UiT The Arctic University of Norway

Faculty of Science and Technology
Department of Physics and Technology

Model and Data Diagnosis under Limited Supervision in Modern AI

Rwiddhi Chakraborty

A dissertation for the degree of Philosophiae Doctor

September 2024

This thesis document was typeset using the *UiT Thesis LaTeX Template*.
© 2024 – `http://github.com/egraff/uit-thesis`

"The most amazing combinations can result if you shuffle the pack enough."

–Koroviev, *The Master and Margarita*

Abstract

Deep Learning in modern Artificial Intelligence (AI) has witnessed unprecedented success on a variety of domains over the past decade, ranging from computer vision to natural language reasoning tasks. This success is owed primarily to the availability of large, annotated datasets, the existence of powerful mathematical models, and the mechanism to train large models on such data with advanced resources of compute. However, this success has led to increased scrutiny on the failure points of models trained on suspect data. Issues such as model and data bias, reliance on spurious correlations, and poor generalization capability on challenging test data, to name a few, have surfaced in the research community. As a result, it seems imperative to diagnose such systems for generalization performance on challenging test data, and uncovering potential biases hidden in datasets. In this thesis, we address these key challenges through the following directions: first, in the generalization capabilities with limited labeled data - few-shot learning, semi-supervised learning, and unsupervised learning. Second, towards bias discovery in existing models and datasets, particularly in unsupervised group robust learning, and debiased synthetic data generation. Our two broad directions are encapsulated by a common challenge: the paucity of labeled data, since manually annotating large datasets is a time consuming and expensive process for humans. This motivation is relevant today due to the exponential growth in the sizes of models and datasets in use. It is becoming more and more intractable for humans to annotate billions of data points, leading to large benchmark datasets that are not well calibrated with human expectations on fairness. These issues, if left unchecked, are inevitably exacerbated when models train on such datasets. We consider these two directions, i.e. model generalization with limited labels, and the existence of biased data, to be two sides of the same coin, and thus coin the framework encapsulating such research as Model and Data Diagnosis. This work proposes novel contributions in few-shot learning, semi-supervised learning, unsupervised learning, and in data diagnosis and debiasing techniques. Further, we show that model and data diagnosis do not necessarily exist as disparate entities, and can be viewed in a co-dependent context. Finally, this thesis hopes to amplify the scrutiny surrounding model capabilities, however impressive, trained on datasets, however vast.

Acknowledgements

I would like to thank my supervisor, Professor Michael Kampffmeyer. Three years ago, I did not know how to do research. Three years later, whatever I have learned over this time, big or small, significant or insignificant, I owe to Michael.

I would like to thank my co-supervisors, Associate Professor Benjamin Ricaud and Professor Robert Jenssen, for their research insights in our collaborative work, and for always being available when I needed specific brainstorming sessions.

I would like to thank my colleagues at UiT and CMU, who have made me better than what was possible alone.

I would like to thank my parents, for whom the life of the mind superseded in satisfaction any other life.

I would like to thank Rito Maitra, my greatest friend in the world. He claims to harbor a deep interest in my work (He has only read the abstract).

Finally, to Tiziana, the love of my life. We met, as the movie goes, at a rather strange time in our lives. Come what may, we will always have Pittsburgh.

Rwiddhi Chakraborty,
Kolkata. September 13th, 2024.

Contents

List of Figures

/1 Introduction

We live in an unprecedented era where machines are proving to be capable at a variety of cognitive tasks previously relegated to humans. These tasks such as identifying images, parsing information from text, video understanding, sentiment understanding, etc. were previously thought to be difficult for non-humans owing to a belief that a deeper, erstwhile unknown intuition is at play for such tasks [106, 31, 99, 4]. As a result, near superhuman performance of computer algorithms on a wide variety of real world tasks invites a fresh spectrum of scrutiny from a diverse group of institutions: academic, financial, governmental, policy, and venture capital [17, 1, 13, 100, 57]. While the full bandwidth of public opinion also accommodates claims from science fiction, in essence we observe a general unrest in the zeitgeist - some are excited, some are anxious, others are curious, very few are uninterested. A large part of this progress is attributable to a list of simple ingredients - hardware, programming, data, and the existence of powerful mathematical models. It turns out that all of these ingredients have witnessed an explosive growth in usefulness in the past decade. Graphics Processing Units (GPUs), previously used in high performance computing and gaming, began to be adopted widely to carry out large batches of matrix multiplication, a process at the heart of most of the progress we see today. This was not possible without the programming of the CUDA platform [84], and the subsequent development of useful libraries to programs in, with CUDA running in the backend, such as PyTorch [87]. Finally, it was also possible to curate, annotate, and store large quantities of data in a structured fashion. One early, successful example of data curation is ImageNet [30]: a dataset of images encompassing around 21000 unique objects. The object names were extracted in a structured, pre-existing hierarchy called WordNet [36], which allowed for a convenient way to group together semantically similar classes. ImageNet, in its original form, contains close to 14 million images, all manually annotated by humans. Since then, ImageNet has turned into among the most frequently used datasets in computer vision research. In addition to hardware, programming and data, a set of mathematical models also proved to be useful. For images, and sequential data (such as text) in particular, a variety of older architectures invited renewed interest in the community, owing to the feasibility of programming such architectures. Architectures such as the Convolutional Neural Network (CNN) [70], the Long Short Term Memory (LSTM) [51], and symbolic computation techniques such as autodifferentiation [7], all techniques from previous decades, suddenly proved to be computationally feasible. Recently, more novel architectures have been developed in various modalities, such as the transformer, the vision transformer, and ConvNeXT, to name a few [113, 33, 78]. The *Model*, then, is a parameterized abstraction that is stored as chunks of a large matrix of vectors. This matrix contains the internal model representations of the data. To achieve such representations, the *Model* is *trained* on a dataset: an iterative process that occurs in *epochs*, where the Model sees the same set of images as time passes by, constantly updating its internal parameterization. Thus was born the *Deep Learning* era, where a model could start from a random parameterization and iteratively update its parameters to gain improved guesses on the image object it processes (the learning process). In the case of computer vision, the base network the model used, the CNN, was created by stacking a set of convolutional *layers*, thus leading to the moniker *deep* learning - an architecture with many layers.

AlexNet [65] was among the early successes of the deep learning paradigm, demonstrating significantly improved results on ImageNet when compared to its baselines of a bygone era. Over the years, more such architectures were developed, in addition to more benchmarks, datasets, tasks, domains, and so on. The exponential adoption of this paradigm is attributable to the flexibility of the architectures in use — One did not need to preset a list of handwritten rules to teach a machine how to perform a task. Instead, one could engineer a way to solve this task: a reasonable parameterization, large quantities of data and hardware capabilities. More recently, with performance saturation on standard benchmarks in vision, text, and video [30, 73, 59], the introduction of foundation models [12] has led to a new, post-ImageNet wave in deep learning. The post-ImageNet wave is not simply the introduction of new datasets. It is the introduction of entirely new architectures and tasks as well [16, 77].

In addition to learning and prediction tasks, *generative* tasks are also witnessing an exponential growth [39, 63]. One set of models that have captured the public imagination today are diffusion-based models— given a text prompt, these models can generate images that adhere closely to the prompt, leading to hyperrealistic image generation [50, 93]. Quite clearly, such developments resemble a space akin to an epistemological wild west - is knowledge simply a set of vector embeddings? Does knowledge and reasoning emerge from pure randomness? What are the ontologies learned by modern deep learning techniques vis-a-vis human ontologies? While this thesis is not concerned about these questions, we believe these questions necessitate the remarkable interest generated by such models in the zeitgeist today.

1.1 Learning with limited labels

A natural consequence of such paradigm shifts within the space of a decade is the price to pay for training: models have gotten exponentially larger, and so have datasets [16, 58, 52, 123].

This exponential increase in model and dataset sizes begs the central motivation of the thesis: diagnosis of the model, and the data, i.e. investigating model understanding and generalizability, in addition to understanding key properties of the datasets such models train on, is of tantamount importance. This thesis is also concerned with the economy of learning - the proliferation of deep learning-based architectures in computer vision, language modelling, and graph-based data in the past decade is owed mostly to the availability of large-scale datasets in each of these data modalities. These datasets are often highly curated, and more importantly, *annotated* (assigning labels to data samples), for training large models with billions of parameters. However, the process encompassing dataset collection, curation, and annotation is expensive [107]. In recent years, therefore, there has emerged an increased focus on developing novel methodologies to design architectures and training paradigms that succeed after training on unlabeled/partially labeled data, deeming the process of learning to be 'self/semi'-supervised [25, 42, 46, 19]. This training paradigm involves diverse applications on a variety of downstream tasks such as recognition, detection, and clustering, to name a few. The broad goal of the (semi/self)-supervised learning paradigm is to develop methodologies that encode a generalized representation in a data modality (for example image, text, graph), i.e., a representation that can then be used successfully on a variety of downstream tasks. Additionally, the performance of any such methodology must be carefully calibrated with understanding why a model makes a certain prediction in the first place, the domain of explainability research [92, 80, 74]. Models should not only be powerful, but their predictions should also be explainable to human evaluators in safety-critical applications such as healthcare. A variety of model explainability techniques have been successful in revealing previously unknown biases in deep learning models [98, 6].

1.2 Spurious Correlations and Bias

Models are not necessarily decoupled from the bias of the data they are trained on. Therefore, in addition to model explainability, we also consider the issue of dataset bias. In short, what are certain repetitive patterns found in data that are potentially harmful for a downstream task? These patterns are called *spurious correlations*, or biases, and can exist in a variety of forms: textural [38], shape-based [54], scale-based [103], object-based [37], and social [18, 11]. An example of a spurious correlation in a dataset is a watermark [68]. This is a subtle cue that may be ignored by human evaluators, but as it turned out, models training on such images frequently picked up on the watermark to make their decisions. This is unreasonable since a watermark should not causally predict the object present in an image. It is a spurious feature that should be ignored. This phenomenon exists under various names in the literature—Shortcut learning [37], Clever Hans effect [68], and a large body of literature exists to mitigate such spurious correlations [5]. Another, possibly more serious issue of dataset bias involves the multitude of *social* biases that exist in modern datasets. ImageNet, for instance, has a stagnant concept vocabulary, a limited diversity in objects represented (even though there are about 21000 unique objects), and a string of racial biases [105]. MS-COCO [124], another oft-used dataset in machine learning, has significant gender bias in its images and captions, and CelebA [79], a database of faces, has a set of features that reinforce racial and gender stereotypes [20]. We note that none of these three datasets are fringe datasets or rarely used. These are among the most common datasets being used in a variety of machine learning research areas today. While some of these biases are, until today, still being discovered, a large gamut of unknown biases exist. More recent, larger datasets such as LAION-5B [97], are also under heavy scrutiny for possible biases [9].

1.3 Model and Data Diagnosis

As a result, this thesis is centred around *Model and Data Diagnosis under Limited Supervision* - through our contributions, we hope to paint a holistic, inter-connected picture of how model diagnosis, that includes generalization and robustness with limited labels, interacts with data diagnosis - uncovering the spurious correlations and inherent structure of datasets in a principled examination. As models and datasets grow larger with time, it is imperative to take a closer look into what large datasets in the wild have in store for us, and how this affects models that train on such datasets.

Broadly, we hope such an investigation helps shed light on certain directions of future work in reasoning capabilities and data bias.

1.4 Thesis objectives and contributions

In summary, the objectives of this thesis are the following:

- To develop new techniques in learning with limited labeled data, i.e. few-shot learning, semi-supervised learning, and unsupervised learning.

- To develop novel techniques to perform model and data debiasing to improve generalization and robustness.

- To reframe such objectives within the framework of model and data diagnosis, and to show that apparently disparate research areas such as model generalization, debiasing, explainability, can all be encapsulated in such a framework.

The contributions of this thesis are as follows:

- A novel technique to perform unsupervised group robustness using model explainability heatmaps. This is the subject of Paper I.

- A novel representation learning method on the geometry of the hypersphere to achieve state-of-the-art results in few-shot transductive classification. This is the subject of Paper II.

- A novel data debiasing framework that represents object co-occurrence-based biases in visual datasets. This is the subject of Paper III.

- A novel connection between oversmoothing in graph neural networks and disentangled representations in transductive semi-supervised node classification. This is the subject of Paper IV.

We present a schematic of the thesis in Figure 1.1.

1.5 List of publications

We present the list of papers that form the core of this thesis. These include published papers, submission-ready manuscripts, and submissions currently under review:

Paper I Rwiddhi Chakraborty, Adrian Sletten, and Michael Kampffmeyer. "ExMap: Leveraging Explainability Heatmaps for Unsupervised Group Robust-

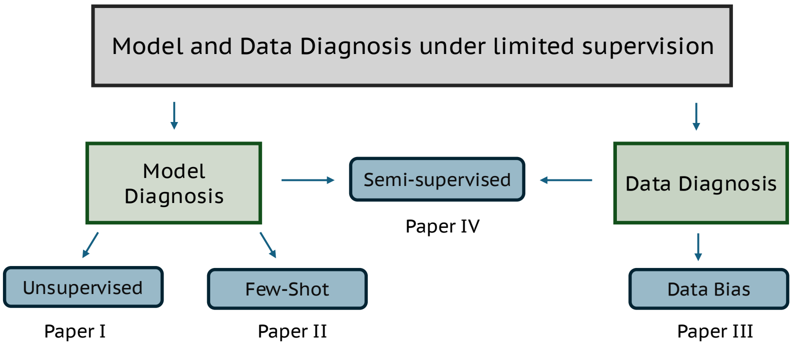


Figure 1.1: An outline of this thesis. Here, we relate how the different research contributions connect to each other in the broader framework of model and data diagnosis under limited supervision.

ness to Spurious Correlations". In: *CVPR*. *2024*.

Paper II Daniel J. Trosten*, Rwiddhi Chakraborty*, Sigurd Løkse, Kristoffer Wickstrøm, Robert Jenssen, and Michael Kampffmeyer. "Hubs and Hyperspheres: Reducing Hubness and Improving Transductive Few-shot Learning with Hyperspherical Embeddings". In: *CVPR*. *2023* [* denotes equal contribution].

Paper III Rwiddhi Chakraborty, Oliver Wang, Jialu Gao, Cheng Zhang, Runkai Zheng, and Fernando de la Torre. "Visual Data Diagnosis and Debiasing with Concept Graphs". In: *(Under Review)*.

Paper IV Rwiddhi Chakraborty, Benjamin Ricaud, Robert Jenssen, and Michael Kampffmeyer. "On Disentangled Representations and the Oversmoothing Problem in Graph Convolutional Networks". *(Submission Ready)*.

The thesis is structured in the following way: the first section, titled *Learning and Data*, discusses relevant topics in statistical learning, deep learning, explainability, and model and data diagnosis. The second section, titled *Summary of Research*, briefly discusses the papers presented in this work. The third section, titled *Conclusion and Future Work*, presents final remarks and interesting areas of future work. The final section, titled *Included Papers*, contains the full papers that form the core of the thesis.

Part I

Learning and Data

2

The Learning Problem

Since this work heavily uses terms from deep learning and statistical learning, here we provide some preliminaries in learning theory. The topics discussed here are relevant to generalization, robustness, and bias as discussed later in the thesis.

A learning machine refers to an algorithm that can learn from data. While a formal, universally accepted definition of learning does not exist, an oft-used definition that serves most purposes was provided by Mitchell [81]: *A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E.*

The Experience E refers to the dataset under investigation, the task T refers to the particular problem at hand, and the performance measure P refers to a quantitative metric to assess the quality of the learning algorithm on the task-specific dataset at hand.

Since the definition of learning is presupposed by the existence of a task, we identify two popular examples of tasks of interest: *classification* and *regression*. In the classification problem, the learning machine aims to identify a correct category of an object from a set of possible categories. For instance, in image classification, the learning machine may be provided an image of an object, and asked to identify the object. In the regression problem, the learning machine aims to output a continuous value relevant to the task at hand, e.g given a set of housing prices of a variety of houses, the machine may be asked to predict the price of a unique house.

In the Classification context, one dataset (Experience) of particular interest is ImageNet [30]. There are 21000 unique objects in this dataset, and the ImageNet Challenge tasked model designers to unlock state-of-the-art performance on the data. In recent years, deep learning approaches have achieved human-parity performance on ImageNet. In the regression context, an example of a dataset would be the Boston Housing Prices Dataset [44], the challenge of predicting house prices based on features such as size, location, etc. In regression problems, tree-based boosting and ensemble methods have proven to be quite popular in large-scale online applications [24, 60].

The definitions of the Experience and the task T allow us to formalize these notions further. We assume we are given a dataset $X = \{x_1, x_2, \ldots x_n\}$ of N points. The learning machine then approximates a function $f(x)$ on the data, and outputs a prediction y. Depending upon T, the performance measure P can now be defined. In the case of classification, given a set of discrete classes C, the cross-entropy loss L_{CE} is commonly used:

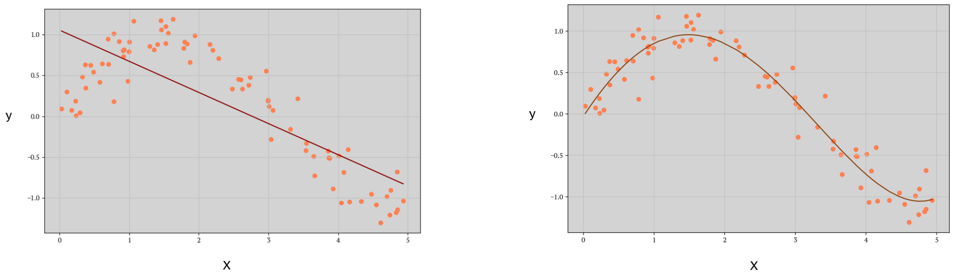
$$L_{CE} = -\frac{1}{N} \sum_{j=1}^{N} \sum_{i=1}^{C} y_{ij} \log(\hat{y}_{ij}). \tag{2.1}$$

where y and $\hat{y}$ are the true labels and predicted labels respectively. The cross-entropy is the expected loss over all datapoints, for all classes in the data. In the case of regression, the *Mean Squared Error (MSE)* is common:

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2. \tag{2.2}$$

Both L_{CE} and MSE can be considered examples of performance measure P. However, these are certainly not the only measures available. The performance measure is defined completely by the existence of data points, a model that approximates a certain function of the given data, and based on this approximation outputs a certain value. The performance measure simply dictates how good this value is in context of the task at hand.

Learning machines are function approximators of data. A model is said to be powerful if its function approximates data in an acceptable fashion. As a result, models are differentiated by the space of functions they encompass. As we will see in the next section, and in a later part of the thesis, the choice of function approximator makes a significant difference in the task at hand.



(a) Linear Fit [w = -0.382, and a = 1.056].

(b) A four degree polynomial fit.

Figure 2.1: Comparison of Linear and Polynomial Fits on a sinusoidal dataset with a random normal noise distribution. Choosing a polynomial model results in a better fit.

Next, we discuss why a correct choice of a function approximation is necessary for learning machines, i.e. we discuss the impact on generalization capabilities of such models on given data.

2.1 Generalization

Once a model takes as input a set of data points (*training data*), how well does it perform on data points that it has not seen before (*test data*)? This is the generalization problem. We illustrate this problem with a simple regression example.

In Figure 2.1, we plot a series of points using a sinusoidal curve with a random normal noise distribution. A natural question to ask is what function can reasonably capture the structure of the data. This process is called *fitting the model to the data*. If we assume a linear model family:

$$f(x) = a + wx. \tag{2.3}$$

where a is a constant (can be interpreted as an error term, or in terms of statistical learning, the *residual*), we can see that such a linear family admits various choices of models, depending upon the parameterization of w. Given that we choose a particular model, its generalization capability could then be measured based on the performance measure (MSE) on the test points, as shown in the figure.

Further, we are not restricted to choosing a linear model family. In fact, in

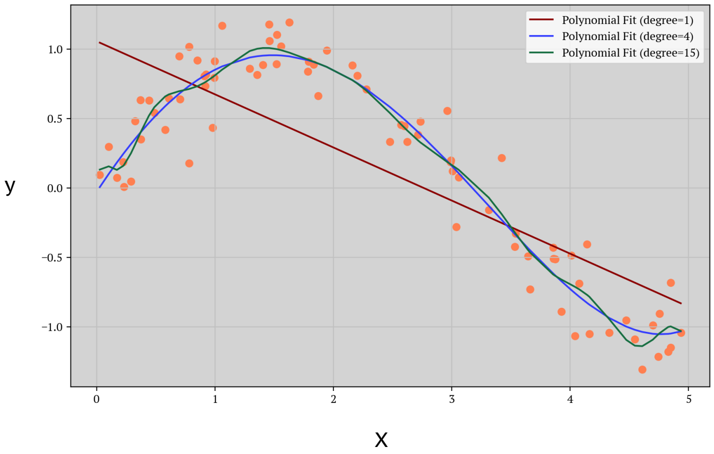


Figure 2.2: The overfitting phenomenon: the highest degree polynomial starts capturing the random noise in the data by trying to fit exact points in the training set. The linear model underfits the data, while the polynomial of degree four seems to be an appropriate compromise.

Figure 2.1, it is intuitively seen that a linear regression may not be appropriate, given the non-linear structure of the data. In this case, we could choose a polynomial based approach:

$$f(x) = a + w_1 x + w_2 x^2. \tag{2.4}$$

We note the increase in the number of parameters, and the subsequent improvement in the quality of the fit. In principle, one could choose parameters for even higher orders of the data, but this approach is not guaranteed to infinitely produce better performance. The reason that over-parameterization fails after a while is due to the *overfitting* issue in statistical learning. In general, fitting an over-parameterized model with low-dimensional data leads to the model capturing all points, even points which are anomalous, or noise, in the training data. This leads to a low MSE value. However, since the model has captured spurious points in the input, it leads to a high MSE when the model is evaluated on the test data. The growing gap between the training and test MSE is the *overfitting* phenomenon, illustrated in Figure 2.2. Conversely, when a model cannot sufficiently capture the information in the input data and performs poorly on the test data, we deem the model to *underfit* the data, and it would be useful to choose more parameters to fit the training data.

This tradeoff between the amount of parameterization chosen for a model (the model *capacity*), and the impact on error-rate is called the *bias-variance tradeoff*, illustrated in Figure 2.3. Essentially, under-parameterized models exhibit higher

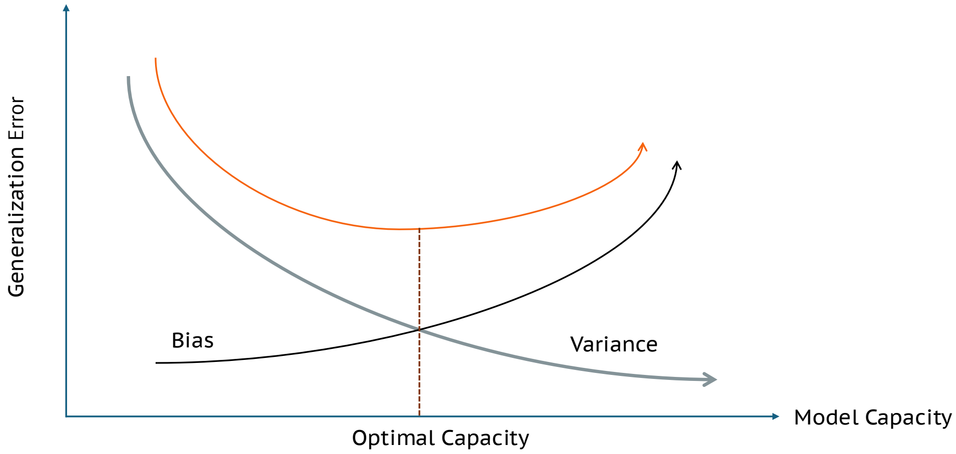


Figure 2.3: The bias-variance tradeoff: beyond an optimal model capacity, generalization capability on test data worsens.

bias (a linear regression model assumes linearity in the data), while over-parameterized models exhibit higher variance (a polynomial of a high order), and the acceptable generalization performance is a tradeoff between these two choices. Given our chosen function approximator $F(x)$, the true function $f(x)$, and a dataset D, this tradeoff can be formulated as the average MSE on the test data:

$$E(f(x) - F(x))^2 = \sigma^2(F(x)) + [Bias(F(x))]^2 + \sigma^2(\epsilon). \qquad (2.5)$$

where $[Bias(F(x))]^2 = f(x) - E(F(x))$, and ϵ is the inevitable error variance in estimation. Given that we desire a minimal test MSE, our ideal model should have *low bias* and *low variance*.

In summary, the bias-variance tradeoff has a significant impact on the classifier's generalization capabilities. A propensity to pick up spurious training points in the form of noise or points that are causally unrelated to the labels, will hinder generalization capabilities.

2.2 Error Bounds

In addition to the expected test error, statistical learning theory has a useful result to provide tight bounds on the empirical (observed) error rate. This is defined by the Vapnik-Chernovennkis (VC) Dimension [112]. The VC dimension measures the complexity of a class of functions. Given the dimensionality of a learned machine F with parameterization w as d, the number of training samples N, the dataset D, the true (unknown) risk is:

$$R(w) = \int \frac{1}{2}|y - F(x, w)|dD. \tag{2.6}$$

where D represents the probability distribution of the universe of all possible samples (training and test). Clearly, $R(w)$ is a purely theoretical measure and thus needs to be upper bounded by the *empirical risk*:

$$R_{emp}(w) = \frac{1}{N}\sum_{1}^{N}|y - F(x, w)|. \tag{2.7}$$

The empirical risk is the actual MSE we measure in our experiments. Given $R(w)$ and $R_{emp}(w)$, and η, where $0 \le \eta \le 1$, the VC-bound provides a tight theoretical bound on generalization performance for a learning machine:

$$R(w) \le R_{emp}(w) + \sqrt{\frac{dlog(\frac{2N}{d}) + 1 - log(\frac{\eta}{4})}{N}}. \tag{2.8}$$

The second term on the right in equation 2.8 provides a confidence on the empirical risk. This bound is significant as it is independent of the dataset being considered, and given a set of functions with computable VC-dimensions, we are guaranteed an estimate of the true risk (unknown apriori) with a certain probability. This clearly gives us a theoretical justification for choosing classifiers with minimal risk. Finally, it is evident from equations 2.7 and 2.8, that the number of *labeled* training samples, and model capacity, have a direct effect on the tightness of the bound. In the absence of a large number of labeled instances therefore, we must be careful to not select models of higher complexity, as this increases the risk of overfitting. This is one of the key challenges in learning with limited labeled data.

2.3 No Free Lunch

The No Free Lunch theorem [118] in statistical learning is a powerful result that shows that, *without underlying assumptions about the structure of the dataset, over all possible probability distributions over the data, every classifier will achieve* *the same test error*. This result signifies two things: first, that the choice of model (classifier) is significantly dependent on the data at hand, and that *no universally superior* model irrespective of data, can exist. Second, our underlying assumptions about the data have a significant impact on model performance and interpretibility. This is why, in our work, we look at models and datasets as co-dependent, and not separate entities, as each informs the other.

The topics discussed in this section provide some theoretical intuition for the work that follows. The growth in dataset sizes and model complexity places intense scrutiny on further overfitting by models today. In fact, this issue is even more pertinent in large models today [109]. These theoretical points help shed more light on our discussions in spurious correlations and bias in later sections.

/3

Learning with limited labels

The typical learning setup assumes access to a dataset $D = \{(x_1, y_1), (x_2, y_2), \ldots\}$, where x_i corresponds to an input data feature, and y_i the corresponding labels. In the *supervised setup*, we assume access to class labels for all data samples in D. However, as discussed before, this assumption is unrealistic, as collecting ground truth labels for all data samples, particularly for large datasets, is time-consuming and expensive. As a result, learning with limited labels assumes the *partially supervised* setup, where a subset $D_{sub} \subset D$ of samples have labels, or the *unsupervised setup*, where the label set for the entire dataset is empty, i.e. No labels are available for the whole dataset. Clearly, these scenarios are more challenging for a learning algorithm, as there is no supervisory signal to rely on for making predictions. As a result, learning with limited labels is essentially a *representation learning* problem, since there is a need to design novel representations of data in lieu of actual labels. In this chapter, we focus on three techniques for learning with limited labels: *few-shot Learning*, *semi-supervised learning*, and *unsupervised learning*.

3.1 Few-Shot Learning

In Paper II, we propose a novel representation learning method for transductive few-shot learning (FSL) [15, 116]. In this section, we provide some background and relevant context for our paper. The objective of few-shot learning (FSL) is to classify a set of novel classes in a test set that a base classifier has not seen during training. Typically, the classifier is provided a small set of such novel samples with labels, called the *support set*, while the test set for evaluation is called the *query set*. The number of samples in the support set is typically one or five, thus giving rise to the nomenclature *one-shot* or *five-shot* learning. For S shots and K novel classes, the scenario is deemed a *S-shot-K-way* problem. In the FSL setting, we assume access to a feature extractor (typically a deep neural network) that has been fit to a set of *base* classes. Note that the novel classes in the support and query sets have no commonality with the base classes.

The evaluation of the FSL classifier proceeds in *episodes*: in each episode, n_s support samples, and n_q query samples are sampled from each of the k classes. The FSL classifier makes the class predictions on the n_q samples based on the support. The average performance (mean and confidence intervals) over a large number of episodes is computed to evaluate the FSL classifier performance.

Transductive vs Inductive FSL There are two popular approaches to FSL in image data today - the transductive approach assumes access to the query representations both during training and inference, while the inductive approach assumes access to the query representations only during inference. As a result, there is a clear difference in the quality of representations available to the base classifier during training. As such, transductive approaches outperform inductive approaches on most FSL tasks in ixmage datasets [89, 69]. This is because the query representations can provide additional information to the classifier in addition to the support representations during training. In our work, we assume the transductive setting, i.e. the query representations are assumed to be available during training. As it turns out, the availability of these representations provides significant advantages in the embedding setup, as discussed next.

Normalization in FSL Since the challenge in FSL is to embed representations in a way that leads to accurate approximations of the data samples, a wide variety of normalization techniques have been employed in this area, often to surprisingly strong, state-of-the-art performance. Some examples include the L_2 and Centred L_2 normalizations, and the Z-score normalization [35].

Other Embedding approaches In addition to novel normalizations, there are other, more sophisticated embedding techniques that also demonstrate strong performance on FSL tasks. ReRep [26], for instance, proceeds in two stages: in the first stage, the query samples are combined using an attention mechanism. In the second stage, the support samples are combined linearly with the aggregated query representations. The intuition is that similar support representations in the second step would move closer to distinguished query representations in the first stage. Thus, ReRep is essentially a two-stage, support and query fusion mechanism mediated via attention. Another method, EASE [129], fuses the support and query samples into a single set. Next, it learns a similarity and dissimilarity matrix through an optimization problem where the similar features are encouraged to be embedded closer to each other, while dissimilar features are embedded further away. EASE ends with an L2 normalization of the learned embeddings. Finally, TCPR [120] first runs a k-neighborhood process and filters out the top-k support samples with respect to the task centroid. Next, it projects out the direction of the task centroid from these representations, thus ensuring orthogonality in the support set. By removing feature components in the direction of the task centroid in this fashion, TCPR is able to alleviate *support ambiguity*, i.e. supports lying too close to the centroid decision boundary that leads to harder predictions.

The Hypersphere in FSL One common feature of most embedding methods discussed above is the presence of the normalization process for the learned embeddings. Essentially, such a normalization projects the features on to the unit circle, or the hypersphere, in higher dimensions. This naturally leads us to ask why embedding representations on the hypersphere is useful for FSL. In Paper II, we address this question by elucidating how embedding representations on the hypersphere eliminates the *hubness problem* [91]. The hubness problem is a well known result in computational statistics, wherein certain points embedded in higher dimensional space often appear in the nearest neighbor lists of other points, leading to increased chances of misclassification. This is primarily due to the unreliable nature of Euclidean distances in high dimensional space. In our work in Paper II, we show that the hypersphere not only provably eliminates hubness, but that the elimination of hubness includes certain intuitive advantages for FSL classifiers as well.

3.2 Semi-Supervised Learning

The FSL setup is quite useful for learning with limited labels. However, the framework of support and query images may be unrealistic, since in the real world, such a clear distinction may not be available. To achieve a stronger step in generality, one can loosen the restriction of having support and query sets, and instead look at the case where a dataset D has a subset D_{sub} that contains labeled samples. The task then is to infer the unlabeled samples from these labeled samples, and this is the domain of semi-supervised learning (SSL), or partially supervised learning. Next, we introduce two key approaches in SSL - the first approach, that of *pseudo-labeling*, represents among the earliest methods that leverage unlabeled instances to label test data. The second approach, the *graph-based approaches*, is the domain that has been one of the most successful in modern SSL. In addition, this thesis also provides a key contribution in the domain of graph-based SSL, as we will discuss in a following section.

3.2.1 Pseudo-Labeling in SSL

These methods aim to assign pseudo-labels to the unlabeled instances in the data through some confidence measure. Self-Learning [130, 121], for instance, is an iterative procedure wherein a classifier predicts a set of unlabeled instances, and the predictions with the highest confidence scores are added to the (labeled) training data, proceeding to the next round of assignments. This process continues until all unlabeled instances are classified through this confidence-based filtering mechanism. Co-training [10, 127], on the other hand, presents a multi-classifier approach to generating pseudo-labels. Specifically, only those labels are considered to be estimated accurately if multiple classifiers are in confident agreement about their classifications. Disagreement between classifiers on label assignments leads to lower confidence, leading to the labels being discarded.

3.2.2 Graph-Based SSL

One of the most popular applications of modern SSL is through graph-based approaches [102]. If a dataset can be represented as a graph, where each node encodes a feature of a training instance, and each edge represents a similarity metric between nodes, various methods can be leveraged to estimate the labels of the unlabeled instances based on the labels available. In the context of our thesis, we explore graph-based SSL in the domain of *Transductive Node Classification*, presented next.

3.2.3 Transductive Node Classification

Given a graph $G = (V, E)$, where $|V|$ represents the node set, $|E|$ the edge set, G_L represents the labels of each node, G_{UL} represents the unlabeled nodes, and feature matrix $F \in \mathbb{R}^{(V \times d)}$ where d is the size of each node feature embedding, the transductive node classification task aims to estimate G_{UL} from G_L, F, and the adjacency matrix A of G.

We leverage Graph Neural Networks (GNN) [126] in this task. GNNs are a certain set of architectures for feature learning on graphs, where in addition to the node features, the graph topology as described by the adjacency matrix also comes into play to define the learning rule. In this thesis, Paper IV examines the effect of graph topology and features in the limited labeled setting on a variety of unique datasets. We show that the learning capability of a GNN is heavily mediated by the nature of the dataset, specifically how the graph topology and feature representations interact implicitly in the data.

3.3 Unsupervised Learning

The most general form of learning with limited labeled data is the scenario where no labels for any training instance are available. This is in principle the most general learning scenario, where other than certain fundamental assumptions, the task of estimating structure from unlabeled data is the most challenging. We discuss two such fundamental assumptions here, namely the *Cluster Assumption*, and the *Manifold Assumption*. These two assumptions are important to understand the challenges of estimating labels when no labeled instance exists.

3.3.1 Cluster Assumption

This assumption states that given a set of data instances, points in high density areas (that tend to form clusters) are more likely to belong to the same class. In other words, one could draw short curves that traverse high density regions across the data, effectively capturing the diverse class information present [22]. This assumption is important, because in the learning problem, we assume that points belonging to unique categories must be organized in unique representations. If this assumption does not hold, the learning problem cannot be framed appropriately. Recalling the No Free Lunch theorem discussed in the previous chapter, without underlying assumptions about the data, no classifier is significantly better than the other on average. This is why the cluster assumption is fundamental in the learning problem.

3.3.2 Manifold Assumption

This assumption states that there exists a low dimensional manifold for high dimensional data [21]. This assumption is useful because the learning problem struggles in high dimensions owing to the *curse of dimensionality* - in high dimensional data, Euclidean distances between points tend to become increasingly uniform, leading to a high risk of misclassification. A related topic in the curse of dimensionality is the *hubness problem*, as discussed previously in this chapter. Essentially, if one can project the data onto a low dimensional manifold, uncovering density disparities becomes tractable, in turn making the learning problem tractable.

These two assumptions form the base of all clustering, and SSL algorithms in use today. Broadly, distances between representations is considered to be of utmost importance. If distances are rendered meaningless, one looks to find representations where distances become useful again. Without these two assumptions, learning with limited labeled data is an intractable problem. Papers I, II, and III are all implicitly tied to these assumptions and other theoretical ideas discussed in the previous chapter.

3.3.3 Clusters as Pseudo-Labels

In this thesis, we employ clustering in the field of group robustness, i.e. estimating groups within data that may be biased in the learning task. The group robustness problem presupposes the existence of groups in the training data. It can be shown that modern classifiers are biased towards particular groups in the data, leading to unreliable predictions on the test set. However, assuming the existence of group labels is unrealistic, since collecting labels is a time-consuming and expensive endeavor. As a result, one of the contributions of this thesis is to provide an *unsupervised* method for group robust learning, one that leverages the clustering of explainability heatmaps of classifiers. In Paper I, we show that such a clustering mechanism reliably estimates the underlying groups in the data in lieu of actual group labels. We experiment with both *K-Means* and *Spectral* clustering techniques. A brief description is provided here.

K-Means

One of the most popular clustering methods, K-Means [119] proceeds by randomly assigning K centroids in the data, and then iteratively assigning points to each of these centroids based on the respective Euclidean distances, and updating the centroids. The algorithm terminates when the centroids do not change over a specified number of iterations.

Spectral Clustering

In this technique, instead of directly applying K-Means on the input data, first a similarity matrix is constructed using the data points. The eigenvectors of the resultant laplacian of the similarity matrix leads to a reduced dimensionality of the data. K-Means can be applied on the new embeddings to generate the clusters. Spectral approaches are useful when handling arbitrary data shapes, and when the data is naturally inclined to include graph-like structure [56]. In our thesis, we leverage a spectral clustering method called SPRAY [68], that clusters explainability heatmaps.

In summary, we have presented three key approaches in learning with limited labeled data: *few-shot learning, semi-supervised learning, and unsupervised learning*, and briefly presented how this thesis contributes in each of these particular areas.

4

Deep Learning

Deep Learning is the dominant form of function approximation in the modern age. Its successes have garnered significant attention in popular media and at the institutional levels of banks, governments, and policy institutes. This is due to the emergence of strong capabilities in tasks that humans excel at, such as vision, language, and reasoning. This chapter aims to be an introduction to deep learning, particularly the architectures we use in the thesis. These architectures include the ResNet [48], CLIP [90], Stable Diffusion [93], the Graph Neural Network (GNN) [126], and the multi-layer perceptron (MLP)[45]. We begin by introducing the simplest architecture among these, i.e. the MLP.

4.1 The Multi-Layer Perceptron

While the history of deep learning is vast and densely annotated [96], the key architecture for function approximation was the Multi-Layer Perceptron, which has two fundamental blocks: (i) The *linear transformation*, and (ii) The *Activation Function*.

The linear transformation is a weighted combination:

$$y = w^T x + b. \tag{4.1}$$

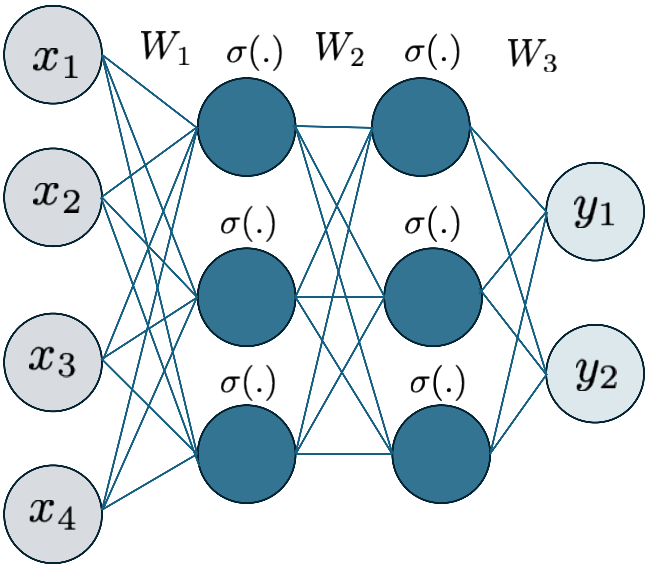


Figure 4.1: The Multilayer Perceptron: x_i represents a single input feature, and y_i represents the output prediction. W_i represents the weight matrix (learned), and $\sigma(.)$ represents a generic activation function. The bias parameters are omitted for clarity of presentation.

where the weight w represents the strength of the scaling, and b is the bias parameter, similar to a linear model as we have seen before. Next, to admit non-linearity, we consider $\phi(x)$ to be a mapping from the input to the transformed, non-linear output. This mapping creates the distinction between linear models or non-parametric models such as kernels, and deep learning. In deep learning, the mapping is chosen in terms of a parameterization θ, i.e. we want to find a mapping $\phi(x; \theta)$, where the parameters θ are *learned* through some optimization process. In the context of Equation 4.1, the parameters θ are thus defined in terms of w and b, and we must learn these parameters through some optimization process. The Activation Function is a non-linear transformation on the input. A common choice of an activation function today is the Rectified Linear Unit [2] (RELU), where the transformation is defined: $h(z) = \max(0, z)$. In summary, for the multi-layer perceptron, the transformation of the input reduces to:

$$f(x, h) = h(\phi(x; \theta)). \tag{4.2}$$

where θ are the parameters to be learned, and ϕ represents the linear transformation over the chosen parameters, i.e. w and b.

The term "deep" learning can now be intuitively explained. In Equation 4.2, we note that the transformation is modular, i.e. it can be repeated multiple times on intermediate outputs using the same transformation and activation functions.

