UiT
THE ARCTIC
UNIVERSITY
OF NORWAY

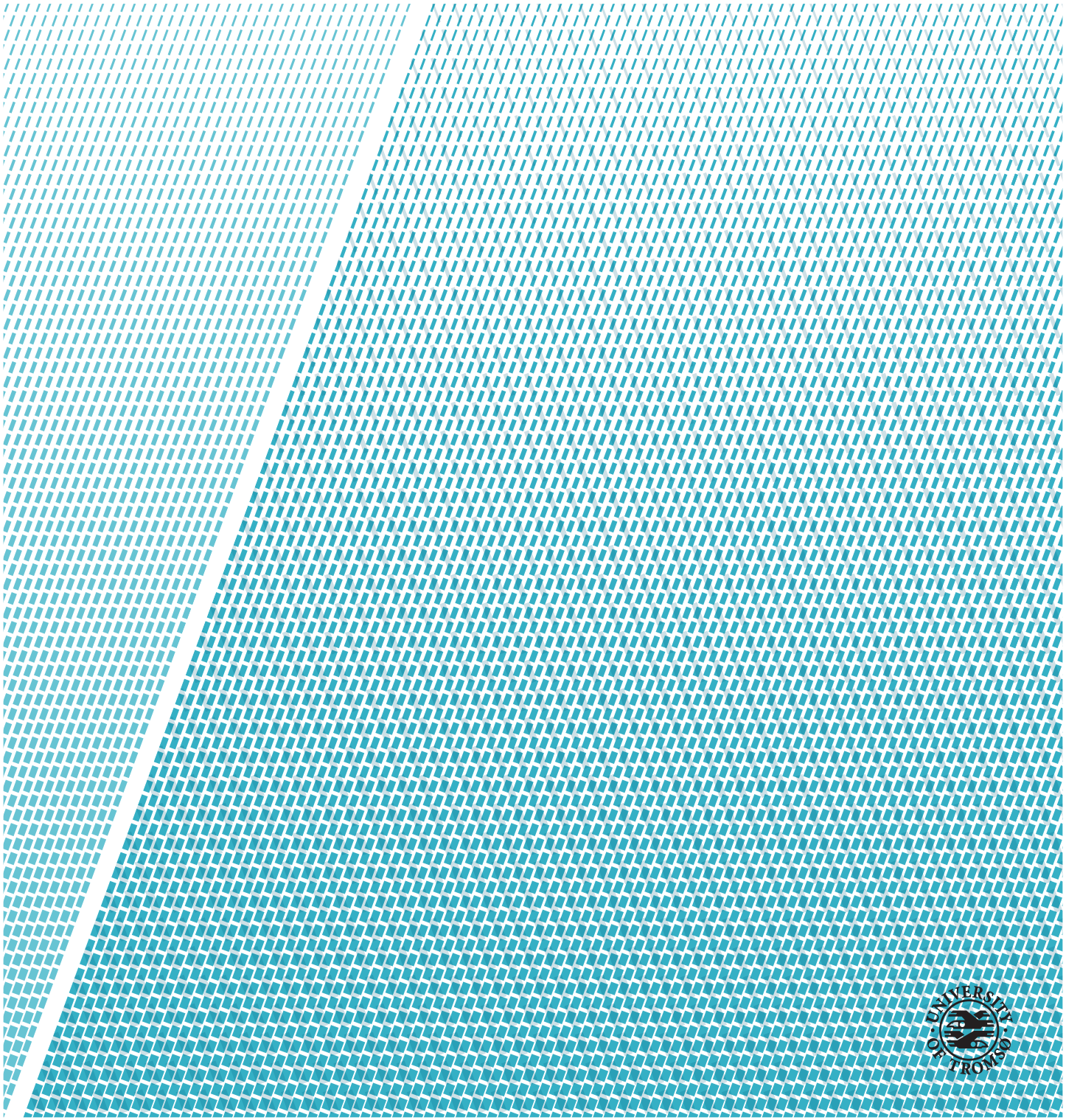
Faculty of Science and Technology
Department of Computer Science

Deep Learning: From Data Extraction to Large-Scale Analysis

—

Mike Voets

INF-3990 Master's thesis in Computer Science May 2018



稲妻の　腕を借らん　草枕

ヘンドリック・ドゥーフ

“Lend me your arms, fast as thunderbolts, for a pillow on my journey.”
–Hendrik Doeff (1764–1837)

Abstract

We aim to give an insight into aspects of developing and deploying a deep learning algorithm to automate biomedical image analyses. We anonymize sensitive data from a medical archive system, attempt to replicate and further improve published methods, and scale out our algorithm to support large-scale analyses. Specifically, our contributions are described as follows.

First, to anonymize and extract mammograms for the development of a breast cancer detection algorithm, we wrote a script for mammograms that reside in a data-locking, sensitive, and proprietary PACS. The script will be used in a larger project to extract mammograms from all screening points in Norway.

Second, because this script is currently being authorized by Helsenord IKT, we instead developed an algorithm for a similar screening problem in the biomedical field. In order not to reinvent the wheel, we investigated earlier work. The high-impact article JAMA 2016; 316(22)[1] describes a high performance deep learning algorithm that detects diabetic retinopathy, reporting a receiver operating characteristic curve (AUC) of 0.99. We attempted to replicate the method. Our AUC of 0.74 and 0.59 did however not reach the reported results, possibly by differences in data, or by missing details in the methodology.

Third, by modifying the data preprocessing methods in the diabetic retinopathy algorithm slightly, the AUC increased to 0.94 and 0.82. These findings emphasize the challenges of replicating deep learning methods that have their source code not published, and do not use publicly available data.

Fourth, benchmarks were run to assess the resources needed to run algorithm development and automated analyses on a national (Norwegian) scale. We estimate that a breast cancer detection algorithm can be trained on 4 GPUs in less than 17 hours, with a sublinear speed-up of 3.36 times compared to 1 GPU. Evaluation with inexpensive GPUs has been shown to perform instantly.

Lastly, with our experiences and lessons learned in mind, we conclude with literature suggestions and recommendations to develop and to deploy an algorithm for breast cancer detection in a large-scale screening program.

Acknowledgements

I would like to express my deep gratitude to Lars Ailo Bongo and Kajsa Møllersen, my supervisors, for their patient guidance, encouragement, useful critiques, and advice for this research work despite occasionally being in different time zones. Their willingness to give their time so generously has been greatly appreciated. My grateful thanks are extended to my co-supervisor Einar Holsbø for his valuable statistical insights.

I would also like to extend my appreciation to Jon Ivar Kristiansen at my department, Gurvinder Singh at UNINETT, Johan Ravn at Medsensio, and Kolbjørn Engeseth at Jupiter System Partner for their help in offering me resources for this research work.

Finally, I wish to acknowledge that my journey would not have been possible without the support of my dear colleagues Are and Cathrin, and friends Helge, Nina, and Sonja.

Contents

List of Figures

List of Tables

1

Introduction

Over the last years, deep learning has emerged as a popular set of machine learning methods based on learning data representations. It has been shown that deep learning algorithms are able to beat state-of-the-art approaches in traditional machine learning problems such as image and sound classification, and it has been stated that they may surpass human-level capabilities in classifying these kinds of data. The remarkable progress in deep learning has been a result of three main factors. First, the collection of massive amounts of data. Second, the development and accessibility of new machine learning frameworks and platforms [2, 3, 4, 5] and algorithms [6, 7, 8, 9] due to advances in parallel [3, 10, 11] and scalable software systems [12, 13, 14]. Third, storage costs have been rapidly decreasing [15, 16], and mobile applications, internet of things (IoT), and the importance of data as a resource [17], have all led to further investments in research and development of deep learning technologies [18].

Deep learning is learning data representations by using a network of multiple layers of nonlinear processing units for various kinds of feature extraction and transformation. Each layer's output is the successive layer's input. Most deep learning models and methods attempt to mimic the activity in layers of neurons in the neocortex, i.e. an artificial neural network. It learns, like other machine learning methods, by iteratively classifying a training data set, and updating its parameters slightly into the right direction every time a classification error occurs. Ultimately, the fine-tuned parameters of the algorithm are tested on an evaluation data set to measure the algorithm's performance. Such an algorithm

facilitates the automatic classification of data, which, when deployed, removes the need for a person to classify the data manually.

The main objective when developing a deep learning algorithm is for it to be fundamental or general enough, such that it performs well on unseen data. Although this seems simple, finding the optimal fine-tuned parameters of the algorithm is a task that usually involves many trial and error attempts. Instead of developing an algorithm from scratch, there are dozens of optimization techniques that accelerate convergence [19]. However, the algorithm's performance is ultimately determined by the input data. Retrieving data from public sources is often insufficient, while private data is hard and costly to obtain due to legal regulations and data *lock-in* [20, 21]. Furthermore, even though other studies may have stated high performance for their algorithm, it has been shown that many studies are non-replicable [22, 23, 24], all emphasizing the obstacles that are associated with developing a deep learning algorithm.

We aim to give an insight into various aspects of developing and deploying a deep learning algorithm to automate biomedical image analyses. We anonymize sensitive data from a medical archive system, attempt to replicate and further improve published methods, and scale out our algorithm to support large-scale analyses. Specifically, our contributions on various aspects of developing and deploying a deep learning algorithm are as follows:

1. We wrote an anonymization script for mammograms that reside in a data-locking, sensitive and proprietary PACS archive system.

2. We developed an algorithm that detects diabetic retinopathy, by attempting to replicate the main method of a highly-cited study that was published in JAMA 2016; 316(22)[1]. The study was non-replicable. Our algorithm had an area under the receiver operating characteristic curve (AUC) of 0.74 and 0.59 for two test sets, compared to 0.99 for both test sets in the original study.

3. We increased the algorithm's AUC to 0.94 and 0.82 by introducing some improvements.

4. We evaluated running the algorithm on 1 and 2 NVIDIA GeForce GTX 1080 (Ti) GPUs, compared to running on the multi-GPU *UNINETT DaaS Cluster*, which consists of 4 NVIDIA Titan X Pascal GPUs in 2 worker nodes.

Thesis Statement Although applying a deep learning method appears to be simple, the developer must overcome the limitations of data lock-in and non-replicability of earlier described algorithms within the same domain.

1.1 Three Challenges of Applying Deep Learning

1.1.1 Data Retrieval

With the introduction of competition platforms for machine learning like *Kaggle* [25], data sets are made publicly available for any researcher to analyze data and develop deep learning algorithms. Similarly, other institutions have also published data sets [26, 27, 28, 29, 30]. However, these data sets may have limited quality, or they may not contain sufficient data for developing a deep learning algorithm. Data may for example lack the right labels or grades for classification, or the databases are too limited in size. Besides, public databases are generally more meant as benchmarks for algorithm performance evaluation rather than being used for algorithm training. When there is no (sufficient) public data available, data needs to be retrieved from a private database or archive system.

Many private databases and archive systems are proprietary and do not allow out-of-the-box data exportation or portability. Data or *vendor* lock-in [20, 21] in databases and archive systems hinders developers getting data necessary for developing a deep learning algorithm. Furthermore, databases or archive systems often contain sensitive information that connects with a person's identity. Due to legal regulations, there is often a lengthy application process involved to obtain access to such data. Moreover, the data usually requires anonymization or de-identification before extraction.

1.1.2 Algorithm Development

Along with the increasing amount of investments in research and development of deep learning technologies in the biomedical industry [31, 32, 33, 34, 35], there is an emerging need to automate classification in image and sound classification tasks. To develop a deep learning algorithm that automates these tasks, an *artificial neural network* (ANN) model needs to be designed. It is possible to design a model from scratch, however it requires extensive domain knowledge, and it will take many trial-and-error attempts, or just luck, to design a model that provides a solid base for developing an algorithm with high performance. In other words, there is no generalized solution, even for a given domain. There are practical guidelines on how to design a model architecture, for example for image recognition [36], however usually without references to academic literature. Therefore it is more practical to use a predesigned model. Predesigned models of *convolutional neural networks* (CNNs), like *InceptionV3* [37] or *ResNet-50* [38], have been shown to develop algorithms with high performance for various domains. Such an algorithm will still need to be verified on large amounts of data before it can be used in a clinical setting.

There have been published numerous high-impact articles stating that their proposed deep learning algorithm reach near human-level performance [39, 40, 41]. Nevertheless, there has been raised a shared concern across the biomedical industry that many studies cannot be verified [22, 23, 24, 42], due to insufficient or inaccurate reporting of methodologies [43]. Being able to verify a study by replication, i.e. strictly following the described methods, is essential for the development of medical technologies based on published results [44]. There is also a general lack of funding to support replication research, and when methods have been replicated, the replication results are rarely published [45]. Furthermore, it is often impossible for researchers with low budgets to verify high-impact studies. High-impact studies like [46, 47] receive large funding, and can thus afford to use non-public data and a team of experts for data quality assurance and professional labeling or grading. Ideally, studies should publish their source code and data, so that other researchers can verify the results for their own data. However, this is not always possible, for example for sensitive data, or for methods with commercial value [22, 48].

1.1.3 Algorithm Deployment

After developing and evaluating a deep learning algorithm, the next step is to make it available for others. Deep learning algorithms can be deployed in various ways. The most commonly used deep learning deployment model is a web service (for example a RESTful API). The web service takes in some data and yields a prediction immediately. To improve the deep learning algorithm, the web service can also keep track of the consumed data, and use the data for periodical automatic re-training of the algorithm. The resource usage of deploying a deep learning algorithm as a web service, or in fact the deployment phase in general, has not been described well in literature.

1.2 Objective and Approaches

This thesis is the offspring of a pilot project with its goal to enable automated analyses for BreastScreen Norway[1] [49], with two main objectives: the development of a deep learning algorithm for breast cancer detection, and enabling real-time analysis by deploying the algorithm. However, we have not yet received the anonymized mammograms, hence we were unable to develop an algorithm for breast cancer detection. Our approaches are described as follows.

1. BreastScreen Norway was earlier called the Norwegian Breast Cancer Screening Programme (NBCSP).

First, we wrote an extraction and anonymization script for mammograms residing in a *Picture archiving and communication system* (PACS) at the University Hospital of North Norway (UNN), which provides storage and access to various images types in the medical field. The mammograms were gathered in BreastScreen Norway orchestrated by the Cancer Registry of Norway [50]. Our script anonymizes sensitive meta-data in the form of personal data that reside in and around the mammogram files for two reasons. First, keeping personally identifiable information is generally constrained by strict legal regulations. Second, keeping personal information in the data is generally not necessary for developing a deep learning algorithm.

Second, we gained insights into developing a deep learning algorithm. Since we have not yet received the anonymized mammograms from the PACS system, we developed an algorithm for a similar screening problem in the biomedical field: diabetic retinopathy. Diabetic retinopathy is an eye disease that people with diabetes can develop. The high blood sugar levels due to diabetes cause damage to blood vessels in the retina, causing them to swell, leak, or close, stopping blood from passing through. These conditions can ultimately lead to blindness [51]. The article *Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs*, published in JAMA in 2016 [1], reported a deep learning algorithm for detecting diabetic retinopathy with an area under the receiver operating characteristic curve of 0.99. This high-impact article has been cited many times since, and has consequently become a well-known study in the biomedical field. The authors did however not publish their source code. This is why we attempted to replicate the proposed method for developing a high-performance deep learning algorithm for diabetic retinopathy detection. We further improved our replica algorithm by modifying the original method.

