: https://arxiv.org/abs/1412.6980

VAN NHAN NGUYEN received the B.Eng. degree in computer science and engineering from the Ho Chi Minh City University of Technology, Ho Chi Minh City, Vietnam, in 2014, and the M.S. degree in computer science from Østfold University College, Halden, Norway, in 2016. He is currently pursuing the Ph.D. degree in deep learning with the UiT Machine Learning Group, UiT/The Arctic University of Norway, Tromsø, Norway. His research interests include deep vision (deep learning for computer vision), especially image classification, object detection, semantic segmentation, one-shot learning, zero-shot learning, deep learning-based line detection, and vision-based automatic autonomous inspection of power lines based on unmanned aerial vehicles (UAV) and deep learning.

ROBERT JENSSEN (M'02) received the Ph.D. (Dr. Scient.) degree in electrical engineering from the University of Tromsø, in 2005. He was a Guest Researcher with the Technical University of Denmark, Kongens Lyngby, Denmark, from 2012 to 2013, also with the Technical University of Berlin, Berlin, Germany, from 2008 to 2009, and also with the University of Florida, Gainesville, FL, USA, from 2002 to 2003, spring 2004, and spring 2018. He is currently a Professor with the Department of Physics and Technology, UiT/The Arctic University of Norway, Tromsø, Norway. He directs the UiT Machine Learning Group: http://site.uit.no/ml. He is also a Research Professor with the Norwegian Computing Center, Oslo, Norway. He is on the IEEE Technical Committee on Machine Learning for Signal Processing. He is the President of the Norwegian Section of IAPR (NOBIM - nobim.no) and serves on the IAPR Governing Board. He is an Associate Editor of the Journal *Pattern Recognition*, since 2010.

DAVIDE ROVERSO received the Ph.D. degree in computing science. He has over 25 years' of experience in the field of machine learning and Big Data Analytics, with the applications in diagnostics, prognostics, condition monitoring, and early fault detection in complex processes, in sectors ranging from energy to medicine and environmental monitoring. He has authored over 100 publications in international journals, conference proceedings, and edited books. He is currently the Chief Analytics Officer at eSmart Systems, where he leads the Analytics and Data Science Group.

Appendix C

Paper IV

LS-Net: Fast Single-Shot Line-Segment Detector

Van Nhan Nguyen*†, Robert Jenssen*, and Davide Roverso†
*The UiT Machine Learning Group, UiT The Arctic University of Norway, 9019 Tromsø, Norway
†Analytics Department, eSmart Systems, 1783 Halden, Norway

***Abstract*—In low-altitude Unmanned Aerial Vehicle (UAV) flights, power lines are considered as one of the most threatening hazards and one of the most difficult obstacles to avoid. In recent years, many vision-based techniques have been proposed to detect power lines to facilitate self-driving UAVs and automatic obstacle avoidance. However, most of the proposed methods are typically based on a common three-step approach: (i) edge detection, (ii) the Hough transform, and (iii) spurious line elimination based on power line constrains. These approaches not only are slow and inaccurate but also require a huge amount of effort in post-processing to distinguish between power lines and spurious lines. In this paper, we introduce LS-Net, a fast single-shot line-segment detector, and apply it to power line detection. The LS-Net is by design fully convolutional and consists of three modules: (i) a fully convolutional feature extractor, (ii) a classifier, and (iii) a line segment regressor. Due to the unavailability of large datasets with annotations of power lines, we render synthetic images of power lines using the Physically Based Rendering (PBR) approach and propose a series of effective data augmentation techniques to generate more training data. With a customized version of the VGG-16 network as the backbone, the proposed approach outperforms existing state-of-the-art approaches. In addition, the LS-Net can detect power lines at near real-time (20.4 FPS). This suggests that our proposed approach has a promising role in automatic obstacle avoidance and as a valuable component of self-driving UAVs, especially for automatic autonomous power line inspection.**

Keywords*—Line segment detection, power line detection, power line inspection, deep learning, UAVs.

I. Introduction

Obstacle detection and avoidance are key to ensure low altitude fight safety. Due to their extremely small size, power lines are considered as one of the most threatening hazards and one of the most difficult obstacles for Unmanned Aerial Vehicles (UAVs) to avoid [1].

In automatic autonomous vision-based power line inspection, power line detection is crucial. Not only for ensuring flight safety, and for vision-based navigation of UAVs, but also for inspection to identify faults on power lines (e.g., corroded and damaged power lines) and surrounding objects, such as vegetation encroachment [2].

In recent years, many techniques have been proposed to detect power lines automatically. However, most of the proposed methods are typically based on a common three-step approach: First, an edge detector such as Canny [3] is applied to produce edge maps. Then, the Hough transform [4], the Radon transform, or a line tracing algorithm, are utilized to detect straight lines from the edge maps. Finally, power line constraints, such as parallel lines, are applied to eliminate spurious lines and detect the power lines. These approaches not only are slow and inaccurate but also require a considerable amount of effort in post-processing to distinguish between power lines and spurious lines.

With the aim of facilitating real-time and accurate power line detection for UAV vision-based navigation and inspection, we propose in this paper LS-Net, a fast single-shot line-segment detector, and apply it to power line detection.

The work presented in this paper is part of an ongoing effort involving the exploitation of recent advances in Deep Learning (DL) and UAV technologies for facilitating automatic autonomous vision-based inspection of power lines. In our previous work [2], we first proposed a novel automatic autonomous vision-based power line inspection concept that uses UAV inspection as the main inspection method, optical images as the primary data source, and deep learning as the backbone of data analysis. We then identified six main challenges of DL vision-based UAV inspection: the lack of training data; class imbalance; the detection of small power components and faults; the detection of power lines in cluttered backgrounds; the detection of previously unseen components and faults; and the lack of metrics for evaluating inspection performance.

To move forward, we proposed approaches to address the first three challenges and built a basic automatic vision-based inspection system with two custom-built UAVs and five DL-based models for data analysis and inspection [5].

In this paper, we take this further by addressing the fourth challenge of DL vision-based UAV inspection, which is to detect power lines in cluttered backgrounds, with our proposed LS-Net. The LS-Net is a feed-forward, fully Convolutional Neural Network (CNN) [6], and consists of three modules: (i) a fully convolutional feature extractor, (ii) a classifier, and (iii) a line segment regressor. Due to the unavailability of large datasets with annotations of power lines, we render synthetic images of power lines using the Physically Based Rendering (PBR) approach [7] and propose a series of effective data augmentation techniques to generate more training data. With a customized version of the VGG-16 network [8] as the backbone, the proposed LS-Net outperforms existing state-of-the-art DL-based power line detection approaches and shows the potential to facilitate real-time power line detection for obstacle avoidance in low-altitude UAV flights.

This work is in our opinion paving the way for fully automatic autonomous vision-based power line inspection, in which high-speed UAVs equipped with sensors, cameras, a DL vision-based UAV navigator, and a DL-based model for data analysis, can automatically navigate along power lines to collect data for offline inspections and perform online inspections to identify potential faults quickly.

The remainder of the paper is structured as follows: Section II presents background knowledge and relevant related work in CNN-based image classification models, CNN-based object detection models, and common approaches to power line detection. Then we describe our proposed LS-Net, a fast single-shot line segment detector in Section III. Next, in section IV, we present in detail our experimental results and ablation studies. Further, in Section V, we discuss the potential of our proposed LS-Net in UAV navigation and inspection as well as in detecting other linear structures, such as railway tracks, unburied onshore pipelines, and roads. Finally, in Section VI, we conclude the paper with a summary and an outlook for the future of the field.

II. Background and Related Work

A. Convolutional Neural Networks

In the past few years, Convolutional Neural Networks (CNNs) [9], which are special neural networks designed to take advantage of the 2D structure of image data, have been advancing the state-of-the-art for both high-level tasks, such as image classification, object detection, image segmentation, and low-level vision tasks, for instance edge detection. In this section, we summarize some of the most well-known CNN architectures for those tasks and describe a selection of methods and techniques that will be used in the LS-Net.

1) High-level vision tasks: Since the success of Krizhevsky et al. [9] with an 8-layer CNN (5 convolutional layers + 3 fully-connected layers) in the 2012 ImageNet challenge, CNNs have become a commodity in the computer vision field. In the last few years, many attempts have been made to improve the original architecture of Krizhevsky et al. by, for example, utilizing smaller receptive window size and by increasing the depth of the network.

One of the most recognized such attempts is the VGGNet, which is a CNN architecture that secured the first and the second places in the localization and classification tasks, respectively, in the 2014 ImageNet challenge [8]. The key innovation of the VGGNet is the combination of small filters (3×3 filters) and deep networks (16-19 layers). The authors argued that a stack of three 3×3 convolutional layers has the same effective receptive field as one 7×7 convolutional layer, but is deeper, has more non-linearities and fewer parameters.

With the increasing complexity of image classification problems, deeper CNNs are typically required. However, deep CNNs constructed simply by stacking up many layers are very difficult to train due to the notorious problem of vanishing/exploding gradients. To ease the training of deep CNNs, Residual Networks (ResNets) were proposed [10]. ResNets add "shortcut" connections to the standard CNN layers to allow the gradient signal to travel back directly from later layers to early layers. The "shortcut" connections allowed the authors of ResNets to successfully train very deep CNNs with 50, 101, and even 152 layers.

Inspired by the success of CNNs in image classification, Faster R-CNN (Region-based Convolutional Neural Network) was proposed to solve a more challenging task of object detection [11]. Faster R-CNN is a single, unified network that performs object detection via two main steps: region proposal and region classification. First, a base network (e.g., ResNet [10]) is utilized to extract features from images. Next, the extracted features are fed into a Region Proposal Network (RPN) to find proposals. Then, a CNN-based classifier is applied on top of the extracted feature maps to classify the proposals and refine their bounding boxes. Finally, post-processing is used to refine the bounding boxes further and eliminate duplicate detections. Faster R-CNN is very accurate; however, it is quite slow.

R-FCN (Region-based Fully Convolutional Network) [12] is an accurate and efficient object detection framework proposed to address existing issues of region-based detectors such as Fast R-CNN [13] and Faster R-CNN [11]. Instead of applying a costly per-region sub-network hundreds of times, R-FCN adopts a fully convolutional architecture with almost all computations shared across the entire image. To address the dilemma between translation-invariance in image classification and translation-variance in object detection, R-FCN proposes novel position-sensitive score maps which allow fully convolutional networks to effectively and efficiently perform both classification and detection in a single evaluation. With those novel improvements, R-FCN can run at 2.5-20 times faster and achieve higher accuracy than the Faster R-CNN counterpart. RPN based approaches are accurate; however, they are typically slow due to their complex multi-stage pipelines [14]. With the aim of facilitating real-time object detection, many single-shot object detectors, which take only one single-shot to detect multiple objects in the image, have been proposed. The two most well-known single-shot object detectors are YOLO [14] and SSD [15].

YOLO (You Only Look Once) is a real-time object detection framework that directly predicts bounding boxes and class probabilities with a single network in a single evaluation [14]. To achieve this, YOLO unifies region proposal and region classification into a single neural network and, according to the authors, "frames object detection as a regression problem to spatially separated bounding boxes and associated class probabilities". YOLO divides the input image into a $S \times S$ grid. Each grid cell predicts B bounding boxes, confidence scores for those boxes, and C conditional class probabilities. With a unified architecture, YOLO is extremely fast; it processes images in real-time. However, YOLO is not state-of-the-art in terms of accuracy.

SSD (Single-Shot MultiBox Detector) improves YOLO by adding a series of modifications: (i) a small convolutional filter is utilized to predict object classes and offsets in bounding box locations; separate predictors (filters) are employed for predicting objects at different aspect ratios; predictions are performed at multiple feature maps from the later stages of a network to enable detection at multiple scales [15]. These modifications make SSD both faster and more accurate than the YOLO counterpart.

Lin et at. identified class imbalance during training as the main obstacle preventing one-stage detectors (e.g., SSD and YOLO) from achieving the state-of-the-art accuracy and proposed to address that by introducing a novel loss function named Focal Loss (FL) [16]. FL dynamically scales the

standard cross-entropy loss with a scaling factor that decays to zero as confidence in the correct class increases. By doing that, FC automatically reduces the weights of easy examples during training and allows the model to focus on hard examples.

Feng et al. observed that in multiple points localization problems, such as facial landmark localization, more attention should be paid to the samples with small or medium range errors [17]. To achieve this target, the authors proposed a new loss function, namely, Wing loss. With the aim of restoring the balance between the influence of errors of different sizes, Wing loss was designed to behave as a log function with an offset for small errors and as L1 for large errors. According to the authors, Wing loss is appropriate for dealing with relatively small localization errors.

Ioffe et al. observed that the change in the distributions of layers' inputs during the training of deep neural networks poses a serious problem because the layers need to adapt to the new distribution continuously [18]; this phenomenon was referred to as *internal covariate shift*. To address this problem, the authors proposed a new mechanism, called *Batch Normalization (BN)*, which fixes the means and variances of layer inputs by normalizing each activation independently along the batch dimension.

Although the normalization along the batch dimension allows BN to reduce internal covariate shift and accelerate the training of deep neural nets, it causes many distinct drawbacks. For example, for BN to work properly, it is required to have a sufficiently large batch size (e.g., 32 per worker), which is typically not possible with very deep CNNs and high-resolution images due to GPU memory limitations [19]. With the aim of eliminating the dependence on batch sizes and avoiding batch statistics computation, Wu et al. proposed *Group Normalization (GN)* as a simple alternative to BN [19]. The key innovation of GN is that it divides channels into groups and normalizes the features within each group.

2) Low-level vision tasks: CNNs have been successfully applied to low-level vision tasks such as edge detection. For example, Xie et al. proposed a method, named holistically-nested edge detection (HED), for predicting edges in an image-to-image fashion [20]. The method leverages fully convolutional neural networks and deeply-supervised nets by attaching a side output layer to the last convolutional layer in each stage of the VGGNet and utilizing both weighted-fusion supervision and deep supervision in training. Liu et al. improved the HED method by utilizing richer features from all convolutional layers in the VGGNet [21]. In addition, a novel loss function was proposed to treat training examples properly, and a multi-scale hierarchy was employed to enhance edges.

CNNs have also been successfully applied to semantic line detection. For example, Lee et at. proposed a semantic line detector (SLNet) based on the VGG16 network [22]. First, multi-scale feature maps are extracted from an input image using convolution and max-pooling layers. Then, a line pooling layer is developed to extract a feature vector for each line candidate. Finally, the feature vectors are fed into parallel classification and regression layers to decide whether the lines are semantic or not and to refine their location.

B. Common approaches to power line detection

1) Line-based methods: A straight-forward approach to power line detection is to treat the power line as a straight line and apply line detection algorithms directly. For example, Li et al. utilized the Hough transform to detect straight lines from Pulse Coupled Neural Network filtered images and employed K-means clustering to discriminate power lines from other mistakable linear objects [23, 24].

Although this approach is effective and easy to implement, its strong assumptions on the characteristics of power lines, including, (i) a power line has uniform brightness, (ii) a power line approximates a straight line, and (iii) power lines are approximately parallel to each other, make it a less practical approach. Due to the strong assumptions, line-based methods often mistakenly detect linear objects, such as metallic fence lines [23], as power lines and misdetected power lines that appear as arc curves due to the influence of gravity [1].

2) Piece-wise line segment-based methods: With the aim of detecting both straight power lines and curvy ones, some researchers have proposed to segment a power line into piece-wise line segments so that they can be approximated by straight lines [25, 26]. For example, Yan et al. utilized the Radon transform to extract line segments of a power line, then employed a grouping method and the Kalman filter to link each line segment and connect the linked line segments into a complete line [25]. Song et at. applied matched filter and first-order derivative of Gaussian to detect line segments, then used a graph cut model based on graph theory to group the detected line segments into whole power lines [26].