If the output of the first transformation step is defined as $f^1(x)$, it is easy to see that the next transformation can be a function composition $f^2(f^1(x))$ and so on. A chain of such function compositions can thus be created easily. If each transformation $f^i(x)$ is considered a "layer", then the chain of function compositions can be interpreted as stacking multiple layers on top of each other, as illustrated in Figure 4.1. This is why the MLP is considered to be the first architecture in deep learning. It allowed for the stacking of the same affine transformations on intermediate outputs. In fact, this concept is so fundamental to deep learning, that modern networks are also called *feedforward networks*, and the function compositions are called the *forward pass* step of the network. Regardless of the more advanced architectures we cover in the next few sections, the concept of the forward pass remains the same.

Given our MLP outputs $f(x;\theta)$, we note that there must be a way to learn the parameters θ through an optimization process that fits the data well. In the previous chapter, we have discussed the cross entropy loss as a useful metric to evaluate a network prediction. However, given the loss, how do we update the parameters to minimize the loss over the next forward pass? We propagate the loss backwards through the network, in a process called *Backpropagation*, and update the weights in the direction that minimizes the loss. We repeat this process of a *Forward Pass-Backpropagation* multiple times until our loss converges. This process is called *training* the deep network. The popular mechanism to update the weights through backpropagation is to use *Gradient Descent*, which we discuss next.

4.1.1 Gradient Descent and Backpropagation

Given the loss L on the network output, how do we update the parameters w_{ij} such that the loss equates to zero? Gradient Descent [94] is a line search algorithm that updates a parameter at the next time step $t+1$ using the values at the previous time step w_t. Specifically, at each time step:

$$\delta w_{t+1}^k := w_t^k - \eta \frac{dL}{dw^k}, \delta b_{t+1}^k := b_t^k - \eta \frac{dL}{db^k}. \tag{4.3}$$

where w^k refers to the weights at layer k, and b^k refers to the bias parameters at layer k. Assuming the appropriate derivatives are available, gradient descent provides a rule to update the parameter values at each successive time step. To actually find the derivates, we use the Backpropagation algorithm. This algorithm uses the chain rule of derivatives to calculate $\frac{dL}{dw^k}$ and $\frac{dL}{db^k}$ for each weight connection across the layers with respect to the output loss.

Given the forward pass:

$$z^k = w^k \cdot a^{k-1} + b^k. \tag{4.4}$$

where $a^{k-1} = h(z^{k-1})$, for each layer $k = 1, 2, \ldots K$, we can compute the output error vector e^K as:

$$e^K = \nabla_a L \odot h'(z^K). \tag{4.5}$$

Then, the errors through the layers $K-1, K-2, \ldots, 2$ can be calculated recursively:

$$e^k = w^{k+1} \cdot e^{k+1} \odot h'(z^k). \tag{4.6}$$

Thus, finally $\frac{dL}{dw^k}$ can be calculated as $h^{k-1} \cdot e^k$. More specifically, for a weight connection between neuron i at layer $k-1$ to neuron j at layer k:

$$\frac{dL}{dw_{ij}^k} = a_j^{k-1} \cdot e_i^k. \tag{4.7}$$

Using a similar approach, we can calculate $\frac{dL}{db^k}$ as well:

$$\frac{dL}{db_i^k} = e_i^k. \tag{4.8}$$

This set of operations occurs for every forward pass and backpropagation step during the training process, and based on the derivatives calculated in this manner, the weight updates using gradient descent in Equation 4.3 can be realized.

All modern deep learning networks are trained using this technique. For practical optimization purposes, a variant of gradient descent, the Stochastic Gradient Descent (SGD) [14] is used, owing to stability issues when considering batches

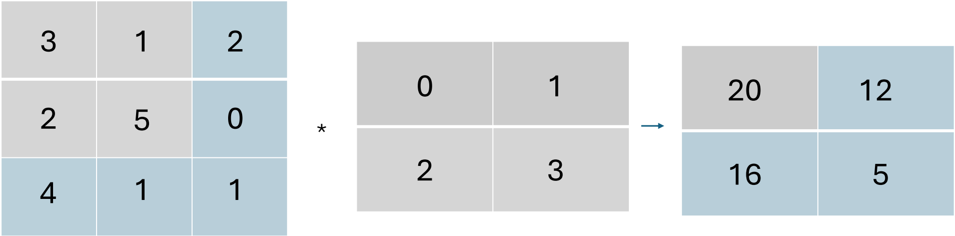


Figure 4.2: A simple convolution operation: the kernel on the right slides progressively over the input features on the left, resulting in a single map of output features. Such convolutions are applied at successive layers for multiple output maps.

of data inputs, but the fundamental principle of parameter updates using backpropagation remains consistent across all architectures. Other backpropagation algorithms are also popular, such as ADAM [62], AdaGrad [34], and RMSProp [67].

4.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) have emerged as the most successful architecture for computer vision tasks [70]. While, in recent years, vision transformers [33] have emerged as strong alternatives, most of the major deep learning successes we observe today can be attributed to the development of CNNs. In this section, we present the convolution operation on images, and define the convolution layer. Then, we present how stacks of such layers lead to multiple levels of granularity in image understanding.

4.2.1 Image Convolutions

An image is a 2D grid of pixels of a certain height and width. An image can also be 3D if the RGB color channel is introduced. The two key ideas behind capturing representations of an image are: (i) Locality, i.e. pixels in a close neighborhood (a patch) should represent similar concepts, and (ii) Equivariance, i.e. translating the objects in the image reflect an equal translation in the output. Given an image, the convolution operation on an image is defined like so:

$$x_j = h\Big(\sum_{i \in W_j} x_i \cdot K_{ij} + b_j\Big). \tag{4.9}$$

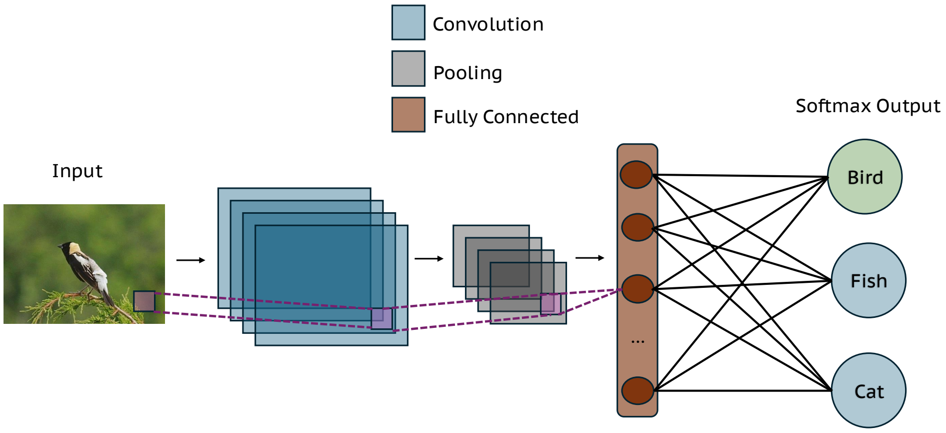


Figure 4.3: A typical CNN consists of a series of convolutional layers, pooling, and finally dense (MLP) connections to generate a probabilistic, softmax output. We begin with an input image, followed by four convolution kernels. Next, we apply pooling that effectively halves the dimensions. Next, we have a fully connected (FC) layer. Finally, we have a softmax output that covers a probabilistic simplex over the known classes. This network is only for illustration purposes. Typical, real world networks are larger and incorporate a combination of these base units.

where K_{ij} represents a kernel function with weights that operate on the image pixels, and W_j represents a set of input feature maps that progressively sliding the kernel generates. An illustration is provided in Figure 4.2. For a single kernel, the weights are *shared* across the whole image. This greatly reduces the computation cost of the linear transformation, as well as captures details in the image at different levels of granularity. For example, a kernel with a larger receptive field captures high level features in the input maps, while a kernel with lower receptive fields captures low level features in the input map.

4.2.2 Convolutional Layers, Pooling, and Dense Connections

The Convolutional Neural Network is defined completely by three broad stages: (i) Image Convolutions, (ii) Pooling, and (iii) Dense Connections. As defined previously, a set of image image convolutions defines a single convolutional layer. To encourage *translation invariance* to features (since convolution only guarantees equivariance), pooling layers are introduced after the convolution operations. A common pooling operation such as *max-pooling*, simply takes an output feature map, and filters out the maximum value in the desired area. This modular application of convolutional and pooling layers is repeated, and the output is attached through a fully connected MLP. This stage is called the

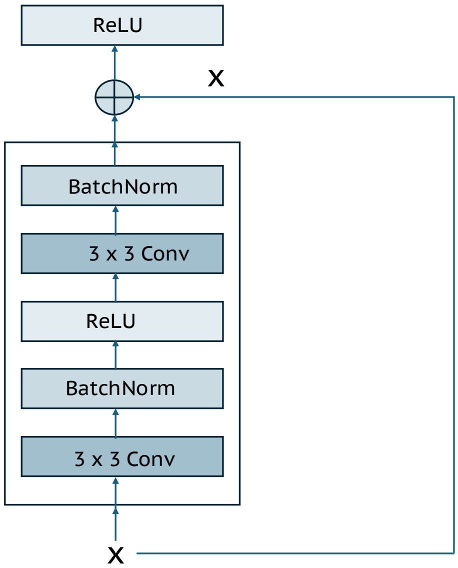


Figure 4.4: The Residual Block is exemplified by its usage of the skip connection, that results in improved long range flow of information through deeper networks. Such blocks are organized in stacks to create different variants of residual networks, such as ResNet-18.

Dense stage. While it is not strictly necessary to use fully connected layers at this stage, for classification tasks, where the output needs to be representated as a set of probabilities, the MLP connections are necessary. An illustrative example is provided in Figure 4.3.

4.2.3 Residual Networks

Originally introduced to solve the *degradation problem*, where deeper convolutional networks saturate in accuracy [47, 104], the core idea of using *skip connections* in residual networks (ResNets) [48] has been implemented in a variety of other architectures such as BERT and GPT [31, 122]. Skip connections, that are simply identity mappings from a previous layer to the current layer, allows long range flow of information through a network, that satisfactorily decreases the learning bottleneck in deep architectures. As a result, residual networks pre-trained on large datasets such as ImageNet, are the de-facto networks in deep computer vision today. Owing to their flexible design choice, there are multiple variants of ResNets at multiple scales of operation available today. An illustration of the residual block is shown in Figure 4.4.

In the context of our thesis, we use ResNet-based backbones in multiple works to investigate model diagnosis, such as our work on few-shot learning and group

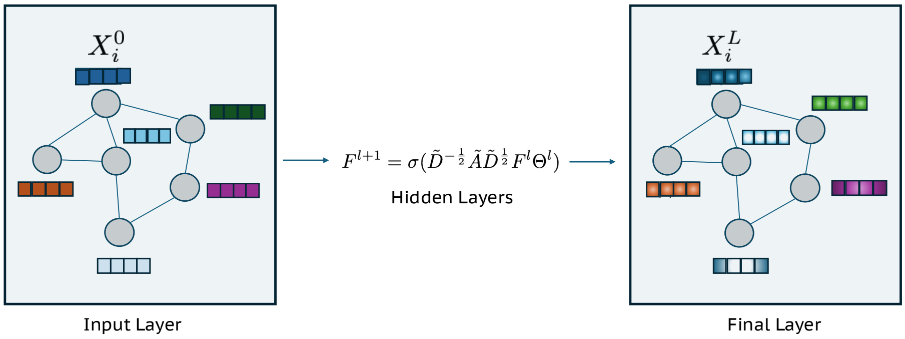

X_i^0
$F^{l+1} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{\frac{1}{2}}F^l\Theta^l)$
Hidden Layers
X_i^L
Input Layer
Final Layer

Figure 4.5: The Graph Neural Network: for node i, the features at the input layer is denoted as X_i^0. After L steps of graph convolution as defined in equation 4.10, the node features are transformed to X_i^L.

robustness. We will discuss these works in more detail in future sections.

4.3 Graph Neural Networks

Graph Neural Networks (GNNs) [126] operate on graph structured data, which are inherently different from images. To begin with, graphs exhibit *permutation invariance*, i.e. any ordering of the set of nodes is equivalent. Second, the notion of similarity in graphs is encoded through the existence of edges (and possibly edge weights). As a result, a different sort of architecture is required to learn effectively on graphs. Given a node set V, an edge set E, an adjacency matrix A, and the feature matrix $F \in \mathbb{R}^{V \times d}$ where d represents the dimensionality of the feature vectors in each of the nodes, the graph convolution operation on a particular layer of the GNN l is defined like so:

$$F^{l+1} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{\frac{1}{2}}F^l\Theta^l). \tag{4.10}$$

where, $\tilde{A}$ is the symmetric normalized adjacency matrix, Θ^l are the model weights at layer l, and σ is a non-linear transformation. Essentially, equation 4.10 represents a weighted mean of features over each node's neighborhoods, with a "layer" defining a single hop over such neighborhoods. In this way, $F^2, F^3, \ldots F^L$ till the final layer of the network can be computed in an iterative fashion. We illustrate the GNN architecture in Figure 4.5.

The Graph Convolutional model was proposed as a result of approximating a spectral filter with Chebysev polynomials, through the k-localized Chebynet [64, 29]. The authors showed that the resultant GCN model was a special case of the Chebysev polynomial-based approximation. This is interesting since one can interpret the GCN operation as a form of polynomial interpolation among the features. As it pertains to classification tasks for graph-based data, GCNs have proven to powerful function approximators.

4.3.1 Oversmoothing in GCNs

The phenomenon of oversmoothing in GNNs was first demonstrated in [71], where it was shown that the propagation rule in a standard GCN was a smoothing (weighted mean) operation equivalent to damping the symmetric normalized laplacian of the signal. As the number of layers in a GCN network increases, the weighted aggregation of k-hop nodes rendered more and more features from nodes with different classes to be similar to each other, adversely affecting classification performance. Over the years, many mitigation techniques have been proposed [86, 76, 125], that aim to build deeper GCNs without adversely affecting performance. In addition to mitigating techniques, proxy metrics to directly measure the effect of oversmoothing on graph networks have also been proposed [127, 23]. This phenomenon is not to be confused with the CNN bottleneck saturation discussed in the previous section - though the consequence of degraded accuracy is similar across the two architectures, oversmoothing refers to a repeated aggregation of features converging to a uniform representation, while the CNN saturation phenomenon was due to the difficulty of propagating early layer features to the deeper layers in the network. In the context of our thesis, we investigate the susceptibility of particular graph datasets to the oversmoothing phenomenon, and its impact on the graph learning procedure in transductive node classification. We focus on graph disentangled representations in particular. As a result, we are able to provide a holistic study into the importance of both the data and the model in the learning process, in line with out broader focus on model and data diagnosis.

4.4 Multimodal Models

The biggest recent shift in deep learning research has occurred in the development of *multimodal models*, models trained on multiple modalities on large datasets with a large number of parameters, and capable of a broad set of downstream generalization tasks [12]. Examples include CLIP [90], Stable Diffusion [93], MiniGPT [128], LLaVA [75] etc. In modern multimodal models, data sizes lie in the order of trillions of data points, while model sizes lie in the order of hundreds of billions of parameters. We briefly present CLIP to illustrate the capabilities of multimodal models in vision tasks.

4.4.1 CLIP

The remarkable capability of training on multi-modal data to exhibit strong performance on a broad range of downstream vision tasks was first demonstrated in CLIP [90]. Using 400 million text-image pairs, the authors use two feature encoders for images and text respectively. The loss function to encode that similar images representations should be assigned to similar text representations (The image of a dog should be aligned with the text "An image of a dog") was called the *CLIP loss*, a contrastive loss. Given the image encoding I, and the text encoding T, they first compute the Euclidean distance (can be interpreted as a form of pairwise similarity) $S = I^T T e^t$, where t is a temperature parameter for scaling. Next given the joint similarity S, two separate losses L_i and L_t are computed for the image and text encoding respectively:

$$L_i = -\frac{1}{N} \sum_{j=1}^{N} \sum_{i=1}^{C} y_{ij} \log(S_{ij}). \tag{4.11}$$

with L_t computed in a similar fashion. The final CLIP loss is simply the mean of L_i and L_t.

Given the contrastive loss, and the existence of a large dataset with text-image pairs, CLIP demonstrated significant advances in computer vision benchmarks [90]. This was a remarkable moment since it demonstrated the usefulness of pairing large quantities of data and multiple modalities, with no significant complexities involved in the encoder architectures or the loss function used.

4.4.2 Diffusion Models

While most of the models we have discussed up to this point are discriminative, i.e. they are primarily focused on creating decision rules around input data via a training mechanism, another group of foundation models today are generative, i.e. these models aim to learn a latent representation of the input data to *generate* new samples from this learned distribution. While the literature on generative models is vast [63], here we focus on a particular kind of generative model that models a *diffusion* process to learn the latent representation of an image dataset. Broadly, given an input image x_0, the diffusion process proceeds in three steps:

- The forward process: at each time step t, a certain noise distribution is added to the image with a set variance schedule. This step proceeds iteratively to transform an image into pure noise.

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t} \cdot x_0, (1-\bar{\alpha}_t) \cdot \mathbf{I}), \alpha_t = 1-\beta_t, \quad \bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s \quad (4.12)$$

where α and β are simply parameterizations that define the noise variance schedule.

- The denoising process: this is the reverse procedure, i.e The objective here is to learn the noise distribution added at each step to iteratively reconstruct the input image.

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)). \quad (4.13)$$

Similarly, the parameters $\mu_\theta(x_t, t)$ and $\Sigma_\theta(x_t, t)$ are learned by the model.

- Sampling: note that the learned distribution p_θ captures the noise added at each time step. As a result, to *generate* new images, one starts from pure noise, and iteratively denoises using distribution p_θ:

$$x_{t-1} \sim p_\theta(x_{t-1}|x_t). \quad (4.14)$$

The sampling repeats until we reach x_0, the desired input image.

Diffusion models have demonstrated stunning success in modern AI [88], but have also invited fresh scrutiny from multiple key institutions, such as the media, entertainment, and ethics boards [49, 9].

In the context of our thesis, we use both CLIP and Stable Diffusion in our work on data diagnosis, where we wish to generate debiased data using a diagnostic concept graph (Paper III). In the generation phase, we use a stable diffusion-based inpainting procedure, coupled with a CLIP-based filter to eliminate unreliable generations.

4.5 Risks and Pitfalls

This presentation of a broad range of architectures serves to support our broader goal of model and data diagnosis: deep Learning models are not supposed to be treated as infallible. Like all models, they encode certain inductive biases that are suitable for the task at hand. However, there should be a stringent focus on the weakness of such models - how robust are they? What do they base their predictions on? How consistent are these predictions? How biased are these predictions? What are some mitigating strategies for bias? Particularly for generative models such as diffusion, how do we controllably generate images with ethical guardrails? How do we quantify such notions of controllability, i.e. how do we score the outputs of generative models?

The importance of these questions is amplified when considering the models in use today, and the datasets these models train on, both of which are growing exponentially in volume, making it more and more difficult for humans to continuously press for checks and balances. In the next section, we discuss two approaches to model diagnosis: *explainability* - explaining classifier predictions, and *robustness* - how robust classifier predictions are to different groups of inputs. We also discuss data diagnosis in the context of generating synthetic, fair data.

5

Model and Data Diagnosis

In the introductory chapter, we presented *Model and Data Diagnosis* as fundamental, co-dependent frames of reference to reliable AI. In this chapter we present concrete techniques to understand how this diagnostic framework is used in modern deep learning. Particularly, we discuss *model explainability, spurious correlations, robustness, and fair data.* All the papers presented in this thesis, are encapsulated within the purview of model and data diagnosis.

5.1 Model Explainability

Why does a model predict what it predicts? How does a human evaluate a model's prediction, checking it for correctness and reliability? These questions are important since no classifier can be deployed in real world tasks without a degree of reliability (as measured by human evaluators) and strict testing to check for potential failure cases. The idea of *attributing* classifier predictions to neurons, layers, and pixels in the data, include what we call *Model Explainability* techniques. The research in this area is vast, including but not limited to adversarial techniques [40], heatmap-based approaches [108, 43, 6, 98], layer and neuron activation approaches [82], and so on. In the context of our thesis, we focus on heatmap-based attributions, and present two popular approaches: GradCAM [98], and LRP [6].

5.1.1 GradCAM

If we were to consider the task of image classification, a 'good' classification is when a classifier exhibits a reasonable explanation of a correct prediction. Unreasonable explanations for correct predictions are not considered 'good' since it is hard to determine the factors that caused the classifier to take its decision. GradCAM is a popular technique to attribute *visual explanations* to CNN-based classifiers, which represent the dominant architecture in computer vision classification tasks. GradCAM proceeds in three simple steps: first, the image of interest (and its class label) is forward propagated through the network. This results in a network prediction (after the application of a set of fully connected layers). Second, only the positive gradients associated with prediction are considered, and all other gradients are set to zero. These gradients are then backpropagated (*guided backpropagation*) through the intermediate convolutional feature maps. Finally, the heatmap is pointwise multiplied with the image of interest, resulting in a fine grained explanation. This technique provides human-intuitive explanations for classifer predictions. The GradCAM procedure is detailed below in a step-by-step fashion. We assume we are given the score S (pre-softmax) of an input image I, where, and feature maps F_l at a particular layer l.

- Compute the gradients $\frac{\partial S}{\partial F_l}$ for the score with respect to each feature map.
- Compute importance weights $w_k = \frac{1}{H \times W} \sum_i \sum_j \frac{\partial S}{\partial F^k_{ij}}$, where H and W are the height and width of the image respectively.
- Finally, the GradCAM attributions $A = ReLU(\sum_k w_k F^k)$.

5.1.2 Layerwise Relevance Propagation (LRP)

The basic idea in LRP is to enforce a score conservation property across neurons at a particular layer, for all layers. By doing so, LRP uncovers how much a particular neuron activation contributes to another neuron activation in the successive layer. By redistributing neuron relevance scores in this way backwards through the network, one can generate heatmap-based attributions on input images as a result of classifier predictions. Specifically, for each data point x, the relevance score is defined on a per-neuron-per layer basis. For an input neuron n_k at layer k, and an output neuron n_l at the following layer l, the relevance score R_k is intuitively a measure of how much this particular input n_k contributed to the output value n_l:

$$R_k = \sum_{l:k \longrightarrow l} \frac{z_{kl}}{z_l + \epsilon \cdot \text{sign}(z_l)} \tag{5.1}$$

where $z_l = n_k w_{kl}$ for the weight connection w_{kl}. R_k is computed for all the neurons at layer k, and backpropagated from the output layer to the input layer to generate pixel level relevance scores r_x for each data input x. In our work on group robustness (discussed below), we cluster explainability heatmaps using LRP to generate pseudo-labels for groups within the training data. Our method ExMap, however, is not constrained by the choice of an explainability method, and any other popular technique such as GradCAM would also apply.

5.2 Spurious Correlations and Group Robustness

A spurious correlation is any feature that a classifier uses to make a prediction, even though the feature is *causally unrelated* to the task. Following our discussion before on model explainability, a reliable analogy would be the following: assuming that the task is classify a dog in an image, if a significantly high number of images in the training set contain dogs in urban backgrounds, the classifier should not rely on background cues to make an object prediction. Interestingly enough, modern CNN-based image classifiers are significantly susceptible to such spurious cues [38] - the reason for this is baked into the CNN architecture. Since the idea of locality is important in the convolution operation, feature maps produced from image kernels will frequently contain spurious features, which are then aggregated with the object level features deeper into the network. As a result of this reliance, a wide variety of biases have been uncovered in CNN-based classifiers, such as texture bias [38], shortcut learning [37], color bias [28, 61], and so on. Many works over the recent years have aimed to mitigate such issues, usually by encoding some *shape*-based feature information into the network over simply textures [53, 72].

In addition to such mitigation techniques, a separate field of research has emerged in recent years that also investigate the reliance on spurious correlations: *group robustness* [55, 83]. The assumption here is that, a dataset can be segmented into groups of interest, with each group representing a particular spurious correlation of interest. For example, in the dataset Waterbirds [95], the task is to distinguish between a landbird and a waterbird (Figure 5.1). The spurious correlation is the *background*, which is either Land or Water. As a result, there are four groups of interest: [(Landbird, Land), (Landbird, Water), (Waterbird, Land), (Waterbird, Water)]. If there is a strong imbalance in the number of images in the training set for each group, this imbalance would constitute a bias, and a classifier would pick up on this bias, leading to worse

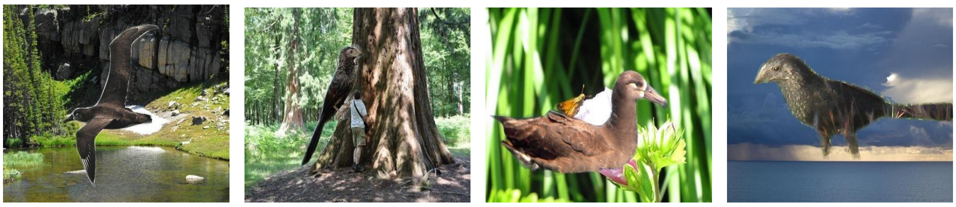


Figure 5.1: The Waterbirds Dataset consists of landbirds and waterbirds in land and water-based backgrounds. The reliance of pretrained classifiers on the spurious background feature can be reliably tested on this dataset. (Left to Right) Waterbird on water, Landbird on land, Waterbird on land, and Landbird on water respectively.

generalization capabilities. In fact, the Waterbirds dataset is intentionally biased - 95% images of Landbirds have land backgrounds, and 95% images of Waterbirds have water backgrounds, and any ResNet trained pre-trained model fails on a test set where this imbalance is absent [95].

As a result, group robustness strategies aim to mitigate classifier reliance on imbalanced groups, and encourages a balanced performance across all groups within the data during evaluation. In the context of our thesis, our contribution is in the domain of *unsupervised group robustness*, i.e. developing a technique for group robustness when the group labels are absent. Assuming access to group labels in the training data is unrealistic, since the collection and segmentation of data into such pre-defined labels would be time-consuming and expensive.

5.3 Fair Data

At this point, we have discussed model diagnosis in the limited labeled setting - few-shot learning, explainability, semi-supervised learning, and robustness. However, we note that in the end, every model trains on data, and any discussion on model capabilities on generalization tasks is constrained by the data it trains on. As a simple example, consider the *tench* class in ImageNet. A simple, manual browsing of the dataset immediately points to a certain discrepancy: most images have the tench being held by a man, in the centre of the frame, surrounded by a rural environment. This strong correlation may be irrelevant at first to the human curator, but a model picks up on all cues it can efficiently find to reach a prediction (the shortcut learning mechanism, as described in [37]). In Figure 5.2, we show what happens when we manually remove the fish from the man, and yet the model predicts the class to be fish with high confidence. In fact, on inspecting the explainability heatmaps, we find that the model focuses on the man in the background to make the prediction. This leads to a fascinating question of how the model will pick up on spurious

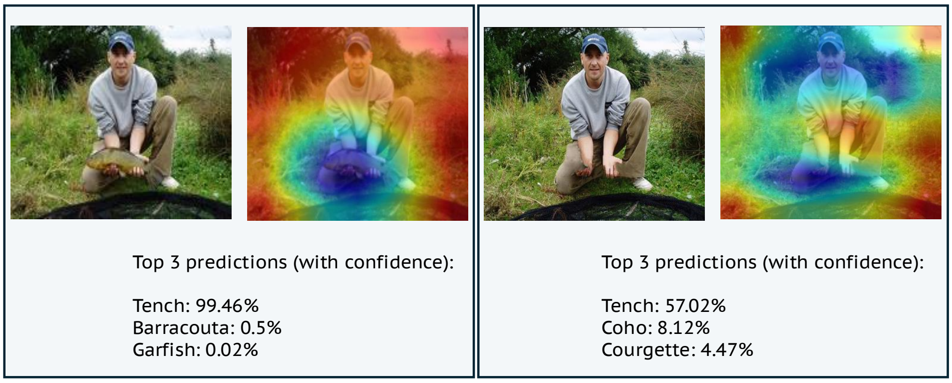


Figure 5.2: Simple object-based bias in ImageNet: the *tench* class images appear frequently with men holding them at the centre of the frame. We use a pretrained ResNet18 to infer each image, one with the fish (Left), and the same image without the fish (Right). We observe that the model, in addition to predicting *tench* with high confidence in both cases, also focuses on the spurious feature in the image, i.e. the man. Blue represents higher relevance scores, and Red represents lower relevance scores.

correlations that are not readily apparent to the human eye. Note the major constraint in this case. The dataset, ImageNet, contains millions of images from tens of thousands of classes. What other such spurious correlations exist in this dataset that may hinder model generalization capabilities? In short, how does *dataset diagnosis* affect model performance? This is not a strictly recent question. The early works on dataset bias [111, 110] elucidate the ways in which visual datasets can be biased in multiple, nefarious ways. Such biases, such as object location, object co-occurrence and scale [85, 101], may not be readily apparent to the human eye, but may be acting as confounders for the learning process. Placed in the current context of datasets of increasing size, fair and debiased datasets are therefore crucial for model generalizability. Some tools for dataset diagnosis exist today [115], and dataset benchmarks are growing more popular by the day [32]. As part of this thesis, keeping in mind the limited labeled data context, we present a work on *generating* de-biased data based on a certain framework of data diagnosis. We show that this framework of dataset diagnosis and debiasing, is successful in significantly improving the state-of-the-art in model generalizability across classification and robustness metrics.

Part II

Summary of Research

6

Paper I

ExMap: Leveraging Explainability Heatmaps for Unsupervised Group Robustness to Spurious Correlations

Rwiddhi Chakraborty, Adrian Sletten, Michael Kampffmeyer

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 2024.

- Code: https://github.com/rwchakra/exmap
- Paper: https://shorturl.at/Howm3
- Talk: https://t.ly/qlmkR

In Paper I, our contribution includes proposing a new, unsupervised method for group robust learning in deep learning classifiers that mitigates the reliance on spurious correlations in the data. Spurious correlations are any features in the datasets that the model relies on to make a prediction, but these features are causally unrelated to the task. Our unsupervised method to mitigate such spurious correlations proceeds in two steps:

Extract Heatmaps We use LRP to extract heatmaps of images in the validation dataset. LRP outputs pixel-wise relevance scores, which illustrate the relevant image regions for a frozen ResNet model pre-trained on ImageNet.

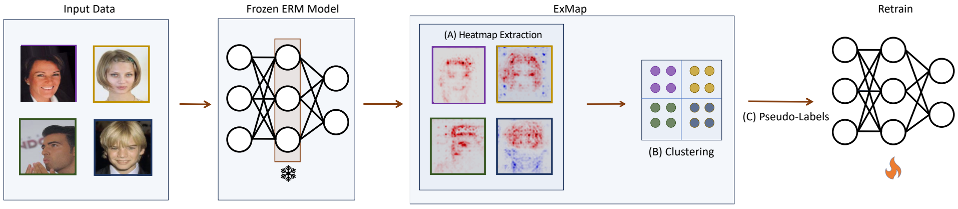


Figure 6.2: Our proposed method: ExMap achieves group-robustness by first extracting explainability heatmaps from the frozen base ERM model for the validation data (A). Next, we cluster the heatmaps (B) to obtain pseudo-labels for the underlying groups. These labels are used for the retraining strategy (C).

Clustering Given the explainability heatmaps, we use a spectral clustering method called SPRAY [68], automatically estimating the clusters based on the eigengap heuristic [56]. The clustering module aims to highlight the dominant model strategies adopted towards the classification task. The outputs of the clustering module are the underlying groups as discovered by our method. Then, using a balanced sampling technique, we retrain the base classifier on the given features, using the estimated pseudo-labels as the guiding validation loss. We provide an illustration in Figure 6.2.

As we demonstrate in the work, this unsupervised strategy is quite successful in estimating underlying groups in the data.

In both single shortcut and multi-shortcut (datasets where there are multiple *types* of spurious correlations), our method results in state-of-the-art performance on a variety of datasets in the literature.

Additional findings from our experiments include:

- Demonstration of how our method circumvents background reliance of the classifier on Waterbirds.
- Demonstration of how our method improves model explanations.
- Robustness to the choice of clustering algorithm and learning strategy.

Contributions

- I proposed the research direction and conducted the initial literature review.

- The methodology was developed in collaboration with all the co-authors.
- I jointly ran experiments in the paper with AS.
- I wrote the first draft of the manuscript. The final polished version was achieved with the help of AS, and MK.

/7

Paper II

Hubs and Hyperspheres: Reducing Hubness and Improving Transductive Few-shot Learning with Hyperspherical Embeddings

Daniel Trosten, Rwiddhi Chakraborty,*, Sigurd Lokse, Kristoffer Wickstrom, Robert Jenssen, Michael Kampffmeyer*

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 2023.

- Code: https://github.com/rwchakra/exmap
- Paper: https://shorturl.at/Howm3
- Talk: https://t.ly/qlmkR
- * denotes equal contribution

In Paper I, we investigate the hubness problem in transductive few-shot learning (FSL).

The hubness problem is a consequence of the curse of dimensionality [8], where certain exemplar points embedded in higher dimensional space often appear in the nearest neighbor lists of other points. These exemplar points are called *hubs*. A negative consequence of hubness is that points from different classes may appear in the same nearest neighbor lists, leading to increased misclassifications.

Since many approaches in FSL adopt distance-based approaches, the motivation was to design a new embedding space where hubness is eliminated.

Consequently, we proved that an uniform distribution of points on the hypersphere are hubness-free, and as a result, hold intuitive advantages in the few shot setup. Motivated by this finding, we propose two methods that at once reduce hubness (by imposing uniformity on the hypersphere), and preserve the inherent class structure of the embedded points, leading to a novel tradeoff that can be encoded as a loss function. We describe the two methods below:

noHub We first provably optimize a tradeoff between local similarity preservation (points in similar classes should remain close in the embedded space), and uniformity (points in different classes must be separated in the embedding space). We show that the frequently-used Kullback-Leibler divergence [66] decomposes naturally into these two properties.

noHub-S Recalling that the support labels are available in the FSL setup, we extend noHub by modifying the loss function to contain the signal from these labels. In this way, our method becomes partially supervised, which intuitively promises stronger results.

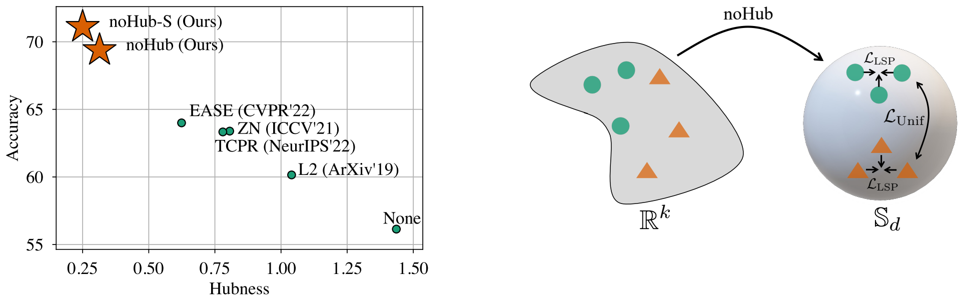
Since noHub and noHub-S are both embedding methods, they are flexible. They can be used on any off-the shelf FSL classifier in use today, on a variety of backbone architectures. These are the major results in our work, i.e. we show that by reducing hubness using our embedding method, we achieve state-of-the-art FSL classification accuracy on a wider variety of backbone architectures and FSL classifiers. An illustration of the method is provided in Figure 7.2.

Other experimental findings from our work include:

- Ablating the amount of tradeoff between local similarity preservation and uniformity.

- Showing that the features learned through our methods results in a clearer separation between representations of different classes.

- Demonstrating the usefulness of incorporating the partial supervision from the support labels.



(a) Reducing hubness improves FSL classifier accuracy. Our methods (noHub and noHub-S) compared with baseline embedding methods.

(b) Our method optimizes a tradeoff between local similarity preservation and uniformity ($\mathcal{L}_{LSP}$ and $\mathcal{L}_{Unif}$) of points on the hypersphere.

Figure 7.2: The two key takeways of our paper - (Left): we show the importance of eliminating hubness, and (Right): we show the principle guiding our method.

Contributions

- The design of the methodology was achieved in collaboration with all the co-authors in the paper.
- I conducted initial exploratory analysis, and jointly conducted all experiments in the paper with DT.
- I jointly wrote the first draft of the manuscript with DT. The final polished version was achieved with DT and MK, and I.

8

Paper III

Visual Data Diagnosis and Debiasing with Concept Graphs

Rwiddhi Chakraborty, Yinong Wang, Jialu Gao, Runkai Zheng, Cheng Zhang, Fernando de la Torre

Under Review

In Paper III, we address the issue of data diagnosis, i.e. directly probing the dataset for inherent biases, rather than using the model that trains on it as a proxy. We propose a novel end-to-end framework that simultaneously diagnoses the data for spurious correlations, discovers imbalanced (biased) concept correlations, and generates a synthetic dataset using a uniform distribution of the discovered concepts, effectively debiasing the data. Our results show that training the classifier on the augmented, debiased dataset results in state-of-the-art performance on a variety of benchmark datasets.

Briefly, our method proceeds in three stages:

- Concept Graph Construction: we construct a knowledge graph from the dataset based on object co-occurrences.

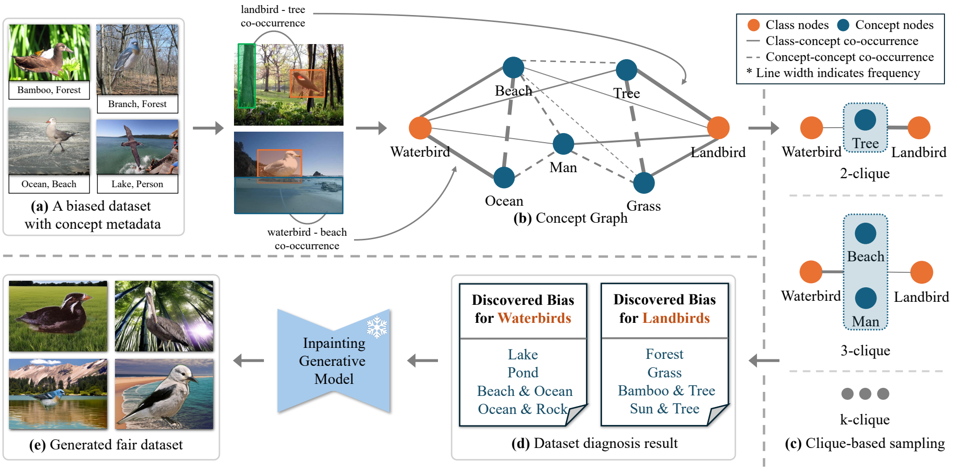

Bamboo, Forest
Branch, Forest
Ocean, Beach
Lake, Person
(a) A biased dataset with concept metadata
landbird - tree co-occurrence
waterbird - beach co-occurrence
Waterbird
Beach
Tree
Man
Ocean
Grass
Landbird
(b) Concept Graph
Class nodes
Concept nodes
Class-concept co-occurrence
Concept-concept co-occurrence
* Line width indicates frequency
Waterbird
Tree
Landbird
2-clique
Beach
Waterbird
Man
Landbird
3-clique
k-clique
(c) Clique-based sampling
Discovered Bias for Waterbirds
Lake
Pond
Beach & Ocean
Ocean & Rock
Discovered Bias for Landbirds
Forest
Grass
Bamboo & Tree
Sun & Tree
(d) Dataset diagnosis result
Inpainting Generative Model
(e) Generated fair dataset

Figure 8.2: ConBias: given a visual dataset (a), we build a concept graph from ground truth metadata such as captions/segmentation labels (b). Line thickness indicates the weight of a co-occurrence. Next, we diagnose the concept graph by discovering class-concept imbalances, using a graph-clique enumeration (c). The imbalanced cliques are precisely the biases in the data. Given the diagnosis, we generate a uniform concept-class distribution of images using a generative model with inpainting (d), to output the final debiased data (e). The base classifier retraining operates on this dataset.

- Concept Graph Diagnosis: we analyze the graph for combinations of classes and concepts that are heavily imbalanced. These are the co-occurrence-based biases in the data.

- Concept Graph Debiasing: upon discovery of biased combinations, we enumerate graph cliques to sample balanced concept-class combinations from the graph. By imposing a balanced sampling strategy, we ensure a uniform representation of concepts and classes in the new data.

This method, ConBias, allows for a principled and controllable way to simultaneously diagnose a visual dataset for biases, and correct such biases by generating concept balanced data. We show that retraining on the generated dataset significantly improves upon existent data debiasing baseline methods. An illustration of our method is provided in Figure 8.2.

Other experimental findings from our work include:

- Demonstrating the usefulness of the method on multi-shortcut tasks as well.

- Showing how ConBias helps the classifier to learn relevant features for the task, ignoring the spurious features.

- Demonstrating the use of the graph structure as opposed to using simple object co-occurrence statistics.

Contributions

- The methodology was developed in collaboration with all the co-authors.
- I ran all the experiments jointly with YW, and help from JG.
- I wrote the first draft of the manuscript. The final polished version was achieved with the help of YW, CZ.

/9

Paper IV

On Disentangled Representations and the Oversmoothing Problem in Graph Convolutional Networks

Rwiddhi Chakraborty, Benjamin Ricaud, Robert Jenssen, Michael Kampffmeyer

Submission Ready

In Paper IV, we address the issue of transductive node classification in graph convolutional networks (GCNs), which is a semi-supervised learning problem. Specifically, we show that graph datasets can be encoded as a tradeoff between feature informativeness, and structure informativeness, i.e. there are varying degrees of information held in the node features of a graph dataset relative to the topology of the graph. Subsequently, we show that such a tradeoff has an inherent effect on the oversmoothing phenomenon in GCNs, a well studied phenomenon that adversely affects GCN performance as the number of layers in the model progressively increases. Our analysis provides novel insights into the learning mechanism of GCNs on different types of graph datasets.