Third, to assess the resource usage of developing and real-time analysis of a biomedical deep learning algorithm on a national (Norwegian) scale, we measured training and analysis speed of our diabetic retinopathy detection algorithm on 1 and 2 NVIDIA GeForce GTX 1080 (Ti) GPUs, compared to the multi-GPU *UNINETT DaaS Cluster*, which consists of 4 NVIDIA Titan X Pascal GPU in 2 worker nodes.

1.3 Summary of Results

The anonymization script is currently being authorized and will be used to extract mammograms from UNN. The source code for the anonymization script is available at: `https://github.com/mikevoets/dicom_anonymizer`.

We were not able to replicate the JAMA 2016; 316(22) study. Our algorithm's area under the receiver operating characteristic curve (AUC) of 0.74 and 0.59 on two independent test sets did not come close to the reported AUC of 0.99 in the original study (see Figure 1.1a). This may be caused by the use of a single grade per image, or different hyper-parameter settings. By changing the preprocessing methods, our replica algorithm's AUC increased to 0.94 and 0.82, respectively (see Figure 1.1b). Our replication attempt shows the challenges of replicating deep learning, and the need for more replication studies to validate deep learning methods, especially for medical image analyses.

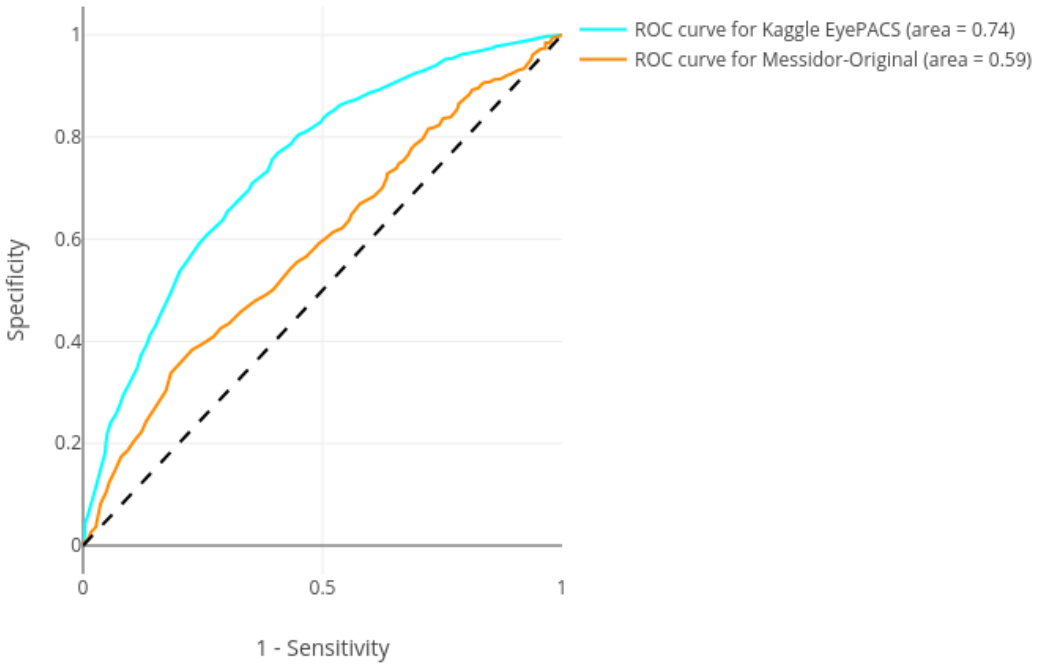
Our source code and instructions for our replication are available at: `https://github.com/mikevoets/jama16-retina-replication`. This repository has gained a significant interest. As of May 2018, the repository had been forked 9 times and gathered 16 stars. We also archived this work as a stand-alone article on *arXiv* [52], gained feedback from the deep learning community on Twitter and Facebook, and submitted it to JAMA Network Open.

Environment	**Max. training speed (images/sec)**	**Max. prediction speed (images/sec)**
Deep1 *(1x NVIDIA GeForce GTX 1080)*	91.1 ± 0.36	328 ± 0.53
UNINETT DaaS Cluster *(4x NVIDIA Titan X Pascal)*	461 ± 5.6	868 ± 2.4[a]

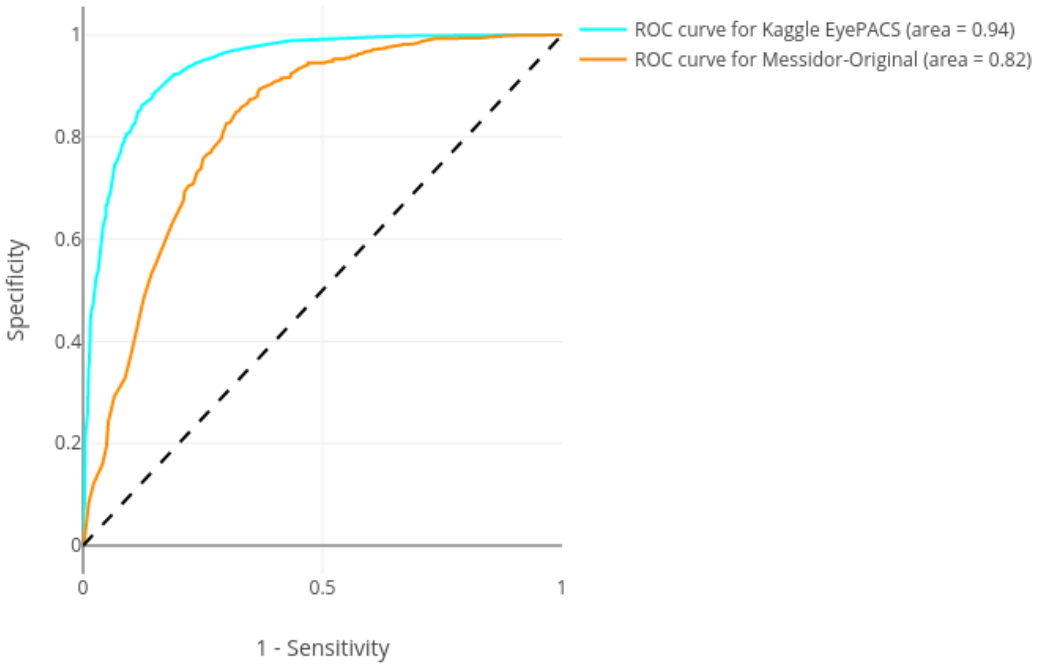
[a] This result was achieved with 2 NVIDIA Titan X Pascal GPUs.

Table 1.1: Training and prediction speed in retinal fundus images per second with one NVIDIA GeForce GTX 1080 compared to 4 NVIDIA Titan X Pascal GPUs.

In Table 1.1, we show the best results of measuring training and evaluation (prediction) speed in images per second, in a minimal single-GPU environment, compared to the *UNINETT DaaS cluster* with 4 NVIDIA Titan X Pascal GPUs. Instructions for running the benchmarks can be found in the replication repository.



(a) JAMA 2016; 316(22) replica algorithm.



(b) Improved algorithm.

Figure 1.1: Area under receiver operating characteristic curve (AUC) for the replica algorithm (a) and with improved preprocessing methods (b) trained with using only gradable retinal fundus images and stochastic gradient descent.

1.4 Thesis Structure

The rest of this work is covered as follows. Chapter 2 covers the implementation of the anonymization script for extracting mammograms from a PACS system. Chapter 3 covers the JAMA 2016; 316(22) replication and development of the deep learning algorithm for diabetic retinopathy, and the improvements we introduced. Chapter 4 measures data throughput during algorithm training, and analysis performance, by comparing the algorithm in three environments. Chapter 5 then discusses the limitations and extensions of our findings, and finally, we conclude and discuss future work.

2

Data Retrieval from PACS

2.1 Introduction

We implemented an extraction and anonymization script for mammograms in Norwegian hospitals for BreastScreen Norway. From a practical point of view, it is easiest to execute the script on the hospital ICT infrastructure and to let the script consume a list of identifiers from the Cancer Registry of Norway [50] (from here referred to as the Cancer Registry). We use the script in a pilot project to extract anonymized data from the University Hospital of North Norway (UNN). We collaborate with Helse Nord IKT. Helse Nord IKT operates the ICT infrastructure of the North Norwegian health region. The project is part of a larger ICT research project at UiT - The Arctic University of Norway aiming to develop the needed infrastructure for integrated analysis of medical data.

The mammography data set includes mammograms taken in the period from 2012 until 2018, and excludes people who have opted-out of their images being used for research. Estimations are that this part of the data set contains 280 000 images from 70 000 screenings. The mammograms reside in a *Sectra* PACS system. This system does not provide an API for extraction or anonymization of files, making the objective of our script to extract mammograms from the part of the file system used by the PACS. The mammograms are *DICOM* files. They contain personal meta-data, requiring the script to anonymize these files since there is no explicit need and allowance to use person-related data to develop a deep learning algorithm. The Cancer Registry has provided a variable

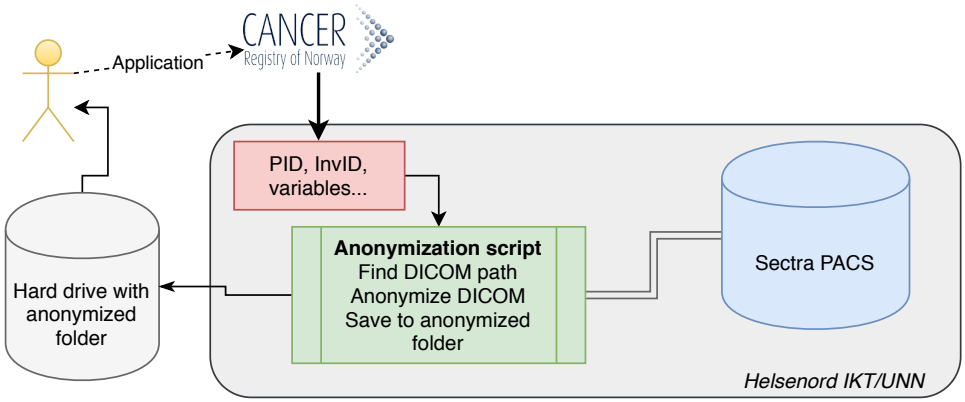


Figure 2.1: Overview of the mammograms anonymization process.

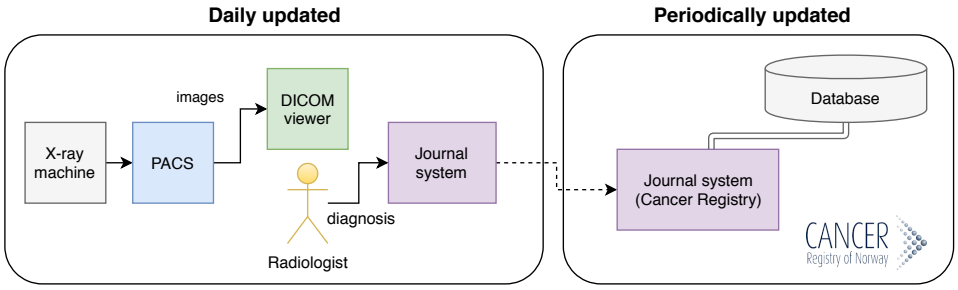
specification for additional meta-data associated with each mammogram that are not part of the DICOM file, but are essential to develop a deep learning algorithm. Because these meta-data contain personal data as well, the script also anonymizes these meta-data. The script further assures that the anonymized data cannot be linked back to the original personal data.

Retrieving the variable specification and confirmation on being able to extract mammograms is an ongoing part of a lengthy application process. We do not gain access to the PACS system to extract mammograms directly, instead we wrote a prepared script that Helse Nord IKT will verify and finish before executing it in the *Sectra* PACS environment at UNN (see Figure 2.1). Ultimately, the script will be used in the larger ICT project to extract mammograms from all screening points in Norway.

2.1.1 Breast Cancer Screening

Figure 2.2 shows the process of breast cancer screening until periodical import into the Cancer Registry. Every business day in the North Norwegian region, about 65 people attend mammography screening at UNN, and an additional 65 people are screened remotely. The Cancer Registry imports data from UNN periodically, typically once a year. The Cancer Registry imported data from 435 000 people who were screened for breast cancer in the screening rounds of the year 2014-2015.

Figure 2.2: Visualization of breast cancer screening and periodic retrieval by the Cancer Registry.



2.2 Implementation

The anonymization script was written in Python 2.7, and can be used from the command line. The script accepts several parameters. First, the root directory path where the DICOM files containing digital mammograms and meta-data reside. Second, the path to the *csv* file with meta-data from the Cancer Registry. Third, the directory path where anonymized DICOM files should be placed to. Fourth, the path where the anonymized variables (cleaned file) from the Cancer Registry should be placed to.

2.2.1 Anonymization of DICOM files

DICOM, *Digital Imaging and Communications in Medicine*, is a standard digital file format for medical images [53, 54]. These files contain raw image data and other meta-data related to the image. This meta-data usually consists of personal information, information about the owner of the image, and information about for what purpose, when, and with what equipment the image was taken. To anonymize the personal data in the DICOM files, the script uses the *dicom-anon* Python tool [55]. Dicom-anon has been implemented by the Children's Hospital of Philadelphia (CHOP) [56]. See Table 2.2 in Section 2.2.3 for an overview on how DICOM files in our script are anonymized. Dicom-anon attempts to be compliant with the Basic Application Confidentiality Profile as specified in DICOM 3.15 Annex E document [57]. These specifications define what values in the meta-data should be anonymized based on their modality. Modality represents the DICOM file type. For mammography, the modality is *mg*. Dicom-anon further removes all attributes from the DICOM file that are not specified in Annex E. The tool creates a *sqlite3* database file with a table containing the original and cleaned version of every attribute. This file will be removed after running our anonymization script.

2.2.2 Anonymization of Cancer Registry Meta-Data

The scripts accepts a *csv* file with a list of variables from the Cancer Registry. See Table 2.1 in Section 2.2.3 for an overview on how the values from this file are anonymized and placed to a cleaned file. The first two variables per line represent the personal identifier (PID), and the invitation or screening identifier (InvID), respectively. PID can be linked to many InvIDs. The third value represents the screening date. The seventh value represents the diagnosis date. Other examples of values in the file are annotations for ground truth, and where the image was taken. These other values do not need to be modified, because they cannot be linked with a person's identity. The PID and InvID are not included in the anonymized meta-data file, but are used to identify the association between the person and screenings later. The screening date and diagnosis date are originally formatted as *15.mmm.yyyy*. The anonymization script re-formats the screening date to *m-yyyy*, and the diagnosis date is converted to the amount of days after the screening date. The screening date is used for the new directory structure of the anonymized DICOM files per person.

2.2.3 Anonymity Assurance Test

The script provides a test to assure that the resulting anonymized mammograms and meta-data cannot be linked back to their original personal information. To facilitate this test, we provided a couple of example DICOM files with fake personal information, together with an example *csv* file that represents the meta-data file from the Cancer Registry. The folder structure for the example DICOM files before running the test is shown in Figure 2.3a. The test can be run by specifying the *-t* flag when executing the script. After running the script in test mode, the *tests* folder is modified as shown in Figure 2.3b. The result of anonymizing a file representing the file from the Cancer Registry is shown in Table 2.1, and the result of anonymizing DICOM files is shown in Table 2.2.