Similar to line-based methods, piece-wise line segment-based methods also often mistakenly detect linear objects with similar line features in the background, such as metallic fence lines and building edges, as power lines [26].

3) Auxiliaries assisted methods: To address the existing problems of line-based and piece-wise line segment-based methods, much effort has been made towards utilizing correlation information and context features provided by auxiliaries. For example, Zhang et al. proposed to use the spatial correlation between the pylon and the power line to improve transmission line detection performance [27]. The proposed method outperforms line-based and piece-wise line segment-based methods; however, the performance drops significantly when the pylon is absent or occluded.

To eliminate the need for manually selecting auxiliaries and defining spatial relationships between auxiliaries and power lines, Shan et at. proposed an optimization-based approach for automatic auxiliaries selection and contexts acquisition [1]. The proposed approach surpasses traditional methods that use manually assigned auxiliaries both in terms of detection accuracy and false alarm probability; however, it is quite slow due to the sliding window-based object extraction and the context representation between the auxiliaries and the hypotheses.

To further improve auxiliaries assisted power line detection accuracy and speed, Pan et al. proposed a metric for measuring the usefulness of an auxiliary in assisting power line detection, named spacial context disparity, based on two factors: spatial context peakedness and spatial context difference and applied

it for automatic selection of optimal auxiliaries [28]. According to the authors, the proposed method is robust and can achieve satisfactory performance for power line detection.

4) DL-based methods: One of the earliest attempts to use deep learning for power line detection was the work of Jayavardhana et al. [29]. The authors proposed a CNN-based classifier that uses Histogram of Gradient (HoG) features as the input and applied it in a sliding window fashion to classify patches of size 32×32 into two classes: "Line present" and "No line present". The authors also finetuned the GoogleNet on patches of original images for the same task. According to the authors, the proposed CNN-based classifier achieves an F-score of 84.6% and outperforms the GoogleNet, which achieves an F-score of 81%.

Ratnesh et al. [30] treated wire detection as a semantic segmentation task and performed a grid search over a finite space of CNN architectures to find an optimal model for the task based on dilated convolutional networks [31]. The model was trained on synthetic images of wires generated by a ray-tracing engine and finetuned on real images of wires from the USF dataset [32]. According to the authors, the proposed model outperforms previous work that uses traditional computer vision and various CNN-based baselines such as FCNs, SegNet, and E-Net; the model achieves an Average Precision (AP) score of 0.73 on the USF dataset and runs at more than 3Hz on the NVIDIA Jetson TX2 with input resolution of 480×640.

Although treating wire detection as a semantic segmentation task has been proved to be a powerful approach for detecting wires [30], its requirement of pixel-level annotated ground-truth data makes it less practical than traditional computer vision approaches. Sang et al. proposed to use weekly supervised learning with CNNs for localizing power lines in pixel-level precision by only using image-level class information [33]. First, a classifier adapted from the VGG19 is applied to classify sub-regions (128×128) from an input image (512×512) by using a sliding window approach. Then, feature maps of intermediate convolutional layers of sub-regions that are classified as "sub-region with power lines" are combined to visualize the location of the power lines. Although the localization accuracy of the proposed approach is still far from an applicable level of industrial fields, it can be applied, according to the authors, to generate ground-truth data in pixel-level roughly.

Yan et al. proposed a power detection pipeline based on pyramidal patch classification in [34]. First, input images are hierarchically partitioned into patches. Next, a CNN classifier is trained to classify the patches into two classes: patches with power lines and patches without power lines. Then, the classified patches are used as inputs for edge feature extraction using steerable filters and line segment detection using the Progressive Probabilistic Hough Transform (PPHT). Finally, the detected line segments are connected using a power line segments correlation module to form complete power lines. The authors concluded that the proposed approach significantly improves the detection rate of the power line detection and largely decreases the false alarm rate.

III. The Line Segment Detector (LS-Net)

A. Data Generation

1) Synthetic Data Generation: Due to the unavailability of large datasets with annotations of power lines, we collaborate with Nordic Media Lab (NMLab)[1] to render synthetic images of power lines using the Physically Based Rendering (PBR) approach [7]. First, we model Aluminium Conductor Steel-Reinforced (ACSR) cables, which are typically composed of one steel center strand and concentric layers of high-purity aluminum outer strands, using the Autodesk 3DS Max program. To increase the realistic appearance of the cables, we utilize the bevel and the twist modifiers together with the metal brushed steel texture. Then, we randomly superimpose the cables on 71 8K High Dynamic Range Images (HDRIs) collected from the internet[2]. Next, to further increase the realistic appearance of the cables, we employ cube mapping to capture the reflection and the lighting data from the HDRIs and apply them to the cables. Finally, we apply a series of effective variations, with respect to the camera angle, the camera distance, out-of-focus blur, cable colors, the number of cables, and the distance between cables, to render more synthetic images.

2) Data Augmentation: Inspired by the success of data augmentation for improving the performance of CNNs in [35], [36], and [37], we propose a series of effective data augmentation techniques to generate more training data by applying transformations in the data-space. These are all implemented using the scikit-image [38] and the OpenCV libraries [39].

The first technique replaces the background of the generated synthetic images with real background images to increase the diversity of the dataset and to account for various types of background variations during the inspection (e.g., different seasons, weather conditions, and lighting conditions).

The second technique adds Gaussian-distributed additive noise to account for noisy image acquisition (e.g., sensor noise caused by poor illumination and/or high temperature, and/or transmission) [40]. The augmented image $f(i,j)$ is the sum of the true image $s(i,j)$ and the noise $n(i,j)$:

$$f(i,j) = s(i,j) + n(i,j). \tag{1}$$

The noise term, $n(i,j)$, follows a Gaussian random distribution:

$$p_G(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}}, \tag{2}$$

where z represents the gray level, μ is the mean value, and σ is the standard deviation.

To account for possible out-of-focus, Gaussian blur is employed by convolving the image with a two-dimensional Gaussian function:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}, \tag{3}$$

where x and y are distances from the origin in the horizontal axis and the vertical axis respectively, and σ is the standard deviation [41].

[1] http://nmlab.no/

[2] https://hdrihaven.com/

Fig. 1: Sample augmented images (from left to right): original image; pixel-level annotation, image with Gaussian-distributed additive noise; Gaussian blurred image; color manipulated image; elastic transformed image, image with new background, cropped and flipped image with new background.

To introduce invariance to changes in lighting and to capture minor color variations, especially in power lines, a series of color manipulations including random brightness, random saturation, random contrast, and random hue are utilized. In addition, to further extend color invariance, we randomly remove colors from RGB images by first converting them to grayscale and then converting the grayscale images back to RGB.

With the aim of training models that can detect not only perfectly straight line segments but also curvy ones, elastic deformations [37] are employed. First, two random displacement fields for the x-axis (Δx) and y-axis (Δy) are generated as follows:

$$\Delta x(x, y) = rand(-1, +1), \tag{4}$$
$$\Delta y(x, y) = rand(-1, +1), \tag{5}$$

where $rand(-1, 1)$ is a random number between -1 and $+1$, generated with a uniform distribution. Next, the fields Δx and Δy are convolved with a two-dimensional Gaussian function similar as shown in Eq. (3) to form elastic deformation fields. Then, the elastic deformation fields are scaled by factor α that controls the intensity of the deformation. Finally, the fields Δx and Δy are applied to images.

To account for various camera distances and viewing angles, zoom and rotation operators are employed [42]. The zoom operator is applied by randomly cropping images and scaling them to their original size. The rotation operator is employed by multiplying images with a rotation matrix R:

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix},$$

where θ is the rotation angle. The final technique flips the images horizontally and vertically.

B. LS-Net Architecture

Inspired by the success of single-shot object detectors such as SSD [15] and YOLO [14] in terms of speed and accuracy, we propose a single-shot line segment detector, named LS-Net. The LS-Net is based on a feed-forward, fully convolutional neural network and consists of three modules: (i) a fully convolutional feature extractor, (ii) a classifier, and (iii) a line segment regressor connected as shown in Fig. 2.

The design of the LS-Net architecture is mainly inspired by state-of-the-art single-shot object detectors such as SSD [15] and YOLO [14]. Specifically, the LS-Net divides the input image of size $W \times H \times C$ into a grid, and each grid cell of size $C \times C$ predicts coordinates and a confidence score for the longest line segment in the cell. The confidence score indicates the probability of the cell containing a line segment, and the coordinates are the normalized distances of the two endpoints of the line segment to the local x-axis and y-axis of the cell.

The traditional one grid approach has been proven to work well for single-shot object detectors such as SSD [15] and YOLO [14]; however, it faces two problems when applied

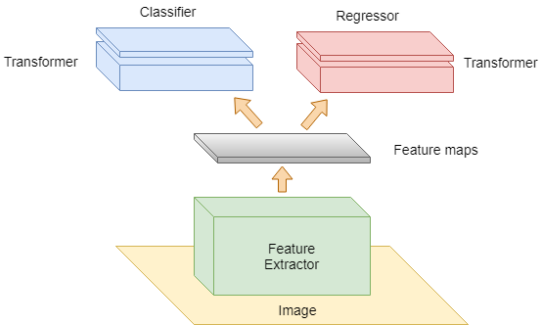


Fig. 2: The LS-Net is a feed-forward, fully convolutional neural network and consists of three modules: (i) a fully convolutional feature extractor, (ii) a classifier, and (iii) a line segment regressor.

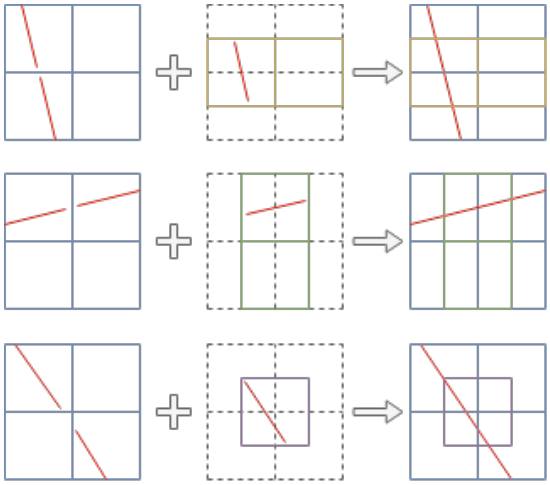


Fig. 3: An illustration of the four-grid approach. LS-Net with the traditional one grid approach (the first column) ignores short line segments at cell corners and create gaps at cell borders in the detected lines. LS-Net with the four-grid approach (the third column) utilizes three additional grids (the second column) to detect line segments ignored by the main grid and close gaps at horizontal and vertical borders, which significantly eliminate the discontinuities in the detected lines.

to line segment detection: (i) discontinuities and gaps at cell borders, and (ii) discontinuities and gaps at cell corners. In the one grid approach, due to regression errors, the detected line segments can be shorter than the ground truths. This can result in discontinuities and gaps in the detected lines at borders of adjacent cells that make regression errors. In addition, the one grid approach ignores short line segments, especially at cell corners, due to the lack of features. This can also lead to discontinuities and gaps in the detected lines (see Fig. 3).

To address the two above-mentioned problems, we propose to replace the one grid approach by a four-grid approach. Specifically, the four-grid LS-Net divides the input image into four overlapping grids: a $S_m \times S_m$ grid (main grid), a $S_m \times S_a$ grid (horizontal grid), a $S_a \times S_m$ grid (vertical grid), and a $S_a \times S_a$ grid (center grid), where $S_a = S_m - 1$ (see Fig. 4). The main grid, which works exactly the same as the grid used by SSD and YOLO for detecting objects, is employed for detecting line segments in grid cells. The horizontal and vertical grids are utilized for closing the gaps at horizontal and vertical borders, respectively. The central grid is used for detecting short line segments at cell corners that were ignored by the main grid. All the detected line segments from the four grids are combined together to form a line segment map. Since the four-grid LS-Net utilizes three additional grids to detect short line segments ignored by the main grid and close gaps at horizontal and vertical borders, the discontinuities in the detected lines are significantly eliminated (see Fig. 4).

1) Fully Convolutional Feature Extractor: The LS-Net feature extractor is inspired by the VGG-16 network [8]. We truncate the network before the last max-pooling layer and substitute the remaining max-pooling layers by strided convolutional layers with stride 2. Max pooling layers have been used extensively in CNNs for image classification; however, they are not an optimal choice for the proposed LS-Net since they throw away spatial information that is useful for predicting line segment end-points.

With the aim of easing the optimization, enabling the network to converge faster, and eliminating the dependence on batch sizes, we adopt Group Normalization [19] before activations in every convolutional layer.

2) Classifier: The classifier sub-network takes feature maps extracted by the fully convolutional feature extractor as input and predicts whether each grid cell contains a line segment or not. The sub-network consists of two layers: The first is a 2×2 convolutional layer with stride 1 that works as a *transformer (transformation layer)* and transforms the input feature maps into four sets of feature maps corresponding to the four overlapping grids. The second layer is a 1×1 convolutional layer that predicts a confidence score for each grid cell.

3) Line Segment Regressor: The line segment regressor sub-network takes feature maps extracted by the fully convolutional feature extractor as input and predict coordinates of the longest line segment in each grid cell. The sub-network also consists of two layers: The first layer is similar to the first layer of the classifier sub-network. The second is a 1×1 convolutional layer that is responsible for predicting line segment coordinates.

4) Summary: With the four-grid approach, the output of the LS-Net is very similar to that of a traditional sliding-window detector of size $C \times C$ with stride $C/2$; however, the LS-Net has to major advantages over the sliding-window approach: The first is that instead of applying a costly forward pass hundreds of times, one for each cell, the LS-Net makes predictions for all cells in a single forward pass, which was made possible thanks to the single-shot detector architecture and the combination of our proposed four-grid approach and our proposed transformation layers. The second advantage is that the LS-Net, with a large effective receptive field, can take into account contextual information when making predictions.

In other words, the LS-Net looks at not only the target cell but also its neighboring cells to make predictions for the cell.

To evaluate the effectiveness of the proposed LS-Net architecture, we train the LS-Net on input images of size $512\times512\times3$ to detect line segments in cells of size 32×32, i.e., $S_m = 16$; however, the proposed LS-Net architecture can be easily generalized to handle images of any sizes and to detect line segments in cells of any sizes. A detailed configuration of the LS-Net used in our experiments in this paper is shown in Table I. All convolutional layers in the feature extractor are padded so that they produce an output of the same size as the input. Padding is not applied in convolutional layers in the classifier and the regressor.

TABLE I: LS-Net's Configuration. The convolutional layer parameters are denoted as "conv(receptive field size)-(number of channels)[-S(stride)]". The default stride is 1.

Input image ($512 \times 512 \times 3$)	
Conv3-64	
Conv3-64	
Conv3-64-S2	
Conv3-128	
Conv3-128	
Conv3-128-S2	
Conv3-256	
Conv3-256	
Conv3-256-S2	
Conv3-512	
Conv3-512	
Conv3-512-S2	
Conv2-512	Conv2-512
Conv1-2	Conv1-4

C. LS-Net Multi-task Loss

The LS-Net has two sibling output layers. The first sibling layer outputs a discrete probability distribution, $p_t^i = (p^i, 1 - p^i)$, for each grid cell, indexed by i, over two classes: *cell with line segments* and *cell without line segments*. The probability distribution p_t^i is computed by a softmax over the two outputs of a 1×1 convolution layer at the i^{th} cell. The second sibling layer outputs coordinates of the two end-points of the longest line segment, $e^i = (e_{x1}^i, e_{y1}^i, e_{x2}^i, e_{y2}^i)$, for each grid cell, indexed by i.

Each training cell is labeled with a ground-truth class label $y^i \in \{\pm1\}$ and a ground-truth end-point regression target $t^i = (t_{x1}^i, t_{y1}^i, t_{x2}^i, t_{y2}^i)$. We use a weighted multi-task loss function, L, to jointly train for cell classification and line segment end-points regression:

$$L(p_t, y, e, t) = L_{cls}(p_t, y) + \lambda[y = 1]L_{reg}(e, t), \qquad (6)$$

where the Iverson bracket indicator function $[y = 1]$ evaluates to 1 when $y = 1$ and 0 otherwise.