This work exists as a bridge between the model and data diagnosis framework

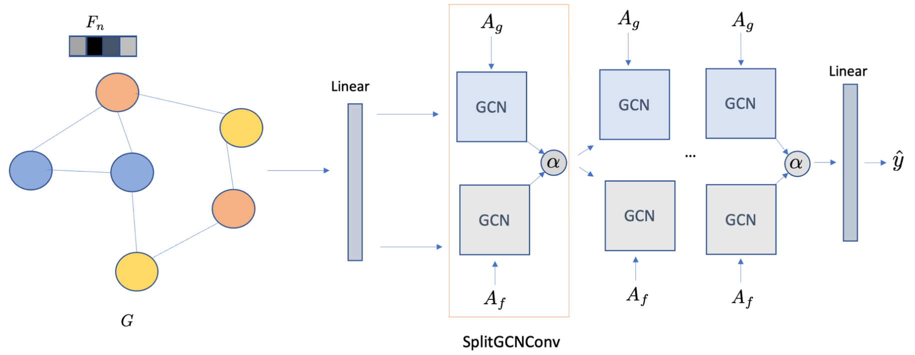


Figure 9.2: Our proposed framework, SplitGCN, frames the graph learning process as a tradeoff between leveraging the node features and the graph structure in the dataset. This tradeoff is quantified by a novel metric that we propose, called the Latent Dirichlet Energy (LDE).

that we have presented in this thesis. The data diagnosis aspect is contained in the proposal of a metric to quantify the tradeoff between feature importance and structure importance in a graph dataset, *without relying on labels*. Our metric, the Latent Dirichlet Energy (LDE), naturally estimates this tradeoff. As a result, we are able to diagnose datasets into high LDE regions, which means that the graph connections do not hold much information, and low LDE regions, where the graph structure can be leveraged for learning.

Next, we propose a novel architecture called the SplitGCN (Figure 9.2), which frames the learning process as a loss function that trades off between the node features and the graph structure in the dataset. We show that such a framework has benefits in graph based learning, and mitigate the phenomenon of oversmoothing in GCNs.

Other experimental findings include:

- Ablating the effect of the tradeoff between node feature information and graph structure information.

- Ablating each branch of our architecture to test the effect on the learning process. We show that *both* the node features and graph structure are important information that need to be leveraged.

- Providing an analysis of oversmoothing mitigation in GCNs using our method.

Contributions

- The methodology was developed in collaboration with all co-authors. BR, in particular, developed one of the main metrics used in the paper.

- I ran all experiments in the paper.

- I wrote the first draft of the manuscript. The final version was achieved with the help of BR and MK.

Part III

Conclusion and Future Work

The success of deep learning-based frameworks in the past decade has impacted diverse areas in modern computing. The impact of this research area extends far beyond performance on certain vision and language benchmarks. Over the last decade, the significant developments in programming libraries, compute capabilities, and the advent of new mathematical models have invited fascinating questions into the nature of human learning, and what it means to truly outperform humans in certain tasks. The next decade is brimming with further optimism: companies today are developing novel chips tailor-made to train AI based models. There is a broad, ambitious vision of revamping the entire LLVM compiler architecture that all modern computers are built on [27]. The success of ChatGPT has, in addition to receiving a plethora of media attention, also led researchers to ask deep questions on the nature of reasoning. The debates are frequent, sometimes heated, almost always interesting. Broad questions on whether reasoning is emergent, inherently stochastic, or pre-programmed in learning machines have invited further scrutiny from the research community. Ethical questions have emerged as well, with nation states scrambling to formalize guidelines and enact regulations [114, 117]. In the introductory section, we referred to this panoply of events as an epistemological “wild west”, where a principled investigation is necessary into how models behave, and what possible issues the datasets these models train on, contain.

The key areas of intrigue in learning machines - generalization, bias, limited supervision, are encapsulated within the purview of model diagnosis. The key areas of inspection for data - bias and spurious correlations, are encapsulated within the purview of data diagnosis. This thesis shed light into this unified framework, eventually demonstrating that they are not necessarily mutually exclusive. In each of the areas inspecting model diagnosis (limited supervision and generalization capabilities), this thesis presented novel contributions with respect to both representation learning, and to model debiasing. In the area inspecting data diagnosis, this thesis proposed a novel contribution towards the automatic debiased generation of data based on intrinsic object co-occurrence based biases in the dataset. Finally, we also demonstrate how model and data diagnosis can inform each other, with our systematic study of disentangled graph representations in the semi-supervised setup.

The need to reframe research objectives within this framework arose primarily due to the observation that, as models and datasets keep growing in scale, the following issues will grow proportionally:

- Annotation of large datasets, which is time-consuming and expensive.

- Generalization capabilities in settings where reliable labels are not available.

- Large datasets may contain unknown spurious correlations and biases that go undetected by human observers.

We briefly summarize our contributions with respect to each of the issues described above.

Generalization capabilities under limited supervision

In Paper I, we present a novel unsupervised mechanism to mitigate spurious correlations in the group robustness framework. We show that clustering explainability heatmaps provides a two-fold improvement in unsupervised group robustness: first, that such heatmaps demonstrate the model focus during classification and second, that the method only focuses on the relevant features as decided by the model, resulting in significant improvements over various datasets in both single and multi-shortcut domains.

In Paper II, we present a novel representation learning method that in principle eliminates the hubness problem by projecting classifier features on the hypersphere. Further, we show that the elimination of hubness harbors the positive consequence of state-of-the-art capabilities in both 1-shot and 5-shot learning settings. Finally, we show the strong correlation between the reduction in hubness and the improvement in few-shot classification accuracy over a variety of classifier backbones and datasets.

In Paper IV, we present a systematic study of the effect of disentangled graph representations on the oversmoothing problem in transductive semi-supervised node classification tasks. We present a novel metric that captures the information trade-off between the node features and the graph topology, which sheds insight on the learning capabilities of graph convolutional models, particularly disentangled models. Further, we provide connections between our proposed metric and common metrics used today such as node homophily. While homophily characteristics are useful, they require node labels to be computed, while our metric only requires the features and the graph laplacian, both of which are always available.

Data Diagnosis and Debiasing

In Paper III, we present a novel concept graph-based framework that encapsulates object co-occurrence-based biases in visual datasets. By inspecting the concept graph, we show that significant spurious correlations emerge in a variety of datasets. Using this diagnostic framework, we are then able to debias such features in a systematic way, by generating data with a uniform combination of such previously biased concepts. By retraining on the new generated data, we observe significant improvements in the current state-of-the-art data debiasing approaches. In fact, our method also leads to improvements in multi-shortcut metrics, which is a second order positive consequence of the framework.

In Paper IV, we address dataset diagnosis by proposing a metric based on node features and the graph laplacian. This metric circumvents the need to access the node labels of the graph, thus providing a flexible approach to understanding how node features and graph structure interact in datasets. In particular, we show the effect of such a tradeoff on disentangled graph models and the oversmoothing phenomenon in transductive semi-supervised node classification.

Future Work

There are several avenues of future work that excite us. Firstly, this thesis was focused on computer vision architectures and datasets. However, as described in the introductory section, we are effectively in a post-ImageNet era where the most capable models are not unimodal, but multimodal. We are excited about extending the model and data diagnosis framework to the multimodal learning perspective, given how this paradigm is the dominant approach in current research. Further, the addition of a new modality such as text, would provide new challenges and insights into the framework.

Second, we are also excited about other approaches to model diagnosis. While this thesis primarily evaluated classification performance and robustness metrics, these are certainly not the exclusive hallmarks of model diagnosis. In fact, with the advent of large multimodal models, it is perhaps amusing that we are witnessing a phenomenon akin to Goodhart's Law in modern AI as well [41]. The vast plethora of benchmarks, and marginal improvements for each successive model, leads to the measure becoming the target, as Goodhart warned all those decades ago. What other forms of model diagnosis could exist? Could there be simple benchmarks to test reasoning capabilities of these models [131]? Could there be simple synthetic data that could be generated to inspect the possibility of deriving universal laws of model behaviour [3]? These are the broad questions that interest us moving forward.

Part IV

Included Papers

/10

Paper I

This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

ExMap: Leveraging Explainability Heatmaps for Unsupervised Group Robustness to Spurious Correlations

Rwiddhi Chakraborty, Adrian Sletten, Michael C. Kampffmeyer
Department of Physics and Technology, UiT The Arctic University of Norway
firstname[.middle initial].lastname@uit.no

Abstract

Group robustness strategies aim to mitigate learned biases in deep learning models that arise from spurious correlations present in their training datasets. However, most existing methods rely on the access to the label distribution of the groups, which is time-consuming and expensive to obtain. As a result, unsupervised group robustness strategies are sought. Based on the insight that a trained model's classification strategies can be inferred accurately based on explainability heatmaps, we introduce ExMap, an unsupervised two stage mechanism designed to enhance group robustness in traditional classifiers. ExMap utilizes a clustering module to infer pseudo-labels based on a model's explainability heatmaps, which are then used during training in lieu of actual labels. Our empirical studies validate the efficacy of ExMap - We demonstrate that it bridges the performance gap with its supervised counterparts and outperforms existing partially supervised and unsupervised methods. Additionally, ExMap can be seamlessly integrated with existing group robustness learning strategies. Finally, we demonstrate its potential in tackling the emerging issue of multiple shortcut mitigation[1].

1. Introduction

Deep neural network classifiers trained for classification tasks, have invited increased scrutiny from the research community due to their overreliance on spurious correlations present in the training data [4, 5, 9, 31, 38]. This is related to the broader aspect of Shortcut Learning [10], or the Clever Hans effect [15], where a model picks the path of least resistance to predict data, thus relying on shortcut features that are not causally linked to the label. The consequence of this phenomenon is that, although such models may demonstrate impressive mean accuracy on the test data, they may still fail on challenging subsets of the data, i.e. the groups [7, 8, 27]. As a result, group robustness is a natural objective to be met to mitigate reliance on spurious correlations. Thus, instead of evaluating models based on mean test accuracy, evaluating them on *worst group accuracy* has been the recent paradigm [12, 21, 25, 40], resulting in the emergence of group robustness techniques. By dividing a dataset into pre-determined groups of spurious correlations, classifiers are then trained to maximize the *worst group accuracy* - As a result, the spurious attribute that the model is most susceptible to is considered the shortcut of interest.

In Figure 1, we illustrate the group robustness paradigm. Given a dataset, a robustness strategy takes as input the group labels and retrains a base classifier (such as Expected Risk Minimization, i.e. ERM) to improve the worst group accuracy (G3 in this case). GroupDRO [28] was one of the early influential works that introduced the group robustness paradigm. Further, it demonstrated a strategy that could indeed improve worst group accuracy. One limitation of this approach was the reliance on group labels in the training data, which was replaced with the reliance on group labels in the validation data in successive works [13, 19]. However, while these efforts have made strides in enhancing the accuracy of trained classifiers for underperforming groups, many hinge on the assumption that the underlying groups are known apriori and that the group labels are available, which is often impractical in real-world contexts. An unsupervised approach, as illustrated in Figure 1, would ideally estimate pseudo-labels that could be inputs to any robustness strategy, leading to improved worst group robustness. An example of such a fully unsupervised worst group robustness approach is (GEORGE) [32]. GEORGE clusters the penultimate layer features in a UMAP reduced space, demonstrating impressive results on multiple datasets. In this work, we instead show that clustering *explainability heatmaps* instead, is more beneficial in improving worst group robustness. Intuitively, this stems from the fact that a pixel-attribution based explainability method in input space focuses only on the relevant image features (pixel space) in the task, discarding other entangled features irrelevant for the final prediction.

In our work, we circumvent the need for a group labeled

[1] Code available at https://github.com/rwchakra/exmap

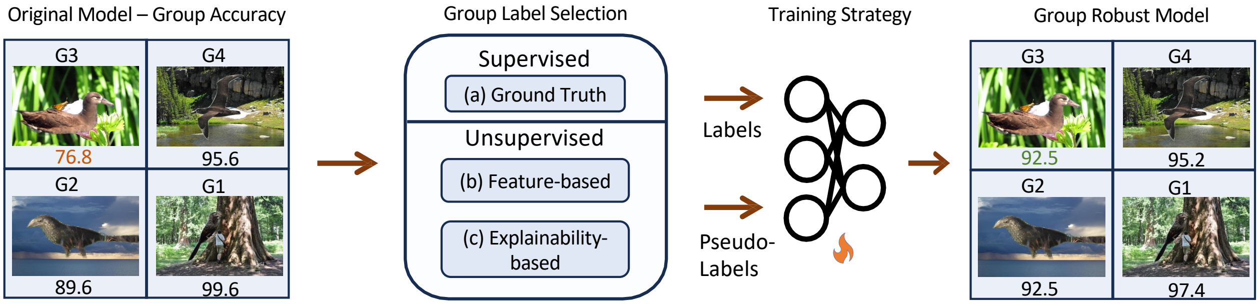


Figure 1. To improve the original models worst group accuracy, most current approaches rely on supervised group labels (a), which requires extensive annotation processes. Unsupervised approaches have relied on extracting pseudo labels based on the models feature representations (b), where information can be highly entangled. ExMap instead infers group pseudo labels based on explainability heatmaps (c), leading to improved worst group performance.

dataset by introducing ExMap, a novel two stage mechanism: First, we extract explainability heatmaps from a trained (base) model on the dataset of interest (we use the validation set *without* group labels). Next, we use a clustering module to produce pseudo-labels for the validation data. The resulting pseudo-labels can then be used for any off-the-shelf group robustness learning strategy in use today. ExMap is also flexible in the choice of clustering algorithm. We show that attaching the ExMap mechanism to baseline methods leads to improved performance over the unsupervised counterparts, and further closes the gap to supervised and partially supervised counterparts. Additionally, we demonstrate that ExMap is also useful in the recent *multiple shortcut* paradigm [18], where current popular supervised approaches have been shown to struggle. We conclude with an extended analysis on why clustering explainability heatmaps is more beneficial than raw features. In summary, our contributions include:

1. ExMap: A simple but efficient unsupervised, strategy agnostic mechanism for group robustness that leverages explainability heatmaps and clustering to generate pseudo-labels for underlying groups.
2. An extended analysis that provides intuition and insight into why clustering explainability heatmaps leads to superior results over other group-robustness baseline methods.
3. Demonstrating the usefulness of ExMap in improving worst group robustness in both the single shortcut and multiple shortcut settings.

2. Related Work

Single shortcut mitigation with group labels The paradigm of taking a frozen base model and proposing a shortcut mitigation strategy to maximise *worst group accuracy* was introduced in Group-DRO (gDRO) [28]. However, the requirement of group labels in both training and validation data motivated the proposal of mitigation strategies without training labels. This has resulted partially supervised approaches [33] that only require a small set of group labels as well as in several methods that only require the validation group labels [13, 19, 26]. One such example is DFR[13], which re-trains the final layer of a base ERM model on a balanced, reweighting dataset. Most relevant to our work, GEORGE [32] proposes an unsupervised mechanism to generate pseudo-labels for retraining by clustering raw features, and can therefore be considered the closest method to our proposed ExMap. We show that clustering heatmaps is a more beneficial and intuitive technique for generating pseudo-labels, as attributing the model performance on the input data pixels leads to a more intuitive interpretation of which features are relevant for the task, and which are not. Our method, ExMap, leverages this insight and clusters the heatmaps instead, leading to improved performance over GEORGE and its two variants - GEORGE(gDRO) trained with the Group-DRO strategy, and GEORGE(DFR), trained with the DFR strategy.

Other Strategies for Shortcut Mitigation There are other extant works that mitigate spurious correlations without adopting the group-label based paradigm directly. MaskTune [2], for instance, learns a mask over discriminatory features to reduce reliance on spurious correlations. CVar DRO [17] proposes an efficient robustness strategy using conditional value at risk (CVar). DivDis [16], on the other hand, proposes to train multiple functions on source and target data, identifying the most informative subset of labels in the target data. Discover-and-Cure (DISC) [37] discovers spurious concepts using a predefined concept bank, then intervenes on the training data to mitigate the spurious concepts, while ULA [35] uses a pretrained self-supervised model to train a classifier to detect and mitigate spurious correlations. While these approaches do not directly adopt the group-label, we show that the proposed explainability heatmap-based approach is more efficient in improving the worst-group accuracy.

Multi-Shortcut Mitigation The single shortcut setting is a simpler benchmark as the label is spuriously correlated with only a single attribute. However, real world datasets are challenging, and may contain multiple spurious attributes correlated with an object of interest. As a result, when one spurious attribute is known, mitigating the reliance on this attribute may exacerbate the reliance on another. The recently introduced Whac-A-Mole [18] dilemma for multiple shortcuts demonstrates this phenomenon with datasets containing multiple shortcuts (e.g. background and co-occurring object). Single shortcut methods fail to mitigate *both* shortcuts at once, leading to a spurious conservation principle, where if one shortcut is mitigated, the other is exacerbated. The authors introduce Last Layer Ensemble (LLE) to mitigate multiple shortcuts in their datasets, by training a separate classifier for each shortcut. However, LLE's reliance on apriori knowledge of dataset shortcuts is impractical in the real world. We evaluate ExMap in this context and show that it is effective as an unsupervised group robustness approach to the multi-shortcut setting.

Heatmap-based Explainability The challenge of attributing learned features to the decision making of a model in the image space has a rich history. The techniques explored can be differentiated on a variety of axes. LIME, SHAP, LRP [3, 11, 22] are early model-agnostic methods, while Grad-CAM and Integrated Gradients[30, 34] are gradient based attribution methods. We use LRP in this work owing to its popularity, but in principle, the heatmap extraction module can incorporate any other method widely in use today. LRP is a backward propagation based technique relying on the relevance conservation principle across each neuron in each layer. The output is a set of relevance scores that can be attributed to a pixel wise decomposition of the input image. Heatmap-based explainability techniques have also been used in conjunction with clustering, in the context of discovering model strategies for classification, and disparate areas such as differential privacy [6, 15, 29].

3. Worst Group Robustness

In this section, we provide notation and brief background of the group robustness problem. We are given a dataset D with image-label pairs being defined as $D = \{(x_i, y_i)\}_{i=1}^{N}$, where x_i represents an image, y_i is its corresponding label, and N is the number of pairs in the dataset. The model's prediction for an image x is $y_{\text{pred}} = \hat{f}(x)$. The cross-entropy loss for true label y and predicted label y_{pred} is given by $L(y, y_{\text{pred}}) = -\sum_{c=1}^{C} y_c \log(y_{\text{pred},c})$, where C is the number of classes. Then, an ERM classifier simply minimizes the average loss over the training data:

$$\hat{f} = \arg\min_{f} \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i)) \tag{1}$$

where $\hat{f}$ is the model obtained after training. Next, given the validation data D, we assume that for the class label set $L = \{c_1, c_2, ..., c_k\}$ there exists a corresponding spurious attribute set $A = \{a_1, a_2, ..., a_m\}$, such that the group label set $G : L \times A$. For example, in CelebA, typically a : Gender (Male/Female), and c: Blonde Hair (Blonde/Not Blonde). In this case $L = \{0, 1\}$, and $A = \{0, 1\}$. Then, the optimization can be described as the worst-expected loss over the validation set, conditioned on the group labels and the spurious attributes:

$$\hat{f}^* = \arg\min_{f} \max_{(c_i, a_j) \in G} \mathbb{E}_{(x,y)\in D}[L(y, f(x))|c_i, a_j] \tag{2}$$

As discussed before, recent works aim to design strategies over the (base) trained model to minimize this objective. For example, JTT collects an error set from the training data, and then upweights misclassified examples during the second training phase. DFR reweights the features responsible for misclassifications during the first phase in its finetuning stage. Note, however, that both these methods rely on the validation set *group labels* G_{val} to finetune the network. We consider the case where $G_{train} = \phi$ and $G_{val} = \phi$. We do not have access to group labels, and must therefore infer pseudo-labels in an unsupervised manner so that existing group robustness methods can be used.

4. Leveraging Explainability Heatmaps for Group Robustness – ExMap

In this section, we describe ExMap, an intuitive and efficient approach for unsupervised group robustness to spurious correlations. ExMap is a two-stage method, illustrated in Figure 2. In the first stage, we extract explainability heatmaps for the model predictions. In the second stage, we cluster the heatmaps to generate pseudo-labels. These pseudo-labels can then be used on any off-the-shelf group robustness strategy in use today. In our work, we demonstrate the strategy agnostic nature of ExMap by running it on two popular strategies - JTT and DFR.

4.1. Explainability Heatmaps

We use LRP [3] in this stage to generate pixel attributions in the input space. This allows us to focus only on the relevant features for the task. Specifically, given the validation data $D_{val} = \{(x_i, y_i)\}_{i=1}^{M}$, we use pixel wise relevance score $r_x = (\mathcal{LRP}(x)) \forall x \in D_{val}$. Specifically, for each data point x, the relevance score is defined on a per-neuron-per layer basis. For an input neuron n_k at layer k, and an output neuron n_l at the following layer l, the relevance score R_k is intuitively a measure of how much this particular input n_k contributed to the output value n_l:

$$R_k = \sum_{l:k \longrightarrow l} \frac{z_{kl}}{z_l + \varepsilon \cdot \text{sign}(z_l)} \tag{3}$$

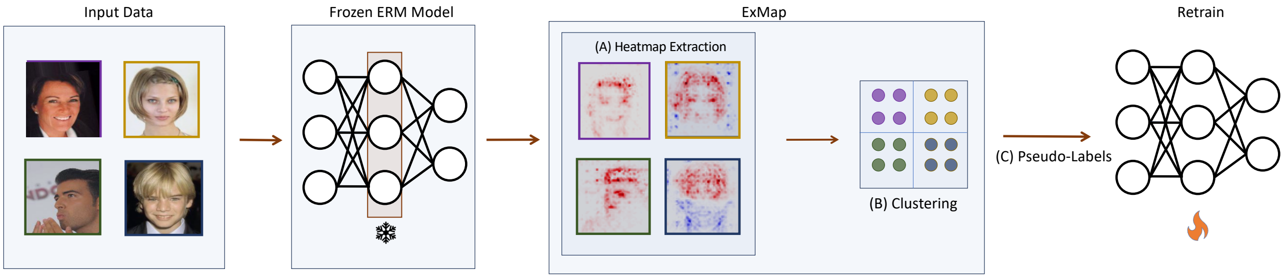


Figure 2. Our Proposed Method: ExMap facilitates group-robustness by extracting explainability heatmaps from the frozen base ERM model for the validation data (A). These heatmaps are then clustered (B) to obtain pseudo-labels for the underlying groups, which are subsequently chosen for the retraining strategy (C).

Algorithm 1 Generating Pseudo-labels using G-ExMap

```
Input: Dataset $D_{val}$, ERM Model $\mathcal{M}$, DataLoader $\mathcal{L}$
Output: Pseudo-labels $\hat{G}$
procedure GENERATEPSEUDOLABELS($D, \mathcal{M}, \mathcal{L}$)
    $R \leftarrow \varnothing$                                ▷ Initialize heatmap set
    for each batch $x$ in $\mathcal{L}$ do
        pred $\leftarrow \arg\max_i \mathcal{M}(x)_i$
        for each layer $k, l$ do
            Compute $z_l \leftarrow n_k w_{kl}$
        end for
        Compute LRP relevance $r_x$ for $x$ using Eq. 3
        Add $r_x$ to $R$
    end for
    Cluster $R$ using G-ExMap method:
    $\hat{A} \leftarrow$ Cluster($R$)                    ▷ Estimated spurious labels
    Combine class labels $L$ with $\hat{A}$
                    $$\hat{G} \leftarrow L \times \hat{A}$$
    return Pseudo-labels $\hat{G}$
end procedure
```

where $z_l = n_k w_{kl}$ for the weight connection w_{kl}. R_k is computed for all the neurons at layer k, and backpropagated from the output layer to the input layer to generate pixel level relevance scores r_x for each data input x. We can thus build the heatmap set $R = \{r_x | x \in D_{val}\}$. This process is summarized in Algorithm 1.

4.2. Clustering

In the second stage, we cluster the LRP representations from the first stage. The intuition here is that over the data, the heatmaps capture the different strategies undertaken by the model for the classification task [15]. The clustering module helps identify dominant model strategies used for the classification task. By identifying such strategies and resampling in a balanced manner, ExMap guides the model to be less reliant on the dominant features across the data, i.e. the spurious features. The heatmaps serve as an effective proxy to describe model focus areas. We have two options in choosing how to cluster: Local-ExMap (L-ExMap), where we cluster heatmaps on a per-class basis, and Global-ExMap (G-ExMap), where we cluster all the heatmaps at once, and segment by class labels. We present the G-ExMap results in this paper, owing to better empirical results.

Specifically, given the Heatmap set R as described in Algorithm 1, the estimated spurious labels are generated by the global clustering method, $\hat{A}$ = Cluster(R) where Cluster (.) represents a clustering function. Now, given class label set L and estimated spurious label set $\hat{A}$, we can generate our pseudo-*group* label set $\hat{G} = L \times A$ by selecting each $a_i \in \hat{A}$, and each $c_k \in L$, to create $\{c_k, a_i\}\ \forall k, i$.

In principle, it doesn't matter what clustering method we use, but that the clustering process itself outputs useful pseudo-labels. For our work, we leverage spectral clustering with an eigengap heuristic, inspired by SPRAY [15]. Later, we show that the choice of the clustering method does not have a significant effect on the results. The outputs, which are the pseudo group labels for the validation data D_{val}, can now be used as labels in lieu of ground truth labels to train any group robustness strategy in use. Note how in principle, *any* method that uses group labels (training or validation) would benefit from this approach. To apply this to the training set D, one would simply repeat Algorithm 1 on D. In this work, we apply ExMap to two common group robustness strategies - JTT [19] and DFR [13]. Thus, we demonstrate the strategy-agnostic nature of our approach that can be applied to any off-the-shelf method using group labels today.

5. Experiments

In this section we first present the datasets, baselines, and experimental setup. Next, we present the results and discussion[2].

[2] A discussion of the limitations and societal impact can be found in the supplementary material.

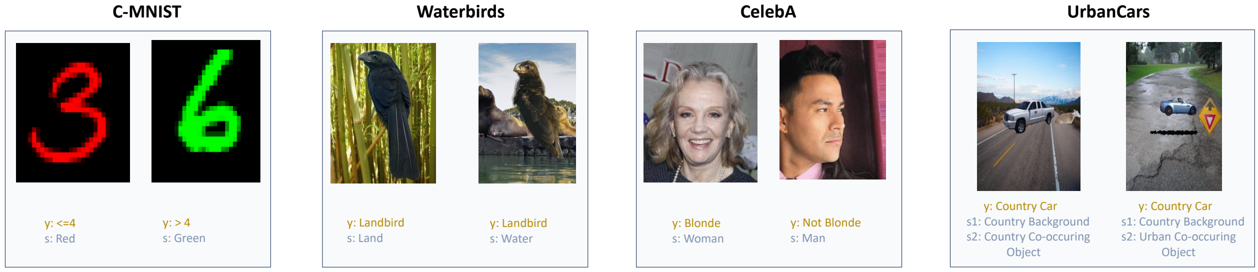


Figure 3. The datasets used in our work, visualized with respect to the class labels, and the shortcuts s. For the complete list of datasets and more details, please refer to the supplementary material.

Datasets We use CelebA [20], Waterbirds [28, 36, 39], C-MNIST [1], and Urbancars [18]. In CelebA, the class label to be predicted is hair colour (Blonde/Not Blonde), and the spurious attribute is gender (Male/Female). For Waterbirds, the class label is the bird type (waterbird/landbird), and the spurious attribute is the background (land/water). In C-MNIST, the class label is if the number is smaller than or equal to four. Any number lesser than or equal to four is assigned blue, while all numbers greater than four are assigned the color red, with a correlation of 99%. Thus, the spurious attribute is the color. For Urbancars, the class label is the car type (country/urban), and the spurious attributes are the background and co-occuring object (both country/urban). We create two variants of Urbancars: The first variant is Urbancars (BG), where only the background object is the spurious attribute. The second variant is Urbancars (CoObj), where the co-occuring object is the spurious attribute. We present single shortcut results on CelebA, Waterbirds, C-MNIST, Urbancars (BG) and Urbancars (CoObj). For the multiple shortcut setting, we use the original UrbanCars dataset with both shortcuts [18]. An overview over the considered datasets can be found in Figure 3. We present more dataset details in the supplementary.

Baselines We use the unsupervised approaches DivDis, MaskTune, and two variants of GEORGE (with gDRO and DFR) as the baselines in our work. We also adapt LfF, JTT, and CVar DRO to the unsupervised setting as additional baselines. We train the ERM model using an Imagenet-pretrained Resnet-50, and use the open source implementations of the baselines to generate our results. Specifically, we implement GEORGE(DFR), ExMap, and JTT. Remaining results are reported from [2], [19], and [16].

Setup We make sure to use the same hyperparameters from the baseline papers to reproduce the results. We utilise a composite of LRP rules to get the explainability heatmaps as recommended by [14, 23]. Following their recommendations we use LRP-ϵ for the dense layers near the output of the model with small epsilon ($\epsilon \ll 1$), followed by LRP-γ for the convolutional layers.

For the spectral clustering, we use the affinity matrix, and cluster-QR [29] to perform the clustering. The eigengap heuristic is applied to the 10 smallest eigenvalues of the Laplacian matrix to select the number of significant clusters to use. We demonstrate later that using a simpler clustering approach such as k-means can also emit reasonable results. For more details on the affinity matrix, clustering and pseudo-label generation, please see the supplementary.

5.1. Results: Single Shortcut

In Table 1 we present the single shortcut results for the datasets. First, we note that with no supervision, ExMap based DFR improves significantly upon ERM. Second, we note the improved performance of ExMap based DFR over the unsupervised baselines, including GEORGE, our closest baseline. Further, since the DFR-based GEORGE and ExMap significantly outperforms the other baselines, we present results comparing these two methods on C-MNIST, Urbancars (BG) and Urbancars (CoObj) in Table 2. In both tables, we demonstrate the superiority of clustering heatmaps to generate pseudo-labels instead of the raw features as in GEORGE. These results also show that the groups inferred by ExMap are indeed useful for worst group robustness to spurious correlations. Third, we note the gap in performance between DFR and ExMap based DFR. Since the former uses validation labels, we expect an increased accuracy, but we can report better performance on Waterbirds, and within 3% 2% 8% and 6% of the DFR results on the remaining datasets. On CelebA, our results are within 5% of Group-DRO, which demonstrates the best overall results. However, note that Group-DRO is a fully supervised approach, using labels from both the training and validation sets. For all datasets, we are able to outperform GEORGE, our closest baseline. As discussed before, while mean accuracy is not the appropriate metric to track in the group robustness setting (ERM has the best overall mean accuracy but the worst overall worst group accuracy), we can still confirm that ExMap based DFR does not witness significant drops in performance.

Methods	Group Info	Waterbirds		CelebA	
	Train/Val	WGA(%)↑	Mean(%)	WGA(%)↑	Mean(%)
Base (ERM)	✗/✗	76.8	98.1	41.1	95.9
Group DRO	✓/✓	91.4	93.5	88.9	92.9
EIIL	✓/✓	87.3	93.1	81.3	89.5
BARACK	✓/✓	89.6	94.3	83.8	92.8
CVar DRO	✗/✓	75.9	96.0	64.4	82.5
LfF	✗/✓	78.0	91.2	77.2	85.1
JTT	✗/✓	86.7	93.3	81.1	88.0
DFR	✗/✓	92.1	96.7	86.9	91.1
GEORGE (gDRO)	✗/✗	76.2	95.7	53.7	94.6
CVar DRO	✗/✗	62.0	96.0	36.1	82.5
LfF	✗/✗	44.1	91.2	24.4	85.1
JTT	✗/✗	62.5	93.3	40.6	88.0
DivDis*	✗/✗	81.0	-	55.0	-
MaskTune	✗/✗	86.4	93.0	78.0	91.3
GEORGE (DFR)	✗/✗	91.7	96.5	83.3	89.2
DFR+ExMap (ours)	✗/✗	**92.5**	96.0	**84.4**	91.8

Table 1. Worst group and mean accuracy on the test sets of the different datasets. The Group Info column showcases for each method whether group labels are used for that split of the data (✗= does not use group labels, ✓= uses group labels). We report the average results over 5 runs after hyperparameter tuning. Gray rows represent supervised approaches. *DivDis does not report mean test accuracy results.

Methods	Group Info	C-MNIST		Urbancars (BG)		Urbancars (CoObj)	
	Train/Val	WGA(%)↑	Mean(%)	WGA(%)↑	Mean(%)	WGA(%)↑	Mean(%)
Base (ERM)	✗/✗	39.6	99.3	55.6	90.2	50.8	92.7
DFR	✗/✓	74.2	93.7	77.5	81.0	84.7	88.2
GEORGE (DFR)	✗/✗	71.7	95.2	69.1	83.6	76.9	91.4
DFR+ExMap (ours)	✗/✗	**72.5**	94.9	**71.4**	93.2	**79.2**	93.2

Table 2. Worst Group accuracy and mean accuracy on C-MNIST, Urbancars (BG), and Urbancars (CoObj). We use GEORGE as the baseline, since both GEORGE and ExMap significantly outperform other unsupervised methods on Waterbirds and CelebA. Gray rows represent supervised approaches.

5.2. Results: Multiple Shortcuts

Here, we present the results on the UrbanCars data, which contains multiple shortcuts in the images - the background and the co-occuring object in the image. This dataset was introduced in the recent work on multiple shortcut mitigation [18], where the authors show that mitigating one shortcut may lead to a reliance on another shortcut in the data, rendering the single shortcut setting incomplete (the Whac-a-Mole problem). The authors introduce a new set of metrics for the task - The **BG Gap**, which is the drop in accuracy between mean and cases when only the background is uncommon, the **CoObj Gap** which is the drop in accuracy between mean and cases when only the co-occurring object is uncommon, and the **BG+CoObj Gap**, the drop when both the background and the co-occurring object are uncommon. A mitigation strategy should witness a smaller drop from the original accuracy when compared to others. In Table 3, we present the ExMap based DFR results with respect to DFR, ERM, and GEORGE(DFR). We also present results of three variants of DFR: DFR (Both), which is retraining on the original UrbanCars data with both shortcuts. DFR(BG) retrains on UrbanCars with only the background shortcut, and DFR(CoObj) retrains with only the co-occuring object shortcut. Red values indicate an increase in gap when compared to ERM, which is undesirable (the Whac-A-Mole dilemma). Note that the first three DFR methods have access to the group labels, while GEORGE and ExMap do not. Table 3 demonstrates some important results: First, that DFR + ExMap consistently posts lower drops than the base ERM model. Second, that ExMap does not witness an increase in gap on any of the metrics compared to ERM, unlike GEORGE(DFR), which

Method	BG Gap ↑	CoObj Gap ↑	BG+CoObj Gap ↑
ERM	-8.2	-14.2	-69.0
DFR (Both)	-4.6	-5.4	-14.2
DFR (BG)	-0.3	-29.2 (× 2.06)	-33.2
DFR (CoObj)	-16.3 (× 1.99)	-0.5	-19.1
GEORGE (DFR)	-7.0	-15.4 (×1.08)	-63.4
DFR+ExMap (ours)	**-5.9**	**-9.9**	**-30.7**

Table 3. Multiple Shortcuts on UrbanCars. Red values indicate the Whac-A-Mole dilemma: Mitigating one shortcut exacerbates reliance on the other (compared to ERM). ExMap proves to be the most robust in this setting, and outperforms GEORGE, its direct unsupervised counterpart.

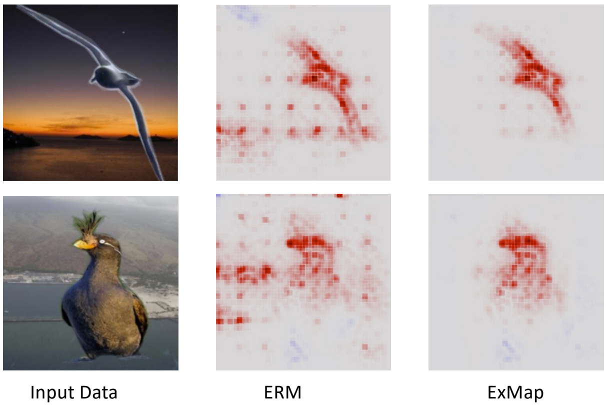


Figure 4. ERM and ExMap Heatmaps - Left: The Input images. Middle: ERM model explanations. Right: Improved group robustness using ExMap. Our method helps improve the focus on relevant attributes, in turn improving the pseudo-label estimation for retraining.

exhibits the Whac-A-Mole dilemma for the CoObj Gap. Finally, the DFR variants exhibit the Whac-A-Mole dilemma: For a DFR variant retrained on a particular shortcut, the reliance on that shortcut is mitigated (e.g. DFR (BG) mitigates the BG Gap), but the other shortcut reliance is exacerbated (DFR (BG) exhibits a higher CoObj Gap than ERM). Note that DFR uses the validation group labels, and hence will be more useful in mitigating shortcuts than our unsupervised setting. In fact, as demonstrated in [18], training separate classifiers for each shortcut is the best approach to mitigating multiple shortcuts, which explains DFR's best overall results. However, this setting assumes availability to the shortcut labels, which ExMap does not assume. Yet, it demonstrates a robust performance for the multi-shortcut setting even in the unsupervised setting, outperforming GEORGE, its closest unsupervised competitor.

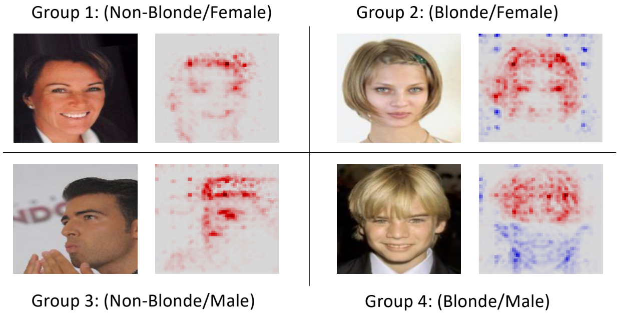


Figure 5. ExMap Heatmaps on CelebA: Each entry represents a group. The positive and negative attributions help interpret which features the model considers spurious (Blue), and which features are helpful (Red).

Methods	Mean (FG-Only %)	Mean (%)	Drop ↓
ERM	44.2	98.1	53.9
DFR	64.7	94.6	29.9
GEORGE (DFR)	73.2	96.5	23.3
DFR+ExMap (ours)	78.5	96.0	**17.5**

Table 4. Waterbirds (FG-Only). All methods exhibit a reliance on the background shortcut in Waterbirds, but ExMap posts the lowest drop, demonstrating its robustness.

6. Analysis

In this section, we present analysis and ablations along five axes: First, we demonstrate how the clustering of heatmaps is more useful than the clustering of features. Second, we demonstrate the usefulness of the ExMap representations when compared to ERM with respect to the classification task. Third, we provide more insight into what the learned clusters by ExMap capture in the data. Fourth, we demonstrate that ExMap is robust to the choice of clustering method, by performing an ablation on the clustering method using k-means instead of spectral clustering. Finally, to demonstrate that ExMap is strategy-agnostic, we use JTT as a retraining strategy using ExMap pseudo-labels, and are able to demonstrate robust performance with respect to JTT trained on true validation labels.

6.1. The benefit of heatmaps over features

In this section we add more insight into why leveraging heatmaps for worst group robustness is more useful over features, as for example done in GEORGE. Specifically, we illustrate how heatmap based clustering mitigates reliance on the image background, the spurious attribute in the Waterbirds dataset. The results illustrate a common intuition - Explainability heatmaps highlight only the features relevant for prediction, ignoring those that are not.

Circumventing background reliance Here, we present results on Waterbirds with the spurious attribute, i.e. background, removed. We call this variant Waterbirds (FG-Only), following [13]. Please refer to the supplementary section for examples. An effective group robustness method would not witness a sharp drop in test accuracy if the model does not rely on the background. In Table 4, we present these results.

We can clearly see that the heatmap clustering strategy

Methods	Group Info	WGA(%) ↑	Mean(%) ↑
Base (ERM)	✗/✗	76.8	98.1
DFR	✗/✓	92.1	96.7
DFR+ExMap (SC)	✗/✗	92.6	96.0
DFR+ExMap (KMeans)	✗/✗	92.5	95.9

Table 5. Worst group accuracy and mean accuracy on Waterbirds with two different clustering methods - kmeans and Spectral.

mitigates the background reliance better than the feature based clustering strategy of GEORGE (lowest drop among all methods). This is also intuitive as the heatmap attributions focus on only the relevant features for prediction, discarding the rest (see Figure 4).

6.2. Qualitative Analysis

ExMap improves explanations upon retraining We visualize the heatmaps and predictions of ERM and Exmap based DFR in Figure 4. This is an image of the Waterbirds dataset that ERM misclassifies. This is reflected on the heatmap, as ERM fails to capture the relevant features. On the other hand, ExMap based DFR correctly classifies the image and focuses on the correct object region of interest (bird), instead of the spurious attribute (background).

ExMap improves Model Strategy In Figure 5, each entry represents a particular group. The positive and negative relevance scores correspond to the features that the model considers relevant and spurious respectively. ExMap uncovers the strategy used to make the prediction: In all four groups, we see ExMap helps the model uncover the hair color as a strategy. In fact, in Group 4, the model also learns that the facial features (Gender) are *negatively* associated with the prediction task (Hair Color), which is what we desire from our method. The model has learned the shortcut between man and not-blonde hair, hence ExMap uncovers the negative relevance in the face, effectively uncovering this shortcut. These examples impart a notion of interpretability to our results, as we are able to explain why the model made a particular prediction, and what shortcuts are uncovered.