2.3 Discussion

To implement the anonymization script, we had to make some assumptions. First, we assume that one PID can be associated with many InvIDs, because a person may be screened for breast cancer multiple times. Second, we assume that the variables in the *csv* file from the Cancer Registry are delimited by white spaces. The delimiter can however be changed in the script. Third, we assume that the variables are delimited in the same order as the variables in the

variable specification received from the Cancer Registry: i.e. the first two values in the line should be *PID* and *InvID*, the third value *O2_Bildetakningsdato*, and the seventh value *Diagnosedato*.

2.3.1 Limitations

DICOM files that are explicitly marked as containing burnt-in data along with files that have a series description of *Patient Protocol*, will be copied to a *quarantine* folder, and cannot be anonymized by our script.

We did not know any details regarding the internal folder structure of the PACS system. Because of this, we have not implemented the method for retrieving the internal path to a specific DICOM file given a PID and InvID. Before executing this script in the *Sectra* PACS environment at UNN, Helse Nord IKT is required to verify the script and implement the method to find the internal DICOM paths for all screened people.

Original values *(in variables.csv)*

PID	**InvID**	**O2_Bildetakingsdato** *Screening date*	**...**	**Diagnosedato** *Diagnosis date*
Example_Patient	Screening_1	15.Jan.2016	...	15.Feb.2016
Example_Patient	Screening_2	15.Dec.2017	...	15.Jan.2018

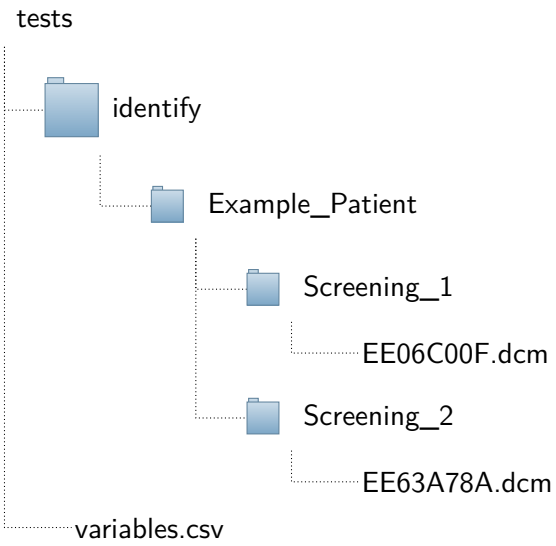
Anonymized values *(in cleaned_variables.csv)*

Anonymized PID	**Screening date**	**...**	**Diagnosis** *Days offset*
e3b23d103c4342...	12-2017	...	31
e3b23d103c4342...	1-2016	...	31

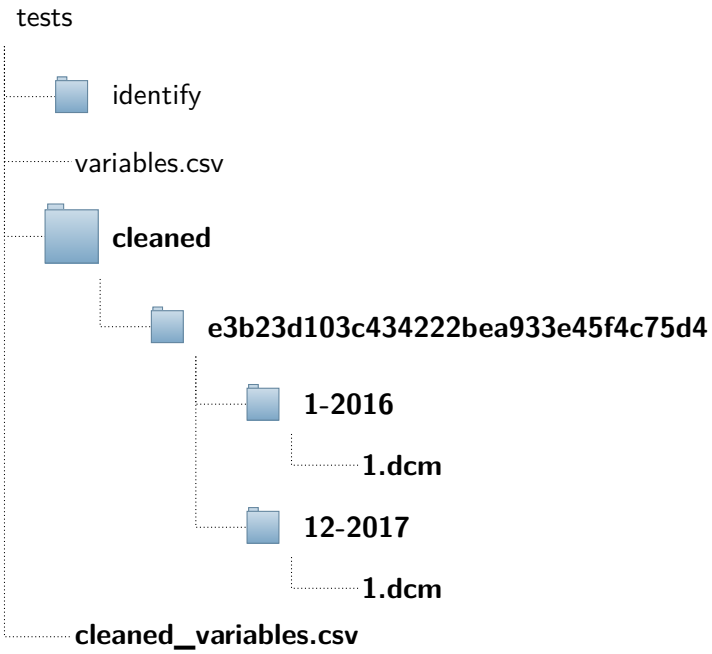
Table 2.1: Overview of anonymization of values of the Cancer Registry. The personal identifier *PID* is anonymized by assigning a pseudo-randomized UUID. The screening identifier *InvID* is removed. Instead, the screening date is used and formatted to *m-yyyy*. Note that this corresponds to the anonymized folder structure in Figure 2.3b. The diagnosis date is converted to the offset in days relative from the screening date. All other variables in the file are unchanged (not shown in this table).

DICOM Attribute Name	**Value in *Screening_1/EE06C00F.dcm***	**Value in *1-2016/1.dcm***
Specific Character Set	'ISO_IR 100'	<removed attribute>
Image Type	['ORIGINAL', 'PRIMARY', '']	['ORIGINAL', 'PRIMARY', '']
Study Date	'20140408'	'19010101'
Content Date	'20140408'	'19010101'
Study Time	'104011'	'000000.00'
Content Time	'104117.000000'	'000000.00'
Accession Number	'R9BF8PC1GE'	'Accession Number 1'
Patient ID	'R9BF8PC1GE'	'Patient ID 1'
[Examination Number]	'E9BF8PC1GE'	<removed attribute>
Patient Name	'Anonymous Female 1959'	"Patient's Name 1"
Patient's Birth Date	'19591221'	'19010101'
Patient's Sex	'F'	'CLEANED'
Patient's Birth Name	'anonymous'	<removed attribute>
Patient's Age	'054Y'	<removed attribute>
Patient's Mother's Birth Name	'anonymous'	<removed attribute>
Medical Alerts	'anonymous'	<removed attribute>
Allergies	'anonymous'	<removed attribute>
	...	
Study ID	'E9BF8PC1GE'	'CLEANED'
Patient Identity Removed	<non-existent>	'YES'
	...	
KVP	'30'	'30'
Distance Source to Detector	'660'	'660'
Distance Source to Patient	'660'	'660'
Estimated Radiographic Magnification	'1'	'1'
Field of View Dimension(s)	['306', '239']	['306', '239']
Exposure Time	'785'	'785'
X-Ray Tube Current	'62'	'62'
Exposure	'49'	'49'
Expore in uAs	'48800'	'48800'
	...	
Pixel Data	Array of 14660856 bytes	Array of 14660856 bytes

Table 2.2: Overview of DICOM meta-data anonymization. Files are anonymized by the *dicom-anon* tool [55]. For this example we used the attribute values of *EE06C00F.dcm* and its anonymized variant *1.dcm*. All person-related meta-data are anonymized by assigning a sequence. All dates are reset to 1901-01-01. *Patient's Sex* and *Study ID* attributes are cleaned. Optional or unrecognized attributes are removed. A new attribute *Patient Identity Removed* is added to the anonymized DICOM file. The actual image represented by *Pixel Data* stays unchanged. We do not show DICOM tag and VR in this table, as they do not provide additional relevant information about the anonymization procedure.



(a) Before executing test. The personal folders reside in the *identify* folder (may be PID), each consisting of one or more screening (may be InvID) folders consisting of DICOM files. The *variables.csv* file represents a possible *csv* file with example variables from the Cancer Registry.



(b) After having executed test. The original *identify* folder structure still exists with its original content, but a new folder *cleaned* has been created. This folder consists of the anonymized data. It consists of folders named with pseudo-randomly generated UUIDs, representing people. Each folder consists of one or more screening folders named with the screening's date formatted by *m-yyyy*, found among the Cancer Registry variables, with one or more renamed anonymized DICOM files for the corresponding screening. The cleaned variables from the Cancer Registry are placed in *cleaned_variables.csv*.

Figure 2.3: Folder structure for the DICOM anonymizer script before and after test run.

2.3.2 Related Work

The Digital Mammography DREAM challenge [58] was a machine learning competition held in 2016, as an attempt to find a machine learning algorithm that improves the predictive accuracy of digital mammography for the early detection of breast cancer, with its main focus on reducing the recall rate for breast cancer screening. It provided a data set consisting of 640 000 de-identified mammograms from 86 000 people with corresponding personal characteristics and outcome measures. This shows that large amounts of annotated data are needed to develop and evaluate a deep learning algorithm, and confirms that de-identification or anonymization of the digital mammograms is necessary.

2.3.3 Conclusion

When the script has been successfully executed, the folder with the anonymized data can be transferred from the sensitive data environment, and the resulting anonymized data can then be used to develop a deep learning algorithm.

3

Replication and Improvement of a High-Impact Study

3.1 Introduction

For this thesis, we make an assessment on the replicability of a deep learning method. We have chosen to attempt to replicate the main method from *Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs*, published in JAMA 2016; 316(22)[1]. As of May 2018, this article had been cited 370 times [59]. We chose to replicate this study because it is a well-known and high-impact study in the medical field, the source code has not been published, and there are as far as we know not any others who have attempted to replicate this study.

The original study describes an algorithm (hereby referred to as the original algorithm) for detection of referable diabetic retinopathy (rDR) in retinal fundus photographs. The algorithm is trained and validated using 118 419 fundus images gotten from EyePACS and from three eye hospitals in India. The original algorithm's performance was evaluated on 2 test sets, and achieved an area under the receiver operating characteristic curve (AUC) for detecting rDR of 0.99 for both the EyePACS-1 and the Messidor-2 test sets. Two operating points were selected for high sensitivity and specificity. The operating point

for high specificity had 90.3% and 87.0% sensitivity and 98.1% and 98.5% specificity for the EyePACS-1 and Messidor-2 test sets, whereas the operating point for high sensitivity had 97.5% and 96.1% sensitivity and 93.4% and 93.9% specificity, respectively.

To assess replicability of the method used to develop the original algorithm for detection of rDR, we used similar images from a publicly available EyePACS data set for training and validation, and we used a subset from the EyePACS data set and images from the public Messidor-Original data set for performance evaluation. Because many of the details regarding the validation procedure were not described in the original study (for example for hyper-parameter optimization), we had to find optimal hyper-parameters ourselves. Our objective is to compare the performance of the original rDR detection algorithm to our result algorithm after trying to replicate, taking into account potential deviations in the data sets, having fewer grades, and potential differences in hyper-parameter settings.

We were not able to replicate the original study. Our algorithm's AUC for detecting rDR for our EyePACS and Messidor-Original test sets were 0.74 and 0.59. The operating point for high specificity had 67.2% and 44.0% sensitivity and 68.2% and 64.8% specificity for our EyePACS and Messidor-Original test sets, and the operating point for high sensitivity had 79.8% and 56.6% sensitivity and 53.7% and 54.3% specificity. The results can differ for four reasons. First, we used public retinal images with only one grade per image, whereas in the original study the non-public retinal images were re-graded multiple times. Second, the original study lacked details regarding the training and validation procedure, and the original algorithm may therefore have been tuned better. Third, there might be errors in the original study or methodology. The last possible reason is that we may have done something wrong with replicating the method by having misinterpreted the methodology. We do not know for sure which of the four reasons has led to our considerably worse performance. In further research, apart from this replication, we improved the algorithm by slightly modifying the preprocessing procedure, and the AUC then increased to 0.94 and 0.82 for the Kaggle EyePACS and the Messidor-Original test sets, respectively.

We believe our failed effort on trying to replicate a highly-cited deep learning paper motivates the need for additional replication studies in deep learning. This result gives a general insight into the challenges of replicating studies that do not use publicly available data and publish source code. We have published our source code with instructions for how to use it with public data. This gives others the opportunity to improve upon the attempted replication.

3.2 Methods

3.2.1 Data Sets

The data sets consist of images of the retinal fundus acquired for diabetic retinopathy screening. Any other information regarding the person is not part of the data sets. Each image is graded according to severity of symptoms (see Section 3.2.2).

The original study obtained 128 175 retinal fundus images from EyePACS in the US and from three eye hospitals in India. 118 419 macula-centered images from this data set were used for algorithm training and validation (referred to as *development set*, divided into *training* and *tuning set* in the original study). To evaluate the performance of the algorithm, the original study used two data sets (referred to as *validation sets* in the original study). For evaluating an algorithm's performance, the term test set is commonly used. The first test set was a randomly sampled set of 9963 images taken at EyePACS screening sites between May 2015 and October 2015. The second test set was the publicly available Messidor-2 data set [60, 61], consisting of 1748 images. We provide an overview of the differences in image distribution used in our replication compared with the original study in Figure 3.2.

We obtained images for training, validation and testing from two sources: EyePACS from a Kaggle competition [62], and the publicly available Messidor-Original set [63]. The Messidor-Original set is a benchmark for algorithms that detect diabetic retinopathy. We randomly sampled the Kaggle EyePACS data set consisting of 88 702 images into a training and validation set of 57 146 images and a test set of 8790 images. The leftover images were mostly images graded as having no diabetic retinopathy and were not used for training the algorithm. The reason for the number of images in our training and validation set is to keep the same balance for the binary rDR class as in the original study's training and validation set. Our EyePACS test set has an identical amount of images and balance for the binary rDR class as in the original study's EyePACS test set. We used all the available 1200 images from Messidor-Original for testing. We removed duplicate images and made corrections from this set as suggested on the Messidor-Original download page, resulting in a test set of 1187 images. Note that we could not use Messidor-2 since they do not provide official grades for diabetic retinopathy. Messidor-Original is a subset of Messidor-2, which means that these data sets are quite similar.

Figure 3.1: Screenshot of grading tool used to assess gradability for all images.

3.2.2 Grading

The images used for the algorithm training and testing in the original study were all graded by ophthalmologists for image quality (gradability), the presence of diabetic retinopathy, and macular edema. We did not have grades for macular edema for all our images, so we did not train our algorithm to detect macular edema.

Kaggle [64] describes that some of the images in their EyePACS distribution may consist of noise, contain artifacts, be out of focus, or be over- or underexposed. [65] states further that 75% of the EyePACS images via Kaggle are estimated gradable. For this replication we graded all Kaggle and Messidor-Original images on their image quality with a simple grading tool (Figure 3.1). We are not licensed ophthalmologists, but we assume fundus image quality can be reliably graded by non-experts. We used the "Grading Instructions" in the Supplement of the original study to assess image quality. We publish the image quality grades with the source code. Images of at least adequate quality were considered gradable.