The first task loss, L_{cls}, is a Focal loss [16] defined as follows:

$$L_{cls}(p_t, y) = -\alpha_t(1 - p_t)^\gamma \log(p_t), \qquad (7)$$

where $\gamma \geq 0$ is a tunable focusing parameter, $\alpha_t \in [0, 1]$ is a weighting factor defined as follows:

$$\alpha_t = \begin{cases} \alpha & \text{if } y = 1 \\ 1 - \alpha & \text{otherwise} \end{cases}, \qquad (8)$$

and $p_t \in [0, 1]$ is the model's estimated probability defined as follows:

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise} \end{cases}. \qquad (9)$$

Since the number of cells without line segments is much larger than the number of cells with line segments, the Focal loss is employed instead of a standard Cross-Entropy loss [43] to address the class imbalance during training.

The second task loss, L_{reg}, is a Wing loss [17] defined as follows:

$$L_{reg}(e, t) = \begin{cases} w \ln(1 + d/\epsilon) & \text{if } d < w \\ d - C & \text{otherwise} \end{cases}, \qquad (10)$$

where w is a non-negative upper bound that sets the range of the nonlinear part to $(-w, w)$, ϵ is a constant that limits the curvature of the nonlinear region, $C = w - w \ln(1 + w/\epsilon)$ is a constant that smoothly links the piecewise-defined linear and nonlinear parts, and d is our proposed error function, which computes the minimum absolute difference between the predicted end-points $e = (e_{x1}, e_{y1}, e_{x2}, e_{y2})$ and the target end-points $t = (t_{x1}, t_{y1}, t_{x2}, t_{y2})$ defined as follows:

$$d(e, t) = \min(\sum(|t - e|), \sum(|t - swap(e)|)), \qquad (11)$$

where $swap(e) = (e_{x2}, e_{y2}, e_{x1}, e_{y1})$ is a function that swaps the order of the two end-points.

The error function d is employed to allow the LS-Net to predicts the two end-points of a line segment regardless of the order, and the Wing loss is utilized instead of standard L_2 [44] or smooth L_1 [13] losses to restore the balance between the influence of errors of different sizes and to allow the model to regress the line segment end-points more accurately.

D. Training and Testing

The LS-Net can be trained end-to-end by backpropagation and Stochastic Gradient Descent (SGD) [45]. We implement the LS-Net using the Tensorflow framework [46]. We train the LS-Net from scratch using the Adam optimizer [47] with initial learning rate 0.0001, 0.9 momentum1, 0.999 momentum2, and batch size 8 (due to memory limitation) on a GeForce GTX 1080 Ti GPU. We use early stopping to prevent the network from overfitting. Our network converges after 3.5 epochs, which takes around 48 hours of training time.

Before training, we augment our dataset by generating five random crops and their flipped versions from each image; we further augment the dataset by replacing the background from each image with five randomly selected backgrounds from our background image dataset.

During training, we apply data augmentation on-the-fly by adding Gaussian-distributed additive noise, by applying Gaussian blur, by performing a series of color manipulations,

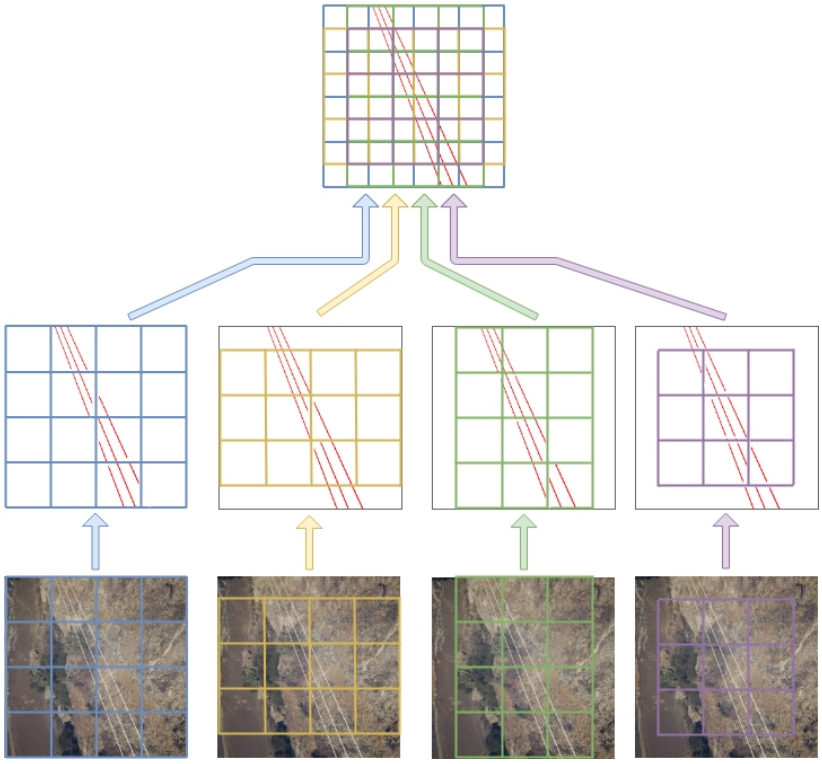


Fig. 4: An illustration of the four overlapping grids approach. LS-Net with one grid (the leftmost branch) ignores short line segments at cell corners and create gaps at cell borders in the detected lines. LS-Net with four overlapping grids approach utilizes three additional grids to detect line segments ignored by the first grid and close gaps in horizontal and vertical lines, which significantly eliminate the discontinuities in the detected lines.

and by employing elastic deformations. All the on-the-fly data augmentation techniques are applied with a probability of α. We use $\alpha = 0.25$ in our experiments.

For $512 \times 512 \times 3$ input, the LS-Net runs at 20.4 Frames Per Second (FPS) on a GeForce GTX 1080 Ti GPU at test time. However, the speed can be further increased by employing a shallower, thinner feature extractor and by decreasing the input size.

IV. Experiments

A. Comparisons with the State-of-the-Art Results

As presented in Section II-B, there are very few relevant DL-based approaches for power line detection. In addition, two approaches among the four reviewed ones apply deep learning for patch classification only, while the line detection step is still addressed by a traditional line detection or line segment detection algorithm such as the Progressive Probabilistic Hough Transform (PPHT) [34] or the Line Segment Detector (LSD) [29]. This typically results in low analysis speed and a need for post-processing to distinguish between power lines and spurious lines. Since our goal is to facilitate real-time power line detection and avoidance in low-altitude UAV flights with deep learning, in this section, we compare our proposed LS-Net only to state-of-the-art DL-based approaches for power line detection that offer high analysis speed and require minimal effort in post-processing.

First, we compare our proposed LS-Net with the weakly supervised learning with CNNs (WSL-CNN) approach proposed in [33] on the publicly available Ground Truth of Power line dataset (Infrared-IR and Visible Light-VL) [48], which is one of the most widely used power line datasets. The LS-Net and the WSL-CNN approaches share a similar objective that is to localize power lines by using cheaper ground-truth data (GTD) than pixel-level GTD (e.g., image-level class information, line end-points information). For a fair comparison, we convert line segment maps generated by the LS-Net to pixel-level segmentation maps using a similar procedure as applied in [33]. First, the pixel-level segmentation maps, S, are generated as follows:

$$conf(x,y) = max(\{conf(LS_i) \mid (x,y) \in LS_i\}), \quad (12)$$

$$S(x,y) = \begin{cases} 0 & \text{if } (x,y) \notin LS_i \ \forall i \in [1,L] \\ conf(x,y) & \text{otherwise} \end{cases}, \quad (13)$$

where L is the number of detected line segments, LS_i is the list of all pixels belonging to the i^{th} line segment, and $conf(LS_i)$ is a function that returns the confidence score of

the i^{th} line segment. Since each line segment predicted by the LS-Net is represented by a pair of two end-points, we apply the 8-connected Bresenham algorithm [49] to form a close approximation to a straight line between the two end-points. We vary the width of the straight line, W_l, from 1 to 5 and select $W_l = 2$ and $W_l = 3$ since they result in the highest F_1 scores (also known as F-scores or F-measures). We call these models LS-Net-W2 and LS-Net-W3, respectively. Then, the generated segmentation maps are smoothed by convolving with a two-dimensional Gaussian function, as shown in Ep. (3). Finally, the predicted segmentation maps are binarized by using the Otsu's method [50, 51]. The test results are shown in Table II.

Then, we implement the Dilated Convolution Networks for Wire Detection (WD-DCNN) proposed in [30] in Tensorflow. In addition, we improve the WD-DCNN approach by adopting Group Normalization [19] to accelerate the training of the networks and Focal loss [16] for restoring the balance between the influence of errors of different sizes in multiple points regression. We create three improved models. In the first model, we add a group normalization layer after each convolutional layer in the WD-DCNN model (WD-DCNN-GN). We replace the class-balanced Cross-Entropy loss function [31], adopted by the WD-DCNN model, by the Focal loss to train the second model (WD-DCNN-FL). Finally, we combine both Group Normalization and Focal loss to train the third model (WD-DCNN-GNFL). We train the WD-DCNN model and its improved versions on the same training dataset that we use to train our proposed LS-Net. The predicted segmentation maps of the four models are binarized by using the Otsu's method [50, 51]. We compare against our proposed LS-Net-W2 and LS-Net-W3 models, the WD-DCNN model, and its improved versions (WD-DCNN-GN, WD-DCNN-FL, and WD-DCNN-GNFL) on the Ground Truth of Power line dataset (Infrared-IR and Visible Light-VL) [48]. The test results are shown in Table II.

TABLE II: Comparisons of the proposed LS-Net-W2 and LS-Net-W3 models and the state-of-the-art DL-based approaches for power line detection including the WSL-CNN, the WD-DCNN, and its improved versions (WD-DCNN-GN, WD-DCNN-FL, WD-DCNN-GNFL) on the Ground Truth of Power line dataset (Infrared-IR and Visible Light-VL). *Results reported by [33].

	ARR	APR	F_1 Score
WSL-CNN [33]*	0.6256	-	-
WD-DCNN [30]	0.7192	0.4713	0.4835
WD-DCNN-GN	0.8292	0.4148	0.4882
WD-DCNN-FL	0.7514	0.4680	0.5079
WD-DCNN-GNFL	0.7930	0.4690	0.5218
LS-Net-W2 ($W_l = 2$)	0.7972	**0.4874**	**0.5344**
LS-Net-W3 ($W_l = 3$)	**0.8525**	0.4483	**0.5256**

As can be seen from Table II, both our proposed LS-Net-W2 and LS-Net-W3 models achieve state-of-the-art performance in terms of F_1 score. In addition, the LS-Net-W2 model surpasses all the existing state-of-the-art methods in terms of APR, while the LS-Net-W3 model attains state-of-the-art ARR by considerable margins. Visual comparisons of the LS-Net-W2 model and the state-of-the-art DL-based approaches for power line detection including the WD-DCNN, the WD-DCNN-GN, and its improved versions (WD-DCNN-GN, WD-DCNN-FL, and WD-DCNN-GNFL) are shown in Fig. 5.

B. Ablation Study

To investigate the effectiveness of the proposed LS-Net architecture and the loss function, we conducted several ablation studies using the publicly available Ground Truth of Power line dataset (Infrared-IR and Visible Light-VL) [48]. We use the approach presented in Section IV-A to convert line segment maps generated by the LS-Net to pixel-level segmentation maps and compare different variants of the LS-Net in terms of APR, ARR, and F_1 Score. To increase the interpretability of the comparison results, we apply a simple thresholding method ($t = 0.5$) to binarize segmentation maps instead of the Otsu method and set the width of the line segment W_l to 1 when applying the 8-connected Bresenham algorithm. This could result in lower APR, ARR, and F_1 score; however, it is not an issue since improving the performance of the LS-Net is not the primary goal of the ablation studies.

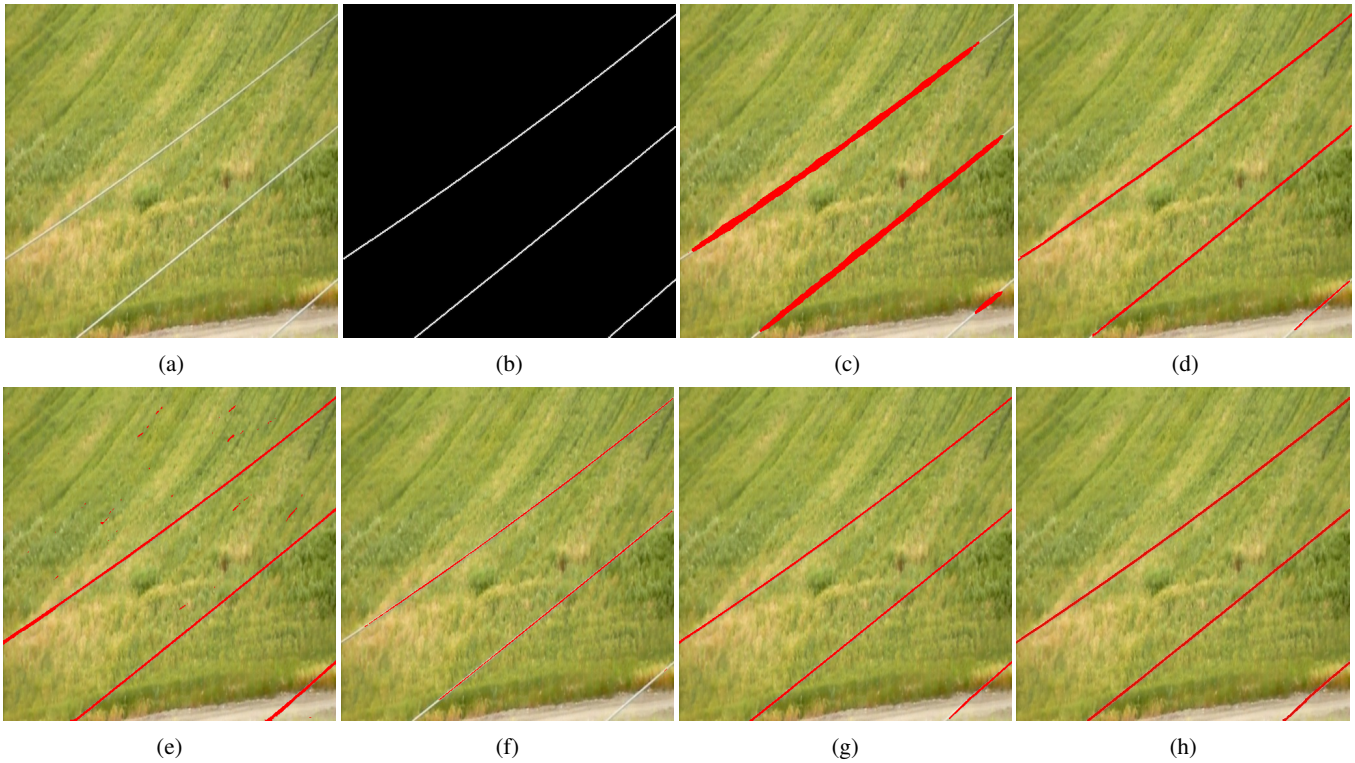
First, we evaluate the effects of replacing max-pooling layers by strided convolution layers. To do this, we compare the proposed LS-Net with strided convolutional layers (LS-Net-S) with an LS-Net with max-pooling layers (LS-Net-P), which is constructed by replacing each stride-2 convolutional layer in the LS-Net-S by a stride-1 convolutional layer followed by a max-pooling layer. The comparisons between the LS-Net-S' and the LS-Net-P' performances and losses are shown in Table III and Fig. 6, respectively. As can be seen from Table III, the LS-Net-S architecture outperforms the LS-Net-P architecture in terms of APR and F_1 Score.