6.3. Ablation Analysis

Robustness to choice of clustering method Our proposed method does not depend on any particular clustering algorithm. Although we used spectral clustering, one can also use the simpler K-means [24] to capture the clusters for pseudo-labelling. In Table 5, we present the results on Waterbirds. We are able to demonstrate that there is no significant difference in the worst group robustness performance for the clustering method we choose[3]. Both improve upon the base ERM and DFR models, and hence, both are useful. Thus, ExMap is more about demonstrating the usefulness of a heatmap clustering pseudo-labelling module rather than the specifics of the clustering method itself.

Methods	Group Info	WGA(%) ↑	Mean(%) ↑
Base (ERM)	✗/✗	41.1	95.9
DFR	✗/✓	92.1	96.7
DFR+ExMap (ours)	✗/✗	92.6	96.0
JTT	✗/✓	86.7	93.3
JTT+ExMap (ours)	✗/✗	86.9	90.0

Table 6. Worst group and mean accuracy on Waterbirds for two different retraining strategies - JTT and DFR.

Robustness to choice of learning strategy All the results presented until now focus on the DFR backbone for shortcut mitigation. We mention previously that ExMap is strategy-agnostic, meaning that it can be applied to any off-the shelf method in use today. In Table 6, we show the results after applying ExMap to the JTT method on Waterbirds. We demonstrate similar performance to using JTT (originally uses validation labels) simply by using the pseudo labels proposed by ExMap. Additionally, we are able to improve over ERM's poor worst group accuracy as well.

7. Conclusion

The group robustness paradigm for deep learning classifiers raises important questions for when deep learning models succeed, but more importantly, *when they fail*. However, most of current research focuses on the setting where group labels are available. This assumption is impractical for real-world scenarios, where the underlying spurious correlations in the data may not be known apriori. While recent work investigating unsupervised group robustness mechanisms have shown promise, we show that further improvements are possible. In our work, we propose ExMap, where we cluster explainable heatmaps to generate pseudo-labels for the validation data. These pseudo-labels are then used on off-the-shelf group robustness learning mechanisms in use today. In addition to showing why using heatmaps over raw features is useful in this setting, our results demonstrate the efficacy of this approach on a range of benchmark datasets, in both the single and multi-shortcut settings. We are able to further close the gap to supervised counterparts, and outperform partially supervised and unsupervised baselines. Finally, ExMap opens up interesting avenues to further leverage explainability heatmaps in group robust learning.

Acknowledgements

This work was financially supported by the Research Council of Norway (RCN) FRIPRO grant no. 315029. It was further funded through the RCN Centre for Research-based Innovation funding scheme (Visual Intelligence, grant no. 309439), and Consortium Partners.

[3] Note, empirical results illustrated that k-means results were robust to the number of clusters, K, given that K was chosen sufficiently large.

References

[1] Martín Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *ArXiv*, abs/1907.02893, 2019. 5

[2] Saeid Asgari, Aliasghar Khani, Fereshte Khani, Ali Gholami, Linh Tran, Ali Mahdavi-Amiri, and Ghassan Hamarneh. Masktune: Mitigating spurious correlations by forcing to explore. *Advances in Neural Information Processing Systems*, 2022. 2, 5

[3] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10, 2015. 3

[4] Hyojin Bahng, Sanghyuk Chun, Sangdoo Yun, Jaegul Choo, and Seong Joon Oh. Learning de-biased representations with biased representations. *International Conference on Machine Learning*, 2019. 1

[5] Wieland Brendel and Matthias Bethge. Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. *International Conference on Learning Representations*, 2018. 1

[6] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. Detecting backdoor attacks on deep neural networks by activation clustering. *Workshop on Artificial Intelligence Safety*, 2019. 3

[7] John Duchi, Tatsunori Hashimoto, and Hongseok Namkoong. Distributionally robust losses for latent covariate mixtures. *Operations Research*, 71(2):649–664, 2023. 1

[8] John C Duchi, Peter W Glynn, and Hongseok Namkoong. Statistics of robust optimization: A generalized empirical likelihood approach. *Mathematics of Operations Research*, 46(3):946–969, 2021. 1

[9] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *International Conference on Learning Representations*, 2018. 1

[10] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard S. Zemel, Wieland Brendel, Matthias Bethge, and Felix Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2:665 – 673, 2020. 1

[11] Xiaojie Guo, Yu Li, and Haibin Ling. Lime: Low-light image enhancement via illumination map estimation. *IEEE Transactions on Image Processing*, 26:982–993, 2017. 3

[12] Pavel Izmailov, Polina Kirichenko, Nate Gruver, and Andrew G Wilson. On feature learning in the presence of spurious correlations. *Advances in Neural Information Processing Systems*, 35:38516–38532, 2022. 1

[13] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last layer re-training is sufficient for robustness to spurious correlations. *International Conference on Learning Representations*, 2022. 1, 2, 4, 7

[14] Maximilian Kohlbrenner, Alexander Bauer, Shinichi Nakajima, Alexander Binder, Wojciech Samek, and Sebastian Lapuschkin. Towards best practice in explaining neural network decisions with lrp. *International Joint Conference on Neural Networks*, pages 1–7, 2020. 5

[15] Sebastian Lapuschkin, Stephan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Unmasking clever hans predictors and assessing what machines really learn. *Nature Communications*, 10, 2019. 1, 3, 4

[16] Yoonho Lee, Huaxiu Yao, and Chelsea Finn. Diversify and disambiguate: Out-of-distribution robustness via disagreement. *International Conference on Learning Representations*, 2023. 2, 5

[17] Daniel Levy, Yair Carmon, John C Duchi, and Aaron Sidford. Large-scale methods for distributionally robust optimization. *Advances in Neural Information Processing Systems*, 33:8847–8860, 2020. 2

[18] Zhiheng Li, I. Evtimov, Albert Gordo, Caner Hazirbas, Tal Hassner, Cristian Cantón Ferrer, Chenliang Xu, and Mark Ibrahim. A whac-a-mole dilemma: Shortcuts come in multiples where mitigating one amplifies others. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20071–20082, 2023. 2, 3, 5, 6, 7

[19] Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. Just train twice: Improving group robustness without training group information. *International Conference on Machine Learning*, pages 6781–6792, 2021. 1, 2, 4, 5

[20] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. *International Conference on Computer Vision*, 2015. 5

[21] Vishnu Suresh Lokhande, Kihyuk Sohn, Jinsung Yoon, Madeleine Udell, Chen-Yu Lee, and Tomas Pfister. Towards group robustness in the presence of partial group labels. *ICML 2022: Workshop on Spurious Correlations, Invariance and Stability*, 2022. 1

[22] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017. 3

[23] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-wise relevance propagation: an overview. *Explainable AI: interpreting, explaining and visualizing deep learning*, pages 193–209, 2019. 5

[24] Shi Na, Liu Xumin, and Guan Yong. Research on k-means clustering algorithm: An improved k-means clustering algorithm. *Third International Symposium on Intelligent Information Technology and Security Informatics*, pages 63–67, 2010. 8

[25] Junhyun Nam, Jaehyung Kim, Jaeho Lee, and Jinwoo Shin. Spread spurious attribute: Improving worst-group accuracy with spurious attribute estimation. *International Conference on Learning Representations*, 2021. 1

[26] Jun Hyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: De-biasing classifier from biased classifier. *Neural Information Processing Systems*, 2020. 2

[27] Yonatan Oren, Shiori Sagawa, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust language modeling. *Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 4227–4237, 2019. 1

[28] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks. *International Conference on Learning Representations*, 2019. 1, 2, 5

[29] Lukas Schulth, Christian Berghoff, and Matthias Neu. Detecting backdoor poisoning attacks on deep neural networks by heatmap clustering. *ArXiv*, abs/2204.12848, 2022. 3, 5

[30] Ramprasaath R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128:336 – 359, 2016. 3

[31] Sahil Singla and Soheil Feizi. Causal imagenet: How to discover spurious features in deep learning? *CoRR*, abs/2110.04301, 2021. 1

[32] Nimit Sohoni, Jared Dunnmon, Geoffrey Angus, Albert Gu, and Christopher Ré. No subclass left behind: Fine-grained robustness in coarse-grained classification problems. *Advances in Neural Information Processing Systems*, 33: 19339–19352, 2020. 1, 2

[33] Nimit Sharad Sohoni, Maziar Sanjabi, Nicolas Ballas, Aditya Grover, Shaoliang Nie, Hamed Firooz, and Christopher Re. Barack: Partially supervised group robustness with guarantees. *ICML 2022: Workshop on Spurious Correlations, Invariance and Stability*, 2022. 2

[34] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. *International Conference on Machine Learning*, 2017. 3

[35] Christos Tsirigotis, Joao Monteiro, Pau Rodriguez, David Vazquez, and Aaron C Courville. Group robust classification without any group information. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[36] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. Caltech-ucsd-birds-2011. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 5

[37] Shirley Wu, Mert Yuksekgonul, Linjun Zhang, and James Zou. Discover and cure: Concept-aware mitigation of spurious correlation. *International Conference on Machine Learning*, 2023. 2

[38] Michael Zhang, Nimit S Sohoni, Hongyang R Zhang, Chelsea Finn, and Christopher Re. Correct-n-contrast: a contrastive approach for improving robustness to spurious correlations. *International Conference on Machine Learning*, 162:26484–26516, 2022. 1

[39] Bolei Zhou, Agata Lapedriza, Antonio Torralba, and Aude Oliva. Places: An image database for deep scene understanding. *Journal of Vision*, 17(10):296–296, 2017. 5

[40] Chunting Zhou, Xuezhe Ma, Paul Michel, and Graham Neubig. Examining and combating spurious features under distribution shift. *International Conference on Machine Learning*, pages 12857–12867, 2021. 1

ExMap: Leveraging Explainability Heatmaps for Unsupervised Group Robustness to Spurious Correlations

Supplementary Material

In this supplementary material, we present additional details about the following:

- The datasets used - C-MNIST, Waterbirds, CelebA, Urbancars, Urbancars single shortcut variants, Waterbirds (FG-Only).
- Experimental Setup - The details on the heatmap extraction and clustering phase in ExMap.
- Additional results providing further intuition on how ExMap captures underlying group information.
- More results on the robustness of our method with standard errors.
- The connection between group robustness and fair clustering.
- Limitations and Societal Impact.

1. Datasets

We present the number of examples from each group for all the datasets, and the process of generating them. For C-MNIST, we used the same setup as in [2]. For Waterbirds and CelebA, we use the same setup as in [6, 8]. For Urbancars we use the same setup as in [7].

1.1. C-MNIST

We create a dataset where we have control of the number of elements in each group and what the spurious attribute is.

The Colored-MNIST dataset is a synthetic dataset based on the well-known MNIST. The MNIST dataset is a collection of several thousands of examples of handwritten digits (0-9). The images are single-channelled (black and white) and have a size of 28x28 pixels, and are accompanied by a label giving the ground truth.

We use the original data split, 60000 train and 10000 test. Since the original dataset does not have a validation set, we use the last 10000 images of the training set as the validation set.

We convert the dataset into a 2 class problem by modifying the task. This is done by simply going over to classify the numbers as smaller or equal to 4 ($y = 0$: value $<= 4$), and larger than 4 ($y = 1$: value > 4). To create the spurious attributes we make use of colors. Red is used as the first spurious attribute ($s = 0$: RGB $= (255, 0, 0)$), and green is used as the second spurious attribute ($s = 1$: RGB $= (0, 255, 0)$). Naturally, the images will need to be made 3-channeled to account for this change.

As we are interested in combating spurious correlations we create the dataset in a way such that there are correlations between the classes and spurious attributes. We use 99% correlation. That means that 99% of images from one class will have the same colour, while the remaining 1% will have the other colour. The amount of correlation was deliberately chosen so that ERM worst group accuracy is low. Table 1 shows the number of images in each group for each split.

Split	Total Data	Groups			
		Group 0 (y=0, s=0)	Group 1 (y=0, s=1)	Group 2 (y=1, s=0)	Group 3 (y=1, s=1)
Train	50,000	254	25,284	24,231	231
Val	10,000	45	5,013	4,893	49
Test	10,000	48	5,091	4,815	46

Table 1. Data splits in the Colored-MNIST dataset.

1.2. Waterbirds

Waterbirds [10] is a synthetic dataset created with the purpose of testing a model's reliance on background. The dataset consists of RGB images depicting different types of birds on different types of backgrounds. The different types of birds are divided into 2 classes, landbirds ($y = 0$) and waterbirds ($y = 1$). The different backgrounds are also divided into 2 and represent the spurious attributes of this dataset: land background ($s = 0$) and water background ($s = 1$). The group distributions across the different splits are presented in Table 2.

The Waterbirds dataset is created by using 2 other datasets, the Caltech-UCSD Birds-200-2011 (CUB) dataset [13] and the Places dataset [14]. The CUB dataset contains images of birds labelled by species and their segmentation masks. To construct the Waterbirds dataset the labels in the CUB dataset are split into 2 groups, where waterbirds are made up of seabirds (albatross, auklet, cormorant, frigatebird, fulmar, gull, jaeger, kittiwake, pelican, puffin, or tern) and waterfowls (gadwall, grebe, mallard, merganser, guillemot, or Pacific loon), while the remaining classes are labelled as landbirds. The birds are cropped using the pixel-level segmentation masks and pasted onto a water background (categories: ocean or natural lake) or land background (categories: bamboo forest or broadleaf forest) from the Places dataset.

The official train-test split of the CUB dataset is used, and 20% of the training set is used to create the validation set. The group distribution for the training set is such that

Split	Total Data	Groups			
		Group 0 (y=0, s=0)	Group 1 (y=0, s=1)	Group 2 (y=1, s=0)	Group 3 (y=1, s=1)
Train	4,795	3,498	184	56	1,057
Val	1,199	467	466	133	133
Test	5,794	2,255	2,255	642	642

Table 2. Data splits in the Waterbirds dataset.

most images (95%) depict bird types with corresponding backgrounds, to represent a distribution that may arise from real-world data. This distribution turns the background into a spurious feature. Take note that there is a distribution shift from the training split to the validation and test splits which are both more balanced, and include many more elements for the minority group. The creators of the dataset argue that they do this to more accurately gauge the performance of the minority groups, something that might be difficult if there are too few examples. They also do this to allow for easier hyperparameter tuning.

1.3. Celeb-A

CelebA here is a reference to a part of the CelebA celebrity face dataset [9] that was introduced by [10] as a group robustness dataset. From the original dataset, the feature *Blond_Hair* is used as the class, meaning that the images are divided into people who are not blonde ($y = 0$) and blonde ($y = 1$). Meanwhile, as a spurious attribute, we use the feature *Male* from the original dataset, which divides into female ($s = 0$) and male ($s = 1$). The official train-val-test split of the CelebA dataset is used. Note in Table 3, that the splits are likely randomly created, which results in equally group-distributed splits. Across all splits the group (blonde, male) is the smallest.

This dataset tests for model reliance on strongly correlated features in a real-world dataset. Observe in Table 3 that $g_3 = (y = 1, s = 1)$ which represents blonde males is severely underrepresented compared to the other groups, hence we expect the model to learn gender as a spurious feature for the class blonde.

Split	Total Data	Group 0 (y=0, s=0)	Group 1 (y=0, s=1)	Group 2 (y=1, s=0)	Group 3 (y=1, s=1)
Train	162,770	71,629	66,874	22,880	1,387
Val	19,867	8,535	8,276	2,874	182
Test	19,962	9,767	7,535	2,480	180

Table 3. Data splits in the CelebA dataset.

1.4. Urbancars

We use Urbancars, as proposed by [7]. There are 4000 images per target class, i.e. 8000 images in total. The target class is the car type (country/urban), while the two shortcuts are the background type (country/urban), and co-occurring object (country/urban). For the exact list of the cars, objects, and background, please see [7].

1.5. Urbancars single shortcut variants

The original Urbancars data has eight group combinations due to two classes, and two shortcuts (Background and Co-Occurring object). For the single shortcut variants, we merge the 4 extra groups for one particular shortcut, to leave 4 groups for the other. For example: To create Urbancars (BG), we merge the 4 groups from the other shortcut (CoObj), to create four groups containing the single shortcut of background for each of the two classes. A similar procedure is adopted to create Urbancars (CoObj).

1.6. Waterbirds (FG-Only)

This dataset is created to evaluate how well the trained models circumvent background reliance on the Waterbirds dataset, since background is the shortcut in the data. We remove the backgrounds in all the images only on the test set. In Figure 1, we present some examples.

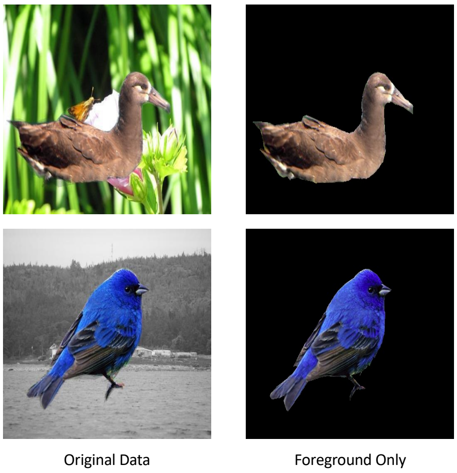


Figure 1. Waterbirds (FG Only)

2. Experimental Setup

In this section, we present more details on the heatmap extraction phase, clustering choice, and the hyperparameters used.

2.1. Heatmap Extraction

Following SPRAY [3], which reports good results across different downsizing of heatmaps, we sweep predominantly

Methods	**Group Info**	**C-MNIST**		**Urbancars (BG)**		**Urbancars (CoObj)**	
	Train/Val	WGA(%)↑	Mean(%)	WGA(%)↑	Mean(%)	WGA(%)↑	Mean(%)
Base (ERM)	✗/✗	39.6	99.3	55.6	90.2	50.8	92.7
GEORGE (DFR)	✗/✗	71.7±0.1	95.2±0.3	69.1±0.9	83.6±1.0	76.9±0.9	91.4±1.0
DFR+ExMap (ours)	✗/✗	**72.5**±0.2	94.9±0.3	**71.4**±0.8	93.2±0.2	**79.2**±0.7	93.2±0.3

Table 4. Group/mean test accuracy with std. Results over 5 runs.

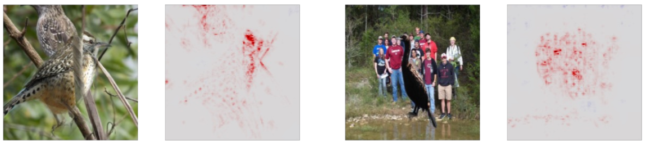


Figure 2. ExMap based misclassifications on challenging examples (Waterbirds): In each of these images, the object of interest (bird) is co-habited by dominant peripheral objects such as humans and other birds. These situations are challenging for the classifier to discern the relevant object from the irrelevant ones.

over the following downsizings: [224, 112, 100, 56, 28, 14, 7, 5, 3]. The downsizing of heatmaps additionally helps in speeding up the clustering process and mitigating potential out-of-memory issues.

2.2. Clustering choice

Since ExMap if flexible to the choice of clustering algorithm, we experiment with spectral clustering, UMap reduced KMeans [6], and KMeans. We use the eigengap heuristic with spectral to automatically choose the number of clusters, and sweep over different cluster sizes for the KMeans based methods. We look for the largest gap in among the first 10 eigenvalues. Otherwise we test for 2-15 clusters for kmeans (overclustering as practiced in [12]).

2.3. Hyperparameters

We use the same hyperparameters for DFR and JTT as in the original papers [6, 8].

For DFR, we perform the following steps:

- Given (pseudo)group labels we create a retraining set by subsampling each group to the size of the smallest group. These are then used to retrain the last layer. After being passed through the feature extractor, each sample is normalised based on the data used to retrain the last layer.
- Similar to [6], we use logistic regression with L1-loss.
- The strength of L1 is swept over [1.0, 0.7, 0.3, 0.1, 0.07, 0.03, 0.01]. The sweep is performed by randomly splitting the retraining dataset in 2, and performing retraining with one half and evaluating the performance with the other. This is performed 5 times with different splits and the best strength is chosen based on highest worst (pseudo)group accuracy.

Method	**Accuracy (%)**	
	Waterbirds	**CelebA**
	WGA / Mean	WGA / Mean
Base (ERM)	76.8 / 98.1	41.1 / 95.9
GEORGE (DFR)	91.7 ± 0.2 / 96.5 ± 0.1	83.3 ± 0.2 / 89.2 ± 0.2
DFR+ExMap	**92.5 ± 0.1** / 96.0 ± 0.3	**84.4 ± 0.5** / 91.8 ± 0.2

Table 5. Group / mean test accuracy with std. Results over 5 runs.

- When L1 strength has been selected, we retrain using the whole retrain set. This is performed 20 times with different subsamplings. The weights from each subsampling are averaged (this is viable according to DFR authors) to yield the final last layer weights. The normalisation of data is also averaged across the 20 runs.

For the ERM model, we perform the following steps:

- We use Resnet-18 for CMNIST, Resnet-50 for the others. We start with imagenet-pretrained Resnet-50 similar to previous work as it was observed to perform better. For all settings we replace the final fully connected layer to reflect the nature of our problems, i.e. 2 classes.
- Learning rate: 3e-3, weight decay: 1e-4, cosine learning rate scheduler.
- Batch size:We use batch size of 32 for Waterbirds and Urbancars, 100 for CelebA, and 128 for C-Mnist.
- Epochs: We train for 100 epochs on Waterbirds and Urbancars, 20 for CelebA, and 10 for C-Mnist.
- We use early stopping using the best mean (weighted) validation accuracy

For GEORGE, we perform the following steps:

- Acquire feature extractor (base ERM) outputs.
- Max normalise features.
- Cluster features as exmap or using UMAP+kmeans. We use 2 dimensions for the UMAP reduction, and high number of clusters (overclustering regime following [12]).

3. Capturing of Group Information

In addition to why ExMap representations are better for downstream group robustness over raw classifier features, we are also interested in what kind of group information the ExMap representations capture. The advantage of heatmaps are that they capture only the relevant features, while previous approaches that cluster in the feature space are prone to be effected by features that are irrelevant for the final prediction. To further substantiate our findings, we generate additional results to demonstrate that ExMap in-

Methods	Group Info	Waterbirds		CelebA	
	Train/Val	WGA(%)↑	Mean(%)	WGA(%)↑	Mean(%)
Base (ERM)	✗/✗	76.8	98.1	41.1	95.9
BPA	✗/✗	71.3	87.1	83.3	90.1
DFR+ExMap (ours)	✗/✗	**92.5**	96.0	**84.4**	91.8

Table 6. Comparison with Fair Clustering: Worst group and mean accuracy on Waterbirds and CelebA.

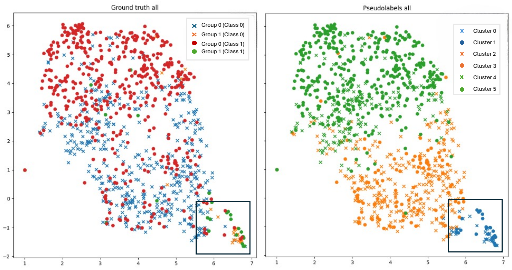


Figure 3. Groups in UrbanCars: (Left) Ground truth group labels per class. We observe minority groups (the spurious correlations) in the highlighted bottom right corner. (Right) Pseudo-labels learned by ExMap based clustering reveals a similar overall structure, conserving the dominant groups (green and yellow), while capturing the minority groups (blue cross and circle) as well.

deed captures the underlying group information. In Figure 3, we plot the pseudo-labels for UrbanCars (CoObj) after ExMap based clustering. ExMap captures both the dominant groups and the minority groups in the dataset, as indicated by the pseudo-labels learned. We also note that ExMap does not necessarily learn the same number of groups as in the ground truth data, since this information is assumed unavailable. The key observation from this figure is that ExMap is successful in identifying the dominant and minority group structure in the data. The group robust learner (such as DFR) can then sample across these groups in a balanced manner while retraining, leading to mitigation against spurious correlations.

4. Robustness Analysis

Our results in Table 1 and Table 2 in the main text are presented as the average of five runs. To illustrate the robustness of the compared approaches, we further provide the standard deviation for the ExMap and the main competitor in Table 4 and Table 5. We observe that results are robust across runs.

5. Connections to Fair Clustering

Given the close relationship between group robustness and the domain of fair clustering [1, 4, 5], we briefly comment on their connection and the potential of the insights of ExMap in the fair clustering setting. The domains of fair clustering and group robustness differ slightly, with the former aiming to improve mean accuracy independent of sensitive attributes, while the latter aim to maximize worst group accuracy. Therefore, there is a natural connection between these two research areas. Sensitive attributes in fair clustering can be regarded as a special type of spurious correlation, causally unrelated to the task. Recent work in fair clustering has therefore adopted some of the insights from the field of group robustness [11]. However, these approaches adopt a GEORGE inspired approach (cluster in raw features space), which we demonstrate to be sub-optimal in the context of group robustness. While an in-depth exploration of this is out-of-scope for this work, it could present an interesting avenue of future work. In Table 6, we present the ExMap results on Waterbirds and CelebA with respect to the method introduced in [11].

6. Limitations and Societal Impact

There are certain intuitive failure cases where the ExMap approach is not as efficient. This occurs when the images themselves are quite challenging to discern the objects of interest (the class), from other peripheral objects in the scene. In Figure 2, we present some examples of misclassifications by ExMap based DFR. In these images, we can see that the object of interest (bird), is co-habited by other dominant objects in the scene, such as humans and other birds. This creates an exceptionally challenging task for the classifier to discern the relevant features for the task. We recognise the need for robustness across challenging examples in datasets as motivation for future work. With regard to social impact, we recognise that model robustness to spurious correlations is an important first step in ensuring fair, transparent, and reliable AI that can be deployed in safety critical domains in the real world. Elucidating why models classify as they do, and specific failure cases uncovers shortcomings in exclusively choosing mean test accuracy as a metric. As a result, probing models for their weaknesses is as important as exemplifying their strengths.

References

[1] Sara Ahmadian, Alessandro Epasto, Ravi Kumar, and Mohammad Mahdian. Clustering without over-representation. *International Conference on Knowledge Discovery & Data Mining*, pages 267–275, 2019. 4

[2] Martín Arjovsky, Léon Bottou, Ishaan Gulrajani, and

David Lopez-Paz. Invariant risk minimization. *ArXiv*, abs/1907.02893, 2019. 1

[3] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10, 2015. 2

[4] Ioana O Bercea, Martin Groß, Samir Khuller, Aounon Kumar, Clemens Rösner, Daniel R Schmidt, and Melanie Schmidt. On the cost of essentially fair clusterings. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019)*, 2019. 4

[5] Anshuman Chhabra, Karina Masalkovaitė, and Prasant Mohapatra. An overview of fairness in clustering. *IEEE Access*, 9:130698–130720, 2021. 4

[6] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last layer re-training is sufficient for robustness to spurious correlations. *International Conference on Learning Representations*, 2022. 1, 3

[7] Zhiheng Li, I. Evtimov, Albert Gordo, Caner Hazirbas, Tal Hassner, Cristian Cantón Ferrer, Chenliang Xu, and Mark Ibrahim. A whac-a-mole dilemma: Shortcuts come in multiples where mitigating one amplifies others. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20071–20082, 2023. 1, 2

[8] Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. Just train twice: Improving group robustness without training group information. *International Conference on Machine Learning*, pages 6781–6792, 2021. 1, 3

[9] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. *International Conference on Computer Vision*, 2015. 2

[10] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks. *International Conference on Learning Representations*, 2019. 1, 2

[11] Seonguk Seo, Joon-Young Lee, and Bohyung Han. Unsupervised learning of debiased representations with pseudo-attributes. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16742–16751, 2022. 4

[12] Nimit Sohoni, Jared Dunnmon, Geoffrey Angus, Albert Gu, and Christopher Ré. No subclass left behind: Fine-grained robustness in coarse-grained classification problems. *Advances in Neural Information Processing Systems*, 33: 19339–19352, 2020. 3

[13] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. Caltech-ucsd-birds-2011. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 1

[14] Bolei Zhou, Agata Lapedriza, Antonio Torralba, and Aude Oliva. Places: An image database for deep scene understanding. *Journal of Vision*, 17(10):296–296, 2017. 1

11

Paper II

This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Hubs and Hyperspheres: Reducing Hubness and Improving Transductive Few-shot Learning with Hyperspherical Embeddings

Daniel J. Trosten[*,†] Rwiddhi Chakraborty[*,†] Sigurd Løkse[†] Kristoffer Knutsen Wickstrøm[†]
Robert Jenssen[†,‡,§,¶] Michael C. Kampffmeyer[†,‡]
Department of Physics and Technology, UiT The Arctic University of Norway
firstname[.middle initial].lastname@uit.no

Abstract

Distance-based classification is frequently used in transductive few-shot learning (FSL). However, due to the high-dimensionality of image representations, FSL classifiers are prone to suffer from the hubness problem, where a few points (hubs) occur frequently in multiple nearest neighbour lists of other points. Hubness negatively impacts distance-based classification when hubs from one class appear often among the nearest neighbors of points from another class, degrading the classifier's performance. To address the hubness problem in FSL, we first prove that hubness can be eliminated by distributing representations uniformly on the hypersphere. We then propose two new approaches to embed representations on the hypersphere, which we prove optimize a tradeoff between uniformity and local similarity preservation – reducing hubness while retaining class structure. Our experiments show that the proposed methods reduce hubness, and significantly improves transductive FSL accuracy for a wide range of classifiers[1].

1. Introduction

While supervised deep learning has made a significant impact in areas where large amounts of labeled data are available [6, 11], few-shot learning (FSL) has emerged as a promising alternative when labeled data is limited [3, 12, 14, 16, 21, 26, 28, 31, 33, 39, 40]. FSL aims to design classifiers that can discriminate between novel classes based on a few labeled instances, significantly reducing the cost of the labeling procedure.

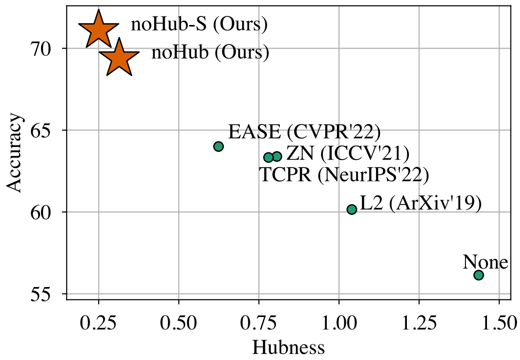


Figure 1. Few-shot accuracy increases when hubness decreases. The figure shows the 1-shot accuracy when classifying different embeddings with SimpleShot [33] on mini-ImageNet [29].

In transductive FSL, one assumes access to the entire query set during evaluation. This allows transductive FSL classifiers to learn representations from a larger number of samples, resulting in better performing classifiers. However, many of these methods base their predictions on distances to prototypes for the novel classes [3, 16, 21, 28, 39, 40]. This makes these methods susceptible to the hubness problem [10, 22, 24, 25], where certain exemplar points (hubs) appear among the nearest neighbours of many other points. If a support sample is a hub, many query samples will be assigned to it regardless of their true label, resulting in low accuracy. If more training data is available, this effect can be reduced by increasing the number of labeled samples in the classification rule – but this is impossible in FSL.

Several approaches have recently been proposed to embed samples in a space where the FSL classifier's performance is improved [4, 5, 7, 17, 33, 35, 39]. However, only one of these directly addresses the hubness problem. Fei *et al.* [7] show that embedding representations on a hypersphere with zero mean reduces hubness. They advocate the use of Z-score normalization (ZN) along the feature axis of each representation, and show empirically that ZN can reduce hubness in FSL. However, ZN does not guarantee a data mean of zero, meaning that hubness can still occur after ZN.

[*] Equal contributions.
[†] UiT Machine Learning group (machine-learning.uit.no) and Visual Intelligence Centre (visual-intelligence.no).
[‡] Norwegian Computing Center.
[§] Department of Computer Science, University of Copenhagen.
[¶] Pioneer Centre for AI (aicentre.dk).
[1] Code available at https://github.com/uitml/noHub.

In this paper we propose a principled approach to embed representations in FSL, which both reduces hubness and improves classification performance. First, we prove that hubness can be eliminated by embedding representations uniformly on the hypersphere. However, distributing representations uniformly on the hypersphere without any additional constraints will likely break the class structure which is present in the representation space – hurting the performance of the downstream classifier. Thus, in order to both reduce hubness and preserve the class structure in the representation space, we propose two new embedding methods for FSL. Our methods, Uniform Hyperspherical Structure-preserving Embeddings (noHub) and noHub with Support labels (noHub-S), leverage a decomposition of the Kullback-Leibler divergence between representation and embedding similarities, to optimize a tradeoff between Local Similarity Preservation (LSP) and uniformity on the hypersphere. The latter method, noHub-S, also leverages label information from the support samples to further increase the class separability in the embedding space.

Figure 1 illustrates the correspondence between hubness and accuracy in FSL. Our methods have both the *least hubness* and *highest accuracy* among several recent embedding techniques for FSL.

Our contributions are summarized as follows.

- We prove that the uniform distribution on the hypersphere has zero hubness and that embedding points uniformly on the hypersphere thus alleviates the hubness problem in distance-based classification for transductive FSL.

- We propose noHub and noHub-S to embed representations on the hypersphere, and prove that these methods optimize a tradeoff between LSP and uniformity. The resulting embeddings are therefore approximately uniform, while simultaneously preserving the class structure in the embedding space.

- Extensive experimental results demonstrate that noHub and noHub-S outperform current state-of-the-art embedding approaches, boosting the performance of a wide range of transductive FSL classifiers, for multiple datasets and feature extractors.

2. Related Work

The hubness problem. The hubness problem refers to the emergence of *hubs* in collections of points in high-dimensional vector spaces [22]. Hubs are points that appear among the nearest neighbors of many other points, and are therefore likely to have a significant influence on *e.g.* nearest neighbor-based classification. Radovanovic *et al.* [22] showed that points closer to the expected data mean are more likely be among the nearest neighbors of other points, indicating that these points are more likely to be hubs. Hubness can also be seen as a result of large density gradients [9], as points in high-density areas are more likely to be hubs. The hubness problem is thus an intrinsic property of data distributions in high-dimensional vector spaces, and not an artifact occurring in particular datasets. It is therefore important to take the hubness into account when designing classification systems in high-dimensional vector spaces.

Hubness in FSL. Many recent methods in FSL rely on distance-based classification in high-dimensional representation spaces [1, 3, 19, 33, 36, 38, 40], making them vulnerable to the hubness problem. Fei *et al.* [7] show that hyperspherical representations with zero mean reduce hubness. Motivated by this insight, they suggest that representations should have zero mean and unit standard deviation (ZN) *along the feature dimension*. This effectively projects samples onto the hyperplane orthogonal to the vector with all elements $= 1$, and pushes them to the hypersphere with radius $\sqrt{d}$, where d is the dimensionality of the representation space. Although ZN is empirically shown to reduce hubness, it does not guarantee that the data mean is zero. The normalized representations can therefore still suffer from hubness, potentially decreasing FSL performance.

Embeddings in FSL. FSL classifiers often operate on embeddings of representations instead of the representations themselves, to improve the classifier's ability to generalize to novel classes [5, 33, 35, 39]. Earlier works use the L2 normalization and Centered L2 normalization to embed representations on the hypersphere [33]. Among more recent embedding techniques, ReRep [5] performs a two-step fusing operation on both the support and query features with an attention mechanism. EASE [39] combines both support and query samples into a single sample set, and jointly learns a similarity and dissimilarity matrix, encouraging similar features to be embedded closer, and dissimilar features to be embedded far away. TCPR [35] computes the top-k neighbours of each test sample from the base data, computes the centroid, and removes the feature components in the direction of the centroid. Although these methods generally lead to a reduction in hubness and an increase in performance (see Figure 1), they are not explicitly designed to address the hubness problem resulting in suboptimal hubness reduction and performance. In contrast, our proposed noHub and noHub-S directly leverage our theoretic insights to target the root of the hubness problem.

Hyperspherical uniformity. Benefits of uniform hyperspherical representations have previously been studied for contrastive self-supervised learning (SSL) [32]. Our work differs from [32] on several key points. First, we study a non-parametric embedding of support and query samples for FSL, which is a fundamentally different task from contrastive SSL. Second, the contrastive loss studied in [32] is a combination of different cross-entropies, making it different from our KL-loss. Finally, we introduce a tradeoff-parameter between uniformity and LSP, and connect our theoretical results to hubness and Laplacian Eigenmaps.

3. Hyperspherical Uniform Eliminates Hubness

We will now show that hubness can be eliminated completely by embedding representations *uniformly* on the hypersphere[2].

Definition 1 (Uniform PDF on the hypersphere.)**.** *The uniform probability density function (PDF) on the unit hypersphere* $\mathbb{S}_d = \{\boldsymbol{x} \in \mathbb{R}^d \mid ||\boldsymbol{x}|| = 1\} \subset \mathbb{R}^d$ *is*

$$u_{\mathbb{S}_d}(\boldsymbol{x}) = A_d^{-1}\delta(||\boldsymbol{x}|| - 1) \tag{1}$$

where $A_d = \frac{2\pi^{d/2}}{\Gamma(d/2)}$ *is the surface area of* $\mathbb{S}_d$*, and* $\delta(\cdot)$ *is the Dirac delta distribution.*

We then have the following propositions[3] for random vectors with this PDF.

Proposition 1. *Suppose* $\boldsymbol{X}$ *has PDF* $u_{\mathbb{S}_d}(\boldsymbol{x})$*. Then*

$$\mathbb{E}(\boldsymbol{X}) = 0 \tag{2}$$

Proposition 2. *Let* $\Pi_{\boldsymbol{p}}$ *be the tangent plane of* $\mathbb{S}_d$ *at an arbitrary point* $\boldsymbol{p} \in \mathbb{S}_d$*. Then, for any direction* $\boldsymbol{\theta}^*$ *in* $\Pi_{\boldsymbol{p}}$ *the directional derivative of* $u_{\mathbb{S}_d}$ *along* $\boldsymbol{\theta}^*$ *is*

$$\nabla_{\boldsymbol{\theta}^*} u_{\mathbb{S}_d} = 0 \tag{3}$$

These two propositions show that the hyperspherical uniform has (i) zero mean; and (ii) zero density gradient along all directions tangent to the hypersphere's surface, at all points on the hypersphere. The hyperspherical uniform thus provably eliminates hubness, both in the sense of having a zero data mean, and having zero density gradient everywhere. We note that the latter property is un-attainable in Euclidean space, as it is impossible to define a uniform distribution over the whole space. It is therefore necessary to embed points on a non-Euclidean sub-manifold in order to eliminate hubness.

4. Method

In the preceding section, we proved that uniform embeddings on the hypersphere eliminate hubness. However, naïvely placing points uniformly on the hypersphere does not incorporate the inherent class structure in the data, leading to poor FSL performance. Thus, there exists a tradeoff between uniformity on the hypersphere and the preservation of local similarities. To address this tradeoff, we introduce two novel embedding approaches for FSL, namely noHub and noHub-S. noHub (Sec. 4.1) incorporates a novel loss function for embeddings on the hypersphere, while noHub-S (Sec. 4.2), guides noHub with additional label information, which should act as a supervisory signal for a class-aware embedding that leads to improved classification performance. Figure 2 provides an overview of the proposed noHub method. We also note that, since our approach generates embeddings, they are compatible with most transductive FSL classifier.

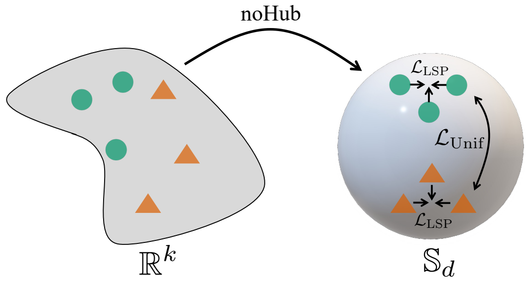

noHub
$\mathcal{L}_{\text{LSP}}$
$\mathcal{L}_{\text{Unif}}$
$\mathcal{L}_{\text{LSP}}$
$\mathbb{R}^k$
$\mathbb{S}_d$

Figure 2. Illustration of the noHub embedding. Given representations $\in \mathbb{R}^k$, $\mathcal{L}_{\text{LSP}}$ preserves local similarities. $\mathcal{L}_{\text{Unif}}$ simultaneously encourages uniformity in the embedding space $\mathbb{S}_d$. This feature embedding framework helps reduce hubness while improving classification performance.

Few-shot Preliminaries. Assume we have a large labeled *base* dataset $\mathcal{X}_{\text{Base}} = \{(\boldsymbol{x}_i, y_i) \mid y_i \in \mathcal{C}_{\text{Base}};\ i = 1, \dots, n_{\text{Base}}\}$, where $\boldsymbol{x}_i$ and y_i denotes the raw features and labels, respectively. Let $\mathcal{C}_{\text{Base}}$ denote the set of classes for the base dataset. In the few–shot scenario, we assume that we are given another labeled dataset $\mathcal{X}_{\text{Novel}} = \{(\boldsymbol{x}_i, y_i) \mid y_i \in \mathcal{C}_{\text{Novel}};\ i = 1, \dots, n_{\text{Novel}}\}$ from *novel*, previously unseen classes $\mathcal{C}_{\text{Novel}}$, satisfying $\mathcal{C}_{\text{Base}} \cap \mathcal{C}_{\text{Novel}} = \emptyset$. In addition, we have a test set $\mathcal{T}$, $\mathcal{T} \cap \mathcal{X}_{\text{Novel}} = \emptyset$, also from $\mathcal{C}_{\text{Novel}}$.