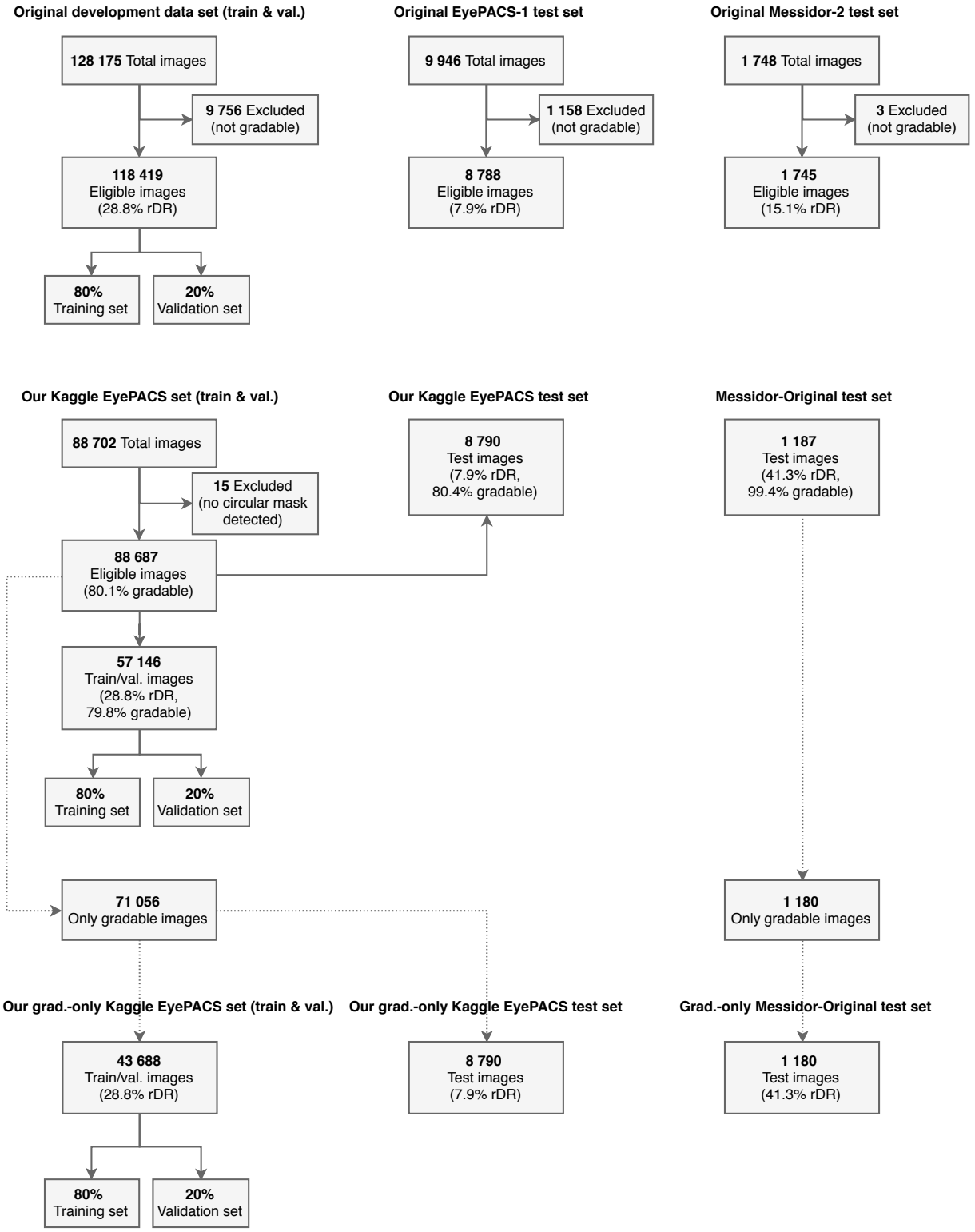


Figure 3.2: Data set distribution in original study compared to this replication.

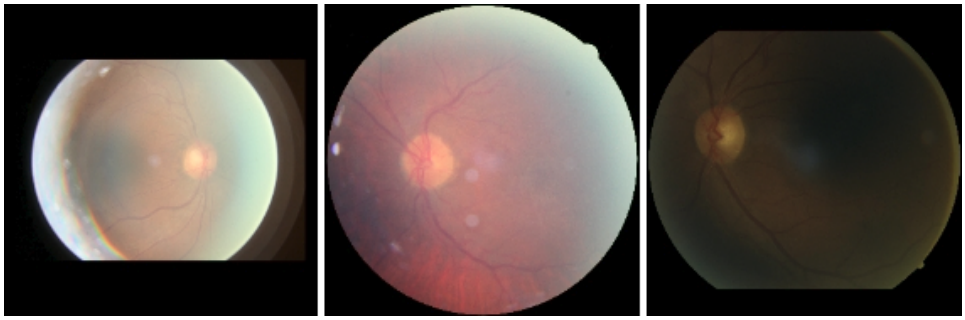


Figure 3.3: Examples of ungradable images because they are either out of focus, under-, or overexposed.

In the original study, diabetic retinopathy was graded according to the International Clinical Diabetic Retinopathy scale [66], with no, mild, moderate, severe or proliferative severity.

The Kaggle EyePACS set had been graded by one clinician for the presence of diabetic retinopathy using the same international scaling standard as used in the original study. We have thus only one diagnosis grade for each image. Kaggle does not give more information about where the data is from. The Messidor-Original test set was graded by medical experts for both the presence of diabetic retinopathy, and for the risk of macular edema. Since we do not have grades for the risk of macular edema in our training set, we did not use these grades in our algorithm. In Messidor-Original, diabetic retinopathy was also graded using a different scale, so we converted the grades to the International Clinical Diabetic Retinopathy scale by using the scale's definitions [66]. Fundus images with one to five microaneurysms and no hemorrhages were considered mild; 6 to 14 microaneurysms or up to 5 hemorrhages and no neovascularization were considered moderate; and more than 15 microaneurysms, more than 5 hemorrhages, or the presence of neovascularization were considered severe or worse diabetic retinopathy. See Table 3.1 for an overview. As in the original study, we converted the final diabetic retinopathy grade to a binary grade indicating referable diabetic retinopathy, which presents moderate or worse diabetic retinopathy.

3.2.3 Algorithm Training

The objective of this replication is to assess replicability of the original study. We try to replicate the method by following the original study's methodology as accurately as possible. As in the original study, our algorithm is created through deep learning, which involves a procedure of training a neural network to perform the task of classifying images. We trained the algorithm with the

Kaggle EyePACS grading *(International Clinical Diabetic Retinopathy scale)*	**Messidor-Original grading**	**rDR grade**
No diabetic retinopathy	Normal: no microaneurysms and no hemorrhages	0
Mild diabetic retinopathy	1 to 5 microaneurysms and no hemorrhages	0
Moderate diabetic retinopathy	6 to 14 microaneurysms, or up to 5 hemorrhages and no neovascularization	1
Severe diabetic retinopathy	More than 15 microaneurysms, more than 5 hemorrhages, or neovascularization	1
Proliferative diabetic retinopathy	-	1

Table 3.1: Interpretation of referable diabetic retinopathy (rDR) grades from the grading used in Kaggle EyePACS and Messidor-Original.

same neural network architecture as in the original study: the InceptionV3 model proposed by Szegedy et al [37]. This neural network consists of a range of convolutional layers that transforms pixel intensities to local features before converting them into global features.

The fundus images from both training and test sets were preprocessed as described by the original study's protocol for preprocessing. In all images the center and radius of the each fundus were located and resized such that each image gets a height and width of 299 pixels, with the fundus center in the middle of the image. We also scale-normalized the images before passing them to the neural network, as in the original study.

The original study used distributed stochastic gradient descent proposed by Dean et al [3] as the optimization function for training the parameters (i.e. weights) of the neural network. This suggests that their neural network was trained in parallel, although the paper does not describe it. We did not conduct any distributed training for our replica neural network. Therefore, we used the non-distributed stochastic gradient descent [67] as our optimization procedure. Using a different optimization procedure affects the time consumption, but not the final performance of the algorithm. The original study did not describe any learning rate for their training. Therefore we had to experiment with several settings for the learning rate.

As in the original study, we used batch normalization layers [68] after each convolutional layer. Our weights were also pre-initialized using weights from the neural network trained to predict objects in the ImageNet data set [69].

The neural network in the original study was trained to output multiple binary predictions: 1) whether the image was graded moderate or worse diabetic retinopathy (i.e. moderate, severe, or proliferative grades); 2) severe or worse diabetic retinopathy; 3) referable diabetic macular edema; or 4) fully gradable. The term referable diabetic retinopathy was defined in the original study as an image associated with either or both category 1) and 3). For the training data obtained in this replication, only grades for diabetic retinopathy were present. That means that our neural network outputs only one binary prediction: moderate or worse diabetic retinopathy (referable diabetic retinopathy).

For this replication, the training and validation sets were split like in the original study: 80% was used for training and 20% was used for validating the neural network. It is estimated that 25% of the Kaggle EyePACS set consists of ungradable images [65]. Therefore, we also assessed image gradability for all Kaggle EyePACS images, and we trained an algorithm with only gradable images. In the original study, the performance of an algorithm trained with only gradable images was also summarized. We do not use the image quality grades as an input for algorithm training.

Hyper-parameter settings for the optimization and validation procedure were not specified, so we conducted experiments to find hyper-parameter settings that worked well for training and validating the algorithms.

3.2.4 Algorithm Validation

We validate the algorithm by measuring the performance of the resulting neural network by the area under the receiver operating characteristic curve (AUC) on a validation set, as in the original study. We find the area by thresholding the network's output predictions, which are continuous numbers ranging from 0 to 1. By moving the operating threshold on the predictions, we obtain different results for sensitivity and specificity. We then plot sensitivity against 1–specificity for 200 thresholds. Finally, the AUC of the validation set is calculated, and becomes an indicator for how well the neural network detects referable diabetic retinopathy. The original study did not describe how many thresholds were used for plotting AUC, so we used the de facto standard of 200 thresholds.

The original paper describes that the AUC value of the validation set was used for the early-stopping criterion [70]; training is terminated when a peak AUC on the validation set is reached. This prevents overfitting the neural network on the training set. In our validation procedure, we also use the AUC calculated from the validation set as an early stopping criterion. To determine if a peak AUC is reached, we compared the AUC values between different

validation checkpoints. To avoid stopping at a local maximum of the validation AUC function, our network may continue to perform training up to n epochs (i.e. patience of n epochs). Since the original paper did not describe details regarding the validation procedure, we had to experiment with several settings for patience. One epoch of training is equal to running all images through the network once.

We used ensemble learning [39] by training 10 networks on the same data set, and using the final prediction computed by taking the mean of the predictions of the ensemble. This was also done in the original study.

In the original study, additional experiments were conducted to evaluate the performance of the resulting algorithm based on the training set, compared with performance based on subsets of images and grades from the training set. We did not replicate these experiments for two reasons. First, we chose to focus on replicating the main results of the original paper. That is, the results of an algorithm detecting referable diabetic retinopathy. Second, we cannot perform subsampling of grades, as we only have one grade per image.

3.3 Results

We found that a static learning rate of 0.003 performed well under training the algorithm. For Nesterov's accelerated gradient descent we used a momentum value of 0.9. As for our early-stopping criterion at a peak AUC, we introduced a patience of 10 epochs. Our chosen requirement for a new peak AUC was a value of AUC that is larger than the previous peak value, with a minimum difference of 0.01.

The replica algorithm's performance was evaluated on two independent test sets. We provide an overview of the differences in image distribution used in our replication compared with the original study in Figure 3.2 in Section 3.2.2. Our replica algorithm yielded an AUC of 0.74 and 0.59 on our Kaggle EyePACS test data set and Messidor-Original (Figure 3.4 and Table 3.2).

We observe mainly three things from Table 3.2. First, there is a large discrepancy between the AUC of our replication and the original study. Second, the AUC did not change substantially when excluding non-gradable images. Third, the AUC increased substantially when altering the preprocessing method (see Section 3.4.3), but it is still low compared to the original study.

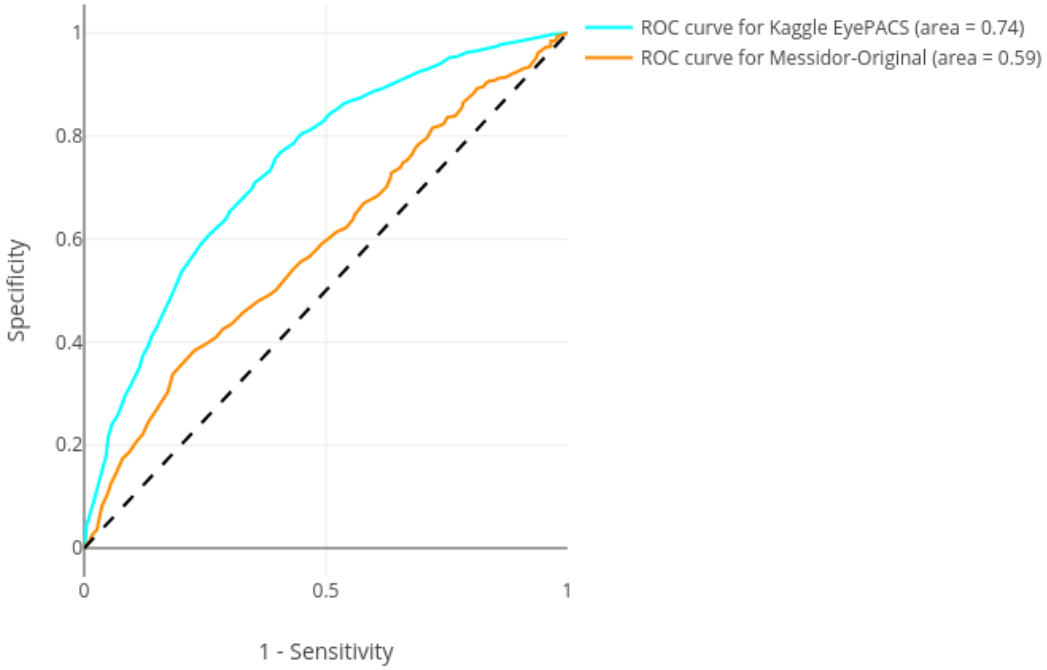


Figure 3.4: Area under receiver operating characteristic curve (AUC) for the replica algorithm trained with only gradable fundus images and stochastic gradient descent.

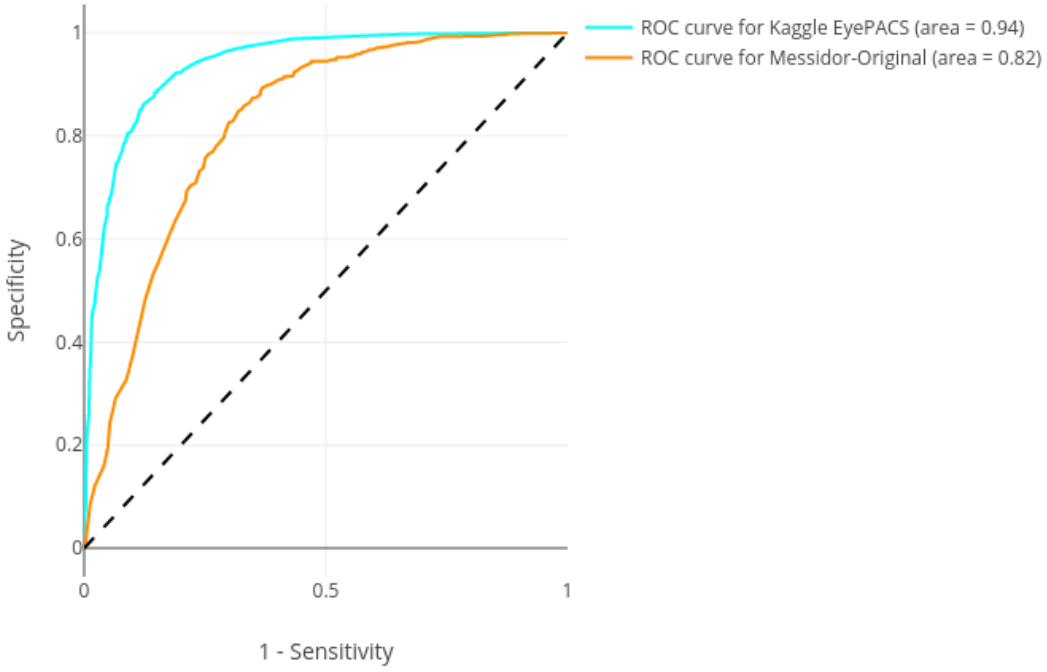


Figure 3.5: Area under receiver operating characteristic curve (AUC) for the improved algorithm trained with only gradable fundus images and stochastic gradient descent.