TABLE III: Comparisons between the LS-Net with strided convolutional layers (LS-Net-S) and the LS-Net with max pooling layers (LS-Net-P).

Method	APR	ARR	F_1 Score
LS-Net-P	0.7828	**0.5378**	0.5885
LS-Net-S	**0.8004**	0.5368	**0.5940**

We observe that both LS-Net-S and LS-Net-P perform similarly on the classification sub-task; however, the LS-Net-S outperforms the LS-Net-P on the line segment regression sub-task (see Fig. 6). This indicates that strided convolution is a more suitable choice for our proposed LS-Net architecture than the standard max pooling.

Then, we investigate the impact of the four-grid approach. We compare against LS-Net with one, two, three, and four grids, respectively. Table IV shows that as the number of grids increases, APR decreases slightly, but ARR increases dramatically. This results in an increase of F_1 score as the number of grids increases. Since LS-Net with more grids makes more predictions then LS-Net with fewer grids, their APRs are slightly lower than that of LS-Net with fewer grids. However, as the additional grids detect short line segments ignored by the main grid at cell corners and close gaps at horizontal



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 5: Visual comparisons of the LS-Net-W2 model and the state-of-the-art methods. From from left to right, top to bottom are respectively (a) the original image, (b) the ground truth, (c) the WSL-CNN's detection results, (d) the WD-DCNN's detection results, (e) the WD-DCNN-GN's detection results, (f) the WD-DCNN-FL's detection results, (g) the WD-DCNN-GNFL's detection results, and (h) the LS-Net's detection results.

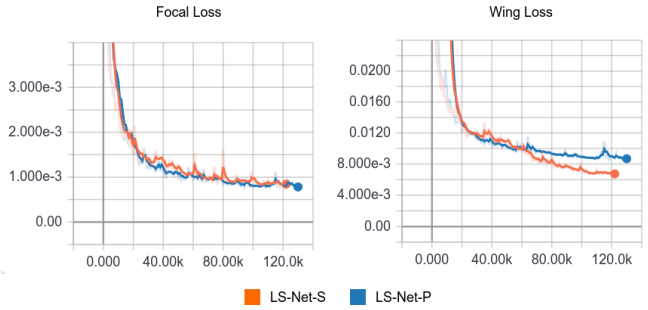


Fig. 6: Comparisons between LS-Net-S' and LS-Net-P' test losses. LS-Net-S and LS-Net-P perform similarly on the classification sub-task (Focal loss); however, the LS-Net-S outperforms the LS-Net-P on the line segment regression sub-task (Wing loss)

and vertical borders, the ARR of LS-Net with more girds is significantly higher than that of LS-Net with fewer grids. As can be seen from Table IV and Fig. 7, the LS-Net with four grids outperforms LS-Net with one, two, and three grids in terms of ARR and F_1 score and significantly eliminates the discontinuities in the detected lines. This suggests that our proposed four-grid approach is more suited for our proposed LS-Net architecture.

TABLE IV: Performance of LS-Net with the one, two, three, and four grids respectively. The methods are denoted as "'LS-Net-⟨number of grids⟩-⟨grids⟩". M, H, V, C represent main, horizontal, vertical, and central grids respectively.

Method	APR	ARR	F1 Score
1 Grid			
LS-Net-1-M	**0.8312**	0.3791	0.4847
2 Grids			
LS-Net-2-MH	0.8174	0.4717	0.5540
LS-Net-2-MV	0.8173	0.4703	0.5533
LS-Net-2-MC	0.8165	0.4776	0.5574
3 Grids			
LS-Net-3-MHC	0.8080	0.5121	0.5792
LS-Net-3-MVC	0.8080	0.5117	0.5791
LS-Net-3-MVH	0.8064	0.5165	0.5826
4 Grids			
LS-Net-4-MHVC	0.8004	**0.5368**	**0.5940**

Next, we show the effects of the Wing loss on line segment regression performance. We compare against LS-Net trained

Fig. 7: Test results of the LS-Net with (from left to right) one, two, three, and four grids, respectively (the width of the line segments is increased to 5 pixels for better visualizations). The one-grid LS-Net approach (the leftmost image) ignores short line segments at cell corners and leaves gaps at cell borders in the detected lines. The four-grid LS-Net approach (the rightmost image) detects line segments ignored by the first grid and close gaps in horizontal and vertical lines, which significantly eliminate the discontinuities in the detected lines.

with Wing loss (LS-Net-W) and its variants: LS-Net trained with L2 loss (LS-Net-2), L1 loss (LS-Net-1), and Smooth L1 loss (LS-Net-S) [13], respectively. Table V shows that LS-Net trained with Wing loss outperforms its variants in terms of APR; however, it performs worse in terms of ARR. Wing loss biases the optimizer towards minimizing small regression errors at the end of the training by increasing the gradient, given by $1/x$, as the errors approach zero error. This results in lower regression errors that lead to a significantly higher APR compared to the LS-Net-1, LS-Net-2, and LS-Net-S. However, this causes the classification errors to increase as we use a fixed weight in the multi-task loss (see Eq. (6)). This leads to a slightly lower ARR compared to the LS-Net-1, LS-Net-2, and LS-Net-S. Since the increase in APR is much more than the decrease in ARR, the F_1 score of LS-Net trained with Wing loss is higher than LS-Net trained with the standard regression losses such as L2, L1, and Smooth L1. This indicates that the Wing loss is a more suitable choice for training the line segment regressor in our proposed LS-Net architecture; however, an adaptive weighting approach is needed for balancing the training of the line segment regressor and the cell classifier. We leave this for future work.

TABLE V: Performance of the LS-Net trained with Wing loss (LS-Net-W), L2 loss (LS-Net-2), L1 loss (LS-Net-1), and L1 smooth loss (LS-Net-S).

Method	APR	ARR	F1 Score
LS-Net-2 (L2)	0.7277	**0.5789**	0.5866
LS-Net-1 (L1)	0.7032	0.5694	0.5765
LS-Net-S (Smooth L1)	0.7317	0.5495	0.5728
LS-Net-W (Wing loss)	**0.8004**	0.5368	**0.5940**

Finally, we evaluate the effect of the Focal loss on cell classification performance. We compare between LS-Net trained with Focal Loss (LS-Net-FL) and LS-Net trained with standard Cross-Entropy loss (LS-Net-CE). As can be seen from Table VI, LS-Net trained with Focal loss outperforms LS-Net trained with standard Cross-Entropy loss in terms of APR, ARR, and F_1 score. This indicates that the Focal loss is a more suitable choice for training the cell classifier in our proposed LS-Net architecture than the standard Cross-Entropy loss.

TABLE VI: Comparisons between LS-Net trained with Focal Loss (LS-Net-FL) and LS-Net trained with standard Cross-Entropy loss (LS-Net-CE).

Method	APR	ARR	F1 Score
LS-Net-CE (Cross Entropy loss)	0.7946	0.5353	0.5899
LS-Net-FL (Focal Loss)	**0.8004**	**0.5368**	**0.5940**

V. Discussion

With the ability to detect power line segments at near real-time (20.4 FPS), the LS-Net shows the potential to facilitate real-time power line detection and avoidance in low-altitude UAV flights to ensure flight safety. During UAV flights, power line segment maps produced by the LS-Net can be employed to detect power lines and identify dangerous zones quickly, and these information sources can be used as additional inputs to improve the performance of obstacle avoidance and path recovery algorithms.

In addition, the LS-Net can be utilized for vision-based UAV navigation and for vision-based inspection of power lines. In automatic autonomous power line inspection, the UAV needs to flight along the power lines to take pictures for offline inspections and performs online inspection to identify faults on the power lines (e.g., corroded and damaged power lines) and surrounding objects, such as vegetation encroachment. When GPS-based navigation is not possible, power line segment maps produced by the LS-Net can be employed to navigate the UAV along the power lines. Besides, the power line segment maps can be used for steering the cameras mounted on the UAV to take higher quality pictures of the power lines to improve the performance and reduce the costs of both online and offline inspections.

Since the LS-Net can be trained end-to-end and performs very well even when trained only on synthetic images, it can potentially be adapted for detecting other linear structures. One example is railway track detection. In recent years, the need for automatic vision-based inspection of railway tracks using UAVs has been increasing since UAVs do not require separate tracks for data acquisition as in traditional inspection methods [52]. Similar to power line inspection, the LS-Net can be potentially applied for detecting railway tracks from images taken from UAVs. These detections can be utilized both for navigating the UAVs along the railway tracks and for steering the cameras mounted on the UAVs to take pictures of the railway tracks for offline inspections. Another example is unburied onshore pipeline detection in automatic UAV-based gas leak inspection [53]. Since the width of gas pipelines is relatively big in images taken from UAVs, the LS-Net can not be applied directly to detect gas pipelines. However, this problem can potentially be addressed by casting the gas pipeline detection as a gas pipeline edge detection problem. The LS-Net can be applied for detecting the edges of gas pipelines. The edge detection results can be used for navigating the UAVs along the pipelines, for steering other sensors such as thermal cameras for detecting gas leaks, and even for sizing the pipelines.

In addition to railway track detection and unburied onshore pipeline detection, the LS-Net can potentially be applied for road detection in low- and mid-altitude aerial imagery which facilitates many applications of UAVs such as traffic monitoring and surveillance, path planning, and inspection [54]. In UAV images, roads are usually very wide; hence, the edge detection approach as used in unburied onshore pipeline detection can be applied. Roads in satellite images, on the other hand, are usually very narrow and thus can be modeled as lines or curves; this means that the LS-Net can potentially be applied directly for detecting roads in satellite images.

VI. Conclusion

This paper introduces LS-Net, a fast single-shot line segment detector. The LS-Net is by design fully convolutional and consists of three modules: (i) a fully convolutional feature extractor, (ii) a classifier, and (iii) a line segment regressor. The LS-Net can be trained end-to-end by backpropagation and stochastic gradient descent (SGD) via a weighted multi-task loss function. The proposed loss function is a combination of Focal loss for addressing the class imbalance in classification and Wing loss for restoring the balance between the influence of errors of different sizes in multiple points regression.

With a customized version of the VGG-16 network as the backbone, the proposed approach outperforms existing state-of-the-art DL-based power line detection approaches. In addition, the LS-Net can run at near real-time (20.4 FPS), which can facilitate real-time power line detection for obstacle avoidance in low-altitude UAV flights, for vision-based UAV navigation and inspection in automatic autonomous power line inspection. Since the LS-Net can be trained end-to-end and performs very well even when trained only on synthetic images, it can potentially be adapted for detecting other linear structures, such as railway tracks, unburied onshore pipelines, and roads from low- and mid-altitude aerial images.

Acknowledgment

The authors would like to thank eSmart Systems and UiT Machine Learning Group for support in the work with this paper. This work was supported by the Research Council of Norway [RCN NÆRINGSPHD grant no. 263894 (2016-2018) on Power Grid Image Analysis] and eSmart Systems.

References

[1] H. Shan, J. Zhang, X. Cao, X. Li, and D. Wu, "Multiple auxiliaries assisted airborne power line detection," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 6, pp. 4810–4819, 2017.

[2] V. N. Nguyen, R. Jenssen, and D. Roverso, "Automatic autonomous vision-based power line inspection: A review of current status and the potential role of deep learning," *International Journal of Electrical Power & Energy Systems*, vol. 99, pp. 107–120, 2018.

[3] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679–698, 1986.

[4] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[5] V. N. Nguyen, R. Jenssen, and D. Roverso, "Intelligent monitoring and inspection of power line components powered by uavs and deep learning," *IEEE Power and Energy Technology Systems Journal*, vol. 6, no. 1, pp. 11–21, 2019.

[6] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. A. Riedmiller, "Striving for simplicity: The all convolutional net," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6806

[7] M. Pharr, W. Jakob, and G. Humphreys, *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. [Online]. Available: http://arxiv.org/abs/1409.1556

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017. [Online]. Available: http://doi.acm.org/10.1145/3065386

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, 2016, pp. 770–778. [Online]. Available: https://doi.org/10.1109/CVPR.2016.90

[11] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, 2017. [Online]. Available: https://doi.org/10.1109/TPAMI.2016.2577031

[12] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: object detection via region-based fully convolutional networks," in *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, 2016, pp. 379–387.

[13] R. Girshick, "Fast R-CNN," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[14] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, 2016, pp. 779–788.

[15] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, "SSD: single shot multibox detector," in *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part I*, 2016, pp. 21–37.

[16] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 2999–3007.

[17] Z. Feng, J. Kittler, M. Awais, P. Huber, and X. Wu, "Wing loss for robust facial landmark localisation with convolutional neural networks," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, 2018, pp. 2235–2245.

[18] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, 2015, pp. 448–456.

[19] Y. Wu and K. He, "Group normalization," in *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIII*, 2018, pp. 3–19. [Online]. Available: https://doi.org/10.1007/978-3-030-01261-8_1

[20] S. Xie and Z. Tu, "Holistically-nested edge detection," *International Journal of Computer Vision*, vol. 125, no. 1-3, pp. 3–18, 2017.

[21] Y. Liu, M. Cheng, X. Hu, J. Bian, L. Zhang, X. Bai, and J. Tang, "Richer convolutional features for edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 8, pp. 1939–1946, 2019.

[22] J. Lee, H. Kim, C. Lee, and C. Kim, "Semantic line detection and its applications," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 3249–3257.

[23] Z. Li, Y. Liu, R. Hayward, J. Zhang, and J. Cai, "Knowledge-based power line detection for uav surveillance and inspection systems," in *Image and Vision Computing New Zealand, 2008. IVCNZ 2008. 23rd International Conference*. IEEE, 2008, pp. 1–6.

[24] Z. Li, Y. Liu, R. A. Walker, R. Hayward, and J. Zhang, "Towards automatic power line detection for a UAV surveillance system using pulse coupled neural filter and an improved hough transform," *Machine Vision and Applications*, vol. 21, no. 5, pp. 677–686, 2010.

[25] G. Yan, C. Li, G. Zhou, W. Zhang, and X. Li, "Automatic extraction of power lines from aerial images," *IEEE Geoscience and Remote Sensing Letters*, vol. 4, no. 3, pp. 387–391, 2007.

[26] B. Song and X. Li, "Power line detection from optical images," *Neurocomputing*, vol. 129, pp. 350–361, 2014.

[27] J. Zhang, H. Shan, X. Cao, P. Yan, and X. Li, "Pylon line spatial correlation assisted transmission line detection," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 50, no. 4, pp. 2890–2905, 2014.

[28] C. Pan, H. Shan, X. Cao, X. Li, and D. O. Wu, "Leveraging spatial context disparity for power line detection," *Cognitive Computation*, vol. 9, no. 6, pp. 766–779, 2017.

[29] J. Gubbi, A. Varghese, and P. Balamuralidhar, "A new deep learning architecture for detection of long linear infrastructure," in *Fifteenth IAPR International Conference on Machine Vision Applications, MVA 2017, Nagoya, Japan, May 8-12, 2017*, 2017, pp. 207–210.

[30] R. Madaan, D. Maturana, and S. Scherer, "Wire detection using synthetic data and dilated convolutional networks for unmanned aerial vehicles," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017*, 2017, pp. 3487–3494.

[31] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. [Online]. Available: http://arxiv.org/abs/1511.07122

[32] R. Kasturi and O. I. Camps, "Wire detection algorithms for navigation," *NASA Technical Report*, 2002.

[33] S. J. Lee, J. P. Yun, H. Choi, W. Kwon, G. Koo, and S. W. Kim, "Weakly supervised learning with convolutional neural networks for power line localization," in *2017 IEEE Symposium Series on Computational Intelligence, SSCI 2017, Honolulu, HI, USA, November 27 - Dec. 1, 2017*, 2017, pp. 1–8.