In a K–way N_S–shot FSL problem, we create randomly sampled *tasks* (or episodes), with data from K randomly chosen novel classes. Each task consists of a *support* set $\mathcal{S} \subset \mathcal{X}_{\text{Novel}}$ and a *query* set $\mathcal{Q} \subset \mathcal{T}$. The support set contains $|\mathcal{S}| = N_S \cdot K$ random examples (N_S random examples from each of the K classes). The query set contains $|\mathcal{Q}| = N_Q \cdot K$ random examples, sampled from the same K classes. The goal of FSL is then to predict the class of samples $\boldsymbol{x} \in \mathcal{Q}$ by exploiting the labeled support set $\mathcal{S}$, using a model trained on the base classes $\mathcal{C}_{\text{Base}}$. We assume a fixed feature extractor, trained on the base classes, which maps the raw input data to the representations $\boldsymbol{x}_i$.

4.1. noHub: Uniform Hyperspherical Structure-preserving Embeddings

We design an embedding method that encourages uniformity on the hypersphere, and simultaneously preserves local similarity structure. Given the support and query rep-

[2] Our results assume hyperspheres with unit radius, but can easily be extended to hyperspheres with arbitrary radii.

[3] The proofs for all propositions are included in the supplementary.

resentations $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n \in \mathbb{R}^k$, $n = K(N_S + N_Q)$, we wish to find suitable embeddings $\boldsymbol{z}_1, \dots, \boldsymbol{z}_n \in \mathbb{S}_d$, where local similarities are preserved. For both representations and embeddings, we quantify similarities using a softmax over pairwise cosine similarities

$$p_{ij} = \frac{p_{i|j} + p_{j|i}}{2}, \quad p_{i|j} = \frac{\exp(\kappa_i \frac{\boldsymbol{x}_i^\top \boldsymbol{x}_j}{||\boldsymbol{x}_i|| \cdot ||\boldsymbol{x}_j||})}{\sum\limits_{l,m} \exp(\kappa_i \frac{\boldsymbol{x}_l^\top \boldsymbol{x}_m}{||\boldsymbol{x}_l|| \cdot ||\boldsymbol{x}_m||})} \tag{4}$$

and

$$q_{ij} = \frac{\exp(\kappa \boldsymbol{z}_i^\top \boldsymbol{z}_j)}{\sum\limits_{l,m} \exp(\kappa \boldsymbol{z}_l^\top \boldsymbol{z}_m)}, \tag{5}$$

where κ_i is chosen such that the effective number of neighbours of $\boldsymbol{x}_i$ equals a pre-defined perplexity[4]. As in [27, 30], local similarity preservation can now be achieved by minimizing the Kullback-Leibler (KL) divergence between the p_{ij} and the q_{ij}

$$KL(P||Q) = \sum_{i,j} p_{ij} \log \frac{p_{ij}}{q_{ij}}. \tag{6}$$

However, instead of directly minimizing $KL(P||Q)$, we find that the minimization problem is equivalent to minimizing the sum of two loss functions[5]

$$\underset{\boldsymbol{z}_1, \dots, \boldsymbol{z}_n \in \mathbb{S}_d}{\arg\min} \; KL(P||Q) = \underset{\boldsymbol{z}_1, \dots, \boldsymbol{z}_n \in \mathbb{S}_d}{\arg\min} \; \mathcal{L}_{\text{LSP}} + \mathcal{L}_{\text{Unif}} \tag{7}$$

where

$$\mathcal{L}_{\text{LSP}} = -\kappa \sum_{i,j} p_{ij} \boldsymbol{z}_i^\top \boldsymbol{z}_j, \tag{8}$$

$$\mathcal{L}_{\text{Unif}} = \log \sum_{l,m} \exp(\kappa \boldsymbol{z}_l^\top \boldsymbol{z}_m). \tag{9}$$

In Sec. 5 we provide a thorough theoretical analysis of these losses, and how they relate to LSP and uniformity on the hypersphere. Essentially, $\mathcal{L}_{\text{LSP}}$ is responsible for the local similarity preservation by ensuring that the embedding similarities ($\boldsymbol{z}_i^\top \boldsymbol{z}_j$) are high whenever the representation similarities (p_{ij}) are high. $\mathcal{L}_{\text{Unif}}$ on the other hand, can be interpreted as a negative entropy on $\mathbb{S}_d$, and is thus minimized when the embeddings are uniformly distributed on $\mathbb{S}_d$. This is discussed in more detail in Sec. 5.

Based on the decomposition of the KL divergence, and the subsequent interpretation of the two terms, we formulate the loss in noHub as the following tradeoff between LSP and uniformity

$$\mathcal{L}_{\text{noHub}} = \alpha \mathcal{L}_{\text{LSP}} + (1 - \alpha) \mathcal{L}_{\text{Unif}} \tag{10}$$

where α is a weight parameter quantifying the tradeoff. $\mathcal{L}_{\text{noHub}}$ can then be optimized directly with gradient descent. The entire procedure is outlined in Algorithm 1.

```
Input: Features $\in \mathbb{R}^k$, $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\}$; perplexity, $P$;
       number of iterations, $T$ ; learning rate, $\eta$.
Output: Embeddings $\in \mathbb{S}_d$, $\{\boldsymbol{z}_1, \dots, \boldsymbol{z}_n\}$
Compute $p_{ij}$ from Eq (4)
Initialize solution $\boldsymbol{Z}^0 = \{\boldsymbol{z}_1, \dots, \boldsymbol{z}_n\}$ with PCA
for $i \leftarrow 1$ to $T$ do
    Compute $q_{ij}$ from Eq. (5)
    Compute gradients $\frac{\mathrm{d}\mathcal{L}_{\text{noHub}}}{\mathrm{d}\boldsymbol{Z}}$, using loss from Eq. (10)
    Update $\boldsymbol{Z}^t$ using the ADAM optimizer with learning
      rate $\eta$ [15]
    Re-normalize elements of $\boldsymbol{Z}^t$ using $L_2$ normalization
end
return $\boldsymbol{Z}^T$
```

Algorithm 1: noHub algorithm for embeddings on the hypersphere

4.2. noHub-S: noHub with Support labels

In order to strengthen the class structure in the embedding space, we modify $\mathcal{L}_{\text{LSP}}$ and $\mathcal{L}_{\text{Unif}}$ by exploiting the additional information provided by the support labels. For $\mathcal{L}_{\text{LSP}}$, we change the similarity function in p_{ij} such that

$$p_{i|j} = \frac{\exp(\kappa_i s_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{x}_j))}{\sum\limits_{l,m} \exp(\kappa_i s_{\boldsymbol{x}}(\boldsymbol{x}_l, \boldsymbol{x}_m))} \tag{11}$$

where

$$s_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \begin{cases} 1 & \text{if } \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{S}, \text{ and } y_i = y_j \\ -1 & \text{if } \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{S}, \text{ and } y_i \neq y_j \\ \boldsymbol{x}_i^\top \boldsymbol{x}_j & \text{otherwise} \end{cases}. \tag{12}$$

With this, we encourage embeddings for support samples in the *same class* to be maximally similar, and support samples in *different classes* to be maximally dissimilar. Similarly, for $\mathcal{L}_{\text{Unif}}$

$$\mathcal{L}_{\text{Unif}} = \log \sum_{l,m} \exp(\kappa s_{\boldsymbol{z}}(\boldsymbol{z}_i, \boldsymbol{z}_j)) \tag{13}$$

where

$$s_{\boldsymbol{z}}(\boldsymbol{z}_i, \boldsymbol{z}_j) = \begin{cases} -\infty, & \text{if } \boldsymbol{z}_i, \boldsymbol{z}_j \in \mathcal{S}, \text{ and } y_i = y_j \\ \varepsilon \, \boldsymbol{z}_i^\top \boldsymbol{z}_j, & \text{if } \boldsymbol{z}_i, \boldsymbol{z}_j \in \mathcal{S}, \text{ and } y_i \neq y_j \\ \boldsymbol{z}_i^\top, \boldsymbol{z}_j & \text{otherwise} \end{cases} \tag{14}$$

where ε is a hyperparameter. This puts more emphasis on between-class uniformity by weighting the similarity higher

[4] Details on the computation of the κ_i are provided in the supplementary.

[5] Intermediate steps are provided in the supplementary.

for embeddings belonging to different classes ($\varepsilon > 1$), and ignoring the similarity between embeddings belonging to the same class[6]. The final loss function is the same as Eq. (10), but with the additional label-informed similarities in Eqs. (11)–(14).

5. Theoretical Results

In this section we provide a theoretical analysis of $\mathcal{L}_{\text{LSP}}$ and $\mathcal{L}_{\text{Unif}}$. Based on our analysis, we interpret these losses with regards to the Laplacian Eigenmaps algorithm and Rényi entropy, respectively.

Proposition 3. *Let $W_{ij} = \frac{1}{2}\kappa p_{ij}$, where $\sum\limits_{i,j} p_{ij} = 1$, and let $\boldsymbol{z}_1, \dots, \boldsymbol{z}_n \in \mathbb{S}_d$. Then we have*

$$\mathcal{L}_{\text{LSP}} = \sum_{i,j} \|\boldsymbol{z}_i - \boldsymbol{z}_j\|^2 W_{ij} - \kappa. \tag{15}$$

Proposition 4 (Minimizing $\mathcal{L}_{\text{Unif}}$ maximizes entropy)**.** *Let $H_2(\cdot)$ be the 2-order Rényi entropy, estimated with a kernel density estimator using a Gaussian kernel. Then*

$$\underset{\boldsymbol{z}_1,\dots,\boldsymbol{z}_n \in \mathbb{S}_d}{\arg\min} \; \mathcal{L}_{\text{Unif}} = \underset{\boldsymbol{z}_1,\dots,\boldsymbol{z}_n \in \mathbb{S}_d}{\arg\max} \; H_2(\boldsymbol{z}_1, \dots, \boldsymbol{z}_n). \tag{16}$$

Definition 2 (Normalized counting measure)**.** *The normalized counting measure associated with a set B on A is*

$$\nu_B(A) = \frac{|B \cap A|}{|B|} \tag{17}$$

Definition 3 (Normalized surface area measure on $\mathbb{S}_d$)**.** *The normalized surface area measure on the hypersphere $\mathbb{S}_d \subset \mathbb{R}^d$, of a subset $S' \subset \mathbb{S}_d$ is*

$$\sigma_d(S') = \frac{\int_{S'} \mathrm{d}S}{\int_{\mathbb{S}_d} \mathrm{d}S} = A_d^{-1} \int_{S'} \mathrm{d}S \tag{18}$$

where A_d is defined as in Eq. (1), and $\int \mathrm{d}S$ denotes the surface integral on $\mathbb{S}_d$.

Definition 4 (Weak* convergence of measures [32])**.** *A sequence of Borel measures $\{\mu_n\}_{n=1}^{\infty}$ in $\mathbb{R}^d$ converges weak* to a Borel measure μ, if for all continuous functions $f : \mathbb{R}^d \to \mathbb{R}$,*

$$\lim_{n \to \infty} \int f(x) \mathrm{d}\mu_n(x) = \int f(x) \mathrm{d}\mu(x) \tag{19}$$

Proposition 5 (Minimizer of $\mathcal{L}_{\text{Unif}}$)**.** *For each $n > 0$, the n point minimizer of $\mathcal{L}_{\text{Unif}}$ is*

$$\boldsymbol{z}_1^\star, \dots, \boldsymbol{z}_n^\star = \underset{\boldsymbol{z}_1,\dots,\boldsymbol{z}_n \in \mathbb{S}_d}{\arg\min} \; \mathcal{L}_{\text{Unif}}. \tag{20}$$

Then $\nu_{\{\boldsymbol{z}_1^\star,\dots,\boldsymbol{z}_n^\star\}}$ converge weak to σ_d as $n \to \infty$.*

Interpretation of Proposition 3–5. Proposition 3 states an alternative formulation of $\mathcal{L}_{\text{LSP}}$, under the hyperspherical assumption. We recognize this formulation as the loss function in Laplacian Eigenmaps [2], which is known to produce *local similarity-preserving* embeddings from graph data. When unconstrained, this loss has a trivial solution where the embeddings for all representations are equal. This is avoided in our case since $\mathcal{L}_{\text{noHub}}$ (Eq. (10)) can be interpreted as the Lagrangian of minimizing $\mathcal{L}_{\text{LSP}}$ subject to a specified level of *entropy*, by Proposition 4.

Finally, Proposition 5 states that the normalized counting measure associated with the set of points that minimize $\mathcal{L}_{\text{Unif}}$, converges to the normalized surface area measure on the sphere. Since $u_{\mathbb{S}_d}$ is the density function associated with this measure, the points that minimize $\mathcal{L}_{\text{Unif}}$ will tend to be uniform on the sphere. Consequently, minimizing $\mathcal{L}_{\text{LSP}}$ also minimizes hubness, by Propositions 1 and 2.

6. Experiments

6.1. Setup

Implementation details. Our implementation is in PyTorch [20]. We optimize noHub and noHub-S for $T = 150$ iterations, using the Adam optimizer [15] with learning rate $\eta = 0.1$. The other hyperparameters were chosen based on validation performance on the respective datasets[7]. We analyze the effect of α in Sec. 6.2. Analyses of the κ and ε hyperparameters are provided in the supplementary.

Initialization. Since noHub and noHub-S reduce the embedding dimensionality ($d = 400$), we initialize embeddings with Principal Component Analysis (PCA) [13], instead of a naïve, random initialization. The PCA initialization is computationally efficient, and approximately preserves global structure. It also resulted in faster convergence and better performance, compared to random initialization.

Base feature extractors. We use the standard networks ResNet-18 [11] and Wide-Res28-10 [37] as the base feature extractors with pretrained weights from [28] and [18], respectively.

Datasets. Following common practice, we evaluate FSL performance on the *mini-ImageNet (mini)* [29], *tiered-ImageNet (tiered)* [23], and *CUB-200 (CUB)* [34] datasets.

Classifiers. We evaluate the baseline embeddings and our proposed methods using both established and recent FSL classifiers: *SimpleShot* [33], *LaplacianShot* [40], *$\alpha-$TIM* [28], *Oblique Manifold (OM)* [21], *iLPC* [16], and *SIAMESE* [39].

Baseline Embeddings. We compare our proposed method with a wide range of techniques for embedding the base features: *None* (No embedding of base features), *L2* [33], *Centered L2* [33], *ZN* [7], *ReRep* [5], *EASE* [39], and *TCPR* [35].

[6] Although any constant value would achieve the same result, we set the similarity to $-\infty$ in this case to remove the contribution to the final loss.

[7] Hyperparameter configurations for all experiments are included in the supplementary.

		mini		tiered		CUB	
Embedding	Feature Extractor	1-shot↑	5-shot↑	1-shot↑	5-shot↑	1-shot↑	5-shot↑
None	ResNet-18	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*
L2 (ARXIV'19 [33])	ResNet-18	73.77 (0.24)	83.14 (0.14)	80.46 (0.26)	87.04 (0.16)	83.1 (0.23)	89.48 (0.12)
CL2 (ARXIV'19 [33])	ResNet-18	75.56 (0.26)	84.04 (0.15)	82.1 (0.26)	87.9 (0.16)	84.35 (0.24)	90.14 (0.12)
ZN (ICCV'21 [7])	ResNet-18	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*
ReRep (ICML'21 [5])	ResNet-18	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*
EASE (CVPR'22 [39])	ResNet-18	76.05 (0.27)	84.61 (0.15)	82.57 (0.27)	88.33 (0.16)	85.24 (0.24)	90.42 (0.12)
TCPR (NEURIPS'22 [35])	ResNet-18	75.99 (0.26)	84.39 (0.15)	82.65 (0.26)	88.26 (0.16)	85.34 (0.23)	90.5 (0.11)
noHub (OURS)	ResNet-18	76.65 (0.28)	84.05 (0.16)	82.94 (0.27)	87.87 (0.17)	**85.88 (0.24)**	90.34 (0.12)
noHub-S (OURS)	ResNet-18	**76.68 (0.28)**	**84.67 (0.15)**	**83.09 (0.27)**	**88.43 (0.16)**	85.81 (0.24)	**90.52 (0.12)**
None	WideRes28-10	45.69 (0.31)	58.82 (0.31)	75.29 (0.28)	82.56 (0.22)	61.36 (0.55)	82.22 (0.37)
L2 (ARXIV'19 [33])	WideRes28-10	80.2 (0.23)	87.11 (0.13)	80.89 (0.26)	87.34 (0.15)	91.98 (0.18)	94.15 (0.1)
CL2 (ARXIV'19 [33])	WideRes28-10	75.23 (0.27)	83.99 (0.16)	79.59 (0.27)	86.71 (0.16)	92.17 (0.18)	94.48 (0.09)
ZN (ICCV'21 [7])	WideRes28-10	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*	20.0 (0.0)*
ReRep (ICML'21 [5])	WideRes28-10	36.69 (0.28)	36.41 (0.3)	67.41 (0.29)	76.49 (0.24)	57.62 (0.56)	60.36 (0.6)
EASE (CVPR'22 [39])	WideRes28-10	81.19 (0.25)	87.82 (0.13)	82.04 (0.26)	88.06 (0.16)	91.99 (0.19)	94.36 (0.09)
TCPR (NEURIPS'22 [35])	WideRes28-10	81.27 (0.24)	87.8 (0.13)	81.89 (0.26)	87.95 (0.16)	91.91 (0.18)	94.25 (0.1)
noHub (OURS)	WideRes28-10	81.97 (0.25)	87.78 (0.14)	82.8 (0.27)	87.99 (0.17)	92.53 (0.18)	94.56 (0.09)
noHub-S (OURS)	WideRes28-10	**82.0 (0.26)**	**88.03 (0.13)**	**82.85 (0.27)**	**88.31 (0.16)**	**92.63 (0.18)**	**94.69 (0.09)**

Table 1. Accuracies (Confidence interval) with the SIAMESE [39] classifier for different embedding approaches. Best and second best performance are denoted in **bold** and underlined, respectively. *The SIAMESE classifier is sensitive to the norm of the embedding, thus leading to detrimental performance for some of the embedding approaches.

Evaluation protocol. We follow the standard evaluation protocol in FSL and calculate the accuracy for 1-shot and 5-shot classification with 15 images per class in the query set. We evaluate on 10000 episodes, as is standard practice in FSL. Additionally, we evaluate the hubness of the representations after embedding using two common hubness metrics, namely the skewness (Sk) of the k-occurrence distribution [22] and the hub occurrence (HO) [8], which measures the percentage of hubs in the nearest neighbour lists of all points.

6.2. Results

Comparison to the state-of-the-art. To illustrate the effectiveness of noHub and noHub-S as an embedding approach for FSL, we consider the current state-of-the-art FSL method, which leverages the EASE embedding and obtains query predictions with SIAMESE [39]. We replace EASE with our proposed embedding approaches noHub and noHub-S, as well as other baseline embeddings, and evaluate performance on all datasets in the 1 and 5-shot setting. As shown in Table 1, noHub and noHub-S outperform all baseline approaches in both settings across all datasets, illustrating noHub's and noHub-S' ability to provide useful FSL embeddings, and updating the state-of-the-art in transductive FSL.

Aggregated FSL performance. To further evaluate the general applicability of noHub and noHub-S as embedding approaches, we perform extensive experiments for all classifiers and all baseline embeddings on all datasets. Tables 2a and 2b provide the results averaged over classifiers[8]. To clearly present the results, we aggregate the accuracy and a ranking *score* for each embedding method across all classifiers. The ranking score is calculated by performing a paired Student's t-test between all pairwise embedding methods for each classifier. We then average the ranking scores across all classifiers. A high ranking score then indicates that a method often significantly outperforms the competing embedding methods. We set the significance level to 5%. noHub and noHub-S consistently outperform previous embedding approaches – sometimes by a large margin. Overall, we further observe that noHub-S outperforms noHub in most settings and is particular beneficial in the 1-shot setting, which is more challenging, given that fewer samples are likely to generate noisy embeddings.

Hubness metrics. To further validate noHub's and noHub-S' ability to reduce hubness, we follow the same procedure of aggregating results for the hubness metrics and average over classifiers. Compared to the current state-of-the-art embedding approaches, Table 3 illustrates that noHub and noHub-S consistently result in embeddings with lower hubness.

Visualization of similarity matrices. As discussed in Sec. 4, completely eliminating hubness by distributing points uniformly on the hypersphere is not sufficient to obtain good FSL performance. Instead, representations need to also capture the inherent class structure of the data. To further evaluate the embedding approaches, we therefore compute the pairwise inner products for the embeddings of a random 5-shot episode on tiered-ImageNet with ResNet-18 features in Figure 3. It can be observed that the block structure is considerably more distinct for noHub and noHub-S, with

[8] The detailed results for all classifiers are provided in the supplementary.

		mini		tiered		CUB	
	Embedding	Acc↑	Score↑	Acc↑	Score↑	Acc↑	Score↑
ResNet18	None	55.74	0.17	62.61	0.0	63.78	0.17
	L2 (arXiv'19 [33])	68.22	2.33	75.94	2.17	78.09	2.33
	CL2 (arXiv'19 [33])	69.56	2.83	76.97	3.0	78.26	2.83
	ZN (ICCV'21 [7])	60.0	2.33	66.21	2.5	67.43	2.67
	ReRep (ICML'21 [5])	60.76	4.0	67.07	3.67	69.6	4.17
	EASE (CVPR'22 [39])	69.63	3.67	77.05	4.0	78.84	3.67
	TCPR (NeurIPS'22 [35])	69.97	4.0	77.18	3.33	78.83	4.0
	noHub (ours)	72.58	6.83	79.77	6.83	81.91	6.83
	noHub-S (ours)	**73.64**	**7.67**	**80.6**	**7.67**	**83.1**	**7.67**
WideRes28-10	None	63.59	1.0	71.29	0.83	79.23	1.17
	L2 (arXiv'19 [33])	74.3	3.0	76.19	2.67	88.61	3.5
	CL2 (arXiv'19 [33])	71.32	1.33	75.17	2.0	88.52	3.33
	ZN (ICCV'21 [7])	64.27	2.5	65.64	2.5	76.0	1.5
	ReRep (ICML'21 [5])	65.51	3.0	71.83	3.17	83.1	3.5
	EASE (CVPR'22 [39])	74.95	4.33	76.59	3.67	88.51	3.5
	TCPR (NeurIPS'22 [35])	75.64	4.83	76.51	4.0	88.22	2.5
	noHub (ours)	78.22	7.0	79.76	7.0	90.25	5.67
	noHub-S (ours)	**79.24**	**7.67**	**80.46**	**7.67**	**90.82**	**7.67**

(a) 1-shot

		mini		tiered		CUB	
	Embedding	Acc↑	Score↑	Acc↑	Score↑	Acc↑	Score↑
ResNet18	None	69.83	0.83	74.38	0.67	76.01	1.17
	L2 (arXiv'19 [33])	81.58	2.33	86.05	1.83	88.43	2.83
	CL2 (arXiv'19 [33])	81.95	2.67	86.43	3.0	88.49	2.5
	ZN (ICCV'21 [7])	71.49	4.0	75.32	3.83	76.92	3.5
	ReRep (ICML'21 [5])	70.25	2.5	74.52	1.83	76.43	2.5
	EASE (CVPR'22 [39])	81.84	3.5	86.4	3.17	88.57	3.5
	TCPR (NeurIPS'22 [35])	82.1	4.0	86.54	3.83	88.79	4.33
	noHub (ours)	82.58	5.5	86.9	4.5	**89.13**	**6.0**
	noHub-S (ours)	**82.61**	**6.5**	**87.13**	**6.67**	88.93	5.33
WideRes28-10	None	78.77	1.5	84.1	1.67	89.49	1.67
	L2 (arXiv'19 [33])	85.65	4.0	86.29	3.83	93.47	3.67
	CL2 (arXiv'19 [33])	83.14	1.33	85.47	1.5	93.49	4.0
	ZN (ICCV'21 [7])	74.61	4.33	75.34	5.0	81.02	3.17
	ReRep (ICML'21 [5])	73.86	1.83	81.51	1.67	87.2	2.0
	EASE (CVPR'22 [39])	85.51	3.5	86.29	3.33	93.34	3.5
	TCPR (NeurIPS'22 [35])	86.03	**6.0**	86.37	4.0	93.3	3.0
	noHub (ours)	**86.44**	5.67	**87.07**	5.5	93.65	4.17
	noHub-S (ours)	85.95	5.5	87.05	**5.83**	**93.76**	**5.0**

(b) 5-shot

Table 2. Aggregated FSL performance for all embedding approaches on the mini-ImageNet, tiered-ImageNet, and CUB-200 datasets. Results are averaged over FSL classifiers. Best and second best performance are denoted in **bold** and underlined, respectively.

		mini		tiered		CUB	
		Sk↓	HO↓	Sk↓	HO↓	Sk↓	HO↓
ResNet18	None	1.349	0.407	1.211	0.408	0.887	0.341
	L2 (arXiv'19 [33])	0.937	0.301	0.812	0.265	0.691	0.236
	CL2 (arXiv'19 [33])	0.667	0.233	0.679	0.249	0.549	0.201
	ZN (ICCV'21 [7])	0.68	0.231	0.698	0.264	0.564	0.216
	ReRep (ICML'21 [5])	3.655	0.548	3.604	0.549	3.565	0.513
	EASE (CVPR'22 [39])	0.521	0.16	0.479	0.158	0.466	0.153
	TCPR (NeurIPS'22 [35])	0.651	0.228	0.65	0.25	0.532	0.204
	noHub (ours)	0.315	**0.095**	0.303	**0.102**	0.32	**0.112**
	noHub-S (ours)	**0.276**	0.13	**0.283**	0.127	**0.296**	0.162
WideRes28-10	None	1.6	0.459	1.81	0.494	1.073	0.369
	L2 (arXiv'19 [33])	0.781	0.296	0.737	0.275	0.475	0.228
	CL2 (arXiv'19 [33])	0.981	0.288	0.817	0.307	0.52	0.267
	ZN (ICCV'21 [7])	0.73	0.287	0.769	0.302	0.517	0.263
	ReRep (ICML'21 [5])	3.56	0.704	3.55	0.777	3.026	0.47
	EASE (CVPR'22 [39])	0.47	0.177	0.477	0.175	0.437	0.213
	TCPR (NeurIPS'22 [35])	0.589	0.236	0.685	0.264	0.477	0.231
	noHub (ours)	0.29	**0.111**	0.301	**0.111**	0.188	**0.108**
	noHub-S (ours)	**0.258**	0.148	**0.274**	0.135	**0.162**	0.13

(a) 1-shot

		mini		tiered		CUB	
		Sk↓	HO↓	Sk↓	HO↓	Sk↓	HO↓
ResNet18	None	1.436	0.422	1.339	0.432	0.987	0.364
	L2 (arXiv'19 [33])	1.04	0.318	0.914	0.287	0.812	0.263
	CL2 (arXiv'19 [33])	0.786	0.264	0.821	0.28	0.698	0.236
	ZN (ICCV'21 [7])	0.806	0.264	0.839	0.296	0.716	0.25
	ReRep (ICML'21 [5])	1.631	0.863	1.721	0.872	1.432	0.869
	EASE (CVPR'22 [39])	0.624	0.186	0.598	0.183	0.607	0.186
	TCPR (NeurIPS'22 [35])	0.78	0.259	0.796	0.283	0.687	0.235
	noHub (ours)	0.286	0.096	0.289	0.104	**0.329**	0.12
	noHub-S (ours)	**0.25**	**0.074**	**0.213**	**0.078**	0.433	**0.097**
WideRes28-10	None	1.709	0.473	1.937	0.51	1.16	0.395
	L2 (arXiv'19 [33])	0.887	0.322	0.86	0.305	0.632	0.266
	CL2 (arXiv'19 [33])	1.12	0.318	0.956	0.337	0.701	0.31
	ZN (ICCV'21 [7])	0.858	0.32	0.912	0.335	0.699	0.305
	ReRep (ICML'21 [5])	1.597	0.819	1.617	0.846	1.299	0.549
	EASE (CVPR'22 [39])	0.579	0.199	0.585	0.193	0.572	0.241
	TCPR (NeurIPS'22 [35])	0.717	0.27	0.815	0.294	0.634	0.264
	noHub (ours)	**0.294**	0.115	**0.298**	**0.115**	**0.195**	**0.1**
	noHub-S (ours)	0.494	**0.103**	0.407	0.12	0.421	0.127

(b) 5-shot

Table 3. Aggregated hubness metrics for all embedding approaches on the Mini-ImageNet, Tiered-ImageNet and CUB-200 dataset. Results are averaged over FSL classifiers. Best and second best performance are denoted in **bold** and underlined, respectively.

noHub-S slightly improving upon noHub. These results indicate that (i) samples are more uniform, indicating the reduced hubness; and (ii) classes are better separated, due to the local similarity preservation.

Tradeoff between uniformity and similarity preservation. We analyze the effect of α on the tradeoff between LSP and Uniformity in the loss function in Eq. (10), on tiered-ImageNet with ResNet-18 features in the 5-shot setting and with the SIAMESE [39] classifier. The results are visualized in Figure 4. We notice a sharp increase in performance when we have a high emphasis on uniformity. This demonstrates the impact of hubness on accuracy in FSL performance. As we keep increasing the emphasis on LSP, however, after a certain point we notice a sharp drop off in performance. This is due to the fact that the classifier does not take into account the uniformity constraint on the features, resulting in a large number of misclassifications. In general, we observe that noHub-S is slightly more robust compared to noHub.

Increasing number of classes. We analyze the behavior of noHub and noHub-S for an increasing number of classes (ways) on the tiered-ImageNet dataset with SIAMESE [39]

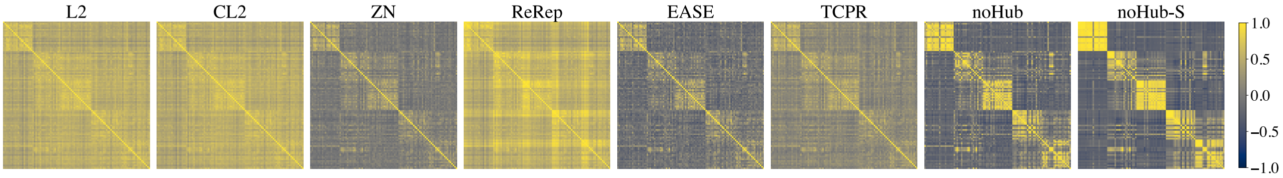


Figure 3. Inner product matrices between features for a random episode for all embedding approaches.

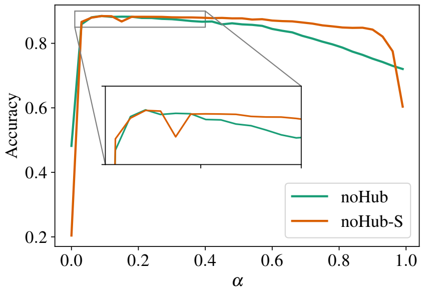


Figure 4. Accuracies for different values of the weighting parameter, α, which quantifies the tradeoff between $\mathcal{L}_{\text{LSP}}$ and $\mathcal{L}_{\text{Unif}}$.

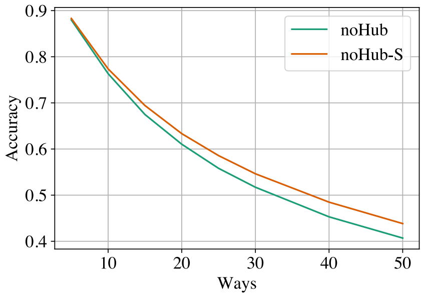


Figure 5. Accuracies for an increasing number of classes (ways) for noHub and noHub-S.

as classifier. While classification accuracy generally decreases with an increasing number of classes, which is expected, we observe from Figure 5 that noHub-S has a slower decay and is able to leverage the label guidance to obtain better performance for a larger number of classes.

Effect of label information in $\mathcal{L}_{\text{LSP}}$ and $\mathcal{L}_{\text{Unif}}$. To validate the effectiveness of using label guidance in noHub-S, we study the result of including label information in $\mathcal{L}_{\text{LSP}}$ and $\mathcal{L}_{\text{Unif}}$ (Eqs. (11)–(14)). We note that the default setting of noHub is that none of the two losses include label information. Ablation experiments are performed on tiered-ImageNet with the ResNet-18 feature extractor and the SimpleShot and SIAMESE classifier [39]. In Table 4, we generally see improvements of noHub-S when *both* the loss terms are label-informed, indicating the usefulness of label guidance.

We further observe that incorporating label information in $\mathcal{L}_{\text{Unif}}$ tends to have a larger contribution than doing the same for $\mathcal{L}_{\text{LSP}}$. This aligns with our observations in Figure 4, where a small α yielded the best performance.

	Label-informed		SimpleShot [33]		SIAMESE [39]	
	$\mathcal{L}_{\text{LSP}}$	$\mathcal{L}_{\text{Unif}}$	1-shot↑	5-shot↑	1-shot↑	5-shot↑
noHub	–	–	76.72 (0.23)	**86.31 (0.16)**	82.94 (0.27)	87.87 (0.17)
noHub-S	✓	–	78.25 (0.24)	85.46 (0.16)	82.56 (0.28)	88.07 (0.17)
noHub-S	–	✓	78.33 (0.23)	86.15 (0.15)	82.81 (0.27)	**88.43 (0.16)**
noHub-S	✓	✓	**78.35 (0.23)**	86.22 (0.15)	**83.09 (0.27)**	**88.43 (0.16)**

Table 4. Ablation study with the label-informed losses in noHub-S. Check marks (✓) indicate that the loss uses information from the support labels.

7. Conclusion

In this paper we have addressed the hubness problem in FSL. We have shown that hubness is eliminated by embedding representations uniformly on the hypersphere. The hyperspherical uniform distribution has zero mean and zero density gradient at all points along all directions tangent to the hypersphere – both of which are identified as causes of hubness in previous work [9, 22]. Based on our theoretical findings about hubness and hyperspheres, we proposed two new methods to embed representations on the hypersphere for FSL. The proposed noHub and noHub-S leverage a decomposition of the KL divergence between similarity distributions, and optimize a tradeoff between LSP and uniformity on the hypersphere – thus reducing hubness while maintaining the class structure in the representation space. We have provided theoretical analyses and interpretations of the LSP and uniformity losses, proving that they optimize LSP and uniformity, respectively. We comprehensively evaluate the proposed methods on several datasets, features extractors, and classifiers, and compare to a number of recent state-of-the-art baselines. Our results illustrate the effectiveness of our proposed methods and show that we achieve state-of-the-art performance in transductive FSL.

Acknowledgements

This work was financially supported by the Research Council of Norway (RCN), through its Centre for Research-based Innovation funding scheme (Visual Intelligence, grant no. 309439), and Consortium Partners. It was further funded by RCN FRIPRO grant no. 315029, RCN IKTPLUSS grant no. 303514, and the UiT Thematic Initiative “Data-Driven Health Technology”.

References

[1] Kelsey R. Allen, Evan Shelhamer, Hanul Shin, and Joshua B. Tenenbaum. Infinite mixture prototypes for few-shot learning. In *ICML*, 2019. 2

[2] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 2003. 5

[3] Malik Boudiaf, Ziko Imtiaz Masud, Jérôme Rony, José Dolz, Pablo Piantanida, and Ismail Ben Ayed. Transductive Information Maximization For Few-Shot Learning. In *NeurIPS*, 2020. 1, 2

[4] Philip Chikontwe, Soopil Kim, and Sang Hyun Park. CAD: Co-Adapting Discriminative Features for Improved Few-Shot Classification. In *CVPR*, 2022. 1

[5] Wentao Cui and Yuhong Guo. Parameterless Transductive Feature Re-representation for Few-Shot Learning. In *ICML*, 2021. 1, 2, 5, 6, 7

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 1

[7] Nanyi Fei, Yizhao Gao, Zhiwu Lu, and Tao Xiang. Z-Score Normalization, Hubness, and Few-Shot Learning. In *ICCV*, 2021. 1, 2, 5, 6, 7

[8] Arthur Flexer and Dominik Schnitzer. Choosing ℓ^p norms in high-dimensional spaces based on hub analysis. *Neurocomputing*, 2015. 6

[9] Kazuo Hara, Ikumi Suzuki, Kei Kobayashi, Kenji Fukumizu, and Milos Radovanovic. Flattening the Density Gradient for Eliminating Spatial Centrality to Reduce Hubness. In *AAAI*, 2016. 2, 8

[10] Kazuo Hara, Ikumi Suzuki, Masashi Shimbo, Kei Kobayashi, Kenji Fukumizu, and Milos Radovanovic. Localized Centering: Reducing Hubness in Large-Sample Data. In *AAAI*, 2015. 1

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 5

[12] Shell Xu Hu, Da Li, Jan Stuhmer, Minyoung Kim, and Timothy M Hospedales. Pushing the Limits of Simple Pipelines for Few-Shot Learning: External Data and Fine-Tuning Make a Difference. In *CVPR*, 2022. 1

[13] Ian T Jolliffe. *Principal component analysis for special types of data*. Springer, 2002. 5

[14] Jongmin Kim, Taesup Kim, Sungwoong Kim, and Chang D Yoo. Edge-labeling graph neural network for few-shot learning. In *CVPR*, 2019. 1

[15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 4, 5

[16] Michalis Lazarou, Tania Stathaki, and Yannis Avrithis. Iterative Label Cleaning for Transductive and Semi-Supervised Few-Shot Learning. In *ICCV*, 2021. 1, 5

[17] Duong H Le, Khoi D Nguyen, and Khoi Nguyen. POODLE: Improving Few-shot Learning via Penalizing Out-of-Distribution Samples. In *NeurIPS*, 2021. 1

[18] Puneet Mangla, Mayank Singh, Abhishek Sinha, Nupur Kumari, Vineeth N Balasubramanian, and Balaji Krishnamurthy. Charting the Right Manifold: Manifold Mixup for Few-shot Learning. In *WACV*, 2020. 5

[19] Van Nhan Nguyen, Sigurd Løkse, Kristoffer Wickstrøm, Michael Kampffmeyer, Davide Roverso, and Robert Jenssen. Sen: A novel feature normalization dissimilarity measure for prototypical few-shot learning networks. In *ECCV*, 2020. 2

[20] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 5

[21] Guodong Qi, Huimin Yu, Zhaohui Lu, and Shuzhao Li. Transductive Few-Shot Classification on the Oblique Manifold. In *ICCV*, 2021. 1, 5

[22] Miloš Radovanovic, Alexandros Nanopoulos, and Mirjana Ivanovic. Hubs in Space: Popular Nearest Neighbors in High-Dimensional Data. *JMLR*, 2010. 1, 2, 6, 8

[23] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. Meta-learning for Semi-supervised Few-shot Classification. In *ICLR*, 2018. 5

[24] Yutaro Shigeto, Ikumi Suzuki, Kazuo Hara, Masashi Shimbo, and Yuji Matsumoto. Ridge Regression, Hubness, and Zero-Shot Learning. In *ECML-PKDD*, 2015. 1

[25] Ikumi Suzuki, Kazuo Hara, Masashi Shimbo, Marco Saerens, and Kenji Fukumizu. Centering Similarity Measures to Reduce Hubs. In *EMNLP*, 2013. 1

[26] Ran Tao, Han Zhang, Yutong Zheng, and Marios Savvides. Powering Finetuning in Few-Shot Learning: Domain-Agnostic Bias Reduction with Selected Sampling. In *AAAI*, 2022. 1

[27] Laurens van der Maaten and Geoffrey E. Hinton. Visualizing data using t-sne. *JMLR*, 2008. 4

[28] Olivier Veilleux, Malik Boudiaf, Pablo Piantanida, and Ismail Ben Ayed. Realistic evaluation of transductive few-shot learning. In *NeurIPS*, 2021. 1, 5

[29] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching Networks for One Shot Learning. In *NeurIPS*, 2016. 1,

5

[30] Mian Wang and Dong Wang. VMF-SNE: embedding for spherical data. *arxiv:1507.08379 [cs]*, 2015. 4

[31] Ruohan Wang, Massimiliano Pontil, and Carlo Ciliberto. The Role of Global Labels in Few-Shot Classification and How to Infer Them. In *NeurIPS*, 2021. 1

[32] Tongzhou Wang and Phillip Isola. Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere. In *ICML*, 2020. 2, 5

[33] Yan Wang, Wei-Lun Chao, Kilian Q. Weinberger, and Laurens van der Maaten. SimpleShot: Revisiting Nearest-Neighbor Classification for Few-Shot Learning. *arXiv:1911.04623 [cs]*, 2019. 1, 2, 5, 6, 7, 8

[34] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-ucsd birds 200. Technical Report CNS-TR-201, Caltech, 2010. 5

[35] Jing Xu, Xu Luo, Xinglin Pan, Wenjie Pei, Yanan Li, and Zenglin Xu. Alleviating the Sample Selection Bias in Few-shot Learning by Removing Projection to the Centroid. In *NeurIPS*, 2022. 1, 2, 5, 6, 7

[36] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set functions. In *CVPR*, 2020. 2

[37] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016. 5

[38] Manli Zhang, Jianhong Zhang, Zhiwu Lu, Tao Xiang, Mingyu Ding, and Songfang Huang. IEPT: Instance-level and episode-level pretext tasks for few-shot learning. In *ICLR*, 2021. 2

[39] Hao Zhu and Piotr Koniusz. EASE: Unsupervised Discriminant Subspace Learning for Transductive Few-Shot Learning. In *CVPR*, 2022. 1, 2, 5, 6, 7, 8

[40] Imtiaz Ziko, Jose Dolz, Eric Granger, and Ismail Ben Ayed. Laplacian Regularized Few-Shot Learning. In *ICML*, 2020. 1, 2, 5

Supplementary material – Hubs and Hyperspheres: Reducing Hubness and Improving Transductive Few-shot Learning with Hyperspherical Embeddings

Daniel J. Trosten[*][†], Rwiddhi Chakraborty[*][†], Sigurd Løkse[†], Kristoffer Knutsen Wickstrøm[†],
Robert Jenssen[†][‡][§][¶], Michael C. Kampffmeyer[†][‡]
Department of Physics and Technology, UiT The Arctic University of Norway
firstname[.middle initial].lastname@uit.no

1. Introduction

Here we provide proofs for our theoretical results on the hyperspherical uniform and hubness; the decomposition of the KL divergence; and the minima of our methods' loss functions. We also give additional details on the implementation and hyperparameters for noHub and noHub-S– and include the complete tables of 1-shot and 5-shot results for all classifiers, datasets and feature extractors. Finally, we briefly reflect on potential negative societal impacts of our work.