Replication results			
Operating threshold	***High sens.***	***High spec.***	***AUC score***
Kaggle EyePACS (orig. EyePACS-1)	75.4% sens. 55.4% spec.	65.7 (90.1)% sens. 67.6 (98.2)% spec.	0.71
Messidor-Original (orig. Messidor-2)	57.6% sens. 54.6% spec.	42.2 (86.6)% sens. 68.8 (98.4)% spec.	0.60
Operating threshold	***High sens.***	***High spec.***	***AUC score (orig.)***
Only grad. Kaggle EyePACS test (orig. EyePACS-1)	79.8 (97.5)% sens. 53.7 (93.4)% spec.	67.2 (90.3)% sens. 68.2 (98.1)% spec.	0.74 (0.99)
Only grad. Messidor-Original (orig. Messidor-2)	56.6 (96.1)% sens. 54.3 (93.9)% spec.	44.0 (87.0)% sens. 64.8 (98.5)% spec.	0.59 (0.99)
Improved results			
Operating threshold	***High sens.***	***High spec.***	***AUC score***
Kaggle EyePACS test (orig. EyePACS-1)	87.0% sens. 81.9% spec.	80.6 (90.1)% sens. 88.1 (98.2)% spec.	0.93
Messidor-Original (orig. Messidor-2)	76.0% sens. 70.7% spec.	70.1 (86.6)% sens. 78.3 (98.4)% spec.	0.81
Operating threshold	***High sens.***	***High spec.***	***AUC score (orig.)***
Only grad. Kaggle EyePACS test (orig. EyePACS-1)	90.0 (97.5)% sens. 81.4 (93.4)% spec.	83.3 (90.3)% sens. 90.5 (98.1)% spec.	0.94 (0.99)
Only grad. Messidor-Original (orig. Messidor-2)	77.0 (96.1)% sens. 70.8 (93.9)% spec.	70.1 (87.0)% sens. 82.6 (98.5)% spec.	0.82 (0.99)

Table 3.2: Overview of performance on test sets of replication and improved ensemble models trained with stochastic gradient descent, compared to results from the original study. In the first two rows, we summarize results for training on all images, and in the last two rows we summarize results for training on only gradable images. The results of the original study are depicted in parenthesizes. In columns without parenthesis values, the original study did not report results for the algorithm and/or operating point.

3.4 Discussion

The results show substantial performance differences between the original study's algorithm and our replica algorithm. Even though we followed the methodology of the original study as closely as possible, our algorithm did not seem to "learn" how to recognize lesions in fundus images as local features. This is probably because our algorithms were trained under different hyper-parameters, and because in the original study ophthalmologic experts re-graded all their images. According to the original study, the validation and test sets should have multiple grades per image, because it will provide a more reliable measure of a model's final predictive ability. Their results on experimenting with only one grade per image show that their algorithm's performance declines with 36%.

Some of the details regarding the methods in the original study were not specified. First, the details on hyper-parameter settings for the validation procedure, or for the optimization function are missing. The original study also briefly mentions that image preprocessing is performed in the validation procedure, but it does not further elaborate on this. Second, it is unclear how the algorithm's predictions for diabetic retinopathy or macular edema are interpreted in case of ungradable images. The image quality grades might have been used as an input for the network, or the network might be concatenated with another network that takes the image quality as an input. Third, apart from the main algorithm that detects referable diabetic retinopathy and outputs 4 binary classifications, other algorithms seem to have been trained as well. An example is the described algorithm that only detects referable diabetic retinopathy for gradable images, and an algorithm that detects all-cause referable diabetic retinopathy, which presents moderate or worse diabetic retinopathy, referable macular edema, and ungradable images. Details on how these other algorithms are built are however not reported. It is unclear whether the main network has been used or if the original study trained new networks. Lastly, the original paper did not state how many iterations it took for their proposed model to converge during training, or describe how to find a converging model.

3.4.1 Hyper-Parameters

The main challenge in this replication was to find hyper-parameters, which were not specified in the original paper, such that the algorithm does not converge on a local maximum of the validation AUC function. To understand how we should adjust the hyper-parameters, we measured the Brier score on the training set and the AUC value on the validation set after each epoch of training. We observed the following. First, during the first 15 epochs, the AUC value on the validation set increases and stabilizes at approximately 0.65. From

then, the validation AUC does not increase, but stays around the same value. The Brier score measured on the training set gradually decreases, indicating that the algorithm is learning features from the images in the training set. This scenario continues for many epochs: the validation AUC stays around 0.65, with the Brier score of the training set gradually decreasing for every epoch. After about 50 epochs, the validation AUC decreases again, and the algorithm clearly overfits on the training data. One possible reason for the algorithm to not converge may be the dimensions of the fundus images. As the original study suggests, the original fundus images were preprocessed and scaled down to a width and height of 299 pixels to be able to initialize the InceptionV3 network with ImageNet pre-trained weights, which have been trained with images of 299 by 299 pixels. We believe it is difficult for ophthalmologists to find lesions in fundus images of this size, so we assume the algorithm has difficulties with detecting lesions as well. [65] also points out this fact, and suggests re-training an entire network with larger fundus images and randomly initialized weights instead. And as mentioned before, it seems like the original study extended the InceptionV3 model architecture for their algorithm to use image gradability as an input parameter.

3.4.2 Kaggle Images

A potential drawback with the images from Kaggle is that it contains grades for diabetic retinopathy for all images. We found that 19.9% of these images is ungradable, and it is thus possible that the algorithm will "learn" features for ungradable images, and make predictions based on anomalies. This is likely to negatively contribute to the algorithm's predictive performance, but we were not able to show a significant difference of performance between an algorithm trained on all images and an algorithm trained on only gradable images.

3.4.3 Improvements

We made minor changes to the replicated method. First, we modified the preprocessing procedure. In the original study the images were scale-normalized, which we assumed meant normalizing the image values by scaling them down to being in a range from 0 to 1 [71]. We have seen from many entries in the Kaggle-competition that as a preprocessing procedure image standardization was performed, subtracting mean from each image and then dividing by the standard deviation. Therefore, we attempted to standardize the images instead of scale-normalizing them, and we re-trained all algorithms. This resulted in a substantial increase in performance (Figure 3.5 and Table 3.2). Why this small difference in preprocessing yielded such a large increase in performance is unclear.

Second, we re-trained the algorithms with Nesterov's accelerated gradient descent instead of stochastic gradient descent. This did however not affect the performance, but fewer epochs were needed to find the peak validation AUC value.

3.5 Conclusion

We attempted to replicate the main method from JAMA 2016; 316(22), but we were not able to get the same performance as reported in that study. The findings of this replication confirm the need for additional deep learning replication studies.

4

Scalability Evaluation

4.1 Introduction

To assess resources required to develop and to deploy a deep learning algorithm for automated analyses, we used the deep learning algorithm from the replication in Chapter 3. We believe that, by scaling out our diabetic retinopathy detection algorithm, we can make a rough estimation on the resources required to develop and deploy an algorithm for large-scale screening programs such as BreastScreen Norway. We compare algorithm training and evaluation on three test environments. The first environment is a machine with 1 NVIDIA GeForce GTX 1080 GPU (*Deep1*). We set results from this environment in contrast with running the same experiments on the multi-GPU *Medsensio1*, which consists of 2 NVIDIA GeForce GTX 1080 Ti GPUs, and *UNINETT DaaS Cluster*, which consists of 4 NVIDIA Titan X Pascal GPUs in 2 worker nodes. To get an insight into whether it is worth to invest in a more powerful set-up to conduct algorithm training or not, we find the training speed given by images per second on all test environments in our first experiment. In the second experiment, we measured prediction speed for unseen data, to assess resource requirements for running a deep learning model in production as a service. In both experiments, we used the retinal fundus images from our replication. We ran the experiments in TensorFlow, since we implemented the deep learning algorithm in TensorFlow as well. We run the benchmark tests non-distributedly for one NVIDIA GeForce GTX 1080, in parallel for 2 Nvidia GeForce GTX 1080 Ti GPUS, and we run in a distributed fashion for 3 and 4 NVIDIA Titan X Pascal GPUs.

Details for Deep1 (NVIDIA GeForce GTX 1080)
Instance Deep1
CPU Intel Core i5-7600K @ 3.80GHz
GPU 1x NVIDIA GeForce GTX 1080
OS Ubuntu 16.04
TensorFlow / CUDA / cuDNN 1.9 / 9.0 / 7
Disk 500 GB HHD

Details for Medsensio1 (NVIDIA GeForce GTX 1080 Ti)
Instance Medsensio1
CPU AMD Ryzen 7 1700
GPU 2x NVIDIA GeForce GTX 1080 Ti
OS Ubuntu 16.04
TensorFlow / CUDA / cuDNN 1.9 / 9.0 / 7
Disk 1 TB SSD

Details for UNINETT DaaS Cluster (NVIDIA Titan X Pascal)
Instance UNINETT DaaS Cluster
CPU Intel Xeon E5-2683 v4
GPU 4x NVIDIA Titan X Pascal (2 per node)
OS Ubuntu 16.04
TensorFlow / CUDA / cuDNN 1.9 / 9.0 / 7
Disk 800 GB local SDD per node

Figure 4.1: Details for test environments on which the benchmarks were run.

TensorFlow supports large-scale, distributed computation operations [5, 72]. TensorFlow's documentation describes results from their own benchmarks of parallel and distributed training on data clusters with many GPUs [73]. The benchmark results have been confirmed independently on other data clusters like Amazon EC2 [74]. The benchmarks were run with various neural network architectural models. The *InceptionV3* [37] neural network architecture was trained on data from *ImageNet* [39]. This is the same architecture we used for our deep learning algorithm. In the replication repository (see Section 1.3), we added functionality for the benchmarks to run with the retinal fundus images and preprocessing used in our replication, since they are different than ImageNet and its preprocessing. According to the official benchmark results, training InceptionV3 with ImageNet data on 8 NVIDIA Tesla K80 GPUs distributedly gives a 7.6x speed-up (30 images/sec on 1 GPU compared to 229 images/sec on 8 GPUs). We therefore expect that distributed training of InceptionV3 with our retinal fundus images by using 4 NVIDIA Titan X Pascal GPUs will give a sub-linear speed-up as well.

Table 4.1: Settings for running the benchmarks on all test environments.

Environment	Training batch size/GPU	Evaluation batch size/GPU	variable_update
UNINETT DaaS Cluster (4x NVIDIA Titan X Pascal)	64	256	1,2 GPUs: parameter_server 3,4 GPUs: distributed_replicated
Medsensio1 (2x NVIDIA GeForce GTX 1080 Ti)	32	256	parameter_server
Deep1 (1x NVIDIA GeForce GTX 1080)	32	256	parameter_server

Environment	local_parameter_device	cross_replica_sync	optimizer function
UNINETT DaaS Cluster (4x NVIDIA Titan X Pascal)	1,2 GPUs: cpu 3,4 GPUs: n/a	1,2 GPUs: False 3,4 GPUs: True	sgd
Medsensio1 (2x NVIDIA GeForce GTX 1080 Ti)	cpu	n/a	sgd
Deep1 (1x NVIDIA GeForce GTX 1080)	cpu	n/a	sgd

4.2 Experiments

See Table 4.1 for all benchmark settings. In the first experiment, we measured training speed with 32 images per batch per GPU and stochastic gradient descent. There are two types of benchmark processes during training and evaluation: 1) workers, which run the model, send their local gradients to the parameter servers, and receive updated variables back; and 2) parameter servers (PS), which host trainable variables and update them with values sent by the workers. To run the benchmarks on one and two GPUs on all environments, we used local parameter servers using the CPU from the same node as the worker. Since the GPUs reside on the same node, we do not have inter-node communication. For running the benchmarks with 3 and 4 GPUs on *UNINETT DaaS Cluster*, the training needs to be split on at least two nodes, since one single node on this cluster has 2 GPUs. Distributed training can be conducted in several ways. The most common approach, and the approach we used, is *data parallelism* (*between-graph replication* in TensorFlow). Each node has an instance of the model and reads different training samples. To reduce overhead of inter-node network communication, each node ran both a parameter server and a worker.

The second experiment measures prediction speed in images per second. A deployed model can either evaluate unseen data in batches or real-time. For evaluation in batches, we used a batch size of 256. To assess maximum real-time evaluation performance, we used a batch size of 1, on 1 GPU, to simulate a deployed model serving many independent evaluation requests. We extended this experiment by measuring the performance on 1 CPU only.

Training and batch prediction speed are presented in Figure 4.2a and Figure 4.2b. See Figure 4.3a for benchmarks results for real-time prediction speed. Each experiment was run 5 times and then the times were averaged together.

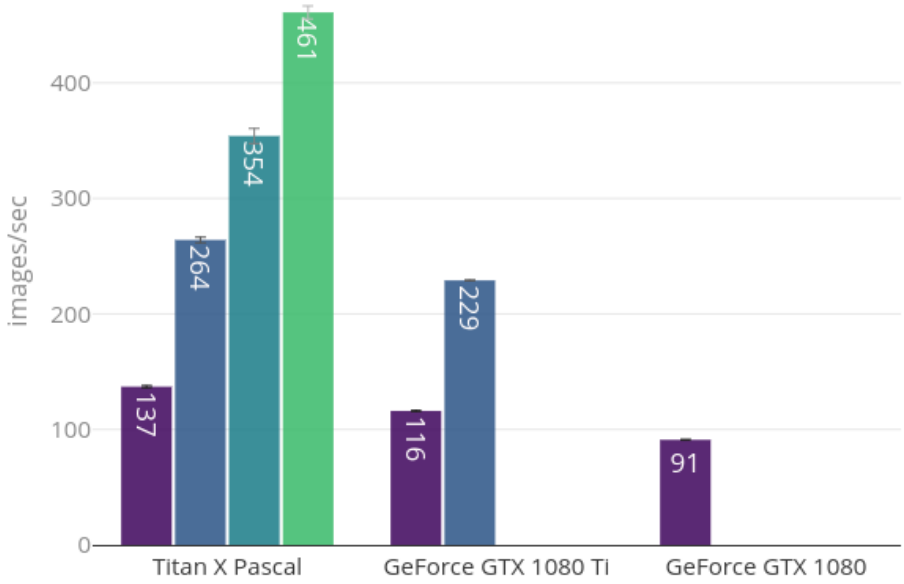
To verify that the added functionality for running the TensorFlow benchmarks with the retinal fundus data run correctly, we compared our benchmark results to results from running the original benchmarks with synthetic ImageNet data. The synthetic data is a set of estimated ImageNet data. The official TensorFlow benchmarks were run with synthetic data to remove disk I/O and input pipeline (to read data from disk) as a variable and to set a baseline for the other benchmark results.