[34] Y. Li, C. Pan, X. Cao, and D. Wu, "Power line detection by pyramidal patch classification," *IEEE Transactions on Emerging Topics in Computational Intelligence*, no. 99, 2018.

[35] S. C. Wong, A. Gatt, V. Stamatescu, and M. D. McDonnell, "Understanding data augmentation for classification: When to warp?" in *2016 International Conference on Digital Image Computing: Techniques and Applications, DICTA 2016, Gold Coast, Australia, November 30 - December 2, 2016*, 2016, pp. 1–6.

[36] A. G. Howard, "Some improvements on deep convolutional neural network based image classification," in *2nd International Conference on Learning*

Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, 2014. [Online]. Available: http://arxiv.org/abs/1312.5402

[37] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," in *7th International Conference on Document Analysis and Recognition (ICDAR 2003), 2-Volume Set, 3-6 August 2003, Edinburgh, Scotland, UK*, 2003, pp. 958–962.

[38] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. Yu, "scikit-image: Image processing in python," *CoRR*, vol. abs/1407.6245, 2014. [Online]. Available: http://arxiv.org/abs/1407.6245

[39] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[40] A. K. Boyat and B. K. Joshi, "A review paper: Noise models in digital image processing," *Signal & Image Processing*, vol. 6, no. 2, p. 63, 2015.

[41] L. Shapiro and G. Stockman, *Computer Vision*, 1st ed. Prentice Hall, 2001, ch. 5.4, p. 154.

[42] G. Wolberg, *Digital image warping*. IEEE, 1990.

[43] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[44] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, 2014, pp. 580–587.

[45] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[46] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 2016, pp. 265–283. [Online]. Available: https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf

[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[48] Ö. E. Yetgin and Ö. N. Gerek, "Ground truth of powerline dataset (infrared-ir and visible light-vl)," *Mendeley Data, v8, http://dx. doi. org/10.17632/twxp8xccsw*, vol. 8, 2017.

[49] J. Bresenham, "Algorithm for computer control of a digital plotter," *IBM Systems Journal*, vol. 4, no. 1, pp. 25–30, 1965.

[50] L. Jianzhuang, L. Wenqing, and T. Yupeng, "Automatic thresholding of gray-level pictures using two-dimension otsu method," in *Circuits and Systems, 1991. Conference Proceedings, China., 1991 International Conference on*. IEEE, 1991, pp. 325–327.

[51] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[52] A. K. Singh, A. Swarup, A. Agarwal, and D. Singh, "Vision based rail track extraction and monitoring through drone imagery," *ICT Express*, 2017.

[53] T. E. Barchyn, C. H. Hugenholtz, S. Myshak, and J. Bauer, "A UAV-based system for detecting natural gas leaks," *Journal of Unmanned Vehicle Systems*, vol. 6, no. 1, pp. 18–30, 2017.

[54] H. Zhou, H. Kong, L. Wei, D. C. Creighton, and S. Nahavandi, "On detecting road regions in a single UAV image," *IEEE Trans. Intelligent Transportation Systems*, vol. 18, no. 7, pp. 1713–1722, 2017.

Van Nhan Nguyen received his B.Eng (2014) in computer science & engineering from Ho Chi Minh City University of Technology, Ho Chi Minh City, Vietnam and a M.S (2016) in computer science from Østfold University College, Halden, Norway. He is currently working on his Ph.D in deep learning at the UiT Machine Learning Group, UiT The Arctic University of Norway, Tromsø, Norway. His research interests include deep vision (deep learning for computer vision), especially image classification, object detection, semantic segmentation, one-shot learning, zero-shot learning, deep learning-based line detection, and vision-based automatic autonomous inspection of power lines based on Unmanned Aerial Vehicles (UAV) and deep learning.

Robert Jenssen (M02) received the PhD (Dr. Scient.) in Electrical Engineering from the University of Tromsø, in 2005. Currently, he is a Professor with the Department of Physics and Technology, UiT The Arctic University of Norway, Tromsø, Norway. He directs the UiT Machine Learning Group: http://site.uit.no/ml. He is also a Research Professor with the Norwegian Computing Center, Oslo, Norway. He was a Guest Researcher with the Technical University of Denmark, Kongens Lyngby, Denmark, from 2012 to 2013, with the Technical University of Berlin, Berlin, Germany, from 2008 to 2009, and with the University of Florida, Gainesville, FL, USA, from 2002 to 2003, spring 2004, and spring 2018. Jenssen is on the IEEE Technical Committee on Machine Learning for Signal Processing (MLSP), he is the president of the Norwegian section of IAPR (NOBIM - nobim.no) and serves on the IAPR Governing Board. He is an associate editor with the journal Pattern Recognition since 2010.

Davide Roverso, PhD
CAO, Chief Analytics Officer
Davide holds a PhD degree in Computing Science. He has over 25 years experience in the field of Machine Learning and Big Data Analytics, with applications in diagnostics, prognostics, condition monitoring, and early fault detection in complex processes, in sectors ranging from energy to medicine and environmental monitoring. He has authored over 100 publications in international journals, conference proceedings and edited books. He is currently Chief Analytics Officer at eSmart Systems where he leads the analytics and data science group.

Appendix D

Paper III

SEN: A Novel Dissimilarity Measure for Prototypical Few-Shot Learning Networks

Nhan van Nguyen*†, Sigurd Løkse*, Kristoffer Wickstrøm*, Michael Kampffmeyer*, Davide Roverso†, and Robert Jenssen*

*Department of Physics and Technology, UiT The Arctic University of Norway, 9019 Tromsø, Norway

†Analytics Department, eSmart Systems, 1783 Halden, Norway

***Abstract*—The (squared) Euclidean distance is the most common option when optimizing embedding distance metrics, for instance via the recent Prototypical Network (PN), for solving the few-shot learning task. However, the Euclidean distance is not optimal in high-dimensional embedding spaces since it typically suffers from the curse of dimensionality. L_2 normalization, which implicitly results in a hyperspherical embedding, can alleviate this problem to some extent. However, L_2 normalization leads to a non-convex loss formulation, which typically results in local minima. With the aim of performing soft feature normalization while preserving the convexity and the simplicity of the loss function, we propose a novel dissimilarity measure in terms of the Squared root of the Euclidean distance and the Norm distance (SEN) combined. The SEN addresses the existing issues of the Euclidean distance combined with L_2 normalization by forcing the data to gradually lie on a scaled unit hypersphere during training. This is done by encouraging the norm of samples to have the same value. We extend the powerful PN by replacing the Euclidean distance by our proposed SEN dissimilarity measure, which we refer to as SEN PN. With minimal modifications, the SEN PN outperforms the original PN by a considerable margin and demonstrates good performance on the miniImageNet dataset with no additional parameters as well as almost no additional computational overhead. We provide analyses showing that SEN indeed explicitly forces all embeddings to have the same norm during training which enables the SEN PN to generate a more robust embedding space. We experimentally show that the proposed SEN dissimilarity measure constantly outperforms the Euclidean distance in PN with different embedding sizes as well as with different embedding networks.**

Keywords*—*Distance Metric Learning, Few-shot Learning, Metric Learning, Deep Learning

I. Introduction

The availability of large datasets (such as ImageNet [1] and Microsoft COCO [2]), advances in GPU-accelerated computing, and streamlined designs of deep neural networks, have enabled deep learning methods to achieve great success in a variety of AI-related tasks. This is especially the case in computer vision, such as image classification, object detection, and image segmentation. However, most of these successes are based on conventional supervised end-to-end learning approaches, which typically require lots of labeled data to train and are prone to overfitting when only a small amount of training data is available. In addition, these approaches are typically not able to generalize well to changing tasks. To avoid overfitting and to improve the generalization ability of conventional deep learning models, many researchers have relied on regularization (e.g., batch normalization [3] and dropout [4]) and data augmentation. These approaches work well on medium-sized (or sometimes even small-sized) datasets. However, they typically fail in extreme cases where only one or a few examples per class are available.

Humans are, on the other hand, capable of learning new concepts quickly from only one or a few examples, i.e., one-shot or few-shot learning, by effectively utilizing prior knowledge and experience. For example, a child who has learned what a horse looks like can rapidly transfer their knowledge to learn what a zebra looks like from just one or a few example images.

Inspired by humans' ability to learn new concepts quickly, there has been a recent resurgence of interest in designing specialized deep learning models for one-shot and few-shot learning tasks. In this paper, we focus mainly on few-shot learning. One of the most common few-shot learning tasks is few-shot classification in which the goal is to adapt a classifier to previously unseen classes from just a handful of labeled examples per class.

In the past few years, many few-shot classification approaches have been proposed. These approaches can be roughly categorized as (i) learning to fine-tune approaches; (ii) sequence-based approaches; (iii) generative modeling-based approaches; (vi) distance metric learning-based approaches; (v) deep distance metric learning-based approaches; and (vi) semi-supervised approaches. Among these categories, distance metric learning-based approaches are typically preferred since they are simpler and more efficient than other few-shot learning approaches, which require complex inference mechanisms, complex Recurrent Neural Network (RNN) architectures, or fine-tuning the target problem.

The basic idea of distance metric learning-based approaches is to learn a non-linear mapping of the input into an embedding space and define a metric distance which maps similar examples close and dissimilar ones distant in the embedding space, so that a query example can be easily classified by, for example, using nearest neighbor methods. The success of these methods relies heavily on the choice of the distance metric function. In the past few years, many fixed metric distance (e.g., the Euclidean distance [5] and the cosine distance [6])

Nguyen, Løkse, Wickstrøm, Kampffmeyer, and Jenssen, are all with the UiT Machine Learning Group: machine-learning.uit.no

and learnable deep metric distance functions, such as [7], have been applied to few-shot classification models.

We build on the distance metric learning line of work due to its simplicity and effectiveness. In distance metric learning-based few-shot learning, the (squared) Euclidean distance is arguably one of the most commonly used distance metrics; however, it has been shown that the Euclidean distance is not an optimal distance function in high-dimensional spaces since it typically suffers from the curse of dimensionality [8]. L_2 normalization, which implicitly results in a hyperspherical embedding, can attenuate the curse of dimensionality to some extent [9]; however, L_2 normalization leads to a non-convex loss formulation, which typically results in local minima. With the aim of performing soft feature normalization while preserving the convexity and the simplicity of the loss function, we propose a novel dissimilarity measure in terms of the Squared root of the Euclidean distance and the Norm distance (SEN) combined. The SEN addresses the existing issues of the Euclidean distance combined with L_2 normalization by forcing the data to gradually lie on a scaled unit hypersphere during training. This is done by encouraging the norm of samples to have the same value.

To evaluate the effectiveness of our proposed SEN dissimilarity measure, we extend the powerful Prototypical Network (PN) by replacing the Euclidean distance by our proposed SEN dissimilarity measure, which we refer to as SEN PN. With minimal modifications, the SEN PN outperforms the original PN by a considerable margin and demonstrates good performance on the miniImageNet dataset with no additional parameters as well as almost no additional computational overhead.

In addition, we provide analyses showing that SEN indeed explicitly forces all embeddings to have the same norm during training which enables the SEN PN to generate a more robust embedding space. Besides, we experimentally show that the proposed SEN dissimilarity measure constantly outperforms the Euclidean distance in PN with different embedding sizes as well as with different embedding networks and is an effective feature normalization technique not only for distance metric learning-based few-shot learning with PN but also potentially for more general tasks such as dimensionality reduction and deep clustering.

The remainder of the paper is structured as follows: Section II presents background knowledge and relevant related work in few-shot learning. Next, we describe our proposed approach in Section III. Then, in section IV, we present in detail experimental results and ablation studies. Finally, concluding remarks are provided in Section V.

II. Background and Related Work

The literature on few-shot learning is vast; we present in this section a short summary of well-known approaches and works most relevant to our proposed approach. We refer the reader to [10] and [11] for more detailed reviews on few-shot learning.

Contemporary approaches for few-shot learning can be roughly categorized as (i) learning to fine-tune approaches; (ii) sequence-based approaches; (iii) generative modeling-based approaches; (vi) distance metric learning-based approaches; (v) deep distance metric learning-based approaches; and (vi) semi-supervised approaches.

A. Learning to Fine-Tune Approaches

Finn et al. [12] propose a Model-Agnostic Meta-Learning approach (MAML) to learn a model's initial parameters such that it can be quickly adapted to a new task through only one or a few gradient update steps. In other words, MAML aims at learning a good internal representation that is broadly suitable for many tasks, and from there, good results can be achieved by simply fine-tuning the model slightly via one or a few weight update steps. Since MAML is model-agnostic by design, it can handle any model representation that is amenable to gradient-based training and is applicable to a variety of different problems such as classification, regression, and reinforcement learning.

Ravi and Larochelle [13] propose a Long Short-Term Memory (LSTM)-based meta-learner model not only to learn a good initialization for another learner (classifier) network that allows for quick training convergence but also to discover the exact optimization algorithm that can be employed for training the learner in the few-shot regime.

Although these approaches can handle many model representations, they both suffer from the need to fine-tune on the target problem, which makes them less appealing to few-shot learning.

B. Sequence Based Approaches

Santoro et al. [14] propose a method for few-shot classification based on Memory-Augmented Neural Networks (MANNs). The authors modify the Neural Turing Machines (NTMs) [15], which have the ability to quickly encode and retrieve new information using external memory, to excel at one-shot learning. The authors introduce a new method for accessing external memory, called the Least Recently Used Access (LRUA), which only uses content-based location.

Mishra et al. [16] formalize meta-learning as a sequence-to-sequence problem and propose a novel class of model architectures, called the Simple Neural AttenIve Learner (SNAIL), to resolve the problem of existing approaches in quickly incorporating and referring to past experience. SNAIL employs a novel combination of temporal convolutions and soft attention, which enables the meta-learner to aggregate contextual information from past experience and allows it to pinpoint specific pieces of information, respectively.

While appealing, these methods typically require complex RNN architectures and complicated mechanisms for storing/retrieving all the historical information of relevance, both long-term and short-term, without forgetting [7].

C. Generative Modeling Based Approaches

Zhang et al. [17] argue that it may be easier to form a decision boundary between objects that look very different, for example, cats and cars, than between objects that look very

similar, such as cats and dogs. Thus, it is difficult for conventional few-shot learning approaches to extract the correct features to separate similar classes (e.g., cats and dogs) if they are not in the training data. Based on this, the authors propose an adversarial training based framework called MetaGAN with the aim of providing additional signals to the classifiers and making the decision boundaries much sharper. MetaGAN casts the classifier in conventional few-shot learning approaches as a discriminator and employs an imperfect generator to provide fake data between the manifolds of different real data classes. Since the discriminator is forced to not only classify real classes but also to distinguish between real/fake classes, it has to extract stronger features that typically lead to much sharper decision boundaries between real classes.

Wang et al. [18] propose a novel approach to few-shot learning based on learning to hallucinate additional examples. The authors combine the "*learning to learn*" [19] and "*learning to augment*" [18] ideas by employing a hallucinator to produce additional training examples to allow the classification algorithm to learn a better classifier. The authors argue that the aim of the hallucinator should be to hallucinate examples that are useful for learning classifiers instead of diversity or realism and propose to train the classification algorithm and the hallucinator jointly.

D. Distance Metric Learning Based Approaches

Snell et al. [5] propose a simple method called Prototypical Networks (PNs) for few-shot learning based on the assumption that there exists an embedding space in which samples from each class cluster around a single prototype representation, which is simply the mean of the individual samples. Specifically, PNs learn a non-linear mapping of the input into the embedding space. A prototype for each class in the embedding space is generated by taking the mean of the embeddings of its support examples. An embedded query point is then classified by simply finding the nearest class prototype based on the squared Euclidean distance metric.