2. Hyperspherical Uniform Eliminates Hubness

Definition 1 (Uniform PDF on the hypersphere.)**.** *The uniform probability density function (PDF) on the unit hypersphere* $\mathbb{S}_d = \{\boldsymbol{x} \in \mathbb{R}^d \mid ||\boldsymbol{x}|| = 1\} \subset \mathbb{R}^d$ *is*

$$u_{\mathbb{S}_d}(\boldsymbol{x}) = A_d^{-1} \delta(||\boldsymbol{x}|| - 1) \tag{1}$$

where $A_d = \frac{2\pi^{d/2}}{\Gamma(d/2)}$ *is the surface area of* $\mathbb{S}_d$*, and* $\delta(\cdot)$ *is the Dirac delta distribution.*

Lemma 1 (Trisection of hypersphere)**.** *The trisection of the hypersphere along coordinate* i *is given by the three-tuple of disjoint sets* $(\mathbb{S}_d^{i,+}, \mathbb{S}_d^{i,-}, \mathbb{S}_d^{i,0})$ *where*

$$\mathbb{S}_d^{i,+} = \{\boldsymbol{x} = [x^1, \dots, x^d]^\top \in \mathbb{S}_d \mid x^i > 0\} \tag{2}$$

$$\mathbb{S}_d^{i,-} = \{\boldsymbol{x} = [x^1, \dots, x^d]^\top \in \mathbb{S}_d \mid x^i < 0\} \tag{3}$$

$$\mathbb{S}_d^{i,0} = \{\boldsymbol{x} = [x^1, \dots, x^d]^\top \in \mathbb{S}_d \mid x^i = 0\} \tag{4}$$

and

$$\mathbb{S}_d^{i,+} \cup \mathbb{S}_d^{i,-} \cup \mathbb{S}_d^{i,0} = \mathbb{S}_d \tag{5}$$

Then we have

$$\mathbb{S}_d^{i,+} = -\mathbb{S}_d^{i,-} = \{-\boldsymbol{x} \mid \boldsymbol{x} \in \mathbb{S}_d^{i,-}\} \tag{6}$$

Proof. Let $\boldsymbol{x} \in \mathbb{S}_d^{i,+}$, then

$$||(-x)|| = ||x|| = 1, \tag{7}$$

and

$$-(x^i) < 0. \tag{8}$$

Hence $\boldsymbol{x} \in -\mathbb{S}_d^{i,-}$, and $\mathbb{S}_d^{i,+} \subseteq -\mathbb{S}_d^{i,-}$.

Similarly, let $-\boldsymbol{x} \in -\mathbb{S}_d^{i,-}$, then

$$|| - (-x)|| = ||x|| = 1, \tag{9}$$

and

$$-(-x^i) = x^i > 0. \tag{10}$$

Hence $\boldsymbol{x} \in \mathbb{S}_d^{i,+}$, and $-\mathbb{S}_d^{i,-} \subseteq \mathbb{S}_d^{i,+}$.

It then follows that $\mathbb{S}_d^{i,+} = -\mathbb{S}_d^{i,-}$. □

Proposition 1. *Suppose* $\boldsymbol{X}$ *has PDF* $u_{\mathbb{S}_d}(\boldsymbol{x})$*. Then*

$$\mathbb{E}(\boldsymbol{X}) = 0 \tag{11}$$

Proof. The expectation $\mathbb{E}(\boldsymbol{X})$ is given by

$$\mathbb{E}(\boldsymbol{X}) = \int_{\mathbb{R}^d} \boldsymbol{x} u_{\mathbb{S}_d}(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \tag{12}$$

Since $u_{\mathbb{S}_d}$ is non-zero only on the hypersphere $\mathbb{S}_d$, the integral can be rewritten as a surface integral over $\mathbb{S}_d$

$$\mathbb{E}(\boldsymbol{X}) = \int_{\mathbb{S}_d} \boldsymbol{x} A_d^{-1} \mathrm{d}S. \tag{13}$$

[*] Equal contributions.

[†] UiT Machine Learning group (machine-learning.uit.no) and Visual Intelligence Centre (visual-intelligence.no).

[‡] Norwegian Computing Center.

[§] Department of Computer Science, University of Copenhagen.

[¶] Pioneer Centre for AI (aicentre.dk).

Decomposing the integral over the trisection of $\mathbb{S}_d$ along coordinate i gives

$$\int_{\mathbb{S}_d} \boldsymbol{x} A_d^{-1} \mathrm{d}S = \tag{14}$$

$$A_d^{-1}\left(\int_{\mathbb{S}_d^{i,+}} \boldsymbol{x}\mathrm{d}S + \int_{\mathbb{S}_d^{i,-}} \boldsymbol{x}\mathrm{d}S + \int_{\mathbb{S}_d^{i,0}} \boldsymbol{x}\mathrm{d}S\right). \tag{15}$$

By Lemma 1 we have

$$\mathbb{S}_d^{i,+} = -\mathbb{S}_d^{i,-} \Rightarrow \int_{\mathbb{S}_d^{i,+}} \boldsymbol{x}\mathrm{d}S = -\int_{\mathbb{S}_d^{i,-}} \boldsymbol{x}\mathrm{d}S. \tag{16}$$

Furthermore, since the set $\mathbb{S}_d^{i,0}$ has zero width along coordinate i, $\int_{\mathbb{S}_d^{i,0}} \boldsymbol{x}\mathrm{d}S = 0$. Hence

$$\mathbb{E}(\boldsymbol{X}) = \tag{17}$$

$$A_d^{-1}\left(\int_{\mathbb{S}_d^{i,+}} \boldsymbol{x}\mathrm{d}S - \int_{\mathbb{S}_d^{i,+}} \boldsymbol{x}\mathrm{d}S + \int_{\mathbb{S}_d^{i,0}} \boldsymbol{x}\mathrm{d}S\right) = 0 \tag{18}$$

□

Proposition 2. *Let $\Pi_{\boldsymbol{p}}$ be the tangent plane of $\mathbb{S}_d$ at an arbitrary point $\boldsymbol{p} \in \mathbb{S}_d$. Then, for any direction $\boldsymbol{\theta}^*$ in $\Pi_{\boldsymbol{p}}$ the directional derivative of $u_{\mathbb{S}_d}$ along $\boldsymbol{\theta}^*$ is*

$$\nabla_{\boldsymbol{\theta}^*} u_{\mathbb{S}_d} = 0 \tag{19}$$

Proof. $u_{\mathbb{S}_d}(\boldsymbol{x})$ can be written in polar coordinates as

$$u_{\mathbb{S}_d}(\boldsymbol{x}(r,\boldsymbol{\theta})) = u_{\mathbb{S}_d}^{\text{Polar}}(r,\boldsymbol{\theta}) = A_d^{-1}\delta(r-1) \tag{20}$$

The gradient of $u_{\mathbb{S}_d}^{\text{Polar}}(r,\boldsymbol{\theta})$ is then

$$\nabla_{(r,\boldsymbol{\theta})} u_{\mathbb{S}_d}^{\text{Polar}}(r,\boldsymbol{\theta}) = \begin{bmatrix} \frac{\partial}{\partial r} u_{\mathbb{S}_d}^{\text{Polar}}(r,\boldsymbol{\theta}) \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{21}$$

For an arbitrary point $\boldsymbol{p} \in \mathbb{S}_d$, an arbitrary unit vector (direction), $\boldsymbol{\theta}^*$, in the tangent plane $\Pi_{\boldsymbol{p}}$ is given by

$$\boldsymbol{\theta}^* = \begin{bmatrix} 0 \\ \theta_1^* \\ \vdots \\ \theta_{d-1}^* \end{bmatrix} \tag{22}$$

The directional derivative of $u_{\mathbb{S}_d}(\boldsymbol{x})$ along $\boldsymbol{\theta}^*$ is then

$$\nabla_{\boldsymbol{\theta}^*} u_{\mathbb{S}_d}(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial}{\partial r} u_{\mathbb{S}_d}^{\text{Polar}}(r,\boldsymbol{\theta}) \\ 0 \\ \vdots \\ 0 \end{bmatrix}^\top \cdot \begin{bmatrix} 0 \\ \theta_1^* \\ \vdots \\ \theta_{d-1}^* \end{bmatrix} = 0 \tag{23}$$

□

3. Method

Computing κ_i. Following [5], we compute κ_i using a binary search such that

$$|\log_2(P) - H(P_i)| \leq 0.1 \cdot \log_2(P) \tag{24}$$

where P is a hyperparameter, and $H(P_i)$ is the Shannon entropy of similarities for representation i

$$H(P_i) = \sum_{j=1}^{n} p_{i|j} \log_2(p_{i|j}). \tag{25}$$

Decomposition of $KL(P||Q)$. Recall that

$$p_{ij} = \frac{p_{i|j} + p_{j|i}}{2}, \quad p_{i|j} = \frac{\exp(\kappa_i \boldsymbol{x}_i^\top \boldsymbol{x}_j)}{\sum\limits_{l,m} \exp(\kappa_i \boldsymbol{x}_l^\top \boldsymbol{x}_m)} \tag{26}$$

and

$$q_{ij} = \frac{\exp(\kappa \boldsymbol{z}_i^\top \boldsymbol{z}_j)}{\sum\limits_{l,m} \exp(\kappa \boldsymbol{z}_l^\top \boldsymbol{z}_m)}. \tag{27}$$

Since p_{ij} is constant w.r.t. q_{ij}, we have

$$\underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n \in \mathbb{S}_d}{\arg\min}\; KL(P||Q) = \underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n \in \mathbb{S}_d}{\arg\min} \sum_{i,j} p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{28}$$

$$= \underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n \in \mathbb{S}_d}{\arg\min} \underbrace{\sum_{i,j} p_{ij} \log p_{ij}}_{\text{constant}} - \sum_{i,j} p_{ij} \log q_{ij} \tag{29}$$

$$= \underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n \in \mathbb{S}_d}{\arg\min} \underbrace{-\sum_{i,j} p_{ij} \log q_{ij}}_{=:\tilde{\mathcal{L}}} \tag{30}$$

Minimizing $KL(P||Q)$ over $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n \in \mathbb{S}_d$ is therefore equivalent to minimizing $\tilde{\mathcal{L}}$.

Decomposing $\tilde{\mathcal{L}}$ gives

$$\tilde{\mathcal{L}} = -\sum_{i,j} p_{ij} \kappa \boldsymbol{z}_i^\top \boldsymbol{z}_j + \tag{31}$$

$$\sum_{i,j}\left(p_{ij} \log \sum_{l,m} \exp(\kappa \boldsymbol{z}_l^\top \boldsymbol{z}_m)\right) \tag{32}$$

$$= -\sum_{i,j} p_{ij} \kappa \boldsymbol{z}_i^\top \boldsymbol{z}_j \tag{33}$$

$$+ \left(\log \sum_{l,m} \exp(\kappa \boldsymbol{z}_l^\top \boldsymbol{z}_m)\right) \cdot \underbrace{\left(\sum_{i,j} p_{ij}\right)}_{=1} \tag{34}$$

$$= \underbrace{-\sum_{i,j} p_{ij} \kappa \boldsymbol{z}_i^\top \boldsymbol{z}_j}_{=:\, \mathcal{L}_{\text{LSP}}} + \underbrace{\log \sum_{l,m} \exp(\kappa \boldsymbol{z}_l^\top \boldsymbol{z}_m)}_{=:\, \mathcal{L}_{\text{Unif}}} \tag{35}$$

Thus, we have shown that

$$\underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n\in\mathbb{S}_d}{\arg\min}\ KL(P||Q) = \underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n\in\mathbb{S}_d}{\arg\min}\ \mathcal{L}_{\text{LSP}} + \mathcal{L}_{\text{Unif}}. \tag{36}$$

4. Theoretical Results

Proposition 3. *Let $W_{ij} = \frac{1}{2}\kappa p_{ij}$, where $\sum_{i,j} p_{ij} = 1$, and let $\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n \in \mathbb{S}_d$. Then we have*

$$\mathcal{L}_{\text{LSP}} = \sum_{i,j} \|\boldsymbol{z}_i - \boldsymbol{z}_j\|^2 W_{ij} - \kappa. \tag{37}$$

Proof. We have

$$\mathcal{L}_{\text{LSP}} = -\kappa \sum_{i,j} p_{ij}\boldsymbol{z}_i^\top \boldsymbol{z}_j \tag{38}$$

$$= -2\sum_{i,j}\frac{1}{2}\kappa p_{ij}\boldsymbol{z}_i^\top \boldsymbol{z}_j + \sum_{i,j} 2\frac{1}{2}\kappa p_{ij} - \kappa \tag{39}$$

$$(\sum_{i,j} p_{ij} = 1)$$

$$= -2\sum_{i,j}\boldsymbol{z}_i^\top \boldsymbol{z}_j W_{ij} + \sum_{i,j}(\|\boldsymbol{z}_i\| + \|\boldsymbol{z}_j\|)W_{ij} - \kappa \tag{40}$$

$$(\|\boldsymbol{z}_i\| = \|\boldsymbol{z}_j\| = 1)$$

$$= \sum_{i,j}(\|\boldsymbol{z}_i\| - 2\boldsymbol{z}_i^\top \boldsymbol{z}_j + \|\boldsymbol{z}_j\|)W_{ij} - \kappa \tag{41}$$

$$= \sum_{i,j}|\boldsymbol{z}_i - \boldsymbol{z}_j\|^2 W_{ij} - \kappa. \tag{42}$$

□

Proposition 4 (Minimizing $\mathcal{L}_{\text{Unif}}$ maximizes entropy)**.** *Let $H_2(\cdot)$ be the 2-order Rényi entropy, estimated with a kernel density estimator using a Gaussian kernel. Then*

$$\underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n\in\mathbb{S}_d}{\arg\min}\ \mathcal{L}_{\text{Unif}} = \underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n\in\mathbb{S}_d}{\arg\max}\ H_2(\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n). \tag{43}$$

Proof. Using a Gaussian kernel, the 2-order Rényi entropy can be estimated as [4, Eq. (2.13)]

$$H_2(\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n) = -\log\left(\frac{1}{n^2}\sum_{l,m}\exp(-\frac{1}{2}\kappa\|\boldsymbol{z}_l - \boldsymbol{z}_m\|^2)\right) \tag{44}$$

Thus, we have

$$\underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n\in\mathbb{S}_d}{\arg\max}\ H_2(\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n) \tag{45}$$

$$= \underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n\in\mathbb{S}_d}{\arg\max}\ -\log\left(\frac{1}{n^2}\sum_{l,m}\exp(-\frac{1}{2}\kappa\|\boldsymbol{z}_l - \boldsymbol{z}_m\|^2)\right) \tag{46}$$

$$= \underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n\in\mathbb{S}_d}{\arg\min}\ \log\left(\sum_{l,m}\exp(-\frac{1}{2}\kappa\|\boldsymbol{z}_l - \boldsymbol{z}_m\|^2)\right) \tag{47}$$

$$= \underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n\in\mathbb{S}_d}{\arg\min}\ \log\left(\sum_{l,m}\exp(-\frac{1}{2}\kappa(\|\boldsymbol{z}_l\|^2\right. \tag{48}$$

$$\left. - 2\boldsymbol{z}_l^\top \boldsymbol{z}_m + \|\boldsymbol{z}_m\|^2)\right) \tag{49}$$

$$= \underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n\in\mathbb{S}_d}{\arg\min}\ \log\left(\sum_{l,m}\exp(-\kappa(1 - \boldsymbol{z}_l^\top \boldsymbol{z}_m))\right) \tag{50}$$

$$(\|\boldsymbol{z}_l\| = \|\boldsymbol{z}_m\| = 1)$$

$$= \underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n\in\mathbb{S}_d}{\arg\min}\ \log\left(\exp(-\kappa)\sum_{l,m}\exp(\kappa\boldsymbol{z}_l^\top \boldsymbol{z}_m)\right) \tag{51}$$

$$= \underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n\in\mathbb{S}_d}{\arg\min}\ \log\sum_{l,m}\exp(\kappa\boldsymbol{z}_l^\top \boldsymbol{z}_m) \tag{52}$$

$$= \underset{\boldsymbol{z}_1,\ldots,\boldsymbol{z}_n\in\mathbb{S}_d}{\arg\min}\ \mathcal{L}_{\text{Unif}}. \tag{53}$$

□

Definition 2 (Normalized counting measure)**.** *The normalized counting measure associated with a set B on A is*

$$\nu_B(A) = \frac{|B \cap A|}{|B|} \tag{54}$$

Definition 3 (Normalized surface area measure on $\mathbb{S}_d$)**.** *The normalized surface area measure on the hyperspehere $\mathbb{S}_d \subset \mathbb{R}^d$, of a subset $S' \subset \mathbb{S}_d$ is*

$$\sigma_d(S') = \frac{\int_{S'} \mathrm{d}S}{\int_{\mathbb{S}_d} \mathrm{d}S} = A_d^{-1}\int_{S'} \mathrm{d}S \tag{55}$$

where A_d is defined as in Eq. (1)*, and $\int \mathrm{d}S$ denotes the surface integral on $\mathbb{S}_d$.*

Definition 4 (Weak* convergence of measures [8])**.** *A sequence of Borel measures $\{\mu_n\}_{n=1}^\infty$ in $\mathbb{R}^d$ converges weak* to a Borel measure μ, if for all continuous functions $f : \mathbb{R}^d \to \mathbb{R}$,*

$$\lim_{n\to\infty}\int f(x)\mathrm{d}\mu_n(x) = \int f(x)\mathrm{d}\mu(x) \tag{56}$$

			mini		tiered		CUB	
Arch.	Param.	Method	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
ResNet18	P	noHub	45	45	45	45	45	45
		noHub-S	45	45	40	45	45	45
	T	noHub	50	50	50	50	50	50
		noHub-S	150	150	150	150	150	150
	α	noHub	0.2	0.2	0.2	0.2	0.2	0.2
		noHub-S	0.3	0.2	0.2	0.2	0.3	0.2
	η	noHub	0.1	0.1	0.1	0.1	0.1	0.1
		noHub-S	0.1	0.1	0.1	0.1	0.1	0.1
	κ	noHub	0.5	0.5	0.5	0.5	0.5	0.5
		noHub-S	0.5	0.5	0.5	0.5	0.5	0.5
	ε	noHub	–	–	–	–	–	–
		noHub-S	8	8	5	8	8	8
	d	noHub	400	400	400	400	400	400
		noHub-S	400	400	400	400	400	400
WideRes28-10	P	noHub	45	45	45	45	45	45
		noHub-S	45	45	40	35	45	30
	T	noHub	50	50	50	50	50	50
		noHub-S	150	150	150	150	150	150
	α	noHub	0.2	0.2	0.2	0.2	0.2	0.2
		noHub-S	0.3	0.2	0.2	0.1	0.3	0.1
	η	noHub	0.1	0.1	0.1	0.1	0.1	0.1
		noHub-S	0.1	0.1	0.1	0.1	0.1	0.1
	κ	noHub	0.5	0.5	0.5	0.5	0.5	0.5
		noHub-S	0.5	0.5	0.5	0.2	0.5	0.2
	ε	noHub	–	–	–	–	–	–
		noHub-S	8	8	5	12	8	8
	d	noHub	400	400	400	400	400	400
		noHub-S	400	400	400	400	400	400

Table 1. Hyperparameter values used in our experiments.

5.2. Results

FSL performance. The complete lists of accuracies and hubness metrics for all embeddings, classifiers, and feature extractors, are given in Tables 2, 3, 4, and 5. The exhaustive results in these tables form the basis of Table 1, Table 2 and Table 3 in the main text. The two proposed approaches consistently outperform prior embeddings across several classifiers, feature extractors and datasets.

Effect of the κ and ε hyperparameters. The plots in Figure 1 show accuracy on *tiered* 5-shot with SIAMESE for increasing κ and ε. Neither method is particularly sensitive to the choice of κ and ε, and noHub-S is less sensitive to variations in κ, than noHub. Choosing $\kappa \in [0.5, 1]$ and $\varepsilon \in [3, 20]$ will result in high classification accuracy

6. Potential Negative Societal Impacts

As is the case with most methodological research in machine learning, the methods developed in this work could be used in downstream applications with potential negative societal impacts. Real world machine learning-based systems that interact with humans, or the environment in general, should therefore be properly tested and equipped with adequate safety measures.

Since our work relies on a large number of labeled examples from the base classes, un-discovered biases from the base dataset could be transferred to the trained models. Furthermore, the small number of examples in the inference stage could make the query predictions biased towards the included support examples, and not accurately reflect the diversity of the novel classes.

Proposition 5 (Minimizer of $\mathcal{L}_{\text{Unif}}$). *For each $n > 0$, the n point minimizer of $\mathcal{L}_{\text{Unif}}$ is*

$$z_1^\star, \dots, z_n^\star = \underset{z_1,\dots,z_n \in \mathbb{S}_d}{\arg\min} \ \mathcal{L}_{\text{Unif}}. \tag{57}$$

Then $\nu_{\{z_1^\star,\dots,z_n^\star\}}$ converge weak to σ_d as $n \to \infty$.*

Proof. We have

$$\underset{z_1,\dots,z_n \in \mathbb{S}_d}{\arg\min} \ \mathcal{L}_{\text{Unif}} \tag{58}$$

$$= \underset{z_1,\dots,z_n \in \mathbb{S}_d}{\arg\min} \ \log \sum_{l,m} \exp(\kappa z_l^\top z_m) \tag{59}$$

$$= \underset{z_1,\dots,z_n \in \mathbb{S}_d}{\arg\min} \ \sum_{l,m} \exp(\kappa z_l^\top z_m) \tag{60}$$

(monotonicity of logarithm)

$$= \underset{z_1,\dots,z_n \in \mathbb{S}_d}{\arg\min} \ \sum_{1 \le l < m \le n} \exp(\kappa z_l^\top z_m) \tag{61}$$

(symmetry of inner product)

$$= \underset{z_1,\dots,z_n \in \mathbb{S}_d}{\arg\min} \ \sum_{1 \le l < m \le n} \underbrace{\exp(-\kappa ||z_l - z_m||_2^2)}_{=: \, G(z_l, z_m)} \tag{62}$$

(multiplication by positive constant)

$$= \underset{z_1,\dots,z_n \in \mathbb{S}_d}{\arg\min} \ \sum_{1 \le l < m \le n} G(z_l, z_m) \tag{63}$$

The result then follows directly from [8, Proposition 2]. □

5. Experiments

5.1. Implementation details

This section covers the additional implementation details not provided in the main paper. These include the initialization of the embeddings in Algorithm 1, hyperparameters, additional transformations wherever required, the architectures used, and a note on accessing the code, datasets, and dataset splits.

Initialization and normalization. Instead of a random initialization of our embeddings Z_0, we follow a PCA based initialization, as in [5]. The weights are computed using the cached features from the base classes, the support and query features are then transformed using these weights. This procedure is also fast as we do not need to compute the PCA weights on every episode. To ensure that the resulting features lie on the hypersphere after each gradient update in noHub and noHub-S, we re-normalize the embeddings using L2 normalization.

Hyperparameters. noHub and noHub-S have the following hyperparameters.

- P – perplexity for computing the κ_i.

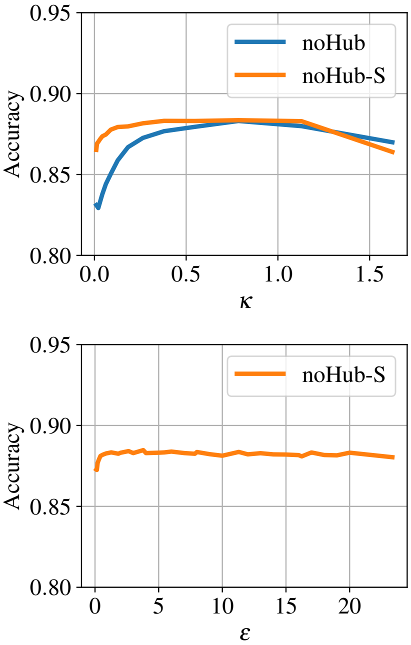


Figure 1. Accuracy for different values for κ and ε. Neither noHub nor noHub-S are particularly sensitive the the choice of these parameters.

- T – number of iterations.
- α – tradeoff parameter in the loss ($\mathcal{L}_{\text{noHub}} = \alpha\mathcal{L}_{\text{LSP}} + (1-\alpha)\mathcal{L}_{\text{Unif}}$).
- η – learning rate for the Adam optimizer.
- κ – concentration parameter for the embeddings.
- ε – exaggeration of similarities between supports from different classes.
- d – dimensionality of embeddings.

All hyperparameter values used in in noHub and noHub-S are given in Table 1

Code. The code for our experiments is available at: `https://github.com/uitml/noHub`

Data splits. Details to access the datasets used with the requisite splits (both are consistent with [6]) are available in the code repository.

Base feature extractors.

- **Resnet-18**: As in [1, 6], we use the weights from [6]. The model is trained using a cross-entropy loss on the base classes.
- **WideRes28-10**: Following [3, 9], we use the weights from [3]. The model is pre-trained using a combination of cross-entropy and rotation prediction [2], and then fine-tuned with Manifold Mixup [7].

				mini			tiered			CUB	
Arch.	Clf.	Emb.	Acc	Skew	Hub. Occ.	Acc	Skew	Hub. Occ.	Acc	Skew	Hub. Occ.
ResNet18	ILPC	None	64.07 (0.28)	1.411 (0.01)	0.408 (0.001)	75.5 (0.28)	1.213 (0.009)	0.41 (0.001)	76.06 (0.27)	0.886 (0.006)	0.34 (0.001)
		L2	69.28 (0.27)	0.966 (0.007)	0.298 (0.001)	77.84 (0.28)	0.811 (0.007)	0.267 (0.001)	79.91 (0.26)	0.688 (0.006)	0.236 (0.001)
		CL2	71.48 (0.27)	0.661 (0.005)	0.229 (0.001)	79.8 (0.27)	0.679 (0.006)	0.249 (0.001)	80.97 (0.26)	0.553 (0.005)	0.203 (0.001)
		ZN	71.48 (0.27)	0.677 (0.006)	0.227 (0.001)	79.95 (0.27)	0.694 (0.006)	0.263 (0.001)	81.49 (0.25)	0.57 (0.005)	0.217 (0.001)
		ReRep	65.49 (0.28)	3.688 (0.007)	0.559 (0.001)	76.75 (0.28)	3.61 (0.01)	0.55 (0.001)	77.73 (0.26)	3.563 (0.007)	0.512 (0.001)
		EASE	71.79 (0.28)	0.515 (0.005)	0.157 (0.001)	80.2 (0.27)	0.48 (0.005)	0.158 (0.001)	81.88 (0.25)	0.463 (0.004)	0.153 (0.001)
		TCPR	71.77 (0.28)	0.647 (0.005)	0.223 (0.001)	80.01 (0.28)	0.652 (0.006)	0.249 (0.001)	81.75 (0.25)	0.534 (0.004)	0.203 (0.001)
		noHub	73.18 (0.28)	0.308 (0.005)	**0.094 (0.001)**	80.76 (0.28)	0.296 (0.004)	**0.101 (0.001)**	82.74 (0.26)	0.32 (0.004)	**0.112 (0.001)**
		noHub-S	**74.02 (0.28)**	**0.276 (0.004)**	0.13 (0.001)	**81.34 (0.27)**	**0.281 (0.004)**	0.127 (0.001)	**83.92 (0.25)**	**0.296 (0.003)**	0.163 (0.001)
	LaplacianShot	None	68.92 (0.23)	1.341 (0.009)	0.408 (0.001)	76.43 (0.25)	1.214 (0.009)	0.41 (0.001)	79.17 (0.23)	0.887 (0.006)	0.34 (0.001)
		L2	69.3 (0.23)	0.945 (0.007)	0.302 (0.001)	77.2 (0.25)	0.808 (0.007)	0.265 (0.001)	79.65 (0.23)	0.682 (0.006)	0.236 (0.001)
		CL2	70.68 (0.23)	0.661 (0.005)	0.231 (0.001)	77.98 (0.24)	0.689 (0.006)	0.248 (0.001)	79.99 (0.22)	0.547 (0.005)	0.201 (0.001)
		ZN	70.51 (0.23)	0.688 (0.006)	0.233 (0.001)	77.51 (0.24)	0.697 (0.006)	0.264 (0.001)	79.86 (0.22)	0.564 (0.005)	0.217 (0.001)
		ReRep	72.75 (0.24)	3.653 (0.007)	0.548 (0.001)	78.95 (0.25)	3.605 (0.011)	0.549 (0.001)	82.38 (0.22)	3.565 (0.007)	0.512 (0.001)
		EASE	72.19 (0.23)	0.526 (0.005)	0.161 (0.001)	79.34 (0.24)	0.481 (0.005)	0.158 (0.001)	81.5 (0.22)	0.459 (0.004)	0.152 (0.001)
		TCPR	71.79 (0.23)	0.654 (0.005)	0.228 (0.001)	78.41 (0.24)	0.651 (0.005)	0.249 (0.001)	80.86 (0.22)	0.537 (0.004)	0.203 (0.001)
		noHub	73.63 (0.25)	0.305 (0.005)	**0.094 (0.001)**	**80.84 (0.25)**	0.3 (0.005)	**0.101 (0.001)**	83.23 (0.22)	0.318 (0.004)	**0.112 (0.001)**
		noHub-S	**73.79 (0.25)**	**0.276 (0.004)**	0.13 (0.001)	80.83 (0.25)	**0.275 (0.004)**	0.125 (0.001)	**83.47 (0.22)**	**0.299 (0.003)**	0.164 (0.001)
	ObliqueManifold	None	68.89 (0.23)	1.412 (0.01)	0.407 (0.001)	77.07 (0.25)	1.21 (0.009)	0.409 (0.001)	79.4 (0.22)	0.887 (0.006)	0.341 (0.001)
		L2	68.92 (0.23)	0.964 (0.007)	0.299 (0.001)	77.17 (0.25)	0.806 (0.007)	0.266 (0.001)	79.32 (0.22)	0.691 (0.006)	0.237 (0.001)
		CL2	70.86 (0.24)	0.66 (0.005)	0.228 (0.001)	78.92 (0.25)	0.68 (0.006)	0.249 (0.001)	80.29 (0.23)	0.547 (0.005)	0.202 (0.001)
		ZN	71.25 (0.24)	0.679 (0.006)	0.227 (0.001)	79.54 (0.25)	0.697 (0.006)	0.263 (0.001)	81.38 (0.23)	0.562 (0.005)	0.216 (0.001)
		ReRep	73.3 (0.25)	3.682 (0.007)	0.559 (0.001)	80.26 (0.26)	3.608 (0.01)	0.551 (0.001)	**83.84 (0.23)**	3.559 (0.008)	0.513 (0.001)
		EASE	68.4 (0.24)	0.516 (0.005)	0.156 (0.001)	77.33 (0.25)	0.477 (0.004)	0.158 (0.001)	79.03 (0.24)	0.461 (0.004)	0.152 (0.001)
		TCPR	70.74 (0.24)	0.646 (0.005)	0.223 (0.001)	78.92 (0.25)	0.649 (0.005)	0.249 (0.001)	80.18 (0.23)	0.537 (0.004)	0.204 (0.001)
		noHub	72.55 (0.26)	0.309 (0.005)	**0.095 (0.001)**	79.97 (0.26)	0.302 (0.005)	**0.102 (0.001)**	82.21 (0.24)	0.319 (0.004)	**0.112 (0.001)**
		noHub-S	**74.24 (0.26)**	**0.274 (0.004)**	0.13 (0.001)	**80.84 (0.26)**	**0.282 (0.004)**	0.127 (0.001)	83.67 (0.23)	**0.294 (0.003)**	0.162 (0.001)
	SIAMESE	None	20.0 (0.0)	1.345 (0.009)	0.407 (0.001)	20.0 (0.0)	1.222 (0.009)	0.41 (0.001)	20.0 (0.0)	0.885 (0.006)	0.339 (0.001)
		L2	73.77 (0.24)	0.949 (0.007)	0.301 (0.001)	80.46 (0.26)	0.811 (0.007)	0.265 (0.001)	83.1 (0.23)	0.691 (0.006)	0.237 (0.001)
		CL2	75.56 (0.26)	0.666 (0.005)	0.232 (0.001)	82.1 (0.26)	0.68 (0.006)	0.248 (0.001)	84.35 (0.24)	0.549 (0.005)	0.201 (0.001)
		ZN	20.0 (0.0)	0.686 (0.006)	0.232 (0.001)	20.0 (0.0)	0.69 (0.006)	0.262 (0.001)	20.0 (0.0)	0.565 (0.005)	0.217 (0.001)
		ReRep	20.0 (0.0)	3.653 (0.007)	0.549 (0.001)	20.0 (0.0)	3.616 (0.01)	0.549 (0.001)	20.0 (0.0)	3.559 (0.007)	0.512 (0.001)
		EASE	76.05 (0.27)	0.529 (0.005)	0.162 (0.001)	82.57 (0.27)	0.485 (0.005)	0.159 (0.001)	85.24 (0.24)	0.464 (0.004)	0.153 (0.001)
		TCPR	75.99 (0.26)	0.655 (0.005)	0.227 (0.001)	82.65 (0.26)	0.651 (0.005)	0.249 (0.001)	85.34 (0.23)	0.535 (0.004)	0.203 (0.001)
		noHub	76.65 (0.28)	0.308 (0.005)	**0.095 (0.001)**	82.94 (0.27)	0.303 (0.004)	**0.101 (0.001)**	**85.88 (0.24)**	0.322 (0.004)	**0.112 (0.001)**
		noHub-S	**76.68 (0.28)**	**0.275 (0.004)**	0.13 (0.001)	**83.09 (0.27)**	**0.281 (0.004)**	0.128 (0.001)	85.81 (0.24)	**0.295 (0.003)**	0.161 (0.001)
	SimpleShot	None	56.14 (0.2)	1.349 (0.009)	0.407 (0.001)	63.34 (0.23)	1.211 (0.009)	0.408 (0.001)	64.02 (0.21)	0.887 (0.006)	0.341 (0.001)
		L2	60.15 (0.2)	0.937 (0.007)	0.301 (0.001)	68.02 (0.23)	0.812 (0.007)	0.265 (0.001)	69.05 (0.21)	0.691 (0.006)	0.236 (0.001)
		CL2	63.1 (0.2)	0.667 (0.005)	0.233 (0.001)	69.76 (0.22)	0.679 (0.006)	0.249 (0.001)	70.16 (0.2)	0.549 (0.005)	0.201 (0.001)
		ZN	63.39 (0.2)	0.68 (0.005)	0.231 (0.001)	70.04 (0.22)	0.698 (0.006)	0.264 (0.001)	71.03 (0.2)	0.564 (0.005)	0.216 (0.001)
		ReRep	66.66 (0.22)	3.655 (0.007)	0.548 (0.001)	73.23 (0.23)	3.604 (0.01)	0.549 (0.001)	76.8 (0.21)	3.565 (0.007)	0.513 (0.001)
		EASE	64.0 (0.2)	0.521 (0.005)	0.16 (0.001)	71.0 (0.21)	0.479 (0.005)	0.158 (0.001)	72.38 (0.2)	0.466 (0.004)	0.153 (0.001)
		TCPR	63.33 (0.2)	0.651 (0.005)	0.228 (0.001)	69.82 (0.22)	0.65 (0.005)	0.25 (0.001)	70.75 (0.2)	0.532 (0.004)	0.204 (0.001)
		noHub	69.38 (0.22)	0.315 (0.005)	**0.095 (0.001)**	76.72 (0.23)	0.303 (0.004)	**0.102 (0.001)**	78.21 (0.21)	0.32 (0.004)	**0.112 (0.001)**
		noHub-S	**71.1 (0.22)**	**0.276 (0.004)**	0.13 (0.001)	**78.35 (0.23)**	**0.283 (0.004)**	0.127 (0.001)	**80.31 (0.21)**	**0.296 (0.003)**	0.162 (0.001)
	α-TIM	None	56.39 (0.2)	1.342 (0.009)	0.406 (0.001)	63.32 (0.23)	1.216 (0.009)	0.411 (0.001)	64.02 (0.22)	0.886 (0.006)	0.341 (0.001)
		L2	67.91 (0.23)	0.942 (0.007)	0.301 (0.001)	74.94 (0.24)	0.814 (0.007)	0.266 (0.001)	77.49 (0.23)	0.694 (0.006)	0.236 (0.001)
		CL2	65.68 (0.21)	0.665 (0.005)	0.232 (0.001)	73.23 (0.23)	0.681 (0.006)	0.248 (0.001)	73.79 (0.21)	0.552 (0.005)	0.202 (0.001)
		ZN	63.36 (0.2)	0.682 (0.005)	0.232 (0.001)	70.19 (0.22)	0.693 (0.006)	0.263 (0.001)	70.85 (0.21)	0.566 (0.005)	0.215 (0.001)
		ReRep	66.37 (0.22)	3.656 (0.007)	0.55 (0.001)	73.24 (0.24)	3.605 (0.011)	0.55 (0.001)	76.86 (0.21)	3.555 (0.007)	0.514 (0.001)
		EASE	65.32 (0.2)	0.526 (0.005)	0.163 (0.001)	71.88 (0.22)	0.477 (0.005)	0.158 (0.001)	73.03 (0.21)	0.459 (0.004)	0.151 (0.001)
		TCPR	66.19 (0.21)	0.65 (0.005)	0.227 (0.001)	73.24 (0.23)	0.649 (0.005)	0.25 (0.001)	74.07 (0.21)	0.532 (0.004)	0.203 (0.001)
		noHub	70.08 (0.23)	0.312 (0.005)	**0.094 (0.001)**	77.39 (0.24)	0.304 (0.004)	**0.101 (0.001)**	79.19 (0.22)	0.319 (0.004)	**0.112 (0.001)**
		noHub-S	**72.04 (0.23)**	**0.273 (0.004)**	0.13 (0.001)	**79.13 (0.24)**	**0.282 (0.004)**	0.126 (0.001)	**81.42 (0.22)**	**0.296 (0.003)**	0.161 (0.001)

Table 2. Resnet-18: 1-shot.