4.3 Discussion

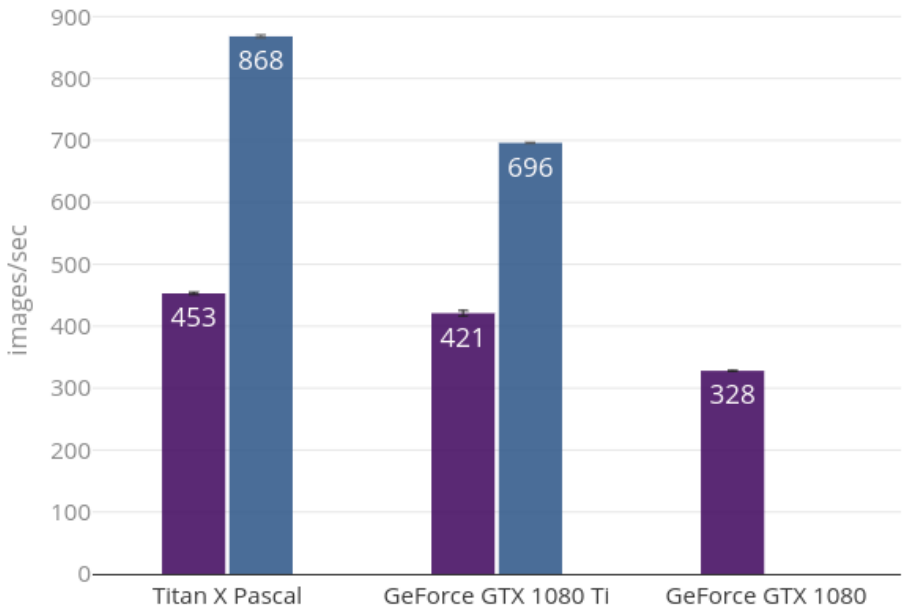
The benchmark results show (sub)linear speed-ups for algorithm training with multiple GPUs (see Figure 4.2a). Scaling up the number of GPUs on a single node from 1 to 2 GPUs gave on average a nearly linear speed-up of 1.95 times (1.93 times on *UNINETT DaaS* and 1.97 times on *Medsensio1*). Moreover, distributing the workload on multiple nodes in the *UNINETT DaaS Cluster* resulted in a speed-up of 2.58 times on 3 GPUs, and 3.36 times on 4 GPUs. By experimenting with various settings, we observed that running the training benchmarks on 4 nodes with 1 GPU each yielded a higher performance than running the benchmarks on 2 nodes with 2 GPUs each. Speed-up was reduced when running the benchmarks in a distributed manner on multiple nodes due to inter-node communication overhead.

The benchmark results show that it is useful to have multiple GPUs available when evaluating data in batches (see Figure 4.2b). Scaling up from 1 to 2 GPUs for batch evaluation resulted in sublinear speed-ups, with an average speed-up of 1.78 times (1.91 times on *UNINETT DaaS* and 1.65 on *Medsensio1*). For real-time evaluation, we measured performance only for 1 GPU to simulate real-time mass-evaluation of single images (see Figure 4.3a). These results show that real-time evaluation performs similarly for all GPU types. Real-time evaluation performance on 1 CPU only was also measured. However, only the AMD Ryzen 7 1700 CPU (*Medsensio1*) was able to run the benchmarks.

Figures 4.3b, 4.4a, and 4.4b point out that similar performance is reached for running the benchmarks with retinal fundus data and synthetic data. As expected, when a larger amount of data is evaluated, the disk I/O and input pipeline overhead becomes visible. Overall, the results show minimal difference between the two types of data, meaning that our retinal fundus data benchmarks yield accurate results.

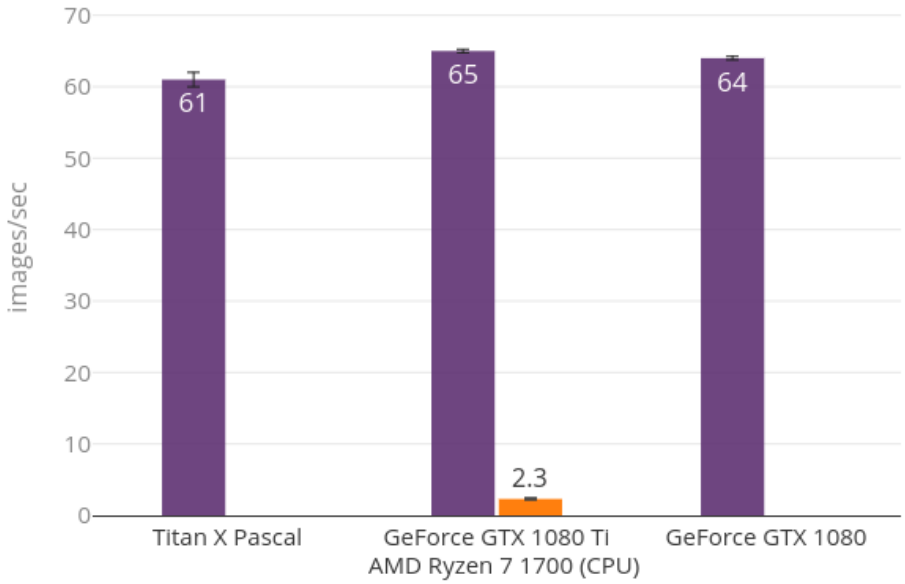


(a) Parallel and distributed training speed.

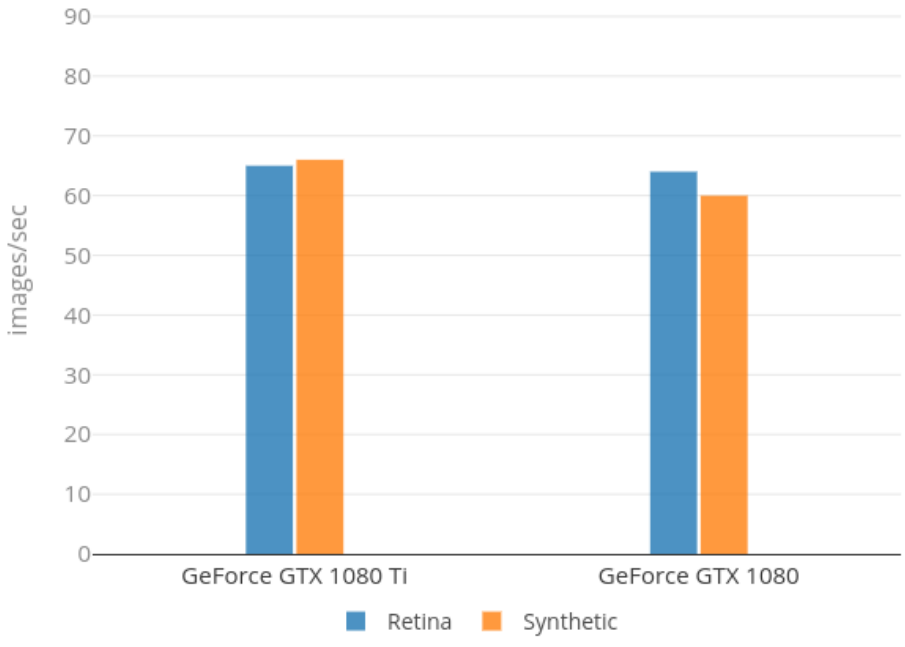


(b) Batch evaluation (prediction) speed.

Figure 4.2: Training (a) and evaluation (b) performance on *UNINETT DaaS Cluster*, *Medsensio1* and *Deep1* GPUs. The number of GPUs are presented from left to right, from 1 to a maximum of 4 GPUs.



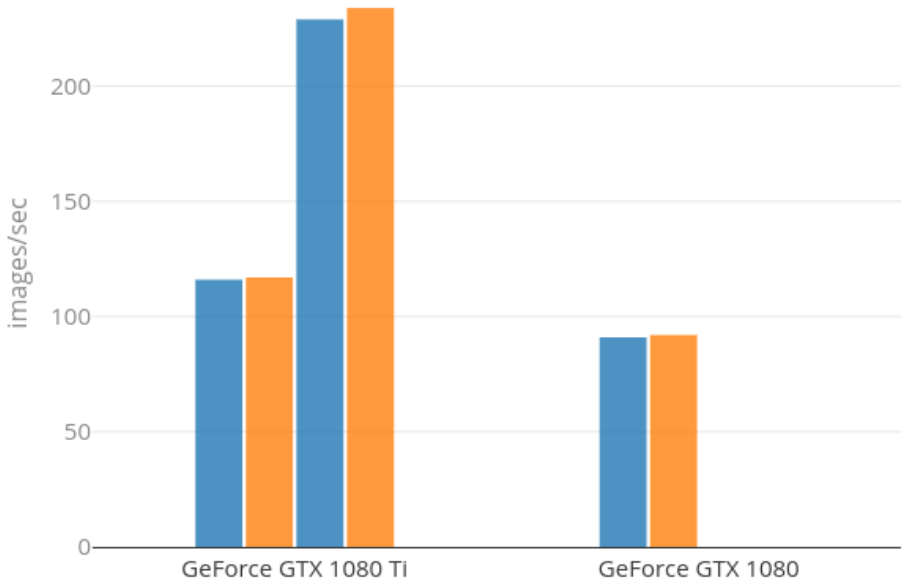
(a) Real-time evaluation (prediction) speed on 1 GPU or 1 CPU on all environments. However, only the AMD Ryzen 7 1700 CPU (*Medsensio1*) was able to run the benchmarks.



(b) Comparison of real-time evaluating retinal fundus data and synthetic ImageNet data on 1 GPU.

Figure 4.3: Real-time evaluation performance (a), and comparing evaluation performance with retinal fundus data and synthetic ImageNet data (b).

(a) Comparison of training with retinal fundus data and synthetic ImageNet data on 1 and 2 GPUs.



(b) Comparison of batch evaluating retinal fundus data and synthetic ImageNet data on 1 and 2 GPUs.

Figure 4.4: Comparisons of training (a) and evaluation (b) of retinal fundus images from our replication and synthetic ImageNet data.

4.4 Conclusion

The benchmark results show that training and batch evaluation can be effectively scaled out to multiple GPUs to gain (sub)linear speed-ups. It is further shown that lower-performance GPUs and even CPUs with high enough specifications are sufficient to run real-time evaluation.

With our results of running benchmarks in our diabetic retinopathy algorithm, we can assess the time and resources needed to train a deep learning algorithm with data from BreastScreen Norway. To be able to make an estimation, we assume that the neural network of the breast cancer detection algorithm has the same amount of layers and parameters as in our diabetic retinopathy detection algorithm. A larger neural network would likely consume more time to train. In addition, we assume that the image dimensions are the same (299 by 299 pixels). To find an estimation of the amount of hours needed to train an algorithm, the following function can be used:

$$hours_{train} \approx \frac{epochs \cdot images}{images/sec \cdot 3600} \tag{4.1}$$

By using 4 GPUs, which have shown to be able to train 461 images per second, training our diabetic retinopathy algorithm with 43 688 images in 100 epochs would take less than 3 hours. Following this function and taking in mind our assumptions, 100 epochs of training a deep learning algorithm for breast cancer detection with 280 000 images from BreastScreen Norway will take approximately 17 hours. If 2 GPUs were used instead of 4, training would take approximately 30 hours instead. If a time constraint would be to train one model in 24 hours, training with 4 GPUs (or more) is recommended. However, if only 4 GPUs are available and two models need to be trained in parallel, for example to create an ensemble for improved performance, it is recommended to train each model with 2 GPUs, due to higher relative speed-up on 2 GPUs.

Our experiments have also shown that all GPUs can yield a visually instant prediction for diabetic retinopathy for one retinal fundus image (in approximately 16 milliseconds), despite the specification differences between the various types of GPUs. High-performance CPUs, like AMD Ryzen 7 1700, have shown to be able to perform real-time evaluation as well, but they perform significantly worse than GPUs due to demanding I/O operations and input computations performed on the CPU. Because low-performance GPUs are cheaper than high-performance CPUs, we recommend the former, to use for performing and scaling out biomedical deep learning analyses.

5

Conclusion

As a first step towards automated analyses for BreastScreen Norway, to develop a deep learning algorithm, we need to extract mammograms from a PACS at the University Hospital of North Norway (UNN). Before extraction, the mammograms must be anonymized. We therefore wrote an extraction and anonymization script, which is being authorized and finalized by Helsenord IKT. Because of the lengthy application process, we did not get the mammograms in time for this project.

Since we did not have a public mammogram data set we could use to develop a breast cancer detection algorithm, we instead developed an algorithm that detects diabetic retinopathy, which is a similar screening problem in the biomedical field. Because, in general, designing a model architecture for a deep learning algorithm from scratch is complicated, we attempted to replicate a method from a high-impact article published in JAMA. We showed that this method is non-replicable due to missing details in the reported methodology, and the usage of different data to develop the algorithm. Our concern of challenges in replication of studies in the biomedical industry is supported by other literature. We believe that replication challenges like we had, are likely to occur when replicating methods from articles that state high-performance of breast cancer detection algorithms as well. In the end, we introduced some improvements to the algorithm by changing the data preprocessing methods. This amplifies the suggestion of that the article's methodology misreported some details, but we cannot say this for certain. All in all, we have demonstrated the importance of replication studies, suggesting to publish the source code

and used data, so that other researchers can verify the results.

We used the diabetic retinopathy detection algorithm as a candidate to be deployed for automated analyses, and we assess the resources necessary to train the algorithm and run evaluation or analyses of unseen data on a larger, national (Norwegian) scale. To orchestrate this, we gained access to several environments with varying GPU set-ups. Our benchmark results have shown that training the algorithm can be effectively scaled out to many GPUs to gain sublinear training speed-ups, suggesting that training of a breast cancer detection algorithm can be performed in less than 17 hours with 4 GPUs. Moreover, additional experiments show that data evaluation that is done batch-wise also benefits sublinear speed-ups on many GPUs, and the higher performance one GPU has, the higher the total performance becomes. However, our results have shown that real-time analyses do not especially benefit from being run on high-performance GPUs, and that lower-performance GPUs are sufficient for this task, with visually instant prediction times of approximately 16 milliseconds.

5.1 Future Work

We make some suggestions for the breast cancer detection algorithm to be developed when the anonymized mammograms have been extracted. We also propose some literature to examine further.

The DREAM challenge [58] formalizes the breast cancer screening problem by stating that out of every 1000 people screened, only 5 will have breast cancer, but 100 are recalled for further testing. In Norway, 8 out of 1000 people screened will have breast cancer, but 30 are recalled for testing. The objective is thus to reduce the amount of false positives in screening. Hence, we suggest to start by investigating earlier attempts that report methods for developing a detection algorithm. There are numerous articles that report a deep learning approach to reduce false positives in breast cancer screening [75, 76, 77, 78, 79, 80, 81, 82, 83, 84]. There are also other approaches that involve a deep learning approach with different data, however with the same goal of reducing false positives [85, 86, 87, 88]. A review study that gathers various deep learning approaches for this problem has been published as well [89]. Furthermore, [90] proposes a breast cancer localization algorithm. [91] describes other machine learning approaches for classifying mammograms. [92, 93] publish general insights into deep learning approaches and learning problems from the biomedical industry. Other automated breast cancer detection methods are also described [94, 95, 96, 97, 98, 99], and give insights into how a deep learning algorithm should function. Lastly, [100, 101, 102] present background information and statistics regarding breast cancer.