Garcia and Bruna [20] argue that few-shot learning, which aims at propagating label information from labeled support examples towards unlabeled query images, can be formalized as a posterior inference over a graphical model determined by the images and labels in the support set and the query set. The authors cast posterior inference as message passing on graph neural networks and propose a graph-based model, which can be trained end-to-end, to solve the task. The authors further extend the algorithm for semi-supervised few-shot learning and active few-shot learning.

With the aim of improving the generalization capacity of metric-based methods for few-shot learning, Wang et al. propose to enforce a large margin between the class centers [21]. To do this, the authors propose to augment a large margin loss function to the standard softmax loss function for classification. The unnormalized triplet loss [22] is chosen to be the large margin distance function. The authors also provide experimental results with other existing large margin distance functions, including the normalized triplet loss, the normalized contrastive loss [23, 24], the normface loss [25], the cosface loss [26], and the arcface loss [27], and conclude that the unnormalized triplet loss is more robust than the above-mentioned loss functions. Experimental results show that the proposed approach slightly improves the performance of existing metric distance learning-based models such as graph neural networks [20] and prototypical networks [5].

E. Deep Distance Metric Learning Based Approaches

To avoid the need of manually choosing the right distance metric (e.g., the Euclidean distance and the cosine distance), Sung et al. [7] propose a two-branch Relation Network (RN) that can learn both a deep embedding and a deep non-linear metric (similarity function) for comparing images in the embedding space. Specifically, the RN consists of two modules: an *embedding* module and a *relation* module. The embedding module works similarly as that of the PN and learns a non-linear mapping of the input into the embedding space, while the relation module learns deep non-linear similarity function for comparing the embedded points.

Although deep distance metric learning-based approaches can avoid the need for manually choosing the right distance metric, they are prone to overfitting and are more difficult to train compared to distance metric learning-based approaches due to the added parameters.

F. Semi-Supervised Approaches

To take advantage of both labeled and unlabeled data, Boney and Ilin [28] propose to extend prototypical networks to address the semi-supervised few-shot learning problem. Based on the observation that prototypical networks tend to produce clustered data representations, the authors cast the semi-supervised few-shot learning problem as a semi-supervised clustering problem and address it by applying guided hard k-means clustering in the embedding space found by prototypical networks at test time. The k-means clustering process is guided by the labeled examples, which are used for initializing the cluster means.

A similar approach was concurrently developed by Ren et al. [29]. However, the authors apply clustering both at testing and at training to refine the prototypes produced by prototypical networks. To keep the inference differentiable, soft k-means is applied instead of hard k-means. In addition, the authors consider a more challenging situation where unlabeled examples can come from *distractor* classes and propose a soft-masking mechanism to learn to include or ignore entirely certain unlabeled examples in the prototype refining process.

III. Few-shot Learning

In this section, we first begin by detailing the general few-shot learning task. Next, we introduce prototypical networks and the Euclidean distance function with special attention paid to highlight its existing challenges. Then, we describe our proposed SEN dissimilarity measure and our SEN PN model. Finally, we provide analyses on the gradient of the SEN PN's loss function and the behavior of the proposed SEN dissimilarity measure during training.

A. Task Description

In the traditional machine learning setting, we are typically given a dataset D. This dataset is usually split into two parts: D_{train} and D_{test}. The former is often used for training the parameters θ of the model, while the latter is typically used for evaluating its generalization. In general few-shot learning, we are dealing with meta-datasets D_{meta} containing multiple regular datasets D [13]. Each dataset $D \in D_{meta}$ has a split of D_{train} and D_{test}; however, they are usually much smaller than that of regular datasets used in the traditional machine learning setting. Let $C = \{1, \ldots, K\}$ be the set of all classes available in D_{meta}. The set C is usually split into two disjoint sets: C_{train} containing training classes and C_{test} containing unseen classes for testing, i.e., $C_{train} \cap C_{test} = \emptyset$. The meta-dataset D_{meta} is often split into two parts: The first is a meta training set $D_{meta-train} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, where $\mathbf{x}_i$ is the feature vector of the i^{th} example, $y_i \in C_{train}$ is its corresponding label, and N is the number of training examples. The second part is a meta testing set $D_{meta-test}$. In a standard M-way K-shot classification task, the meta testing set $D_{meta-test}$ consists of a *support set* and a *query set*. The support set $S = \{(\mathbf{x}_j, y_j)\}_{j=1}^{N_S}$ contains K examples from each of the M classes from C_{test}, i.e., the number of support examples are $N_S = M \times K$ and $y_j \in C_{test}$. The *query set* contains N_Q unlabeled examples $Q = \{(\mathbf{x}_j)\}_{j=N_S+1}^{N_S+N_Q}$. The support set is employed by the model for learning the new task, while the query set is utilized by the model for evaluating its performance.

B. Prototypical Networks

Prototypical networks learn a non-linear embedding function $f_\phi : \mathbb{R}^D \rightarrow \mathbb{R}^E$ parameterized by ϕ that maps a D-dimensional feature vector of an example $\mathbf{x}_i$ to an E-dimensional embedding $\mathbf{z}_i = f_\phi(\mathbf{x}_i)$ [5]. In meta-testing, the embedding function f_ϕ is employed for mapping examples in the support set $S = \{(\mathbf{x}_j, y_j)\}_{j=1}^{N_S}$ into the embedding space. An E-dimensional representation $\mathbf{c}_k$, or *prototype*, of each class is computed by taking the mean of the embedded support points belonging to the class:

$$\mathbf{c}_k = \frac{1}{|S_k|} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_\phi(\mathbf{x}_i) = \frac{1}{|S_k|} \sum_{(\mathbf{x}_i, y_i) \in S_k} \mathbf{z}_i, \qquad (1)$$

where S_k is the support set of class k. An embedded query point $\mathbf{x}_q$ is then classified by simply finding the nearest class prototype in the embedding space based on the squared Euclidean distance metric (see Figure 2).

To train PN, the episodic training strategy proposed in [6, 13] is adopted. In particular, to train PN for the M-way, K-shot classification task, a training episode is formed from the meta training set D_{meta_train} as follows: K examples from each of M randomly selected classes from C_{train} are sampled to form a support set $S = \{S_1, \ldots, S_M\}$. A query set $Q = \{Q_1, \ldots, Q_M\}$ is formed by sampling from the rest of the M classes' samples. Next, for each class k, its support set $S_k \in S$ is used for computing a prototype using Equation 1. Then, a distribution over classes for each query point $\mathbf{x}_q \in Q$ based on a softmax over distances to the prototypes in the embedding space is produced:

$$p_\phi(y = k|\mathbf{x}_q) = \frac{\exp(-d(f_\phi(\mathbf{x}_q), \mathbf{c}_k))}{\sum_{k'} \exp(-d(f_\phi(\mathbf{x}_q), \mathbf{c}_{k'}))}, \qquad (2)$$

where $d = \mathbb{R}^E \times \mathbb{R}^E \rightarrow [0, +\infty)$ is a distance function. Based on that, the PN is trained by minimizing the negative log-probability of the true class k via SGD:

$$J(\phi) = -\frac{1}{M} \sum_{k=1}^{M} \frac{1}{|Q_k|} \sum_{\mathbf{x}_q \in Q_k} \log p_\phi(y = k|\mathbf{x}_q). \qquad (3)$$

The training is repeated with new, randomly generated training episodes until a stopping criterion is met.

C. The Euclidean Distance Function

PN employs the squared Euclidean distance as the distance metric. The squared Euclidean distance between two arbitrary points $\mathbf{z} = (z_1, z_2, \ldots, z_n)$ and $\mathbf{c} = (c_1, c_2, \ldots, c_n)$ is defined as follows:

$$d_{se}(\mathbf{z}, \mathbf{c}) = \|\mathbf{z} - \mathbf{c}\|^2 = \sum_{i=1}^{n} (z_i - c_i)^2. \qquad (4)$$

The (squared) Euclidean distance is arguably one of the most commonly used distance metrics in distance metric learning-based few-shot learning. However, the Euclidean distance typically suffers from the curse of dimensionality. Since the volume of the Euclidean space increases exponentially as the dimension increase, the data becomes very sparse in high-dimensional spaces. Besides, the ratio of an embedding's nearest neighbor over its farthest neighbor approaches one in high-dimensional spaces [8]. In other words, all embeddings are approximately equidistant from each other in high-dimensional Euclidean embedding spaces.

One approach to attenuate the curse of dimensionality is to employ L_2 normalization, which projects a D-dimensional Euclidean space to a high-dimensional hypersphere, S^{D-1}. This causes data to lie on the surface of a unit hypersphere. The area of the unit hypersphere increases as the dimension of the sphere increases initially but then decreases as the dimensionality increases further [30]. This typically results in a more compact embedding space than the Euclidean embedding space. In such an embedding space, the cosine distance is commonly chosen as the distance metric. Many few-shot classification approaches [6, 13] and clustering methods [31] have employed the cosine distance and the hyperspherical embedding space. One of the most well-known example of using cosine distance for clustering data that lies on the surface of a unit hypersphere is the *spherical k-means* algorithm [31], which arises as a special case of *Expectation Maximization (EM)* on *mixture of von Mises-Fisher Distributions (vMF)* [32]. The cosine distance between two arbitrary point $\mathbf{z} = (z_1, z_2, \ldots, z_n)$ and $\mathbf{c} = (c_1, c_2, \ldots, c_n)$ is defined as follows:

$$d_{cs} = 1 - \frac{\mathbf{z} \cdot \mathbf{c}}{\|\mathbf{z}\|\|\mathbf{c}\|} = 1 - \frac{\sum_{i=1}^{n} z_i c_i}{\sqrt{\sum_{i=1}^{n} z_i^2} \sqrt{\sum_{i=1}^{n} c_i^2}}. \qquad (5)$$

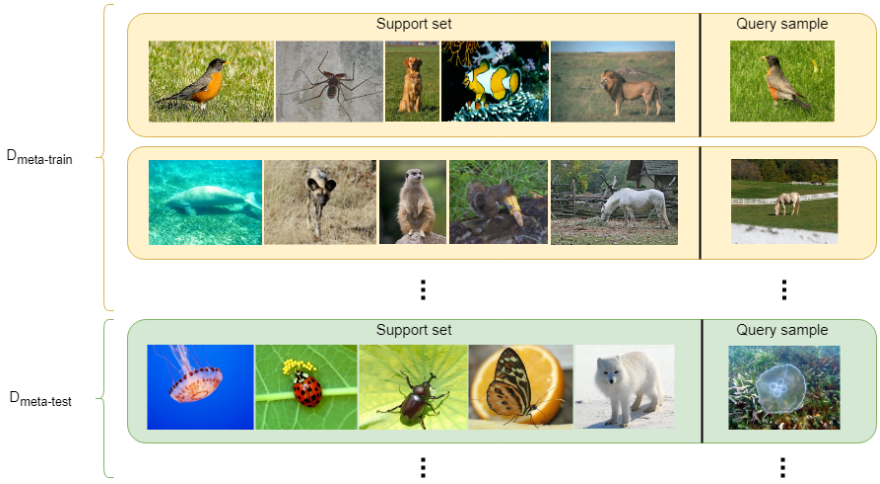


Fig. 1. Illustration of a meta training set $D_{meta-train}$ and a meta testing set $D_{meta-test}$ for the 5-way, 1-shot classification task. In this illustration, for each dataset, we have one example from each of 5 classes in the training set and 1 example for evaluation in the test set.

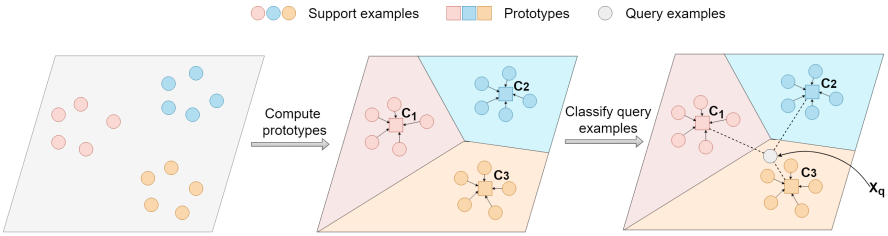


Fig. 2. An illustration of prototypical networks. A prototype for each class in the embedding space is generated by taking the mean of the embeddings of its support examples. An embedded query point is classified by simply finding the nearest class prototype in the embedding space based on the squared Euclidean distance metric.

However, feature normalization through hard normalization operations such as L_2 normalization leads to a non-convex loss formulation, which typically results in local minima [9]. Since the network optimization itself is non-convex, it is important to preserve convexity in loss functions for more effective minimization.

One possible solution is to use Ring loss [9]. The Ring loss introduces an additional term to the primary loss function, which penalizes the squared difference between the norm of samples and a learned target norm value R. The modified loss function is defined as follows:

$$L = L_P + \gamma L_R, \tag{6}$$

where γ is the loss weight w.r.t to the primary loss L_P and L_R is the Ring loss, which is defined as:

$$L_R = \frac{1}{2n}\sum_{i=1}^{n}(\|f_\phi(\mathbf{x}_i)\| - R)^2. \tag{7}$$

Since the Ring loss encourages the norm of samples being value R during training instead of explicit enforcing through a hard normalization operation, the convexity in the loss function is preserved. However, the Ring loss is more difficult to train than the primary loss (e.g., the standard Softmax loss) due to the added term (the norm difference L_R), the added parameter (the target norm R), and the added hyperparameter (the loss weight w.r.t to the primary loss γ).

Intending to encourage the norm of samples to have the same value, in other words, force the data to lie on a scaled unit hypersphere, while preserving the convexity and the simplicity of the loss function, we propose a novel dissimilarity measure, called SEN, to address the existing issues of the traditional Euclidean distance in high dimensional spaces.

D. The SEN Dissimilarity Measure

The SEN dissimilarity measure $d_s(\mathbf{z}, \mathbf{c})$ between two arbitrary points $\mathbf{z} = (z_1, z_2, \ldots, z_n)$ and $\mathbf{c} = (c_1, c_2, \ldots, c_n)$ in D-dimensional space is a combination of the standard squared Euclidean distance d_e and the squared norm distance d_n:

$$d_s(\mathbf{z}, \mathbf{c}) = \sqrt{d_e(\mathbf{z}, \mathbf{c}) + \epsilon d_n(\mathbf{z}, \mathbf{c})}, \tag{8}$$

where ϵ is a tunable balancing hyperparameter and must be chosen such that $d_e(\mathbf{z}, \mathbf{c}) + \epsilon d_n(\mathbf{z}, \mathbf{c})$ is always positive, $d_e(\mathbf{z}, \mathbf{c})$ and $d_n(\mathbf{z}, \mathbf{c})$ are defined as:

$$d_e(\mathbf{z}, \mathbf{c}) = \|\mathbf{z} - \mathbf{c}\|^2, \tag{9}$$

$$d_n(\mathbf{z}, \mathbf{c}) = (\|\mathbf{z}\| - \|\mathbf{c}\|)^2. \tag{10}$$

Algorithm 1 PN's training episode loss computation. $\mathcal{D}_{meta-train} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ is the meta-training set, where $\mathbf{x}_i$ is the feature vector of the i^{th} example, $y_i \in \{1, \ldots, K\}$ is its corresponding label, K is the number of classes in $\mathcal{D}_{meta-traing}$, and N is the number of training examples. $\mathcal{D}_k = \{(\mathbf{x}_j, y_j) \in \mathcal{D} \mid y_j = k\}$ is the meta-training set of class k. $N_C \leq K$, N_S, and N_Q are the number of classes per episode, the number of support examples per class, and the number of query examples per class, respectively. $\mathrm{RS}(S, N)$ is a function that returns a set of N elements chosen uniformly at random from set S, without replacement.