			mini			tiered			CUB		
Arch.	Clf.	Emb.	Acc	Skew	Hub. Occ.	Acc	Skew	Hub. Occ.	Acc	Skew	Hub. Occ.
WideRes28-10	ILPC	None	71.27 (0.28)	1.595 (0.01)	0.46 (0.001)	75.01 (0.28)	1.807 (0.01)	0.494 (0.001)	89.75 (0.19)	1.072 (0.009)	0.367 (0.001)
		L2	76.41 (0.26)	0.773 (0.006)	0.295 (0.001)	78.25 (0.27)	0.731 (0.006)	0.274 (0.001)	90.27 (0.2)	0.473 (0.004)	0.228 (0.001)
		CL2	74.13 (0.27)	0.993 (0.009)	0.29 (0.001)	78.2 (0.27)	0.815 (0.006)	0.306 (0.001)	90.34 (0.2)	0.524 (0.004)	0.267 (0.001)
		ZN	77.76 (0.26)	0.728 (0.005)	0.287 (0.001)	79.42 (0.27)	0.776 (0.006)	0.302 (0.001)	90.21 (0.2)	0.516 (0.004)	0.263 (0.001)
		ReRep	62.51 (0.34)	3.56 (0.002)	0.704 (0.001)	60.66 (0.37)	3.55 (0.002)	0.776 (0.001)	87.44 (0.25)	3.033 (0.008)	0.472 (0.001)
		EASE	78.01 (0.26)	0.47 (0.004)	0.176 (0.001)	79.64 (0.27)	0.479 (0.004)	0.175 (0.001)	90.76 (0.19)	0.437 (0.003)	0.212 (0.001)
		TCPR	78.37 (0.26)	0.584 (0.005)	0.237 (0.001)	79.55 (0.28)	0.683 (0.006)	0.265 (0.001)	90.77 (0.19)	0.476 (0.004)	0.23 (0.001)
		noHub	78.84 (0.27)	0.293 (0.004)	**0.112 (0.001)**	80.75 (0.28)	0.3 (0.004)	**0.112 (0.001)**	90.91 (0.2)	0.189 (0.004)	**0.109 (0.001)**
		noHub-S	**79.77 (0.26)**	**0.262 (0.004)**	0.148 (0.001)	**81.24 (0.27)**	**0.278 (0.004)**	0.135 (0.001)	**91.28 (0.19)**	**0.16 (0.004)**	0.13 (0.001)
	LaplacianShot	None	72.56 (0.23)	1.599 (0.01)	0.459 (0.001)	75.58 (0.25)	1.795 (0.01)	0.495 (0.001)	88.71 (0.19)	1.071 (0.009)	0.369 (0.001)
		L2	75.18 (0.23)	0.777 (0.006)	0.296 (0.001)	77.03 (0.24)	0.732 (0.006)	0.274 (0.001)	89.73 (0.17)	0.474 (0.004)	0.229 (0.001)
		CL2	71.29 (0.24)	0.987 (0.009)	0.29 (0.001)	75.42 (0.25)	0.819 (0.006)	0.309 (0.001)	89.61 (0.18)	0.52 (0.004)	0.268 (0.001)
		ZN	75.18 (0.22)	0.724 (0.005)	0.286 (0.001)	77.0 (0.24)	0.768 (0.006)	0.301 (0.001)	89.22 (0.18)	0.517 (0.004)	0.263 (0.001)
		ReRep	75.25 (0.22)	3.562 (0.002)	0.704 (0.001)	77.12 (0.24)	3.548 (0.002)	0.776 (0.001)	88.98 (0.18)	3.024 (0.008)	0.47 (0.001)
		EASE	77.29 (0.22)	0.473 (0.004)	0.177 (0.001)	78.97 (0.24)	0.475 (0.004)	0.175 (0.001)	90.06 (0.17)	0.435 (0.003)	0.213 (0.001)
		TCPR	76.77 (0.22)	0.593 (0.005)	0.236 (0.001)	77.49 (0.24)	0.686 (0.006)	0.264 (0.001)	89.42 (0.17)	0.475 (0.004)	0.231 (0.001)
		noHub	**79.13 (0.23)**	0.29 (0.004)	**0.111 (0.001)**	80.5 (0.25)	0.302 (0.004)	**0.112 (0.001)**	**90.73 (0.18)**	0.19 (0.004)	**0.109 (0.001)**
		noHub-S	79.13 (0.23)	**0.259 (0.004)**	0.147 (0.001)	**80.59 (0.24)**	**0.277 (0.004)**	0.135 (0.001)	90.61 (0.17)	**0.164 (0.004)**	0.13 (0.001)
	ObliqueManifold	None	76.02 (0.22)	1.599 (0.01)	0.46 (0.001)	77.75 (0.25)	1.801 (0.01)	0.494 (0.001)	90.82 (0.18)	1.07 (0.009)	0.368 (0.001)
		L2	76.11 (0.22)	0.779 (0.006)	0.295 (0.001)	77.74 (0.25)	0.731 (0.006)	0.274 (0.001)	90.89 (0.18)	0.475 (0.004)	0.228 (0.001)
		CL2	74.43 (0.24)	0.985 (0.009)	0.289 (0.001)	77.98 (0.25)	0.816 (0.007)	0.307 (0.001)	90.6 (0.18)	0.523 (0.004)	0.267 (0.001)
		ZN	77.69 (0.23)	0.724 (0.005)	0.286 (0.001)	79.32 (0.24)	0.767 (0.006)	0.301 (0.001)	90.73 (0.18)	0.519 (0.004)	0.263 (0.001)
		ReRep	78.08 (0.23)	3.56 (0.002)	0.703 (0.001)	79.46 (0.25)	3.549 (0.002)	0.777 (0.001)	91.16 (0.18)	3.032 (0.008)	0.471 (0.001)
		EASE	74.77 (0.23)	0.472 (0.004)	0.178 (0.001)	77.07 (0.25)	0.473 (0.004)	0.174 (0.001)	89.2 (0.18)	0.439 (0.003)	0.212 (0.001)
		TCPR	77.39 (0.23)	0.587 (0.005)	0.236 (0.001)	78.75 (0.24)	0.687 (0.006)	0.265 (0.001)	89.93 (0.19)	0.474 (0.004)	0.23 (0.001)
		noHub	78.44 (0.24)	0.292 (0.004)	**0.112 (0.001)**	79.99 (0.26)	0.302 (0.004)	**0.113 (0.001)**	90.59 (0.19)	0.185 (0.004)	**0.108 (0.001)**
		noHub-S	**79.89 (0.24)**	**0.259 (0.004)**	0.148 (0.001)	**80.67 (0.26)**	**0.279 (0.004)**	0.137 (0.001)	**91.37 (0.18)**	**0.162 (0.004)**	0.13 (0.001)
	SIAMESE	None	45.69 (0.31)	1.594 (0.009)	0.459 (0.001)	75.29 (0.28)	1.801 (0.01)	0.495 (0.001)	61.36 (0.55)	1.074 (0.009)	0.37 (0.001)
		L2	80.2 (0.23)	0.776 (0.006)	0.296 (0.001)	80.89 (0.26)	0.735 (0.006)	0.275 (0.001)	91.98 (0.18)	0.476 (0.004)	0.23 (0.001)
		CL2	75.23 (0.27)	0.988 (0.009)	0.289 (0.001)	79.59 (0.27)	0.82 (0.006)	0.307 (0.001)	92.17 (0.18)	0.518 (0.004)	0.266 (0.001)
		ZN	20.0 (0.0)	0.726 (0.005)	0.286 (0.001)	20.0 (0.0)	0.775 (0.006)	0.302 (0.001)	20.0 (0.0)	0.517 (0.004)	0.264 (0.001)
		ReRep	36.69 (0.28)	3.561 (0.002)	0.705 (0.001)	67.41 (0.29)	3.55 (0.002)	0.776 (0.001)	57.62 (0.56)	3.027 (0.008)	0.472 (0.001)
		EASE	81.19 (0.25)	0.474 (0.004)	0.178 (0.001)	82.04 (0.26)	0.476 (0.004)	0.176 (0.001)	91.99 (0.19)	0.436 (0.003)	0.213 (0.001)
		TCPR	81.27 (0.24)	0.582 (0.005)	0.236 (0.001)	81.89 (0.26)	0.681 (0.006)	0.264 (0.001)	91.91 (0.18)	0.477 (0.004)	0.232 (0.001)
		noHub	81.97 (0.25)	0.291 (0.004)	**0.111 (0.001)**	82.8 (0.27)	0.298 (0.004)	**0.112 (0.001)**	92.53 (0.18)	0.189 (0.004)	**0.109 (0.001)**
		noHub-S	**82.0 (0.26)**	**0.258 (0.004)**	0.148 (0.001)	**82.85 (0.27)**	**0.278 (0.004)**	0.137 (0.001)	**92.63 (0.18)**	**0.159 (0.004)**	0.13 (0.001)
	SimpleShot	None	55.66 (0.21)	1.6 (0.01)	0.459 (0.001)	54.71 (0.22)	1.81 (0.01)	0.494 (0.001)	70.92 (0.23)	1.073 (0.009)	0.369 (0.001)
		L2	65.78 (0.2)	0.781 (0.006)	0.296 (0.001)	68.75 (0.22)	0.737 (0.006)	0.275 (0.001)	82.85 (0.19)	0.475 (0.004)	0.228 (0.001)
		CL2	64.33 (0.2)	0.981 (0.009)	0.288 (0.001)	67.66 (0.22)	0.817 (0.006)	0.307 (0.001)	82.8 (0.19)	0.52 (0.004)	0.267 (0.001)
		ZN	67.31 (0.2)	0.73 (0.005)	0.287 (0.001)	69.14 (0.22)	0.769 (0.006)	0.302 (0.001)	82.79 (0.19)	0.517 (0.004)	0.263 (0.001)
		ReRep	67.38 (0.2)	3.56 (0.002)	0.704 (0.001)	70.17 (0.22)	3.55 (0.002)	0.777 (0.001)	84.86 (0.19)	3.026 (0.008)	0.47 (0.001)
		EASE	68.62 (0.2)	0.47 (0.004)	0.177 (0.001)	70.26 (0.21)	0.477 (0.004)	0.175 (0.001)	84.14 (0.18)	0.437 (0.003)	0.213 (0.001)
		TCPR	68.45 (0.2)	0.589 (0.005)	0.236 (0.001)	68.68 (0.22)	0.685 (0.006)	0.264 (0.001)	82.28 (0.19)	0.477 (0.004)	0.231 (0.001)
		noHub	75.06 (0.21)	0.29 (0.004)	**0.111 (0.001)**	76.7 (0.23)	0.301 (0.004)	**0.111 (0.001)**	88.06 (0.18)	0.188 (0.004)	**0.108 (0.001)**
		noHub-S	**76.86 (0.21)**	**0.258 (0.004)**	0.148 (0.001)	**78.4 (0.23)**	**0.274 (0.004)**	0.135 (0.001)	**89.25 (0.18)**	**0.162 (0.004)**	0.13 (0.001)
	α-TIM	None	60.31 (0.2)	1.603 (0.01)	0.458 (0.001)	69.42 (0.25)	1.811 (0.01)	0.494 (0.001)	73.83 (0.21)	1.072 (0.009)	0.369 (0.001)
		L2	72.11 (0.22)	0.778 (0.006)	0.295 (0.001)	74.45 (0.23)	0.73 (0.006)	0.275 (0.001)	85.96 (0.19)	0.476 (0.004)	0.229 (0.001)
		CL2	68.5 (0.21)	0.988 (0.009)	0.29 (0.001)	72.17 (0.23)	0.811 (0.006)	0.306 (0.001)	85.6 (0.18)	0.522 (0.004)	0.267 (0.001)
		ZN	67.69 (0.2)	0.73 (0.005)	0.287 (0.001)	68.94 (0.22)	0.769 (0.006)	0.302 (0.001)	83.03 (0.19)	0.518 (0.004)	0.263 (0.001)
		ReRep	73.15 (0.23)	3.56 (0.002)	0.704 (0.001)	76.19 (0.25)	3.551 (0.002)	0.778 (0.001)	88.55 (0.18)	3.027 (0.008)	0.472 (0.001)
		EASE	69.83 (0.2)	0.468 (0.004)	0.176 (0.001)	71.54 (0.22)	0.481 (0.004)	0.175 (0.001)	84.9 (0.19)	0.436 (0.003)	0.213 (0.001)
		TCPR	71.6 (0.21)	0.586 (0.005)	0.237 (0.001)	72.71 (0.22)	0.689 (0.006)	0.264 (0.001)	84.99 (0.19)	0.479 (0.004)	0.231 (0.001)
		noHub	75.87 (0.22)	0.29 (0.004)	**0.111 (0.001)**	77.83 (0.23)	0.302 (0.004)	**0.112 (0.001)**	88.7 (0.17)	0.189 (0.004)	**0.108 (0.001)**
		noHub-S	**77.76 (0.22)**	**0.259 (0.004)**	0.147 (0.001)	**79.04 (0.24)**	**0.276 (0.004)**	0.136 (0.001)	**89.77 (0.17)**	**0.163 (0.003)**	0.13 (0.001)

Table 3. WideRes28-10: 1-shot.

				mini			tiered			CUB	
Arch.	Clf.	Emb.	Acc	Skew	Hub. Occ.	Acc	Skew	Hub. Occ.	Acc	Skew	Hub. Occ.
ResNet18	ILPC	None	76.46 (0.18)	1.503 (0.01)	0.421 (0.001)	84.46 (0.18)	1.334 (0.008)	0.433 (0.001)	85.86 (0.14)	0.981 (0.005)	0.364 (0.001)
		L2	80.9 (0.16)	1.051 (0.007)	0.314 (0.001)	86.23 (0.17)	0.912 (0.006)	0.289 (0.001)	88.03 (0.13)	0.808 (0.005)	0.264 (0.001)
		CL2	81.64 (0.16)	0.778 (0.005)	0.262 (0.001)	86.88 (0.17)	0.823 (0.006)	0.281 (0.001)	88.44 (0.13)	0.695 (0.005)	0.235 (0.001)
		ZN	81.61 (0.16)	0.793 (0.005)	0.258 (0.001)	86.9 (0.17)	0.841 (0.006)	0.297 (0.001)	88.44 (0.12)	0.717 (0.004)	0.25 (0.001)
		ReRep	74.83 (0.19)	1.623 (0.003)	0.871 (0.001)	83.96 (0.19)	1.722 (0.004)	0.873 (0.001)	84.54 (0.15)	1.432 (0.003)	0.869 (0.001)
		EASE	81.75 (0.16)	0.618 (0.005)	0.182 (0.001)	86.84 (0.17)	0.593 (0.004)	0.181 (0.001)	88.85 (0.12)	0.606 (0.004)	0.186 (0.001)
		TCPR	81.76 (0.16)	0.766 (0.005)	0.254 (0.001)	86.78 (0.17)	0.801 (0.005)	0.284 (0.001)	88.69 (0.13)	0.683 (0.004)	0.237 (0.001)
		noHub	82.09 (0.16)	**0.295 (0.004)**	0.097 (0.001)	86.81 (0.17)	**0.289 (0.004)**	0.102 (0.001)	88.85 (0.13)	**0.333 (0.004)**	0.12 (0.001)
		noHub-S	**82.33 (0.16)**	0.488 (0.006)	**0.086 (0.001)**	**87.05 (0.17)**	0.475 (0.006)	**0.091 (0.001)**	**89.12 (0.13)**	0.438 (0.006)	**0.097 (0.001)**
	LaplacianShot	None	81.97 (0.15)	1.442 (0.009)	0.422 (0.001)	86.17 (0.16)	1.336 (0.008)	0.432 (0.001)	88.58 (0.12)	0.985 (0.005)	0.365 (0.001)
		L2	81.89 (0.14)	1.035 (0.007)	0.319 (0.001)	86.19 (0.16)	0.913 (0.006)	0.289 (0.001)	88.52 (0.11)	0.811 (0.005)	0.264 (0.001)
		CL2	81.93 (0.14)	0.786 (0.005)	0.265 (0.001)	86.16 (0.16)	0.82 (0.006)	0.282 (0.001)	88.46 (0.12)	0.7 (0.005)	0.235 (0.001)
		ZN	82.57 (0.14)	0.803 (0.005)	0.263 (0.001)	86.67 (0.16)	0.838 (0.006)	0.296 (0.001)	88.88 (0.11)	0.714 (0.004)	0.25 (0.001)
		ReRep	82.32 (0.14)	1.633 (0.003)	0.863 (0.001)	86.09 (0.16)	1.721 (0.004)	0.873 (0.001)	88.74 (0.12)	1.431 (0.002)	0.869 (0.001)
		EASE	82.57 (0.14)	0.627 (0.005)	0.186 (0.001)	86.82 (0.15)	0.596 (0.004)	0.182 (0.001)	88.94 (0.11)	0.608 (0.004)	0.185 (0.001)
		TCPR	82.24 (0.14)	0.781 (0.005)	0.259 (0.001)	86.27 (0.16)	0.797 (0.005)	0.284 (0.001)	88.63 (0.11)	0.687 (0.004)	0.236 (0.001)
		noHub	82.55 (0.15)	0.285 (0.004)	0.096 (0.001)	86.75 (0.16)	0.29 (0.004)	0.103 (0.001)	**89.08 (0.11)**	**0.329 (0.004)**	0.12 (0.001)
		noHub-S	**82.81 (0.14)**	**0.25 (0.005)**	**0.073 (0.001)**	**87.12 (0.16)**	**0.214 (0.005)**	**0.077 (0.001)**	88.99 (0.11)	0.438 (0.006)	**0.096 (0.001)**
	ObliqueManifold	None	83.53 (0.15)	1.497 (0.01)	0.421 (0.001)	87.85 (0.15)	1.334 (0.009)	0.433 (0.001)	90.28 (0.11)	0.987 (0.005)	0.364 (0.001)
		L2	83.66 (0.15)	1.051 (0.007)	0.314 (0.001)	87.83 (0.15)	0.922 (0.006)	0.289 (0.001)	90.21 (0.11)	0.81 (0.005)	0.263 (0.001)
		CL2	83.62 (0.15)	0.775 (0.005)	0.261 (0.001)	88.1 (0.15)	0.823 (0.006)	0.281 (0.001)	90.09 (0.11)	0.701 (0.005)	0.236 (0.001)
		ZN	**83.86 (0.15)**	0.795 (0.005)	0.258 (0.001)	**88.47 (0.15)**	0.835 (0.006)	0.296 (0.001)	**90.47 (0.11)**	0.716 (0.004)	0.251 (0.001)
		ReRep	82.44 (0.15)	1.62 (0.003)	0.871 (0.001)	86.85 (0.16)	1.725 (0.004)	0.872 (0.001)	89.83 (0.11)	1.431 (0.003)	0.869 (0.001)
		EASE	82.83 (0.15)	0.628 (0.005)	0.185 (0.001)	87.63 (0.16)	0.597 (0.005)	0.182 (0.001)	89.74 (0.12)	0.609 (0.004)	0.186 (0.001)
		TCPR	83.51 (0.15)	0.766 (0.005)	0.255 (0.001)	88.09 (0.15)	0.795 (0.005)	0.283 (0.001)	90.28 (0.11)	0.687 (0.004)	0.236 (0.001)
		noHub	83.28 (0.15)	**0.287 (0.004)**	0.096 (0.001)	87.58 (0.16)	**0.288 (0.004)**	0.102 (0.001)	89.89 (0.12)	**0.334 (0.004)**	0.121 (0.001)
		noHub-S	83.25 (0.16)	0.487 (0.006)	**0.086 (0.001)**	87.82 (0.16)	0.469 (0.006)	**0.091 (0.001)**	89.38 (0.17)	nan (nan)	**0.097 (0.001)**
	SIAMESE	None	20.0 (0.0)	1.441 (0.009)	0.421 (0.001)	20.0 (0.0)	1.339 (0.009)	0.433 (0.001)	20.0 (0.0)	0.984 (0.005)	0.364 (0.001)
		L2	83.14 (0.14)	1.035 (0.007)	0.319 (0.001)	87.04 (0.16)	0.912 (0.006)	0.288 (0.001)	89.48 (0.12)	0.808 (0.005)	0.264 (0.001)
		CL2	84.04 (0.15)	0.788 (0.005)	0.264 (0.001)	87.9 (0.16)	0.816 (0.006)	0.28 (0.001)	90.14 (0.12)	0.698 (0.005)	0.235 (0.001)
		ZN	20.0 (0.0)	0.8 (0.005)	0.263 (0.001)	20.0 (0.0)	0.84 (0.006)	0.296 (0.001)	20.0 (0.0)	0.713 (0.004)	0.251 (0.001)
		ReRep	20.0 (0.0)	1.633 (0.003)	0.863 (0.001)	20.0 (0.0)	1.724 (0.004)	0.872 (0.001)	20.0 (0.0)	1.428 (0.002)	0.869 (0.001)
		EASE	84.61 (0.15)	0.63 (0.005)	0.187 (0.001)	88.33 (0.16)	0.594 (0.004)	0.182 (0.001)	90.42 (0.12)	0.607 (0.004)	0.185 (0.001)
		TCPR	84.39 (0.15)	0.772 (0.005)	0.259 (0.001)	88.26 (0.16)	0.791 (0.005)	0.283 (0.001)	90.5 (0.11)	0.686 (0.004)	0.235 (0.001)
		noHub	84.05 (0.16)	0.292 (0.004)	0.096 (0.001)	87.87 (0.17)	**0.291 (0.004)**	0.103 (0.001)	90.34 (0.12)	**0.334 (0.004)**	0.12 (0.001)
		noHub-S	**84.67 (0.15)**	**0.247 (0.005)**	**0.074 (0.001)**	**88.43 (0.16)**	0.473 (0.006)	**0.092 (0.001)**	**90.52 (0.12)**	0.443 (0.006)	**0.097 (0.001)**
	SimpleShot	None	78.5 (0.14)	1.436 (0.009)	0.422 (0.001)	83.95 (0.16)	1.339 (0.008)	0.432 (0.001)	85.65 (0.12)	0.987 (0.005)	0.364 (0.001)
		L2	79.89 (0.14)	1.04 (0.007)	0.318 (0.001)	84.5 (0.16)	0.914 (0.006)	0.287 (0.001)	86.46 (0.12)	0.812 (0.005)	0.263 (0.001)
		CL2	80.0 (0.14)	0.786 (0.005)	0.264 (0.001)	84.66 (0.16)	0.821 (0.006)	0.28 (0.001)	86.3 (0.12)	0.698 (0.005)	0.236 (0.001)
		ZN	80.57 (0.14)	0.806 (0.005)	0.264 (0.001)	84.97 (0.16)	0.839 (0.006)	0.296 (0.001)	86.76 (0.12)	0.716 (0.005)	0.25 (0.001)
		ReRep	80.86 (0.14)	1.631 (0.003)	0.863 (0.001)	85.05 (0.16)	1.721 (0.004)	0.872 (0.001)	87.83 (0.12)	1.432 (0.002)	0.869 (0.001)
		EASE	80.13 (0.14)	0.624 (0.005)	0.186 (0.001)	84.74 (0.16)	0.598 (0.004)	0.183 (0.001)	86.76 (0.12)	0.607 (0.004)	0.186 (0.001)
		TCPR	80.15 (0.14)	0.78 (0.005)	0.259 (0.001)	84.86 (0.15)	0.796 (0.005)	0.283 (0.001)	86.8 (0.12)	0.687 (0.004)	0.235 (0.001)
		noHub	**82.13 (0.14)**	0.286 (0.004)	0.096 (0.001)	**86.31 (0.16)**	0.289 (0.004)	0.104 (0.001)	**88.46 (0.11)**	**0.329 (0.004)**	0.12 (0.001)
		noHub-S	81.22 (0.14)	**0.25 (0.005)**	**0.074 (0.001)**	86.22 (0.15)	**0.213 (0.005)**	**0.078 (0.001)**	87.6 (0.12)	0.433 (0.006)	**0.097 (0.001)**
	α-TIM	None	78.51 (0.15)	1.45 (0.009)	0.42 (0.001)	83.86 (0.16)	1.341 (0.009)	0.433 (0.001)	85.7 (0.12)	0.981 (0.005)	0.363 (0.001)
		L2	80.02 (0.16)	1.036 (0.007)	0.318 (0.001)	84.49 (0.18)	0.92 (0.006)	0.288 (0.001)	87.88 (0.13)	0.812 (0.005)	0.264 (0.001)
		CL2	80.46 (0.16)	0.784 (0.005)	0.264 (0.001)	84.86 (0.17)	0.82 (0.006)	0.281 (0.001)	87.53 (0.13)	0.701 (0.005)	0.235 (0.001)
		ZN	80.32 (0.14)	0.802 (0.005)	0.263 (0.001)	84.93 (0.16)	0.834 (0.006)	0.295 (0.001)	86.95 (0.12)	0.715 (0.004)	0.25 (0.001)
		ReRep	81.05 (0.14)	1.63 (0.003)	0.863 (0.001)	85.18 (0.16)	1.718 (0.004)	0.872 (0.001)	87.63 (0.12)	1.43 (0.002)	0.87 (0.001)
		EASE	79.13 (0.15)	0.632 (0.005)	0.188 (0.001)	84.04 (0.17)	0.596 (0.004)	0.181 (0.001)	86.7 (0.13)	0.607 (0.004)	0.186 (0.001)
		TCPR	80.52 (0.16)	0.776 (0.005)	0.259 (0.001)	85.01 (0.17)	0.796 (0.005)	0.283 (0.001)	87.81 (0.13)	0.681 (0.004)	0.234 (0.001)
		noHub	**81.39 (0.15)**	0.29 (0.004)	0.096 (0.001)	86.09 (0.16)	0.292 (0.004)	0.103 (0.001)	**88.16 (0.12)**	**0.336 (0.004)**	0.121 (0.001)
		noHub-S	81.37 (0.15)	**0.253 (0.005)**	**0.074 (0.001)**	**86.14 (0.16)**	**0.219 (0.005)**	**0.078 (0.001)**	87.97 (0.12)	0.437 (0.006)	**0.096 (0.001)**

Table 4. Resnet-18: 5-shot.

			mini			tiered			CUB		
			Acc	Skew	Hub. Occ.	Acc	Skew	Hub. Occ.	Acc	Skew	Hub. Occ.
Arch.	Clf.	Emb.									
WideRes28-10	ILPC	None	81.93 (0.16)	1.717 (0.01)	0.473 (0.001)	84.34 (0.17)	1.927 (0.011)	0.509 (0.001)	93.18 (0.11)	1.164 (0.008)	0.396 (0.001)
		L2	85.74 (0.14)	0.888 (0.005)	0.322 (0.001)	86.26 (0.17)	0.859 (0.005)	0.306 (0.001)	93.77 (0.1)	0.636 (0.004)	0.266 (0.001)
		CL2	83.33 (0.16)	1.12 (0.009)	0.318 (0.001)	85.99 (0.17)	0.957 (0.006)	0.338 (0.001)	93.79 (0.1)	0.703 (0.004)	0.309 (0.001)
		ZN	85.96 (0.14)	0.858 (0.005)	0.32 (0.001)	86.77 (0.16)	0.909 (0.006)	0.335 (0.001)	93.73 (0.1)	0.696 (0.004)	0.305 (0.001)
		ReRep	72.11 (0.27)	1.601 (0.003)	0.819 (0.001)	71.68 (0.3)	1.616 (0.004)	0.845 (0.001)	91.52 (0.13)	1.301 (0.005)	0.548 (0.002)
		EASE	85.89 (0.14)	0.577 (0.004)	0.198 (0.001)	86.83 (0.17)	0.583 (0.004)	0.193 (0.001)	**93.87 (0.1)**	0.576 (0.004)	0.242 (0.001)
		TCPR	86.29 (0.14)	0.715 (0.004)	0.27 (0.001)	86.96 (0.17)	0.819 (0.005)	0.295 (0.001)	93.82 (0.1)	0.634 (0.004)	0.265 (0.001)
		noHub	86.07 (0.15)	**0.295 (0.004)**	0.115 (0.001)	86.75 (0.17)	**0.299 (0.004)**	**0.115 (0.001)**	93.72 (0.1)	**0.2 (0.004)**	**0.101 (0.001)**
		noHub-S	**86.41 (0.14)**	0.499 (0.006)	**0.104 (0.001)**	**87.31 (0.17)**	0.406 (0.005)	0.121 (0.001)	93.79 (0.1)	0.416 (0.005)	0.126 (0.001)
	LaplacianShot	None	85.23 (0.13)	1.711 (0.01)	0.474 (0.001)	86.14 (0.15)	1.921 (0.011)	0.509 (0.001)	92.61 (0.1)	1.164 (0.008)	0.395 (0.001)
		L2	85.9 (0.13)	0.892 (0.006)	0.321 (0.001)	86.47 (0.15)	0.867 (0.006)	0.304 (0.001)	93.17 (0.09)	0.635 (0.004)	0.267 (0.001)
		CL2	82.08 (0.15)	1.112 (0.009)	0.318 (0.001)	84.62 (0.16)	0.954 (0.006)	0.34 (0.001)	93.01 (0.1)	0.702 (0.004)	0.309 (0.001)
		ZN	85.97 (0.13)	0.86 (0.005)	0.319 (0.001)	86.67 (0.15)	0.912 (0.006)	0.335 (0.001)	93.3 (0.1)	0.698 (0.004)	0.305 (0.001)
		ReRep	84.34 (0.14)	1.599 (0.003)	0.819 (0.001)	85.61 (0.16)	1.615 (0.004)	0.845 (0.001)	92.2 (0.1)	1.304 (0.005)	0.549 (0.002)
		EASE	86.24 (0.13)	0.573 (0.004)	0.198 (0.001)	86.74 (0.15)	0.582 (0.004)	0.194 (0.001)	93.31 (0.09)	0.578 (0.004)	0.243 (0.001)
		TCPR	86.16 (0.13)	0.712 (0.004)	0.269 (0.001)	85.72 (0.16)	0.813 (0.005)	0.293 (0.001)	92.99 (0.1)	0.638 (0.004)	0.264 (0.001)
		noHub	**86.25 (0.13)**	**0.292 (0.004)**	0.115 (0.001)	**86.78 (0.16)**	**0.299 (0.004)**	**0.115 (0.001)**	**93.38 (0.09)**	**0.197 (0.004)**	**0.1 (0.001)**
		noHub-S	85.79 (0.13)	0.494 (0.006)	**0.103 (0.001)**	86.44 (0.16)	0.397 (0.005)	0.12 (0.001)	93.36 (0.1)	0.42 (0.005)	0.126 (0.001)
	ObliqueManifold	None	87.46 (0.13)	1.712 (0.01)	0.472 (0.001)	88.16 (0.15)	1.913 (0.01)	0.509 (0.001)	94.75 (0.09)	1.161 (0.008)	0.395 (0.001)
		L2	87.61 (0.13)	0.889 (0.005)	0.321 (0.001)	88.14 (0.15)	0.862 (0.006)	0.306 (0.001)	**94.8 (0.09)**	0.642 (0.004)	0.268 (0.001)
		CL2	86.03 (0.14)	1.112 (0.009)	0.317 (0.001)	87.64 (0.16)	0.949 (0.006)	0.338 (0.001)	94.67 (0.09)	0.703 (0.004)	0.31 (0.001)
		ZN	87.88 (0.13)	0.852 (0.005)	0.32 (0.001)	**88.43 (0.15)**	0.908 (0.006)	0.335 (0.001)	94.77 (0.08)	0.697 (0.004)	0.306 (0.001)
		ReRep	87.62 (0.12)	1.599 (0.003)	0.819 (0.001)	88.15 (0.15)	1.616 (0.004)	0.845 (0.001)	94.48 (0.09)	1.302 (0.005)	0.547 (0.002)
		EASE	86.75 (0.13)	0.573 (0.004)	0.198 (0.001)	87.78 (0.15)	0.583 (0.004)	0.193 (0.001)	94.16 (0.09)	0.57 (0.004)	0.24 (0.001)
		TCPR	**87.94 (0.12)**	0.718 (0.004)	0.271 (0.001)	88.15 (0.15)	0.816 (0.005)	0.294 (0.001)	94.47 (0.09)	0.635 (0.004)	0.265 (0.001)
		noHub	87.23 (0.13)	**0.297 (0.004)**	0.115 (0.001)	87.95 (0.16)	**0.296 (0.004)**	**0.114 (0.001)**	94.13 (0.09)	**0.197 (0.004)**	**0.1 (0.001)**
		noHub-S	87.13 (0.14)	0.495 (0.006)	**0.103 (0.001)**	87.84 (0.16)	0.399 (0.005)	0.12 (0.001)	94.06 (0.09)	0.421 (0.005)	0.126 (0.001)
	SIAMESE	None	58.82 (0.31)	1.722 (0.01)	0.473 (0.001)	82.56 (0.22)	1.93 (0.01)	0.511 (0.001)	82.22 (0.37)	1.154 (0.008)	0.396 (0.001)
		L2	87.11 (0.13)	0.894 (0.006)	0.321 (0.001)	87.34 (0.15)	0.861 (0.005)	0.305 (0.001)	94.15 (0.1)	0.638 (0.004)	0.266 (0.001)
		CL2	83.99 (0.16)	1.107 (0.009)	0.318 (0.001)	86.71 (0.16)	0.953 (0.006)	0.339 (0.001)	94.48 (0.09)	0.704 (0.004)	0.31 (0.001)
		ZN	20.0 (0.0)	0.856 (0.005)	0.319 (0.001)	20.0 (0.0)	0.913 (0.006)	0.334 (0.001)	20.0 (0.0)	0.702 (0.004)	0.305 (0.001)
		ReRep	36.41 (0.3)	1.597 (0.003)	0.818 (0.001)	76.49 (0.24)	1.613 (0.004)	0.846 (0.001)	60.36 (0.6)	1.299 (0.005)	0.547 (0.002)
		EASE	87.82 (0.13)	0.579 (0.004)	0.199 (0.001)	88.06 (0.16)	0.586 (0.004)	0.192 (0.001)	94.36 (0.09)	0.571 (0.004)	0.241 (0.001)
		TCPR	87.8 (0.13)	0.717 (0.004)	0.27 (0.001)	87.95 (0.16)	0.822 (0.005)	0.295 (0.001)	94.25 (0.1)	0.637 (0.004)	0.266 (0.001)
		noHub	87.78 (0.14)	**0.29 (0.004)**	0.114 (0.001)	87.99 (0.17)	**0.297 (0.004)**	**0.115 (0.001)**	94.56 (0.09)	**0.196 (0.004)**	**0.1 (0.001)**
		noHub-S	**88.03 (0.13)**	0.492 (0.006)	**0.103 (0.001)**	**88.31 (0.16)**	0.398 (0.005)	0.12 (0.001)	**94.69 (0.09)**	0.416 (0.005)	0.127 (0.001)
	SimpleShot	None	78.56 (0.14)	1.709 (0.01)	0.473 (0.001)	80.32 (0.16)	1.937 (0.01)	0.51 (0.001)	89.27 (0.11)	1.16 (0.008)	0.395 (0.001)
		L2	83.81 (0.13)	0.887 (0.005)	0.322 (0.001)	84.82 (0.15)	0.86 (0.006)	0.305 (0.001)	92.06 (0.1)	0.632 (0.004)	0.266 (0.001)
		CL2	81.05 (0.14)	1.12 (0.009)	0.318 (0.001)	83.82 (0.16)	0.956 (0.006)	0.337 (0.001)	92.19 (0.1)	0.701 (0.004)	0.31 (0.001)
		ZN	83.92 (0.13)	0.858 (0.005)	0.32 (0.001)	85.1 (0.15)	0.912 (0.006)	0.335 (0.001)	92.17 (0.1)	0.699 (0.004)	0.305 (0.001)
		ReRep	79.26 (0.16)	1.597 (0.003)	0.819 (0.001)	82.7 (0.16)	1.617 (0.004)	0.846 (0.001)	91.48 (0.11)	1.299 (0.005)	0.549 (0.002)
		EASE	83.65 (0.13)	0.579 (0.004)	0.199 (0.001)	84.47 (0.15)	0.585 (0.004)	0.193 (0.001)	92.01 (0.1)	0.572 (0.004)	0.241 (0.001)
		TCPR	83.77 (0.13)	0.717 (0.004)	0.27 (0.001)	84.81 (0.15)	0.815 (0.005)	0.294 (0.001)	91.84 (0.1)	0.634 (0.004)	0.264 (0.001)
		noHub	**85.73 (0.13)**	**0.294 (0.004)**	0.115 (0.001)	**86.58 (0.15)**	**0.298 (0.004)**	**0.115 (0.001)**	93.21 (0.09)	**0.195 (0.004)**	**0.1 (0.001)**
		noHub-S	84.39 (0.13)	0.494 (0.006)	**0.103 (0.001)**	86.38 (0.15)	0.407 (0.005)	0.12 (0.001)	**93.39 (0.09)**	0.421 (0.005)	0.127 (0.001)
	α-TIM	None	80.61 (0.15)	1.711 (0.01)	0.473 (0.001)	83.05 (0.18)	1.928 (0.01)	0.51 (0.001)	84.89 (0.29)	1.153 (0.008)	0.396 (0.001)
		L2	83.71 (0.16)	0.892 (0.005)	0.323 (0.001)	84.69 (0.18)	0.863 (0.005)	0.304 (0.001)	92.88 (0.1)	0.633 (0.004)	0.266 (0.001)
		CL2	82.35 (0.16)	1.111 (0.009)	0.318 (0.001)	84.06 (0.18)	0.949 (0.006)	0.339 (0.001)	92.81 (0.1)	0.7 (0.004)	0.31 (0.001)
		ZN	83.93 (0.13)	0.857 (0.005)	0.321 (0.001)	85.07 (0.15)	0.912 (0.006)	0.336 (0.001)	92.15 (0.1)	0.698 (0.004)	0.306 (0.001)
		ReRep	83.4 (0.14)	1.596 (0.003)	0.82 (0.001)	84.4 (0.16)	1.615 (0.004)	0.845 (0.001)	93.19 (0.09)	1.302 (0.005)	0.547 (0.002)
		EASE	82.72 (0.14)	0.576 (0.004)	0.2 (0.001)	83.86 (0.16)	0.583 (0.004)	0.193 (0.001)	92.31 (0.1)	0.572 (0.004)	0.242 (0.001)
		TCPR	84.21 (0.15)	0.718 (0.004)	0.27 (0.001)	84.63 (0.18)	0.814 (0.005)	0.293 (0.001)	92.44 (0.1)	0.635 (0.004)	0.265 (0.001)
		noHub	**85.56 (0.13)**	**0.293 (0.004)**	0.115 (0.001)	**86.37 (0.16)**	**0.3 (0.004)**	**0.115 (0.001)**	92.89 (0.1)	**0.193 (0.004)**	**0.099 (0.001)**
		noHub-S	83.96 (0.15)	0.496 (0.006)	**0.102 (0.001)**	86.01 (0.16)	0.395 (0.005)	0.12 (0.001)	**93.24 (0.1)**	0.422 (0.005)	0.126 (0.001)

Table 5. WideRes28-10: 5-shot.

References

[1] Malik Boudiaf, Ziko Imtiaz Masud, Jérôme Rony, José Dolz, Pablo Piantanida, and Ismail Ben Ayed. Transductive Information Maximization For Few-Shot Learning. In *NeurIPS*, 2020. 4

[2] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised Representation Learning by Predicting Image Rotations. In *ICLR*, 2018. 4

[3] Puneet Mangla, Mayank Singh, Abhishek Sinha, Nupur Kumari, Vineeth N Balasubramanian, and Balaji Krishnamurthy. Charting the Right Manifold: Manifold Mixup for Few-shot Learning. In *WACV*, 2020. 4

[4] Jose C Principe. *Information theoretic learning: Renyi's entropy and kernel perspectives*. Springer Science & Business Media, 2010. 3

[5] Laurens van der Maaten and Geoffrey E. Hinton. Visualizing data using t-sne. *JMLR*, 2008. 2, 4

[6] Olivier Veilleux, Malik Boudiaf, Pablo Piantanida, and Ismail Ben Ayed. Realistic evaluation of transductive few-shot learning. In *NeurIPS*, 2021. 4

[7] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold Mixup: Better Representations by Interpolating Hidden States. In *ICML*, 2019. 4

[8] Tongzhou Wang and Phillip Isola. Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere. In *ICML*, 2020. 3, 4

[9] Hao Zhu and Piotr Koniusz. EASE: Unsupervised Discriminant Subspace Learning for Transductive Few-Shot Learning. In *CVPR*, 2022. 4

/12

Paper III

Visual Data Diagnosis and Debiasing with Concept Graphs

Rwiddhi Chakraborty[1,2]* **Yinong Wang**[1] **Jialu Gao**[1] **Cheng Zhang**[1]

Runkai Zheng[1] **Fernando de la Torre**[1]

[1]Carnegie Mellon University, Pittsburgh, PA, USA
[2]UiT The Arctic University of Norway, Tromsø, Norway

Abstract

The widespread success of deep learning models today is owed to the curation of extensive datasets significant in size and complexity. However, such models frequently pick up inherent biases in the data during the training process, leading to unreliable predictions. Diagnosing and debiasing datasets is thus a necessity to ensure reliable model performance. In this paper, we present ConBias, a novel framework for diagnosing and mitigating **Con**cept co-occurrence **Bias**es in visual datasets. ConBias represents visual datasets as knowledge graphs of concepts, enabling meticulous analysis of spurious concept co-occurrences to uncover concept imbalances across the whole dataset. Moreover, we show that by employing a novel clique-based concept balancing strategy, we can mitigate these imbalances, leading to enhanced performance on downstream tasks. Extensive experiments show that data augmentation based on a balanced concept distribution augmented by ConBias improves generalization performance across multiple datasets compared to state-of-the-art methods. We will make our code and data publicly available.

1 Introduction

Over the last decade we have witnessed an unparalleled growth in the capabilities of deep learning models across a wide range of tasks, such as image classification [17, 47, 7], object detection [41, 55], semantic segmentation [20, 26, 43], and so on. More recently, with the introduction of large multi-modal models, these capabilities have improved further [25, 15]. However, such models, while demonstrating impressive performance on a wide range of tasks, have been shown to be biased in their predictions [30, 13]. These biases come in various forms, based in texture [14], shape [39, 32], object co-occurrence [51, 52, 48], and so on. In addition to exploring model biases, dataset diagnosis, or evaluating biases directly within the dataset, is particularly crucial as large datasets available today are beyond the scope of human evaluation, owing to their size and complexity. For example, ImageNet [6], a widely used dataset in deep learning literature, is known to have thousands of erroneous labels and a lack of diversity in its class hierarchy [33, 57]. Other popular datasets such as MS-COCO [23] and CelebA [27], have problematic social biases with respect to gendered captions and prejudicial attributes of people from different races. As a result, frameworks that effectively diagnose and debias these datasets are sought.

While multiple works exist in the categorization and exploration of biases in visual data [9, 30], an end-to-end pipeline incorporating both diagnosis and debiasing has received relatively scant attention. ALIA [8] is the closest and most recent work exploring such a data-augmentation-based approach to debiasing, but it has two shortcomings - first, it does not diagnose the dataset which it aims to debias.

[*] Correspondence to: rwiddhi.chakraborty@uit.no

Preprint. Under review.

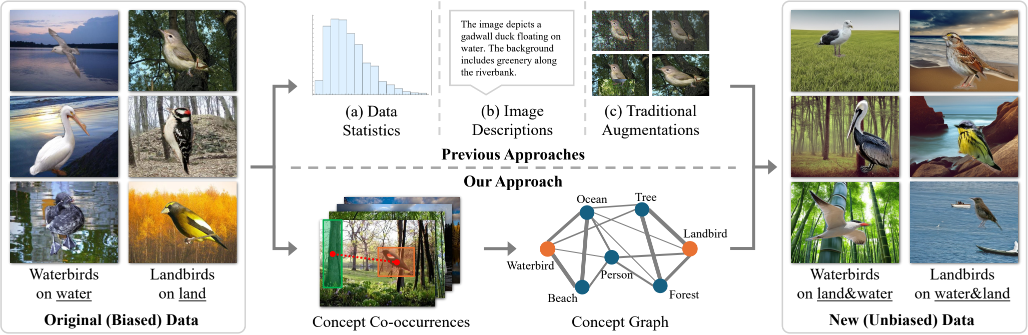


Figure 1: The conventional data diagnosis and augmentation pipeline begins with an original (biased) dataset. Existing methods address these biases via object frequency calibration [52], metadata analysis [8], or traditional augmentation techniques [59, 5]. In contrast, our framework models visual data as a knowledge graph of concepts, with orange nodes representing classes and blue nodes representing concepts, facilitating a systematic diagnosis of class-concept imbalances for debiasing object co-occurrences in vision datasets.

Without such a diagnosis, it is challenging to identify the biases to be mitigated in the first place. Second, the method relies on a large language model (ChatGPT-4 [4]) to generate diverse, unbiased, in-domain descriptions. This approach is potentially confounding since there is no reliable way to ensure that the biases of the large language model itself do not affect the quality of such domain descriptions. In this work, we address both these shortcomings.