To develop the deep learning algorithm, we suggest exploring earlier and particularly the latest approaches. For instance, studying approaches from the Kaggle competition was an efficient way to learn how to develop a deep learning algorithm for diabetic retinopathy detection. Similarly, one could study and get inspired by approaches used in the DREAM challenge. Ideally, some submissions include a link to their source code, which can be used to quickly verify their algorithm and results. Submissions that only include a report may be replicated. The mammography data set from UNN is estimated to contain files from 70 000 screenings, with each screening providing 4 DICOM files. Machine learning frameworks are usually not able to use DICOM files as an input for a computational pipeline, so the data needs to be converted to a usable image file format. The DICOM images can for instance be converted to JPEG, and be resized to a width and height of 512 pixels, which we think is large enough for breast cancer detection. To optimize for disk I/O operations and input pipeline, we recommend to convert the data set to a special data format. In TensorFlow this format is *TFRecord*, and can be elegantly used with their *Dataset* API. Also, images and grades preprocessing should be done before training. Moreover, in our replication, we learned that small changes in preprocessing can affect the resulting performance drastically. We therefore recommend to experiment with various preprocessing methods, and to try to understand why some methods work better than others. Look into how outliers in the data affect the data after normalization. Even though we have not been able to show this ourselves, the quality of the data, including the quality and quantity of grades (annotations), also seem to be important factors. When new data from the Cancer Registry becomes available to use for algorithm development, we suggest to fine-tune the algorithm model on the new data, until a certain validation threshold has been reached. Repeat this process for all subsequent new data, but avoid training on validation data. To evaluate the algorithm, independent data should be used, and preferably from a different screening. In addition to the data from the DREAM challenge, publicly available mammography databases can be used for algorithm evaluation [103].

The developed algorithm can be deployed in various ways. For real-time analyses we suggest to deploy the algorithm in the form of a web service, which assigns prediction tasks to one of a set of GPUs. We found that low-performance GPUs are sufficient and cost-effective for real-time evaluation. If necessary, scale out the number of GPUs and web service servers to balance the load. If data does not need to be analyzed real-time, we recommend to perform the analyses on a multi-GPU cluster, depending on the extent to what time constraints apply, and the amount of data that needs to be evaluated. We believe TensorFlow-Serving [104] is a promising tool to use for algorithm deployment for both batch and real-time evaluation. However, what deployment tools are eligible depend entirely on the framework used for algorithm training, as other frameworks typically offer different solutions.

Bibliography

[1] V. Gulshan, L. Peng, M. Coram, M. C. Stumpe, D. Wu, A. Narayanaswamy, S. Venugopalan, K. Widner, T. Madams, J. Cuadros, R. Kim, R. Raman, P. C. Nelson, J. L. Mega, and D. R. Webster, "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs," *JAMA - Journal of the American Medical Association*, vol. 316, no. 22, pp. 2402–2410, 2016. doi: 10.1001/jama.2016.17216

[2] J. Bergstra, O. Breuleux, F. F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math compiler in Python," in *Proceedings of the Python for Scientific Computing Conference (SciPy)*, no. Scipy, 2010, pp. 1–7.

[3] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, Q. V. Le, and A. Y. Ng, "Large Scale Distributed Deep Networks," in *Advances in Neural Information Processing Systems 25 (NIPS 25)*. Curran Associates Inc., 2012, pp. 1223–1231.

[4] E. R. Sparks, A. Talwalkar, V. Smith, J. Kottalam, X. Pan, J. Gonzalez, M. J. Franklin, M. I. Jordan, and T. Kraska, "MLI: An API for Distributed Machine Learning," *Proceedings - IEEE International Conference on Data Mining, ICDM*, pp. 1187–1192, 2013. doi: 10.1109/ICDM.2013.158

[5] 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 *USENIX Symposium on Operating Systems Design and Implementation*. USENIX Association, 2016. ISBN 978-1-931971-33-1. ISSN 0270-6474 pp. 265–283.

[6] L. Bottow, "Large-Scale Machine Learning with Stochastic Gradient Descent," in *Lechevallier Y., Saporta G. (eds) Proceedings of COMPSTAT'2010*. Heidelberg: Physica-Verlag HD, 2010. doi: 10.1007/978-3-7908-2604-3. ISBN 188065346X. ISSN 0269-2155, 0269-2155 pp. 94–101.

[7] M. Jaggi, V. Smith, M. Takáč, J. Terhorst, S. Krishnan, T. Hofmann, and M. I. Jordan, "Communication-Efficient Distributed Dual Coordinate Ascent," *arXiv:1409.1458 [cs.LG]*, 2014.

[8] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv:1412.6980 [cs.LG]*, 2014.

[9] F. Niu, B. Recht, C. Re, and S. J. Wright, "HOGWILD!: A Lock-Free Approach to Parallelizing Stochastic Gradient Descent," *arXiv:1106.5730 [math.OC]*, 2011.

[10] N. P. Jouppi, A. Borchers, R. Boyle, P.-l. Cantin, C. Chao, C. Clark, J. Coriell, M. Daley, M. Dau, J. Dean, B. Gelb, C. Young, T. V. Ghaemmaghami, R. Gottipati, W. Gulland, R. Hagmann, C. R. Ho, D. Hogberg, J. Hu, R. Hundt, D. Hurt, J. Ibarz, N. Patil, A. Jaffey, A. Jaworski, A. Kaplan, H. Khaitan, D. Killebrew, A. Koch, N. Kumar, S. Lacy, J. Laudon, J. Law, D. Patterson, D. Le, C. Leary, Z. Liu, K. Lucke, A. Lundin, G. MacKean, A. Maggiore, M. Mahony, K. Miller, R. Nagarajan, G. Agrawal, R. Narayanaswami, R. Ni, K. Nix, T. Norrie, M. Omernick, N. Penukonda, A. Phelps, J. Ross, M. Ross, A. Salek, R. Bajwa, E. Samadiani, C. Severn, G. Sizikov, M. Snelham, J. Souter, D. Steinberg, A. Swing, M. Tan, G. Thorson, B. Tian, S. Bates, H. Toma, E. Tuttle, V. Vasudevan, R. Walter, W. Wang, E. Wilcox, D. H. Yoon, S. Bhatia, and N. Boden, "In-Datacenter Performance Analysis of a Tensor Processing Unit," in *Proceedings of the 44th Annual International Symposium on Computer Architecture - ISCA '17*, vol. 45, no. 2. New York, New York, USA: ACM Press, 2017. doi: 10.1145/3079856.3080246. ISBN 9781450348928 pp. 1–12.

[11] R. Raina, A. Madhavan, and A. Y. Ng, "Large-scale deep unsupervised learning using graphics processors," in *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*. New York, New York, USA: ACM Press, 2009. doi: 10.1145/1553374.1553486. ISBN 9781605585161. ISSN 12258687 pp. 1–8.

[12] J. Dean and S. Ghemawat, "MapReduce," *Communications of the ACM*, vol. 51, no. 1, p. 107, 2008. doi: 10.1145/1327452.1327492

[13] J. Gonzalez, Y. Low, and H. Gu, "PowerGraph: Distributed Graph-Parallel Computation on Natural Graphs," in *OSDI'12 Proceedings of the 10th USENIX conference on Operating Systems Design and Implementation*. USENIX Association, 2012. ISBN 978-1-931971-96-6 pp. 17–30.

[14] J. B. Share, "Review of drug treatment for Down's syndrome persons." *American journal of mental deficiency*, vol. 80, no. 4, pp. 388–93, 1 1976.

doi: 10.1111/j.1095-8649.2005.00662.x

[15] S. B. R. I. John C. McCallum, NewEgg, "Average Historic Price of RAM," p. Statistic Brain, 2015. [Online]. Available: https://www.statisticbrain.com/average-historic-price-of-ram/

[16] M. Komorowski, "A History Of Storage Cost," 2015. [Online]. Available: http://www.mkomo.com/cost-per-gigabyte-update

[17] D. Parkins, "The world's most valuable resource is no longer oil, but data," *The Economist*, 2017. [Online]. Available: https://www.economist.com/news/leaders/21721656-data-economy-demands-new-approach-antitrust-rules-worlds-most-valuable-resource

[18] G. Press, "Forrester Predicts Investment In Artificial Intelligence Will Grow 300% in 2017," 2016. [Online]. Available: https://www.forbes.com/sites/gilpress/2016/11/01/forrester-predicts-investment-in-artificial-intelligence-will-grow-300-in-2017/#254fcd185509

[19] P. Domingos, "A few useful things to know about machine learning," *Communications of the ACM*, vol. 55, no. 10, p. 78, 2012. doi: 10.1145/2347736.2347755

[20] B. W. Fitzpatrick and J. Lueck, "The Case Against Data Lock-in," *Queue*, vol. 8, no. 10, p. 20, 2010. doi: 10.1145/1866296.1868432

[21] J. Opara-Martins, R. Sahandi, and F. Tian, "Critical review of vendor lock-in and its impact on adoption of cloud computing," in *International Conference on Information Society (i-Society 2014)*. IEEE, 2014. doi: 10.1109/i-Society.2014.7009018. ISBN 978-1-9083-2038-4. ISSN 2192-113X pp. 92–97.

[22] M. Hutson, "Missing data hinder replication of artificial intelligence studies," 2018. [Online]. Available: http://www.sciencemag.org/news/2018/02/missing-data-hinder-replication-artificial-intelligence-studies

[23] C. G. Begley and L. M. Ellis, "Raise standards for preclinical cancer research," *Nature*, vol. 483, no. 7391, pp. 531–533, 3 2012. doi: 10.1038/483531a

[24] M. Baker, "1,500 scientists lift the lid on reproducibility," *Nature*, vol. 533, no. 7604, pp. 452–454, 2016. doi: 10.1038/533452a

[25] "Kaggle: Your Home for Data Science." [Online]. Available: https://www.kaggle.com/data

[26] W. C. Barker, J. S. Garavelli, P. B. McGarvey, C. R. Marzec, B. C. Orcutt, G. Y. Srinivasarao, L.-S. L. Yeh, R. S. Ledley, H.-W. Mewes, F. Pfeiffer, A. Tsugita, and C. Wu, "The PIR-International Protein Sequence Database," *Nucleic Acids Research*, vol. 27, no. 1, pp. 39–43, 1999. doi: 10.1093/nar/27.1.39

[27] J. A. Blake, J. E. Richardson, M. T. Davisson, and J. T. Eppig, "The Mouse Genome Database (MGD): genetic and genomic information about the laboratory mouse," *Nucleic Acids Research*, vol. 27, no. 1, pp. 95–98, 1999. doi: 10.1093/nar/27.1.95

[28] R. A. Baasiri, S. R. Glasser, D. L. Steffen, and D. A. Wheeler, "The Breast Cancer Gene Database: a collaborative information resource," *Oncogene*, vol. 18, no. 56, pp. 7958–7965, 1999. doi: 10.1038/sj.onc.1203335

[29] T. Kauppi, V. Kalesnykiene, J.-K. Kamarainen, L. Lensu, I. Sorri, A. Raninen, R. Voutilainen, H. Uusitalo, H. Kalviainen, and J. Pietila, "the DIARETDB1 diabetic retinopathy database and evaluation protocol," in *Procedings of the British Machine Vision Conference 2007*. British Machine Vision Association, 2007. doi: 10.5244/C.21.15. ISBN 1-901725-34-0. ISSN 22195491 pp. 1–15.

[30] J. Mohandass, S. Ravichandran, K. Srilakshmi, C. P. Rajadurai, S. Sanmugasamy, and G. R. Kumar, "BCDB - A database for breast cancer research and information," *Bioinformation*, vol. 5, no. 1, pp. 1–3, 2010. doi: 10.6026/97320630005001

[31] S. Liu, S. Liu, W. Cai, S. Pujol, R. Kikinis, and D. Feng, "Early diagnosis of Alzheimer's disease with deep learning," in *2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2014. doi: 10.1109/ISBI.2014.6868045. ISBN 978-1-4673-1961-4. ISSN 1945-7928 pp. 1015–1018.

[32] J.-Z. Cheng, D. Ni, Y.-H. Chou, J. Qin, C.-M. Tiu, Y.-C. Chang, C.-S. Huang, D. Shen, and C.-M. Chen, "Computer-Aided Diagnosis with Deep Learning Architecture: Applications to Breast Lesions in US Images and Pulmonary Nodules in CT Scans," *Scientific Reports*, vol. 6, no. 1, p. 24454, 2016. doi: 10.1038/srep24454

[33] Y. Bar, I. Diamant, L. Wolf, S. Lieberman, E. Konen, and H. Greenspan, "Chest pathology detection using deep learning with non-medical train-

ing," in *2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI)*, vol. 2015-July. IEEE, 2015. doi: 10.1109/ISBI.2015.7163871. ISBN 978-1-4799-2374-8. ISSN 19458452 pp. 294–297.

[34] J. Khan, J. S. Wei, M. Ringnér, L. H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. R. Antonescu, C. Peterson, and P. S. Meltzer, "Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks," *Nature Medicine*, vol. 7, no. 6, pp. 673–679, 2001. doi: 10.1038/89044

[35] A. S. Becker, M. Marcon, S. Ghafoor, M. C. Wurnig, T. Frauenfelder, and A. Boss, "Deep Learning in Mammography," *Investigative Radiology*, vol. 52, no. 7, pp. 434–440, 2017. doi: 10.1097/RLI.0000000000000358

[36] X. Cao, "A practical theory for designing very deep convolutional neural networks." [Online]. Available: https://kaggle2.blob.core.windows.net/forum-message-attachments/69182/2287/A practical theory for designing very deep convolutional neural networks.pdf

[37] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," *arXiv:1512.00567 [cs.CV]*, 2015. doi: 10.1109/CVPR.2016.308

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *arXiv:1512.03385 [cs.CV]*, 2015.

[39] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances In Neural Information Processing Systems*. Advances in Neural Information Processing Systems 25 (NIPS 2012), 2012. doi: http://dx.doi.org/10.1016/j.protcy.2014.09.007. ISBN 9781627480031. ISSN 10495258 pp. 1–9.

[40] T. Ching, D. S. Himmelstein, B. K. Beaulieu-Jones, A. A. Kalinin, B. T. Do, G. P. Way, E. Ferrero, P.-M. Agapow, M. Zietz, M. M. Hoffman, W. Xie, G. L. Rosen, B. J. Lengerich, J. Israeli, J. Lanchantin, S. Woloszynek, A. E. Carpenter, A. Shrikumar, J. Xu, E. M. Cofer, C. A. Lavender, S. C. Turaga, A. M. Alexandari, Z. Lu, D. J. Harris, D. DeCaprio, Y. Qi, A. Kundaje, Y. Peng, L. K. Wiley, M. H. S. Segler, S. M. Boca, S. J. Swamidass, A. Huang, A. Gitter, and C. S. Greene, "Opportunities and obstacles for deep learning in biology and medicine," *Journal of The Royal Society Interface*, vol. 15, no. 141, 2018. doi: 10.1098/rsif.2017.0387

[41] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. van der Laak, B. van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," *Medical Image Analysis*, vol. 42, pp. 60–88, 2017. doi: 10.1016/j.media.2017.07.005

[42] S. J. Morrison, "Time to do something about reproducibility," *eLife*, vol. 3, no. e03981, 2014. doi: 10.7554/eLife.03981

[43] T. M. Errington, E. Iorns, W. Gunn, F. E. Tan, J. Lomax, and B. A. Nosek, "An open investigation of the reproducibility of cancer biology research," *eLife*, vol. 3, no. e04333, 2014. doi: 10.7554/eLife.04333

[44] "The challenges of replication," *eLife*, vol. 6, no. e23693, 2017. doi: 10.7554/eLife.23693

[45] S. Schmidt, "Shall we really do it again? The powerful concept of replication is neglected in the social sciences." *Review of General Psychology*, vol. 13, no. 2, pp. 90–100, 2009. doi: 10.1037/a0015108

[46] L. B. Balsam, A. J. Wagers, J. L. Christensen, T. Kofidis, I. L. Weissman, and R. C. Robbins, "Haematopoietic stem cells adopt mature haematopoietic fates in ischaemic myocardium," *Nature*, vol. 428, no. 6983, pp. 668–673, 4 2004. doi: 10.1038/nature02460

[47] D. E. Wright, "Physiological Migration of Hematopoietic Stem and Progenitor Cells," *Science*, vol. 294, no. 5548, pp. 1933–1936, 2001. doi: 10.1126/science.1064081

[48] "Reality check on reproducibility," *Nature*, vol. 533, no. 7604, p. 437, 2016. doi: 10.1038/533437a

[49] S. S. H. Hofvind, H. Wang, and S. Thoresen, "The Norwegian breast cancer screening program: Re-attendance related to the women's experiences, intentions and previous screening result," *Cancer Causes and Control*, vol. 14, no. 4, pp. 391–398, 2003. doi: 10.1023/A:1023918610664

[50] "The Cancer Registry of Norway." [Online]. Available: https://www.kreftregisteret.no/en/General/About-the-Cancer-Registry/About-the-organization/

[51] Kierstan Boyd, "What Is Diabetic Retinopathy?" 2017. [Online]. Available: https://www.aao.org/eye-health/diseases/what-is-diabetic-retinopathy

[52] M. Voets, K. Møllersen, and L. A. Bongo, "Replication study: Development and validation of deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs," *arXiv:1803.04337 [cs.CV]*, 2018.