Require: Meta-training set $\mathcal{D}_{meta-train} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$.
Ensure: The loss $J(\phi)$ for a randomly generated training episode.
$V \leftarrow \mathrm{RS}(\{1, \ldots, K\}, N_C)$ ▷ Select class indices for an episode
for k in $\{1, \ldots, N_C\}$ **do**
$S_k \leftarrow \mathrm{RS}(\mathcal{D}_{V_k}, N_S)$ ▷ Select support examples
$Q_k \leftarrow \mathrm{RS}(\mathcal{D}_{V_k} \setminus S_k, N_Q)$ ▷ Select query examples
$\mathbf{c}_k \leftarrow \frac{1}{N_S} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_{\boldsymbol{\phi}}(\mathbf{x}_i)$ ▷ Compute prototype from support examples
end for
$J(\phi) = \frac{1}{N_C} \sum_{k=1}^{N_C} \frac{1}{N_Q} \sum_{\mathbf{x}_q \in Q_k} \left[d(f_{\boldsymbol{\phi}}(\mathbf{x_q}), \mathbf{c}_k) + \log \sum_{k'} \exp(-d(f_{\boldsymbol{\phi}}(\mathbf{x_q}), \mathbf{c}_{k'})) \right]$ ▷ Compute loss

E. SEN Prototypical Networks

We modify the PN by replacing the Euclidean distance by our proposed SEN dissimilarity measure. We call this model SEN PN. Specifically, we replace the distance function $d(\mathbf{z}_i, \mathbf{c}_k)$ in Equation 2 by our proposed SEN dissimilarity measure $d_s(\mathbf{z}_i, \mathbf{c}_k) = \sqrt{d_e(\mathbf{z}_i, \mathbf{c}_k) + \epsilon d_n(\mathbf{z}_i, \mathbf{c}_k)}$, where ϵ is a balancing hyperparameter, $\mathbf{z}_i$ is the embedding of the example $\mathbf{x}_i$, and $\mathbf{c}_k$ is the prototype of class k. For simplicity, we consider the setting in which only one query example per class is used; however, the loss function presented in this session and the analysis presented in the next section can be easily generalized for other settings in which more than one query examples class are used. When only one query example per class is used, the updated negative log probability loss is given as:

$$\begin{aligned} J(\phi) &= -\sum_k \log p_\phi(y_i = k | \mathbf{x}_i) \\ &= -\sum_k \log \frac{\exp(-d_s(\mathbf{z}_i, \mathbf{c}_k))}{\sum_{k'} \exp(-d_s(\mathbf{z}_i, \mathbf{c}_{k'}))} \\ &= \sum_k \left(d_s(\mathbf{z}_i, \mathbf{c}_k) + \log \sum_{k'} \exp(-d_s(\mathbf{z}_i, \mathbf{c}_{k'})) \right). \end{aligned} \tag{11}$$

The learning proceeds by minimizing $J(\phi)$ of the true class k via Stochastic Gradient Descent (SGD), which is equivalent to minimizing the SEN dissimilarity measure between the query example $\mathbf{x}_i$ and its prototype $\mathbf{c}_k$: $d_s(\mathbf{z}_i, \mathbf{c}_k)$, and maximizing the SEN dissimilarity measures between the query example $\mathbf{x}_i$ and the other prototypes $\mathbf{c}_{k'}$: $d_s(\mathbf{z}_i, \mathbf{c}_{k'})$. Minimizing $d_s(\mathbf{z}_i, \mathbf{c}_k)$ pulls $\mathbf{z}_i$ to its own class and encourages embeddings of the same class to have the same norm. Maximizing $d_s(\mathbf{z}_i, \mathbf{c}_{k'})$ pushes $\mathbf{z}_i$ away from other classes; however it encourages embeddings of different classes to have different norms.

Since our goal is to force the data to lie on a scaled unit hypersphere, we propose a special balancing hyperparameter ϵ_{ik}, which is defined as:

$$\epsilon_{ik} = \begin{cases} \epsilon_p > 0 & \text{if } y_i = k \\ \epsilon_n < 0 & \text{if } y_i \neq k \end{cases}, \tag{12}$$

where i is the index of the embedding $\mathbf{z}_i$, y_i is the embedding's class label, and k is the class label of the prototype $\mathbf{c}_k$. During training, a positive epsilon ($\epsilon_{ik} = \epsilon_p > 0$) is used for computing the SEN dissimilarity measure between the query example $\mathbf{x}_i$ and its prototype $\mathbf{c}_k$, while a negative epsilon ($\epsilon_{ik} = \epsilon_n < 0$) is used for computing the SEN dissimilarity measures between the query example $\mathbf{x}_i$ and the other prototypes $\mathbf{c}_{k'}$. The negative epsilon ϵ_n will inverse the effect of the norm distance when maximizing $d_s(\mathbf{z}_i, \mathbf{c}_{k'})$. In other words, maximizing $d_s(\mathbf{z}_i, \mathbf{c}_{k'})$ with a negative epsilon ϵ_n pushes $\mathbf{z}_i$ away from other classes and encourages embeddings of all classes to have the same norm.

This suggests that our proposed SEN dissimilarity measure explicitly encourages the norm of samples to have the same value during training, while preserving the convexity and the simplicity of the loss function. At test time, a positive epsilon ($\epsilon_{ik} = \epsilon_p > 0$) is used for computing all dissimilarity measures.

In the next section, we provide analyses showing that our proposed SEN dissimilarity measure together with the special balancing hyperparameter ϵ_{ik} explicitly pulls the data to a scaled unit hypersphere during training.

F. Analysis

The partial derivative of the negative log probability loss $J(\phi)$ with respect to $d_s(\mathbf{z}_i, \mathbf{c}_k)$ is given by:

$$\frac{\partial J(\phi)}{\partial d_s(\mathbf{z}_i, \mathbf{c}_k)} = \sum_k (1[y_i = k] - p_\phi(y_i = k | \mathbf{x})), \tag{13}$$

where the Iverson bracket indicator function $[y_i = k]$ evaluates to 1 when $y_i = k$ and 0 otherwise. The partial derivative of

the SEN dissimilarity measure $d_s(\mathbf{z}_i, \mathbf{c}_k)$ with respect to $\mathbf{z}_i$ is given by:

$$\begin{aligned}\frac{\partial d_s(\mathbf{z}_i, \mathbf{c}_k)}{\partial \mathbf{z}_i} &= \frac{\partial \sqrt{d_e(\mathbf{z}_i, \mathbf{c}_k) + \epsilon_{ik} d_n(\mathbf{z}_i, \mathbf{c}_k)}}{\partial \mathbf{z}_i} \\ &= \frac{(\mathbf{z}_i - \mathbf{c}_k) + \epsilon_{ik}(\|\mathbf{z}_i\| - \|\mathbf{c}_k\|)\frac{\mathbf{z}_i}{\|\mathbf{z}_i\|}}{d_s(\mathbf{z}_i, \mathbf{c}_k)} \\ &= -\frac{(\mathbf{c}_k - \mathbf{z}_i) + \epsilon_{ik}(\|\mathbf{c}_k\| - \|\mathbf{z}_i\|)\frac{\mathbf{z}_i}{\|\mathbf{z}_i\|}}{d_s(\mathbf{z}_i, \mathbf{c}_k)} \\ &= -\frac{v(\mathbf{z}_i, \mathbf{c}_k)}{d_s(\mathbf{z}_i, \mathbf{c}_k)},\end{aligned} \tag{14}$$

where

$$v(\mathbf{z}_i, \mathbf{c}_k) = (\mathbf{c}_k - \mathbf{z}_i) + \epsilon_{ik}(\|\mathbf{c}_k\| - \|\mathbf{z}_i\|)\frac{\mathbf{z}_i}{\|\mathbf{z}_i\|}. \tag{15}$$

Using the chain rule, we get:

$$\begin{aligned}\frac{\partial J(\phi)}{\partial \mathbf{z}_i} &= \frac{\partial J(\phi)}{\partial d_s(\mathbf{z}_i, \mathbf{c}_k)} \frac{\partial d_s(\mathbf{z}_i, \mathbf{c}_k)}{\partial \mathbf{z}_i} \\ &= \sum_k -\frac{1[y_i = k] - p_\phi(y_i = k|x)}{d_s(\mathbf{z}_i, \mathbf{c}_k)} v(\mathbf{z}_i, \mathbf{c}_k) \\ &= \sum_k \frac{\partial J_k(\phi)}{\partial \mathbf{z}_i}.\end{aligned} \tag{16}$$

Thus, there is a gradient contribution from all prototypes. In particular, the gradient contribution with respect to the correct prototype, when $k = k^* = y_i$, is given by:

$$\begin{aligned}\frac{\partial J_{k^*}(\phi)}{\partial \mathbf{z}_i} &= -\frac{1 - p_\phi(y_i = k^*|x)}{d_s(\mathbf{z}_i, \mathbf{c}_{k^*})} v(\mathbf{z}_i, \mathbf{c}_{k^*}) \\ &= -\frac{1 - p_\phi(y_i = k^*|x)}{d_s(\mathbf{z}_i, \mathbf{c}_{k^*})} v_p(\mathbf{z}_i, \mathbf{c}_{k^*}),\end{aligned} \tag{17}$$

where

$$v_p(\mathbf{z}_i, \mathbf{c}_{k^*}) = \underbrace{(\mathbf{c}_{k^*} - \mathbf{z}_i)}_{\text{attractor}} + \underbrace{\epsilon_{ik^*}(\|\mathbf{c}_{k^*}\| - \|\mathbf{z}_i\|)\frac{\mathbf{z}_i}{\|\mathbf{z}_i\|}}_{\text{norm equalizer}}. \tag{18}$$

The gradient contribution with respect to incorrect prototypes, when $k = k' \neq y_i$, is given by:

$$\begin{aligned}\frac{\partial J_{k'}(\phi)}{\partial \mathbf{z}_i} &= -\frac{0 - p_\phi(y_i = k'|x)}{d_s(\mathbf{z}_i, \mathbf{c}_{k'})} v(\mathbf{z}_i, \mathbf{c}_{k'}) \\ &= -\frac{p_\phi(y_i = k'|x)}{d_s(\mathbf{z}_i, \mathbf{c}_{k'})} v_n(\mathbf{z}_i, \mathbf{c}_{k'}),\end{aligned} \tag{19}$$

where

$$v_n(\mathbf{z}_i, \mathbf{c}_{k'}) = \underbrace{(\mathbf{z}_i - \mathbf{c}_{k'})}_{\text{repeller}} + \underbrace{\epsilon_{ik'}(\|\mathbf{z}_i\| - \|\mathbf{c}_{k'}\|)\frac{\mathbf{z}_i}{\|\mathbf{z}_i\|}}_{\text{norm equalizer}}. \tag{20}$$

From the preceding analysis, we observe the following:

1) Each gradient component contains an attractor/repeller, which encourages $\mathbf{z}_i$ to move towards the correct prototype and move away from the incorrect ones.
2) From (18), it is clear that if $\|\mathbf{c}_{k^*}\| > \|\mathbf{z}_i\|$ and $\epsilon_{ik^*} > 0$, $\epsilon_{ik^*}(\|\mathbf{c}_{k^*}\| - \|\mathbf{z}_i\|)\frac{1}{\|\mathbf{z}_i\|} > 0$, such that $\|\mathbf{z}_i\|$ is encouraged to increase (and vice verca for $\|\mathbf{z}_i\| > \|\mathbf{c}_{k^*}\|$).
3) Conversely, from (20), if $\|\mathbf{c}_{k'}\| > \|\mathbf{z}_i\|$ and $\epsilon_{ik'} > 0$, $\epsilon_{ik'}(\|\mathbf{z}_i\| - \|\mathbf{c}_{k'}\|)\frac{1}{\|\mathbf{z}_i\|} < 0$ (and vice verca for $\|\mathbf{z}_i\| > \|\mathbf{c}_{k'}\|$). Thus, we need $\epsilon_{ik'} < 0$ in order to ensure similar behaviours as with the correct prototype.

Observation 2) and 3) shows that the gradient contributions with respect to the correct prototype and the incorrect ones *cooperate* in order to equalize the norms during training when $\epsilon_{ik^*} > 0$ and $\epsilon_{ik'} < 0$.

IV. Experiments

A. Comparisons with the State-of-the-Art Results

To evaluate the effectiveness of the proposed SEN dissimilarity measure, we compare our proposed SEN PN approach with the original PN proposed in [5] as well as six state-of-the-art approaches for few-shot learning, including Large Margin Graph Neural Networks (Large Margin GNN) [21], Large Margin Prototypical Networks (Large Margin PN) [21], SNAIL [16], Relation Network (RN) [7], MetaGAN + RN [17], and Semi-Supervised Prototypical Networks (Semi-Supervised PN) [28]. To test the performance of the SEN dissimilarity measure in more general settings, we employ a more sophisticated network, the Wide Residual Network [33], as the embedding network. We use the same network architecture proposed in [28], which is a network of depth 16 and a widening factor of 6. We train the Prototypical Wide Residual Network with both the traditional Euclidean distance (Resnet PN) and the SEN dissimilarity measure (SEN Resnet PN). The test results are shown in Table I.

As can be seen from Table I, although our implementation of the PN (the baseline model) achieves 0.4% lower in terms of accuracy compared to the original implementation of the PN (67.8% vs 68.2%), the baseline model trained with the proposed SEN dissimilarity measure still outperforms the original PN by 1.6% and achieves an accuracy of 69.8%. In addition, the SEN Resnet PN outperforms the Semi-Supervised Resnet PN by 1.4% and achieves an accuracy of 72.3% with the Wide Residual Network as the embedding network.

Snell et al. [5] evaluated 5-way and 20-way training episodes for the 5-way, 5-shot learning task and concluded that 20-way achieves higher accuracy than 5-way. The authors argued that the increased difficulty of 20-way classification helps the network to generalize better because it forces the model to make more fine-grained decisions in the embedding space. We evaluate our implementation of the PN and our proposed approach, the SEN PN, with both 20-way and 5-way training episodes. The experimental results of the two training settings are shown in Table I. As can be seen from Table I, 5-way training episodes work better.

B. Ablation Study

To investigate the effectiveness and behavior of the proposed SEN dissimilarity measure, we conduct several ablation studies. First, we compare against the original PN trained with

TABLE I. FEW-SHOT CLASSIFICATION ACCURACIES ON MINIIMAGENET (5-WAY 5-SHOT TESTING).

Model	Accuracy
Original PN (20-way training) [5]	**68.2%**
Large Margin GNN (5-way training) [21]	67.6%
Large Margin PN (5-way training) [21]	66.8%
SNAIL (5-way training) [16]	68.9%
RN (5-way training) [7]	65.3%
MetaGAN + RN (5-way training) [17]	68.6%
Semi-Supervised PN (5-way training) [28]	65.5%
Supervised Resnet PN (20-way training) [28]	69.6%
Semi-Supervised Resnet PN (20-way training) [28]	**70.9%**
PN (ours, 20-way training)	65.7%
PN (ours, 5-way training, baseline)	67.8%
SEN PN (ours, 20-way training)	67.9%
SEN PN (ours, 5-way training)	**69.8%**
Resnet PN (ours, 5-way training)	71.0%
SEN Resnet PN (ours, 5-way training)	**72.3%**

the Euclidean distance (PN), the PN trained with the Ring loss (Ring PN), and the PN trained with the SEN dissimilarity measure (SEN PN). The test results are show in Table II. We train the Ring PN with different values of γ, the loss weight w.r.t to the primary loss, in range $[10^{-7}, 1]$ and pick $\gamma = 10^{-7}$ since it results in the highest accuracy. As can be seen from

TABLE II. TEST RESULTS OF PN, RING PN, AND SEN PN.

Model	Accuracy
PN	67.8%
Ring PN	68.6%
SEN PN	**69.8%**

Table II, the Ring loss improves the accuracy of the original PN by 1.2%; however, the Ring PN approach still performs worse than our proposed SEN PN approach, which improves the accuracy of the original PN by 2.0%.

Next, we apply Principal Component Analysis (PCA) to project 1600D embeddings produced by the PN, the Ring PN, and the SEN PN to 2D space and visualize the outputs (see Fig. 3). As can be seen from Fig. 3, the Ring loss forces the prototypes to lie on a scaled unit hypersphere; however, the prototypes produced by the Ring PN are not very well-separated compared to the ones produced by the original PN. Our proposed SEN dissimilarity measure, on the other hand, both forces the prototypes to lie on a scaled unit hypersphere and keeps them well-separated.

Then, we plot the norm of embeddings produced by the PN, the Ring PN, and the SEN PN. As can be seen from Fig. 4, the norm of embeddings produced by the PN and the Ring PN vary a lot, while the norm of embeddings produced by the SEN PN has a very consistent value. This confirms that the proposed SEN dissimilarity measure encourages all embeddings to have the same norm during training. Both the SEN dissimilarity measure and the Ring loss are adopted for explicitly enforcing the norm of embeddings to have the same value during training the PN; however, as can be seen from Fig. 4, the SEN dissimilarity measure is obviously a better choice for the task than the Ring loss. This is partly due to the use of a very small gamma ($\gamma = 10^{-7}$) during training the Ring PN. In our experiments, higher gamma values do encourage the norm of embeddings to have a more consistent value; however, they cause a considerable decrease in the accuracy of the PN. This suggests that the Ring loss is not an optimal choice for enforcing feature normalization in the PN. The proposed SEN dissimilarity measure; on the other hand, both encourages all embeddings to have the same norm and improves the accuracy of the PN. This indicates that the proposed SEN dissimilarity measure is a more suitable choice for feature normalization than the Ring loss in training the PN.

After that, we compare between the PT and SEN PT trained with different embedding sizes (see Fig. 5). As can be seen from Fig 5, in low dimensional spaces, the PT and SEN PT perform very similarly; however, in high dimensional spaces, the SEN PN consistently outperforms the PT by a considerable margin. This suggests that the proposed SEN dissimilarity measure is a more suitable distance metric for metric distance learning-based few-shot learning than the standard Euclidean distance in high dimensional spaces.

Finally, we evaluate the possibility of combining the proposed SEN dissimilarity measure with other distance functions such as the Euclidean distance and the cosine distance in training PN. Specifically, we train the original PN with the SEN dissimilarity measure and test the trained model with both the Euclidean distance and the cosine distance. We compare the two tested models with the original PN, the SEN PN, and the Cosine PN (the PN trained and tested with the cosine distance). The test results are show in Table III. As can be seen from

TABLE III. TEST RESULTS OF THE PN WITH DIFFERENT DISTANCES.

Training Distance	Testing Distance	Accuracy
Cosine	Cosine	53.3%
Euclidean	Euclidean	67.8%
SEN	**SEN**	**69.8%**
SEN	Euclidean	68.8%
SEN	**Cosine**	**69.8%**

Table III, the model trained with the SEN dissimilarity measure achieves the highest accuracy of 69.8% when tested with either the SEN dissimilarity measure or the cosine distance. This happens because the SEN dissimilarity measure explicitly forces all embeddings to have the same norm during training, and, as a result, pulling the prototypes very close to the hypersphere. This turns the standard few-shot classification problem to a *clustering on a hypersphere* task in which the cosine distance has been proven to work well [31, 32].

C. SEN in Dimensionality Reduction and Clustering

The Euclidean distance has been the default choice in many dimensionality reduction techniques [23], in deep clustering approaches, and especially in distance metric learning-based few-shot learning methods. However, as pointed out above, the Euclidean distance is not an optimal distance function in high dimensional spaces since it typically suffers from the curse of dimensionality. Our proposed SEN dissimilarity measure, on the other hand, can attenuate the curse of dimensionality during training by explicitly encouraging the norm of samples to have the same value. This typically results in a more compact embedding space than the Euclidean space.

We have demonstrated that SEN dissimilarity measure outperforms the Euclidean distance in distance metric learning-based few-shot learning with Prototypical Networks. In this

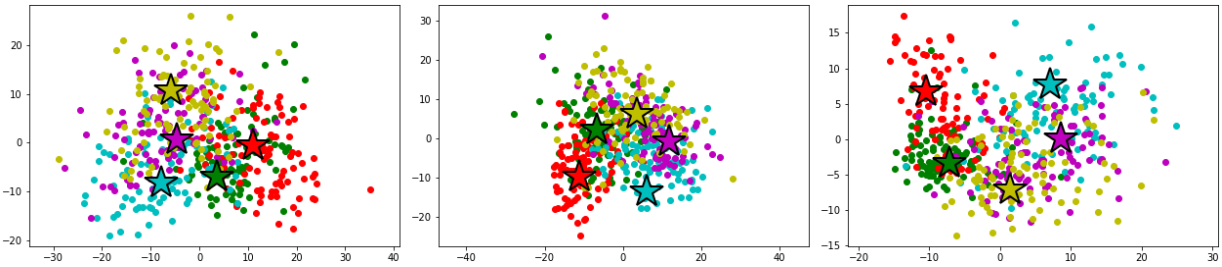


Fig. 3. 2D embeddings produced by the PN (lef), Ring PT (middle) and SEN PN (right). The circles denote query examples, and the stars denotes prototypes.

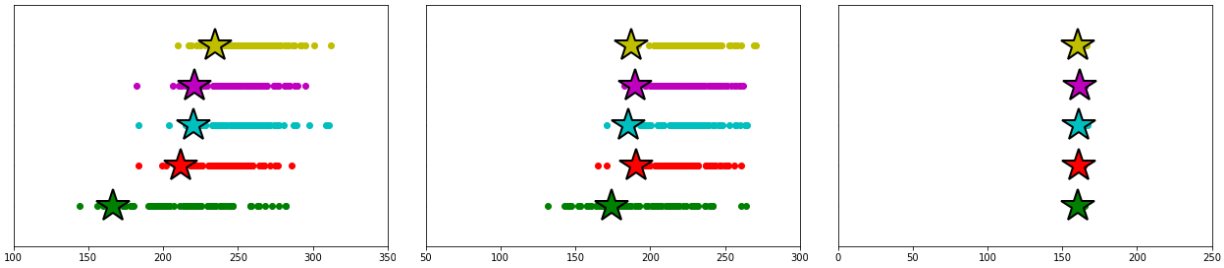


Fig. 4. The norm of embeddings produced by the PN (left), PN trained with the Ring Loss (middle), and the SEN PN (right). The stars denote query examples, and the diamonds denotes prototypes.

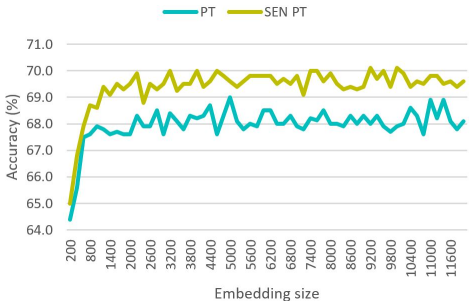


Fig. 5. The PT vs the SEN PT with different embedding sizes.

section, we study the behaviors of the proposed SEN dissimilarity measure in dimensionality reduction and deep clustering via a toy example. To do this, we implement the well-known Siamese network and Contrastive loss [23]; we call this model Siamese Baseline. We augment the Siamese Baseline by replacing the Euclidean distance by our proposed SEN dissimilarity measure (Siamese SEN) and by employing Ring loss (Siamese Ring). We train the three models on the MNIST dataset [34] for dimensionality reduction and clustering. During training the Siamese SEN, a positive epsilon ($\epsilon_{ik} = \epsilon_p > 0$) is used for computing the SEN dissimilarity measures between examples of the same class, and a negative epsilon ($\epsilon_{ik} = \epsilon_n < 0$) is used for computing the SEN dissimilarity measures between examples of different classes. At test time, a positive epsilon ($\epsilon_{ik} = \epsilon_p > 0$) is used for computing all dissimilarity measures.

As can be seen from Fig. 6, the Siamese Ring forces all embeddings to lie on a scaled unit hypersphere; however, embeddings produced by the Siamese Ring are not as well-separated as embeddings produced by the Siamese Baseline. Our proposed SEN dissimilarity measure, on the other hand, both forces all embeddings to lie on a scaled unit hypersphere and keeps the embeddings well-separated. This suggests that our proposed SEN dissimilarity measure is an effective feature normalization technique not only for distance metric learning-based few-shot learning with prototypical networks but also potentially for dimensionality reduction and deep clustering.

A natural direction for future work is to investigate the possibility of employing the proposed SEN dissimilarity measure for improving Euclidean distance-based methods in dimensionality reduction and in deep clustering.

V. Conclusion

In this paper, we propose a novel dissimilarity measure, called SEN, for distance metric learning-based few-shot learning by modifying the traditional Euclidean distance to attenuate the curse of dimensionality in high dimensional spaces. The proposed SEN dissimilarity measure is a combination of the Euclidean distance and the norm distance. We extend the powerful prototypical network by replacing the Euclidean distance by our proposed SEN dissimilarity measure, which we refer to as SEN PN. With minimal modifications, the SEN PN outperforms the original PN by a considerable margin and demonstrates good performance on the miniImageNet dataset with no additional parameters as well as almost no additional computational overhead. We provide analyses showing that the proposed SEN dissimilarity measure encourages the embeddings to have the same norm and enables the SEN PN to generate a hyperspherical embedding space, which is a more compact embedding space than the Euclidean space.

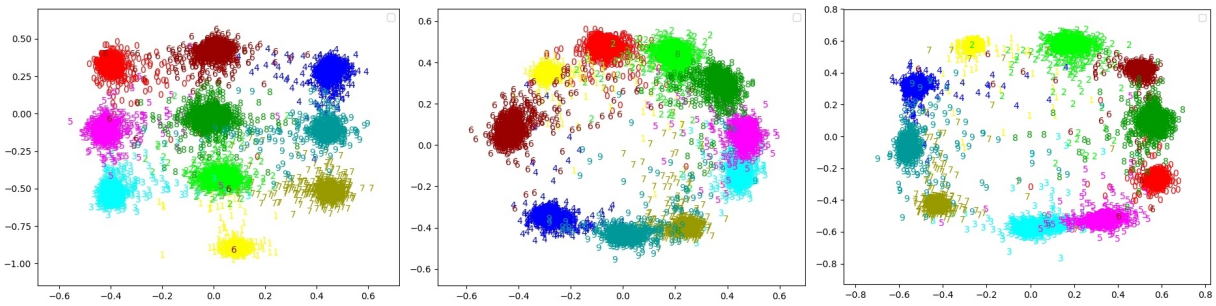


Fig. 6. 2D embeddings produced by the Siamese Baseline (left), the Siamese Ring (middle), and the Siamese SEN (right).

The hyperspherical embedding enables the classification to be performed via either the SEN dissimilarity measure or the cosine distance. We experimentally show that the proposed SEN dissimilarity measure consistently outperforms the Euclidean distance in the PT with different embedding sizes as well as with different embedding networks.

ACKNOWLEDGMENT

The authors would like to thank eSmart Systems and UiT Machine Learning Group for support in the work with this paper. This work was supported by the Research Council of Norway [RCN NÆRINGSPHD grant no. 263894 (2016-2018) on Power Grid Image Analysis] and eSmart Systems.

REFERENCES

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. IEEE, 2009, pp. 248–255.

[2] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[3] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*. JMLR. org, 2015, pp. 448–456.

[4] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[5] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Curran Associates Inc., 2017, pp. 4080–4090.

[6] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*. Curran Associates Inc., 2016, pp. 3637–3645.

[7] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, 2018, pp. 1199–1208.

[8] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the surprising behavior of distance metrics in high dimensional space," in *International conference on database theory*. Springer, 2001, pp. 420–434.

[9] Y. Zheng, D. K. Pal, and M. Savvides, "Ring loss: Convex feature normalization for face recognition," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, 2018, pp. 5089–5097.

[10] Y. Wang and Q. Yao, "Few-shot learning: A survey," *CoRR*, vol. abs/1904.05046, 2019. [Online]. Available: http://arxiv.org/abs/1904.05046

[11] J. Vanschoren, "Meta-learning: A survey," *CoRR*, vol. abs/1810.03548, 2018. [Online]. Available: http://arxiv.org/abs/1810.03548

[12] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1126–1135.

[13] R. Sachin and L. Hugo, "Optimization as a model for few-shot learning," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017. [Online]. Available: https://openreview.net/forum?id=rJY0-Kcll

[14] S. Adam, B. Sergey, B. Matthew, W. Daan, and P. L. Timothy, "One-shot learning with memory-augmented neural networks," *CoRR*, vol. abs/1605.06065, 2016. [Online]. Available: http://arxiv.org/abs/1605.06065

[15] G. Alex, W. Greg, and D. Ivo, "Neural turing machines," *CoRR*, vol. abs/1410.5401, 2014. [Online]. Available: http://arxiv.org/abs/1410.5401

[16] M. Nikhil, R. Mostafa, C. Xi, and A. Pieter, "A simple neural attentive meta-learner," *CoRR*, vol. abs/1707.03141, 2017. [Online]. Available: http://arxiv.org/abs/1707.03141

[17] R. Zhang, T. Che, Z. Ghahramani, Y. Bengio, and Y. Song, "Metagan: an adversarial approach to few-

shot learning," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. Curran Associates Inc., 2018, pp. 2371–2380.

[18] Y.-X. Wang, R. Girshick, M. Hebert, and B. Hariharan, "Low-shot learning from imaginary data," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018, pp. 7278–7286.

[19] S. Thrun, "Lifelong learning algorithms," in *Learning to learn*. Springer, 1998, pp. 181–209.

[20] G. Victor and B. Joan, "Few-shot learning with graph neural networks," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=BJj6qGbRW

[21] W. Yong, W. Xiao-Ming, L. Qimai, G. Jiatao, X. Wangmeng, Z. Lei, and O. K. L. Victor, "Large margin few-shot learning," *CoRR*, vol. abs/1807.02872, 2018. [Online]. Available: http://arxiv.org/abs/1807.02872

[22] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, 2015, pp. 815–823. [Online]. Available: https://doi.org/10.1109/CVPR.2015.7298682

[23] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2. IEEE, 2006, pp. 1735–1742.

[24] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Proceedings of the 27th International Conference on Neural Information Processing Systems-Volume 2*. MIT Press, 2014, pp. 1988–1996.

[25] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille, "Normface: L2 hypersphere embedding for face verification," in *Proceedings of the 25th ACM international conference on Multimedia*. ACM, 2017, pp. 1041–1049.

[26] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, "Cosface: Large margin cosine loss for deep face recognition," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018, pp. 5265–5274.

[27] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4690–4699.

[28] R. Boney and A. Ilin, "Semi-supervised few-shot learning with prototypical networks," *CoRR*, vol. abs/1711.10856, 2017. [Online]. Available: http://arxiv.org/abs/1711.10856

[29] M. Ren, E. Triantafillou, S. Ravi, J. Snell, K. Swersky, J. B. Tenenbaum, H. Larochelle, and R. S. Zemel, "Meta-learning for semi-supervised few-shot classification," *CoRR*, vol. abs/1803.00676, 2018. [Online]. Available: http://arxiv.org/abs/1803.00676

[30] X. Zhe, S. Chen, and H. Yan, "Directional statistics-based deep metric learning for image classification and retrieval," *Pattern Recognition*, vol. 93, pp. 113–123, 2019.

[31] I. S. Dhillon, J. Fan, and Y. Guan, "Efficient clustering of very large document collections," in *Data mining for scientific and engineering applications*. Springer, 2001, pp. 357–381.

[32] A. Banerjee, I. S. Dhillon, J. Ghosh, and S. Sra, "Clustering on the unit hypersphere using von mises-fisher distributions," *The Journal of Machine Learning Research*, vol. 6, pp. 1345–1382, 2005.

[33] S. Zagoruyko and N. Komodakis, "Wide residual networks," *CoRR*, vol. abs/1605.07146, 2016. [Online]. Available: http://arxiv.org/abs/1605.07146

[34] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/