[53] P. Mildenberger, M. Eichelberg, and E. Martin, "Introduction to the DICOM standard," *European Radiology*, vol. 12, no. 4, pp. 920–927, 2002. doi: 10.1007/s003300101100

[54] D. Peck, "Digital Imaging and Communications in Medicine (DICOM): A Practical Introduction and Survival Guide," *Journal of Nuclear Medicine*, vol. 50, no. 8, pp. 1384–1384, 2009. doi: 10.2967/jnumed.109.064592

[55] CHOP, "Dicom-anon: Python DICOM Anonymizer," 2016. [Online]. Available: https://github.com/chop-dbhi/dicom-anon

[56] ——, "Children's Hospital of Philadelphia." [Online]. Available: http://www.chop.edu/about-us

[57] National Electrical Manufacturers Association, "Digital Imaging and Communications in Medicine (DICOM) Part 15: Security and System Management Profiles," pp. 85–92, 2011. [Online]. Available: ftp://medical.nema.org/medical/dicom/2011/11_15pu.pdf

[58] Sage Bionetworks, "The Digital Mammography DREAM Challenge," 2016. [Online]. Available: https://www.synapse.org/#!Synapse:syn4224222/wiki/401745

[59] Google, "Citations for Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs," 2018. [Online]. Available: https://scholar.google.no/scholar?cites=16083985573643781536

[60] E. Decencière, X. Zhang, G. Cazuguel, B. Lay, B. Cochener, C. Trone, P. Gain, R. Ordonez, P. Massin, A. Erginay, B. Charton, and J.-C. Klein, "Feedback on a publicly distributed image database: the Messidor database," *Image Analysis & Stereology*, vol. 33, no. 3, p. 231, 2014. doi: 10.5566/ias.1155

[61] G. Quellec, M. Lamard, P. M. Josselin, G. Cazuguel, B. Cochener, and C. Roux, "Optimal Wavelet Transform for the Detection of Microaneurysms in Retina Photographs," *IEEE Transactions on Medical Imaging*, vol. 27, no. 9, pp. 1230–1241, 2008. doi: 10.1109/TMI.2008.920619

[62] Kaggle, "Diabetic Retinopathy Detection (Data)," 2015. [Online]. Available: https://www.kaggle.com/c/diabetic-retinopathy-detection/data

[63] Messidor, "Methods to evaluate segmentation and indexing techniques in the field of retinal ophthalmology," 2004. [Online]. Available: http://www.adcis.net/en/Download-Third-Party/Messidor.html

[64] Kaggle, "Diabetic Retinopathy Detection," 2015. [Online]. Available: https://www.kaggle.com/c/diabetic-retinopathy-detection

[65] A. Rakhlin, "Diabetic Retinopathy detection through integration of Deep Learning classification framework," *bioRxiv*, 2017. doi: 10.1101/225508

[66] C. Wilkinson, F. L. Ferris, R. E. Klein, P. P. Lee, C. D. Agardh, M. Davis, D. Dills, A. Kampik, R. Pararajasegaram, J. T. Verdaguer, and Global Diabetic Retinopathy Project Group, "Proposed international clinical diabetic retinopathy and diabetic macular edema disease severity scales," *Ophthalmology*, vol. 110, no. 9, pp. 1677–1682, 2003. doi: 10.1016/S0161-6420(03)00475-5

[67] L. Bottou, "Large-Scale Machine Learning with Stochastic Gradient Descent," in *Proceedings of COMPSTAT'2010*. Heidelberg: Physica-Verlag HD, 2010, pp. 177–186. ISBN 978-3-7908-2603-6

[68] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *arXiv:1502.03167 [cs.LG]*, 2015. doi: 10.1007/s13398-014-0173-7.2

[69] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015. doi: 10.1007/s11263-015-0816-y

[70] R. Caruana, S. Lawrence, and L. Giles, "Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping," in *the 13th International Conference on Neural Information Processing Systems*. Advances in Neural Information Processing Systems 13 (NIPS 2000), 2000. doi: 10.1109/IJCNN.2000.857823. ISBN 1049-5258. ISSN 10495258 pp. 402–408.

[71] S. Theodorides and K. Koutroumbas, *Pattern Recognition*. Academic Press, 2009. ISBN 978-1-59749-272-0

[72] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems," *arXiv:1603.04467 [cs.DC]*, 2016.

[73] "TensorFlow: Benchmarks." [Online]. Available: https://www.tensorflow.org/performance/benchmarks

[74] E. Bugliarello, "Distributed Tensorflow Benchmarks," 2017. [Online]. Available: https://github.com/e-bug/distributed-tensorflow-benchmarks

[75] G. D. Tourassi, M. K. Markey, J. Y. Lo, and C. E. Floyd, "A neural network approach to breast cancer diagnosis as a constraint satisfaction problem," *Medical Physics*, vol. 28, no. 5, pp. 804–811, 2001. doi: 10.1118/1.1367861

[76] A. Papadopoulos, D. I. Fotiadis, and A. Likas, "An automatic microcalcification detection system based on a hybrid neural network classifier," *Artificial Intelligence in Medicine*, vol. 25, no. 2, pp. 149–167, 2002. doi: 10.1016/S0933-3657(02)00013-1

[77] T. Ayer, Q. Chen, and E. S. Burnside, "Artificial Neural Networks in Mammography Interpretation and Diagnostic Decision Making," *Computational and Mathematical Methods in Medicine*, vol. 2013, pp. 1–10, 2013. doi: 10.1155/2013/832509

[78] T. Ayer, O. Alagoz, J. Chhatwal, J. W. Shavlik, C. E. Kahn, and E. S. Burnside, "Breast cancer risk estimation with artificial neural networks revisited: Discrimination and calibration," *Cancer*, vol. 116, no. 14, pp. 3310–3321, 2010. doi: 10.1002/cncr.25081

[79] Y. Wu, M. L. Giger, K. Doi, C. J. Vyborny, R. A. Schmidt, and C. E. Metz, "Artificial neural networks in mammography: Application to decision making in the diagnosis of breast cancer," *Radiology*, vol. 187, no. 1, pp. 81–87, 1993. doi: 10.1148/radiology.187.1.8451441

[80] J. A. Baker, P. J. Kornguth, J. Y. Lo, M. E. Williford, and C. E. Floyd, "Breast cancer: prediction with artificial neural network based on BI-RADS standardized lexicon." *Radiology*, vol. 196, no. 3, pp. 817–822, 1995. doi: 10.1148/radiology.196.3.7644649

[81] H. P. Chan, S. C. B. Lo, B. Sahiner, K. L. Lam, and M. A. Helvie, "Computer-aided detection of mammographic microcalcifications: Pattern recognition with an artificial neural network," *Medical Physics*, vol. 22, no. 10, pp. 1555–1567, 1995. doi: 10.1118/1.597428

[82] G. Rezai-rad and S. Jamarani, "Detecting microcalcification clusters in digital mammograms using combination of wavelet and neural network," in *International Conference on Computer Graphics, Imaging and Visualization (CGIV'05)*. IEEE, 2005. doi: 10.1109/CGIV.2005.30. ISBN 0-7695-2392-7 pp. 1–5.

[83] J. Y. Lo, J. A. Baker, P. J. Kornguth, and J. Floyd, "Effect of patient history data on the prediction of breast cancer from mammographic findings with artificial neural networks," *Academic Radiology*, vol. 6, no. 1, pp. 10–15, 1999. doi: 10.1016/S1076-6332(99)80056-7

[84] D. Soriano, C. Aguilar, I. Ramirez-Morales, E. Tusa, W. Rivas, and M. Pinta, "Mammogram Classification Schemes by Using Convolutional Neural Networks," in *Communications in Computer and Information Science*. Springer, Cham, 2018, vol. 798, pp. 71–85. ISBN 9783319727264

[85] L. Zhang, W. Qian, R. Sankar, D. Song, and R. Clark, "A new false positive reduction method for MCCs detection in digital mammography," *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, vol. 2, pp. 1033–1036, 2001. doi: 10.1109/ICASSP.2001.941095

[86] J. L. Jesneck, J. Y. Lo, and J. A. Baker, "Breast Mass Lesions: Computer-aided Diagnosis Models with Mammographic and Sonographic Descriptors," *Radiology*, vol. 244, no. 2, pp. 390–398, 2007. doi: 10.1148/radiol.2442060712

[87] Y. Wu, K. Doi, M. L. Giqer, and R. M. Nishikawa, "Computerized detection of clustered microcalcifications in digital mammograms Applications of artificial neural networks," *Medical Physics*, vol. 19, no. 3, pp. 555–560, 1992. doi: 10.1118/1.596845

[88] Y. Yuan, M. L. Giger, H. Li, N. Bhooshan, and C. A. Sennett, "Multimodality Computer-Aided Breast Cancer Diagnosis with FFDM and DCE-MRI," *Academic Radiology*, vol. 17, no. 9, pp. 1158–1167, 2010. doi: 10.1016/j.acra.2010.04.015

[89] S. Shi, Q. Wang, P. Xu, and X. Chu, "Benchmarking State-of-the-Art Deep Learning Software Tools," in *2016 7th International Conference on Cloud*

Computing and Big Data (CCBD), 8 2016. doi: 10.1109/CCBD.2016.029. ISBN 978-1-5090-3555-7. ISSN 978-1-5090-3555-7 pp. 99–104.

[90] R. G. Stafford, J. Beutel, and D. J. Mickewich, "Application of neural networks to computer-aided pathology detection in mammograms," *Proc.SPIE*, vol. 1905, p. 1898, 1993. doi: 10.1117/12.154606

[91] Y. Jiang, R. M. Nishikawa, D. E. Wolverton, C. E. Metz, M. L. Giger, R. A. Schmidt, C. J. Vyborny, and K. Doi, "Malignant and benign clustered microcalcifications: automated feature analysis and classification," *Radiology*, vol. 198, no. 3, pp. 671–678, 1996. doi: 10.1148/radiology.198.3.8628853

[92] J. V. Tu, "Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes," *Journal of Clinical Epidemiology*, vol. 49, no. 11, pp. 1225–1231, 1996. doi: 10.1016/S0895-4356(96)00002-9

[93] S. C. B. Lo, H. P. Chan, J. S. Lin, H. Li, M. T. Freedman, and S. K. Mun, "Artificial convolution neural network for medical image pattern recognition," *Neural Networks*, vol. 8, no. 7-8, pp. 1201–1214, 1995. doi: 10.1016/0893-6080(95)00061-5

[94] R. H. Nagel, R. M. Nishikawa, J. Papaioannou, and K. Doi, "Analysis of methods for reducing false positives in the automated detection of clustered microcalcifications in mammograms," *Medical Physics*, vol. 25, no. 8, pp. 1502–1506, 1998. doi: 10.1118/1.598326

[95] Z. Huo, M. L. Giger, C. J. Vyborny, and C. E. Metz, "Breast cancer: effectiveness of computer-aided diagnosis observer study with independent database of mammograms." *Radiology*, vol. 224, no. 2, pp. 560–568, 2002. doi: 10.1148/radiol.2242010703

[96] J. Wei, B. Sahiner, L. M. Hadjiiski, H.-P. Chan, N. Petrick, M. A. Helvie, M. A. Roubidoux, J. Ge, and C. Zhou, "Computer-aided detection of breast masses on full field digital mammograms," *Medical Physics*, vol. 32, no. 9, pp. 2827–2838, 2005. doi: 10.1118/1.1997327

[97] M. Kallergi, "Computer-aided diagnosis of mammographic microcalcification clusters," *Medical Physics*, vol. 31, no. 2, pp. 314–326, 2004. doi: 10.1118/1.1637972

[98] H.-P. Chan, B. Sahiner, M. A. Helvie, N. Petrick, M. A. Roubidoux, T. E. Wilson, D. D. Adler, C. Paramagul, J. S. Newman, and S. Sanjay-Gopal, "Im-

provement of Radiologists' Characterization of Mammographic Masses by Using Computer-aided Diagnosis: An ROC Study," *Radiology*, vol. 212, no. 3, pp. 817–827, 1999. doi: 10.1148/radiology.212.3.r99au47817

[99] Katie Planey, "Machine Learning Approaches to Breast Cancer Diagnosis and Treatment Response Prediction," 2011. [Online]. Available: http://cs229.stanford.edu/proj2011/Planey-Machine%20Learning%20Approaches%20to%20Breast%20Cancer%20Diagnosis%20and%20Treatment%20Response%20Prediction.pdf

[100] M. L. Brown, F. Houn, E. A. Sickles, and L. G. Kessler, "Screening mammography in community practice: Positive predictive value of abnormal findings and yield of follow-up diagnostic procedures," *American Journal of Roentgenology*, vol. 165, no. 6, pp. 1373–1377, 1995. doi: 10.2214/ajr.165.6.7484568

[101] M. J. Schell, B. C. Yankaskas, R. Ballard-Barbash, B. F. Qaqish, W. E. Barlow, R. D. Rosenberg, and R. Smith-Bindman, "Evidence-based Target Recall Rates for Screening Mammography 1," *Radiology*, vol. 243, no. 3, pp. 681–689, 6 2007. doi: 10.1148/radiol.2433060372

[102] Y. Fong, J. Evans, D. Brook, J. Kenkre, P. Jarvis, and K. Gower-Thomas, "The Nottingham Prognostic Index: Five- and ten-year data for all-cause survival within a screened population," *Annals of the Royal College of Surgeons of England*, vol. 97, no. 2, pp. 137–139, 2015. doi: 10.1308/003588414X14055925060514

[103] J. Suckling, J. Parker, D. Dance, S. Astley, I. Hutt, C. Boggis, I. Ricketts, E. Stamatakis, N. Cerneaz, S. Kok, P. Taylor, D. Betal, and J. Savage, "The Mammographic Image Analysis Society Digital Mammogram Database," *Experta Medica, International Congress Series*, vol. 1069, no. January 1994, pp. 375–378, 1994.

[104] C. Olston, N. Fiedel, K. Gorovoy, J. Harmsen, L. Lao, F. Li, V. Rajashekhar, S. Ramesh, and J. Soyke, "TensorFlow-Serving: Flexible, High-Performance ML Serving," *arXiv:1712.06139 [cs.DC]*, 2017.