We present CONBIAS, our framework for diagnosis and debiasing of visual data. Our key contribution is in representing a visual dataset as a knowledge graph of concepts. Analyzing this graph for imbalanced class-concept combinations leads to a principled diagnosis of biases present in the dataset. Once identified, we generate images to address under-represented class-concept combinations, promoting a more uniform concept distribution across classes. By using a concept graph, we circumvent the reliance on a large language model to generate debiased data. Figure 1 illustrates the core idea of our approach in contrast with existing methods. We target object co-occurrence bias, a human-interpretable issue known to confound downstream tasks [34, 10]. Object co-occurrence bias refers to any spurious correlation between a label and an object causally unrelated to the label. Representing the dataset as a knowledge graph of object co-occurrences provides a structured and controllable method to diagnose and mitigate these spurious correlations.

Our framework proceeds in three steps: (1) *Concept Graph Construction:* We construct a knowledge graph of concepts from the dataset. These concepts are assumed to come from dataset ground truth such as captions or segmentation masks. (2) *Concept Diagnosis:* This stage then analyzes the knowledge graph for concept imbalances, revealing potential biases in the original dataset. (3) *Concept Debiasing:* We sample imbalanced concept combinations from the knowledge graph using graph cliques, each representing a class-concept combination identified as imbalanced. Finally, we generate images containing under-represented concept combinations to supplement the dataset. The image generation protocol is generic and uses an off-the-shelf inpainting process with a text-to-image generative model. This principled approach ensures that the concept distribution in our augmented data is uniform and less biased. Our experiments validate this approach, showing that data augmentation based on a balanced concept distribution improves generalization performance across multiple datasets compared to existing baselines. In summary, our **contributions** include:

- We propose a new concept graph-based framework to diagnose biases in visual datasets, which represents a principled approach to diagnosing datasets for biases, and to mitigating them.
- Based on our graph construction and diagnosis, we propose a novel clique-based concept balancing strategy to address detected biases.
- We demonstrate that balanced concept generation in data augmentation enhances classifier generalization across multiple datasets, over baselines.

2 Related Work

Bias discovery in deep learning models. The identification of biases in trained deep learning models has a rich history, with early works exploring the texture and shape-bias tradeoff in ImageNet-pretrained ResNets [14, 21, 32, 39]. More recently, the field of worst group robustness has emerged, aiming to generalize classifier performance across multiple groups in the data that correspond to known spurious correlations [49, 45, 24, 42]. Debiasing and concept discovery in the feature space of the learned classifier is also common [1, 54, 58]. Testing model performance sensitivity to the presence of particular attributes has also been explored [53, 36]. With the recent rise in popularity of large language models, efforts have been made to identify learned biases using off-the-shelf captioning models [56], adaptive testing [11], and language guidance [19, 37]. Traditional data augmentation approaches such as CutMix [59], and RandAug [5], are used as baselines as well. Our work intervenes on the dataset directly, instead of operating in the model feature space or testing model sensitivity. This allows for a more intuitive and principled approach to bias discovery.

Data diagnosis. Our work is placed in the context of data diagnosis, i.e. identifying biases directly from the data without using the model as a proxy. One of the early influential works expounding the importance of datasets in deep learning research was a systematic review of the popular datasets in computer vision [51]. A modern appraisal categorizing more diverse types of biases in visual datasets exists in [9]. Additionally, works investigating possible issues with dataset labels have also received interest [33, 57]. Data diagnosis tools such as REVISE [52] compute object statistics (including co-occurrence) to generate high-level insights of the data. However, REVISE is not an end-to-end framework that at once diagnoses and debiases data. It is rather an exploratory tool for an overview of common concepts in the dataset. A more recent method, ALIA, uses a language model to populate diverse descriptions of the given dataset, consequently generating images from such descriptions. A more critical look on dataset bias lies in the field of fairness, particularly with regards to societal bias [12, 16]. Finally, benchmark datasets for data diagnosis have also been proposed [29, 28].

Object co-occurrence bias in visual recognition. Objects are biased in the company they keep. This adage is well known in the computer vision literature, as outlined in [34, 10]. Modern efforts to mitigate object co-occurrence bias involve feature decorrelation [48], object aware contrastive representations [31], causal interventions [38], and fusing object and contextual information via attention [2]. The common theme in tackling contextual and co-occurrence bias lies entirely in using better models (feature representations) rather than intervening in the dataset directly. We place our debiasing method along the data augmentation direction, allowing for better controllability and interpretability of the debiasing stage, rather than relying on semantic features learned by a classifier, which may be difficult for humans to interpret.

3 Approach

Figure 2 illustrates the overall pipeline of our method. In this section, we begin with the problem statement in Section 3.1, and move to the three major stages in our method definition. Section 3.2 describes the procedure of concept graph construction. Section 3.3 illustrates the details of concept diagnosis. Finally, Section 3.4 presents our method for concept debiasing.

3.1 Problem Statement

We are given a dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$, a set of images and their corresponding labels. We also assume access to a concept set $C = \{c_1, c_2, \ldots, c_k\}$ that describes unique objects present in the data. An example concept set looks like the following: `{alley, crosswalk, downtown, ..., gas station}`, i.e. a list of unique objects present in each image in addition to the class label. Finally, we are given a classifier $f_\theta(X)$ parameterized by network parameters θ. The central hypothesis of this work is that the class labels exhibit co-occurring bias with the concept set C, affecting downstream task performance. In this light, we wish to generate an augmented dataset D_{aug} that is debiased with respect to the concepts and their corresponding class labels. Thus, given the new dataset $D' = D \cup D_{\text{aug}}$, we wish to retrain $f_\theta(X)$ in the standard classification setup:

$$\hat{f}^* = \arg\min_f \mathbb{E}_{(x,y)\in D'}[\mathcal{L}(y, f_\theta(x))], \tag{1}$$

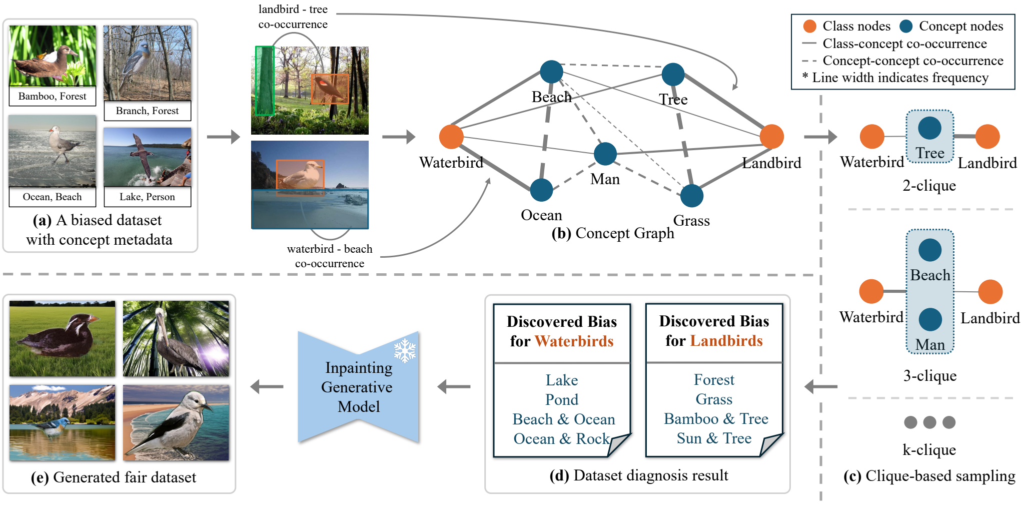

Bamboo, Forest
Branch, Forest
Ocean, Beach
Lake, Person
(a) A biased dataset with concept metadata
landbird - tree co-occurrence
waterbird - beach co-occurrence
Beach
Tree
Waterbird
Man
Landbird
Ocean
Grass
(b) Concept Graph
● Class nodes ● Concept nodes
— Class-concept co-occurrence
- - Concept-concept co-occurrence
* Line width indicates frequency
Waterbird
Tree
Landbird
2-clique
Beach
Waterbird
Landbird
Man
3-clique
k-clique
(c) Clique-based sampling
Discovered Bias for Waterbirds
Lake
Pond
Beach & Ocean
Ocean & Rock
Discovered Bias for Landbirds
Forest
Grass
Bamboo & Tree
Sun & Tree
(d) Dataset diagnosis result
Inpainting Generative Model
(e) Generated fair dataset

Figure 2: **Overview of our framework** ConBias**.** (a) Given a dataset and its concept metadata which contains the objects present in each image, (b) we build the concept graph using object co-occurrences. The line thickness indicates the co-occurrence frequencies of particular concepts with their respective classes. (c) Next, the clique-based sampling strategy generates under-represented class-concept combinations, which yield (d) the dataset diagnosis result. (e) Finally, with biases discovered, we generate images of classes containing under-represented concept combinations in the dataset with a standard text-to-image generative model.

where $\mathcal{L}(y, f_\theta(x))$ is the cross entropy loss between the class label and classifier prediction. Our framework consists of three stages: *Concept Graph Construction*, *Concept Diagnosis*, and *Concept Debiasing*. Next, we provide details on each step.

3.2 Concept Graph Construction

We construct a concept graph $G = (V, E, W)$ from the data, where $|V|$ is the node set of the graph, $|E|$ is the edge set, and $|W|$ is the set of weights for each edge in the graph. We first construct the node set V as a union of the label set Y and concept set C:

$$V = Y \cup C.$$

Next, we construct the edge set E:

$$E = \{(i, j) \mid \exists \, \text{image } D_k \text{ such that both } i \text{ and } j \text{ appear in } D_k\}.$$

Finally, we construct the weight set W by computing the weights w_{ij} for each edge (i, j) in G:

$$w_{ij} = \sum_{n=1}^{N} \mathbb{I}(i \in D_n \text{ and } j \in D_n),$$

where $\mathbb{I}$ is the indicator function that returns 1 if both i and j are present in the n-th image in D, and 0 otherwise, and N is the total number of images in the dataset.

The concept graph G encapsulates co-occurrence counts between nodes, thus providing an alternative representation of the (originally visual) data. As we show in the next section, this representation helps uncover novel imbalances (bias) contained in the dataset.

3.3 Concept Graph Diagnosis

In the previous section, we define how to build the concept graph. Here, we present how to leverage the concept graph for discovering co-occurrence biases. We present a principled approach to discovering concept-combinations across classes that co-occur in an imbalanced fashion.

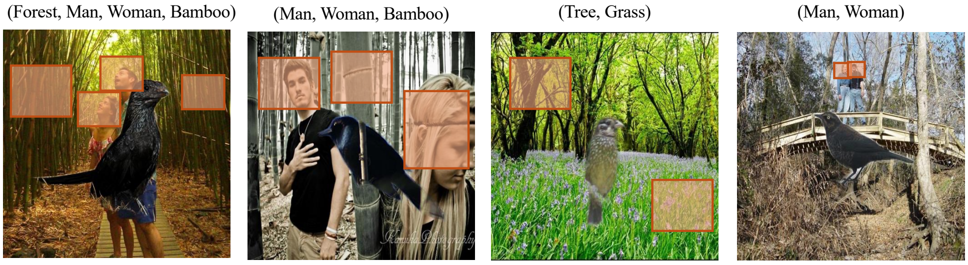


Figure 3: Examples of concept clique sets for `Landbird` class in Waterbirds dataset uncovered by our diagnosis. Concepts such as `Tree, Forest, Man, Woman, Bamboo` are overwhelmingly associated with this class, indicating strong co-occurrence bias. All these concepts are causally unrelated to the bird type.

Definition (Class Clique Sets) For each class $Y_i \in Y$, we construct a set of k-cliques using the concept graph G. The set of all possible k-cliques for class Y_i is denoted as $\mathcal{K}_i^k$:

$$\mathcal{K}_i^k = \{\{c_{j_1}, c_{j_2}, \ldots, c_{j_k}\} \mid c_{j_1}, c_{j_2}, \ldots, c_{j_k} \in C \text{ and } j_1 < j_2 < \ldots < j_k\},$$

where $j_1, j_2, \ldots$ are the indices of concepts in C. Then, $\mathcal{K}_i$ for class Y_i can be successfully constructed for $k = 1, 2, \ldots, K$, where K is the size of the largest clique in G containing Y_i. We construct class clique sets for every class in the dataset. An illustration of concept cliques in the Waterbirds dataset that help in bias diagnosis is provided in Figure 3.

Definition (Common Class Clique Sets) Given $\mathcal{K}_i$ for each class, we then compute the cliques common to all classes. These are the cliques of interest, whose imbalances we want to investigate:

$$\mathcal{K} = \bigcap_i \mathcal{K}_i,$$

where $\mathcal{K}$ encapsulates all common cliques enumerated across the dataset for all classes. Refer to Figure 2 for a broad illustration of the k-clique set construction from the concept graph G.

Definition (Imbalanced Common Cliques) Given the set of common cliques across all classes $\mathcal{K}$, we compute the imbalanced class-concept combinations, i.e. the imbalanced clique set I:

$$I_{[\mathcal{K}]_{m=1}^{M}} = \{(|F_{\mathcal{K}_{y_i}^m} - F_{\mathcal{K}_{y_j}^m}|, \arg\min(F_{\mathcal{K}_{y_i}^m}, F_{\mathcal{K}_{y_j}^m}))\}, \forall i, j,$$

where $F_{\mathcal{K}_{y_i}^m}$ and $F_{\mathcal{K}_{y_j}^m}$ indicates the co-occurrence frequency of concepts in clique m with respect to class y_i and y_j respectively, and the $\arg\min$ operator identifies the underrepresented class for the particular concept clique. Thus, each element in I is a number representing the imbalance of each common clique across all classes. For the special case where the size of clique m is 1, this equates to simply looking up the value w_{ij} in G. For the case where the size of $m > 2$, it is straightforward to compute the co-occurrence of class y_i with respect to concepts in m:

$$w_{ij\ldots k} = \sum_{n=1}^{N} \mathbb{I}(i \in D_n \text{ and } j \in D_n \ldots \text{ and } k \in D_n),$$

for each image D_n in the data. The set I holds rich information about the data. In addition to holding the imbalanced counts of concept combinations across all classes, the set I also holds which is the *underrepresented* class with respect to a particular concept clique.

Intuitively, concept combinations that are common across all the classes, but do *not* co-occur uniformly across the classes are likely biased concept combinations. We provide an example from the *Waterbirds* dataset in Figure 4. The training set in *Waterbirds* is intentionally biased to the background: 95% of landbirds appear with land backgrounds, and 95% of waterbirds appear with water backgrounds. First, we find common cliques of varying sizes across the classes `(Landbird, Waterbird)`. One example of a common clique of size 3 is `(Landbird, Beach, Ocean)` and `(Waterbird, Beach, Ocean)`. We compute the co-occurrence of `(Landbird, Beach, Ocean)` and `(Waterbird, Beach, Ocean)` from the extracted metadata, and the imbalance is clear. Since waterbirds are far more prone to appear on water, there are significantly more images of waterbirds containing concepts `Beach` and `Ocean` than landbirds, which are more prone to be in land-based environments. If we look at the co-occurence of `Landbird` with a land-based concept such as `Grass`, we see the opposite imbalance. There are significantly more images of landbirds containing trees over waterbirds. Similarly, for the water-based concept of `Ocean`, we see a strong imbalance towards the `Waterbird` class. In our debiasing stage, we should therefore generate more images of waterbirds with tree-based backgrounds, and landbirds with beach/water-based backgrounds. Using the clique-based approach, we have successfully uncovered the known background bias in the Waterbirds dataset. This approach is generalizable to multiple classes. All we need are common cliques, and the computation of concept co-occurrences across the dataset. In this way, our concept graph approach uncovers interesting concept combinations across the *whole* dataset that appear in an imbalanced and spurious fashion. More examples of such imbalances are provided in the supplementary material.

3.4 Concept Graph Debiasing

We have, to this point, constructed a knowledge graph of the visual data, and diagnosed it for concept-based co-occurrence biases. Once the imbalanced clique set I is identified in G, we debias the data by generating images containing under-represented concepts across classes.

Recall that $I = \{f_i, Y_i\}$ inherently holds the underrepresented class Y_i and the frequency f_i by which the original dataset needs to be adjusted with new images of class Y_i with respect to concept clique i. Following the example in the previous section, we notice that the concepts `(Beach, Ocean)` are significantly over-represented in the `Waterbird` class than `Landbird`. Similarly, the concept `Tree` is significantly over-represented with the `Landbird` class than the `Waterbird` class. As a result, we sample f_i instances of these under-represented cliques with respect to their classes, and prompt a text-to-image generative model for more images of the `Waterbird` class with the concept `Tree`. Similarly, we would prompt the model to generate images of Landbird with the concept `Beach, Ocean`. We generate images for all class based imbalances following this upsampling protocol.Typical prompts for our image-inpainting model would look like: `An image of a ocean and a beach`, `An image of a tree`, `An image of a forest`, etc. We use an inpainting-based method to make sure that the original object is not modified in the image, and that the new concepts are only injected into the non-object space in the image. See the supplementary material for the generated images and the prompts.

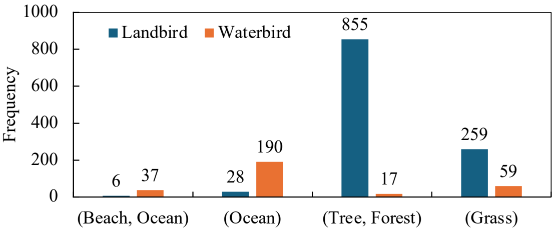


Figure 4: **Examples of concept imbalances in the Waterbirds dataset.** We show the frequencies of concepts cliques as discovered in the dataset. We see imbalances across not only single concepts (e.g., `Ocean`, `Grass`) but also concept combinations (e.g., `(Beach, Ocean)`, `(Tree, Forest)`). These are the biases we aim to mitigate for the downstream task.

Using this upsampling protocol, we generate a set of images that leads to our augmented, debiased dataset D_{aug}. The original training data D can now be augmented using this data, and the classifier $f_\theta(X)$ can be retrained on the dataset $D \cup D_{\text{aug}}$. In the next section, we conduct experiments on three datasets to demonstrate our method's significant improvements of baselines.

4 Experiments

We validate our method on vision datatset diagnosis and debiasing across various scenarios. We begin by introducing the experimental setup including the datasets, baselines, tasks, and implementation details in Section 4.1. Section 4.2 presents the main results of our proposed framework, ConBias, compared with state-of-the-art methods. Finally, Section 4.3 details ablation studies and analyses.

4.1 Setup

Datasets. We use three datasets in our work: Waterbirds [45], UrbanCars [22], and COCO-GB [50], that are commonly used in the bias mitigation domain. We tackle background bias in the Waterbirds dataset, background and co-occuring object bias in the UrbanCars dataset, and finally gender bias in

Table 1: **State-of-the-art comparison on different datasets.** Results are averaged over three training runs. **CB**: class balanced split. **OOD**: out-of-distribution split. Binary class classification accuracy is used as the metric. Our method outperforms previous approaches across multiple datasets.

Method	**Waterbirds** [45]		**UrbanCars** [22]		**COCO-GB** [50]	
	CB ↑	OOD ↑	CB ↑	OOD ↑	CB↑	OOD ↑
Baseline [17]	67.1	44.9	73.5	40.5	58.5	51.9
+ RandAug [5]	73.7	60.2	76.3	46.1	55.8	50.2
+ CutMix [59]	67.9	45.6	74.4	39.3	57.4	51.2
+ ALIA [8]	69.6	48.2	74.0	42.5	58.7	**52.4**
+ ConBias (Ours)	**77.9**	**69.3**	**78.3**	**52.9**	**58.8**	51.4

COCO-GB. All the tasks are binary classification tasks. More details on the training splits and class labels are provided in the supplementary material.

Baseline methods. Our baselines are include a vanilla Resnet-50 model pre-trained on ImageNet, two typical data augmentation based debiasing methods: (1) CutMix, a technique where we cut and paste patches between different training images to generate diverse discriminative features, and (2) RandAug, which creates random transformations on the training data during the learning phase. Finally, we compare against the recently proposed and state-of-the-art ALIA[8], which uses a large language model to generate diverse, in-distribution prompts for a text-to-image generative model.

Evaluation protocols. We compute the mean test accuracy over the class-balanced test data and the out-of-distribution (OOD) test data, similar to [8]. The class-balanced data contains an even distribution of classes and their respective spurious correlations, while the OOD data contains counterfactual concepts. For example, in Waterbirds dataset, for the class-balanced test data 50% images of Landbirds have Land backgrounds, while 50% images of Waterbirds have Water backgrounds. The OOD test set contains Landbirds on Water, and Waterbirds on Land. More details on the test sets are presented in the supplementary material.

Implementation details. We use existing implementations to train our models. Our Base model is a Resnet-50 pretrained on ImageNet [17]. We generate the same number of images per data-augmentation protocol to ensure a fair comparison. For comparison with ALIA on Waterbirds, we directly use their generated dataset available here. For the other datasets, we used the existing ALIA implementation to generate the augmented data. Following previous work, we use validation loss based checkpointing to choose the best model, the Adam optimizer with a learning rate of 10^{-3}, a weight decay of 10^{-5}, and a cosine learning schedule over 100 epochs. To generate images, we use Stable Diffusion [44] with a CLIP [40]-based filtering mechanism to ensure reliable image generation. Finally, we inpaint the object onto the generated image using ground truth masks (available for all datasets). All code was written in PyTorch [35].

4.2 Main Results

In Table 1 we present the main results, averaged over three training runs. First, we note that for Waterbirds and UrbanCars, we observe significant improvements in both the Class-Balanced and OOD test sets over the typical augmentation methods such as CutMix and RandAug. Second, we note the significant improvement in performance over the most recent state-of-the-art augmentation method, ALIA. Third, for COCO-GB, while we notice slightly smaller difference in the CB and OOD accuracies between our method and the baselines, our hypothesis is that this happens because of limited number of samples used for augmentation. ALIA uses a confidence based filtering mechanism to remove generated samples. This leads to a small final number of 260 samples to be added for the retraining part. In the ablation section, we show this hypothesis to be true, and further demonstrate that on adding more images for the retraining step, we progressively increase the performance gap between our method and the baselines. These three observations taken together validate the usefulness of our approach. The next section provides additional insights on the usefulness of our method and the effect of ablating its components.

Table 2: **Benefit of the graph structure in ConBias.** Leveraging the graph structure is beneficial as opposed to simply computing single concept-class frequency counts on UrbanCars.

Model	**CB ↑**	**OOD ↑**
Base	73.4	40.4
Base + ALIA	74.0	42.5
Base (BG)	78.5	51.9
Base (CoObj)	77.0	47.3
Base (Both)	78.1	51.3
Base (ConBias)	**79.4**	**53.2**

Table 3: **Performance for the IP2P variant** of ConBias with respect to base, ALIA, and our original model on Waterbirds. Our method significantly improves over ALIA even when using IP2P, although the best results are still achieved when using the stable diffusion based inpainting protocol.

Models	**CB ↑**	**OOD ↑**
Base	67.1	44.9
Base + ALIA	69.6	48.2
Base + ConBias (IP2P)	72.9	60.5
Base + ConBias	**77.9**	**69.3**

4.3 Ablations and Analyses

We further analyze our method along five axes: (1) The usefulness of the graph structure, (2) Robustness of our method to other evaluation metrics, (3) The impact on CB and OOD performance by increasing the number of added samples for the retraining step, (4) The usefulness of discovered concepts by our method on the trained classifier, and (5) The impact of the generative component in our work compared to ALIA, since the latter uses InstructPix2Pix [3] while we use a Stable Diffusion based inpainting protocol.

Effect of the graph structure. Recalling the definition of Class Clique Sets, in principle one could only use cliques sizes of 1, i.e., the direct neighbors of each class node. This would be equivalent to computing the frequency of co-occurrence over a single hop neighborhood of the class node in the graph. In this ablation we show that one should use larger cliques, i.e. leverage the graph structure, instead of a simple direct neighborhood based frequency calculation. We trained three separate models on three different types of D_{aug}: Ours (BG), trained on images containing only background shortcuts, Ours (CoObj), images containing only the co-occurrence shortcuts, and Ours (Both), images containing both shortcuts, but *not simultaneously*.

Table 2 shows the results. First, our approach of leveraging the graph structure provides improvement over simply using the frequency of a 1-hop neighborhood. Second, we note that *all* the methods outperform the baseline and ALIA, which shows that incorporating frequency based co-occurrences is in a broader sense much more useful than relying on diverse prompts generated by ChatGPT-4, which is the approach taken by ALIA.

Robustness to evaluation metrics. The CB and OOD test accuracies test for generalization capabilities, but more direct evaluators of shortcut learning exist in the literature. In [22], for instance, the authors propose (i) The *ID Accuracy* - which is the accuracy when the test set contains common background and co-occuring objects, (ii) The *BG-GAP* - which is the drop in *ID accuracy* when the test set contains common co-occurring objects, but uncommon background objects, (iii) The *CoObj-GAP*, which is the drop in *ID accuracy* when the test set contains uncommon co-occurring objects, but common background objects, and finally (iv) The *BG + CoObj GAP*, which is the case when both background and co-occurring objects are uncommon in the test set. A multiple shortcut mitigation method should *minimize* the *BG + CoOBj GAP* metric, and also make sure it does not exacerbate any shortcut that the base model relies on. In Table 4, we present results of Base, Base (BG), Base (CoObj), Base(Both), and Base (ConBias) on these metrics for UrbanCars. We are able to post the lowest drops among all baselines on the *CoObj-GAP* and *BG + CoOBj GAP* metrics, suggesting mitigation of multiple shortcut reliance. This places our method in a more realistic context, as it is infeasible to assume that real world data will only have a single type of bias in them.

Scaling the number of images in D_{aug}. In Table 1, we commented on the fact that our method provides marginal improvement over the baselines in the COCO-GB dataset. Our hypothesis was that this was due to the low number of images in the augmented dataset. In Figure 5, we demonstrate the impact of adding more images to D_{aug} for retraining. Clearly, our method benefits from this protocol, leading to significant differences over ALIA as we keep increasing the number of images. Note that, infinite enrichment is not recommended and has been found to be detrimental to classifier performance, as progressive addition of synthetic images will likely lead to addition of out-of-distribution examples

Table 4: **Robustness of our method to evaluation metrics** In addition to CB and OOD performance, we also report metrics evaluating multiple shortcut mitigation. Results on UrbanCars (Average of three training runs).

Model	BG-GAP ↑	CoObj-GAP ↑	BG+CoObj GAP ↑
Base	-11.2	-21.5	-54.8
Base (BG)	**-5.0**	-19.4	-38.0
Base (CoObj)	-6.3	-19.2	-47.3
Base (Both)	-5.6	-23.2	-47.6
Base (ConBias)	-6.0	**-18.4**	**-41.4**

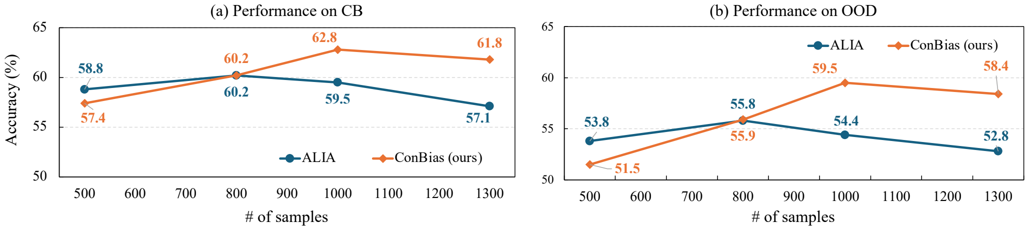


Figure 5: **Performance on COCO-GB.** We show the accuracies on (a) Class-Balanced (CB) and (b) Out-of-Distribution (OOD) splits. We observe that increasing number of images in D_{aug} improves performance up to a certain point (1000 images).

in the training data. This explains why, after an inflexion point, the accuracy suffers from adding more images. Similar observations have been made in [8] and [18].

Discovered concept imbalances and feature attributions. To verify that the model indeed debiases the imbalanced concepts that our method discovers, we present GradCAM [46] attributions of the model predictions after retraining. In Figure 6, we show results on all datasets. While other methods frequently focus on the spurious feature , ConBias helps the model focus only on the relevant, object level features of the data.

The impact of the generative model. ALIA uses an InstructPix2Pix (IP2P) based generation procedure, while we use stable diffusion with a mask-inpainting procedure to make sure the objects remain consistent in the image. To ablate the effect of the generation, we present results of our method with IP2P as the generative model instead, on Waterbirds dataset, in Table 3. First, we note that even with IP2P as the generative component, we are able to outperform ALIA, which suggests that it is actually the superior quality of our concept discovery method that leads to the improved results. Second, our inpainting based method outperforms our IP2P based method, which we argue is due to the objects being preserved in the generated image, as opposed to traditional image editing methods, where the object may transform arbitrarily, hurting the quality of augmented data.

5 Conclusion, Limitations, and Future Works

While ConBias is the first end-to-end pipeline to both diagnose and debias visual datasets, there are some limitations: First, that the enumeration of cliques grows exponentially with the size of the graph. For larger real world graphs, there could be more efficient strategies to find the concept combinations. Second, in this work we focus on biases emanating out of object co-occurrences. A variety of other biases exist in vision datasets, and future work would look to address the same. We add an extended section on broader impact of our work in the supplementary material. In summary, datasets in the real world are biased, and the exponential increase in dataset sizes over the past decade amplifies the challenge of investigating model and dataset biases. While both dataset and model diagnosis are exciting areas of research, an end-to-end diagnosis and debiasing pipeline such as ConBias offers a principled approach to diagnosing and debiasing visual datasets, in turn improving downstream classification performance. Our state-of-the art results open up numerous interesting possibilities for future work - incorporating more novel graph structures, and diagnosis under the regime where concept sets may be wholly or partially unavailable, remain interesting directions to pursue.

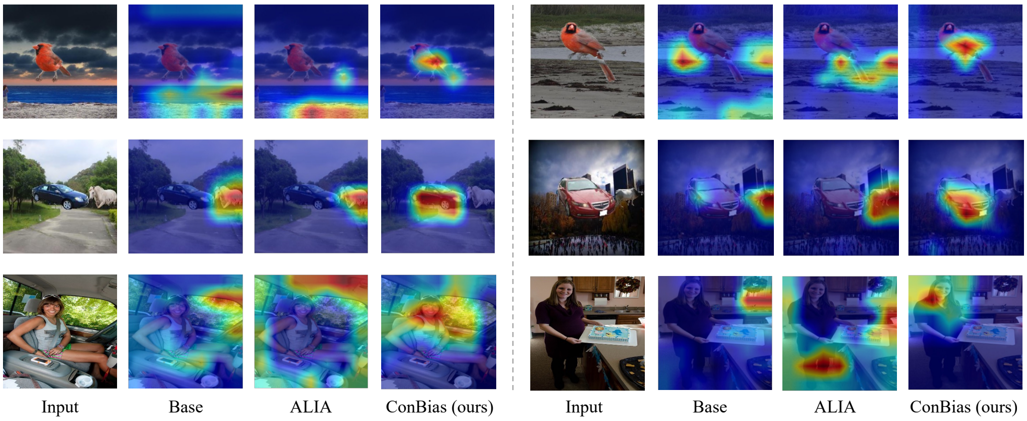


Figure 6: **Visualization of the heatmaps for different methods.** Top row: Waterbirds. Middle row: UrbanCars. Bottom row: COCO-GB. Our method enforces the base model to focus on only the relevant features in the data, and removing reliance on shortcut features, i.e. the background for Waterbirds, the background and co-occurring object for UrbanCars, and the gender for COCO-GB.

References

[1] Md Rifat Arefin, Yan Zhang, Aristide Baratin, Francesco Locatello, Irina Rish, Dianbo Liu, and Kenji Kawaguchi. Unsupervised concept discovery mitigates spurious correlations. *arXiv preprint arXiv:2402.13368*, 2024.

[2] Philipp Bomatter, Mengmi Zhang, Dimitar Karev, Spandan Madan, Claire Tseng, and Gabriel Kreiman. When pigs fly: Contextual reasoning in synthetic and natural scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 255–264, 2021.

[3] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *NeurIPS*, 2020.

[5] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[8] Lisa Dunlap, Alyssa Umino, Han Zhang, Jiezhi Yang, Joseph E Gonzalez, and Trevor Darrell. Diversify your vision datasets with automatic diffusion-based augmentation. *Advances in Neural Information Processing Systems*, 36, 2024.

[9] Simone Fabbrizzi, Symeon Papadopoulos, Eirini Ntoutsi, and Ioannis Kompatsiaris. A survey on bias in visual datasets. *Computer Vision and Image Understanding*, 223:103552, 2022.

[10] Carolina Galleguillos, Andrew Rabinovich, and Serge Belongie. Object categorization using co-occurrence, location and appearance. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[11] Irena Gao, Gabriel Ilharco, Scott Lundberg, and Marco Tulio Ribeiro. Adaptive testing of computer vision models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4003–4014, 2023.

[12] Noa Garcia, Yusuke Hirota, Yankun Wu, and Yuta Nakashima. Uncurated image-text datasets: Shedding light on demographic bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6957–6966, 2023.

[13] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[14] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019.

[15] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14953–14962, 2023.

[16] Laura Gustafson, Chloe Rolland, Nikhila Ravi, Quentin Duval, Aaron Adcock, Cheng-Yang Fu, Melissa Hall, and Candace Ross. Facet: Fairness in computer vision evaluation benchmark. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20370–20382, 2023.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[18] Ruifei He, Shuyang Sun, Xin Yu, Chuhui Xue, Wenqing Zhang, Philip Torr, Song Bai, and XIAOJUAN QI. Is synthetic data from generative models ready for image recognition? In *The Eleventh International Conference on Learning Representations*, 2022.

[19] Younghyun Kim, Sangwoo Mo, Minkyu Kim, Kyungmin Lee, Jaeho Lee, and Jinwoo Shin. Bias-to-text: Debiasing unknown visual biases through language interpretation. *arXiv preprint arXiv:2301.11104*, 2023.

[20] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023.

[21] Sangjun Lee, Inwoo Hwang, Gi-Cheon Kang, and Byoung-Tak Zhang. Improving robustness to texture bias via shape-focused augmentation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 4322–4330, 2022.

[22] Zhiheng Li, Ivan Evtimov, Albert Gordo, Caner Hazirbas, Tal Hassner, Cristian Canton Ferrer, Chenliang Xu, and Mark Ibrahim. A whac-a-mole dilemma: Shortcuts come in multiples where mitigating one amplifies others. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20071–20082, 2023.

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

[24] Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning*, pages 6781–6792. PMLR, 2021.

[25] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.

[26] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

[27] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[28] Aengus Lynch, Gbètondji JS Dovonon, Jean Kaddour, and Ricardo Silva. Spawrious: A benchmark for fine control of spurious correlation biases. *arXiv preprint arXiv:2303.05470*, 2023.

[29] Mark Mazumder, Colby Banbury, Xiaozhe Yao, Bojan Karlaš, William Gaviria Rojas, Sudnya Diamos, Greg Diamos, Lynn He, Alicia Parrish, Hannah Rose Kirk, et al. Dataperf: Benchmarks for data-centric ai development. *Advances in Neural Information Processing Systems*, 36, 2024.

[30] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM computing surveys (CSUR)*, 54(6):1–35, 2021.

[31] Sangwoo Mo, Hyunwoo Kang, Kihyuk Sohn, Chun-Liang Li, and Jinwoo Shin. Object-aware contrastive learning for debiased scene representation. *Advances in Neural Information Processing Systems*, 34:12251–12264, 2021.

[32] Chaithanya Kumar Mummadi, Ranjitha Subramaniam, Robin Hutmacher, Julien Vitay, Volker Fischer, and Jan Hendrik Metzen. Does enhanced shape bias improve neural network robustness to common corruptions? In *International Conference on Learning Representations*, 2020.

[33] Curtis Northcutt, Lu Jiang, and Isaac Chuang. Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70:1373–1411, 2021.

[34] Aude Oliva and Antonio Torralba. The role of context in object recognition. *Trends in cognitive sciences*, 11(12):520–527, 2007.

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[36] Gregory Plumb, Marco Tulio Ribeiro, and Ameet Talwalkar. Finding and fixing spurious patterns with explanations. *Transactions on Machine Learning Research*, 2022.

[37] Viraj Prabhu, Sriram Yenamandra, Prithvijit Chattopadhyay, and Judy Hoffman. Lance: Stress-testing visual models by generating language-guided counterfactual images. *Advances in Neural Information Processing Systems*, 36, 2024.

[38] Wei Qin, Hanwang Zhang, Richang Hong, Ee-Peng Lim, and Qianru Sun. Causal interventional training for image recognition. *IEEE Transactions on Multimedia*, 25:1033–1044, 2023.

[39] Xinkuan Qiu, Meina Kan, Yongbin Zhou, Yanchao Bi, and Shiguang Shan. Shape-biased cnns are not always superior in out-of-distribution robustness. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2326–2335, 2024.

[40] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[41] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[42] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International conference on machine learning*, pages 4334–4343. PMLR, 2018.

[43] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, Zhaoyang Zeng, Hao Zhang, Feng Li, Jie Yang, Hongyang Li, Qing Jiang, and Lei Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 2024.

[44] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[45] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks. In *International Conference on Learning Representations*, 2019.

[46] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

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

[48] Krishna Kumar Singh, Dhruv Mahajan, Kristen Grauman, Yong Jae Lee, Matt Feiszli, and Deepti Ghadiyaram. Don't judge an object by its context: Learning to overcome contextual bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11070–11078, 2020.

[49] Nimit Sohoni, Jared Dunnmon, Geoffrey Angus, Albert Gu, and Christopher Ré. No subclass left behind: Fine-grained robustness in coarse-grained classification problems. *Advances in Neural Information Processing Systems*, 33:19339–19352, 2020.

[50] Ruixiang Tang, Mengnan Du, Yuening Li, Zirui Liu, Na Zou, and Xia Hu. Mitigating gender bias in captioning systems. In *Proceedings of the Web Conference 2021*, pages 633–645, 2021.

[51] A Torralba and AA Efros. Unbiased look at dataset bias. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1521–1528, 2011.

[52] Angelina Wang, Alexander Liu, Ryan Zhang, Anat Kleiman, Leslie Kim, Dora Zhao, Iroha Shirai, Arvind Narayanan, and Olga Russakovsky. Revise: A tool for measuring and mitigating bias in visual datasets. *International Journal of Computer Vision*, 130(7):1790–1810, 2022.

[53] Angelina Wang and Olga Russakovsky. Overwriting pretrained bias with finetuning data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3957–3968, 2023.

[54] Bowen Wang, Liangzhi Li, Yuta Nakashima, and Hajime Nagahara. Learning bottleneck concepts in image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10962–10971, 2023.

[55] Zhenyu Wang, Yali Li, Xi Chen, Ser-Nam Lim, Antonio Torralba, Hengshuang Zhao, and Shengjin Wang. Detecting everything in the open world: Towards universal object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11433–11443, 2023.

[56] Olivia Wiles, Isabela Albuquerque, and Sven Gowal. Discovering bugs in vision models using off-the-shelf image generation and captioning. In *NeurIPS ML Safety Workshop*, 2022.

[57] Kaiyu Yang, Klint Qinami, Li Fei-Fei, Jia Deng, and Olga Russakovsky. Towards fairer datasets: Filtering and balancing the distribution of the people subtree in the imagenet hierarchy. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 547–558, 2020.

[58] Sriram Yenamandra, Pratik Ramesh, Viraj Prabhu, and Judy Hoffman. Facts: First amplify correlations and then slice to discover bias. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4794–4804, 2023.

[59] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.

Visual Data Diagnosis and Debiasing with Concept Graphs

Supplementary Material

Anonymous Author(s)
Affiliation
Address
email

In this supplementary materials, we provide details and additional results omitted in the main text.

- Section A: Broader Impact.
- Section B: Dataset details - Splits and image examples.
- Section C: Full Concept Set for each dataset.
- Section D: Dataset Imbalances.
- Section E: The Generative Model.
- Section F: Examples of Generated Images using ConBias.
- Section G: Results with standard deviations.
- Section H: Compute details and runtime.

A Broader Impact

Fairness in AI is rapidly gaining priority in current research as models and datasets grow exponentially larger, thus making it more and more complicated to diagnose them for biases. It is imperative to focus on understanding and mitigating biases learned by models, and inherent biases in the data, to ensure reliable and transparent predictions in the real world. The advent of generative models in particular, including large language models, and image generative models, invites new questions into how to reliably regulate such technologies. These models are trained on datasets in the order of hundreds of billions of data points. How do we ensure that problematic aspects of the data do not pass onto the models learning from them? How do we ensure that models do not generate synthetic data that is potentially harmful, misleading, and misinformative in nature? How do we evaluate the quality of generated data by such models? These are the pressing questions that our research direction is interested in.

B Dataset Details

We use three datasets in our work - Waterbirds [8], UrbanCars [4], and COCO-GB [9].

For Waterbirds, the class labels are *Landbird, Waterbird*. The Waterbirds dataset has the background bias, i.e. 95% images of landbirds have land-based backgrounds, and 95% images of waterbirds have water-based backgrounds. For the concept set annotations, we use the captions extracted by authors of [2] captions available here.

For UrbanCars, the class labels are *Urban, Country*, defining the type of car. There are multiple biases in UrbanCars - (1) Background Bias, i.e. Urban cars appear with 95% correlation with urban backgrounds, and Country cars appear with 95% correlation with country backgrounds. (2)

Co-Occurring object, i.e. Urban cars appear with 95% correlation with urban objects, and Country cars appear with 95% correlation with country objects.

For COCO-GB, the class labels are *Man, Woman*. The bias for the dataset are the set of objects in the MS-COCO dataset [5]. The authors of [9] find a strong bias of most objects in the data with respect to the "Man" class, and design a secret, gender-balanced test set to evaluate gender bias in classifiers.

B.1 Waterbirds

In Fig A1 we present examples from the Waterbirds training data. The classes are heavily biased to the backgrounds, i.e. Landbirds on Land, Waterbirds on Water.

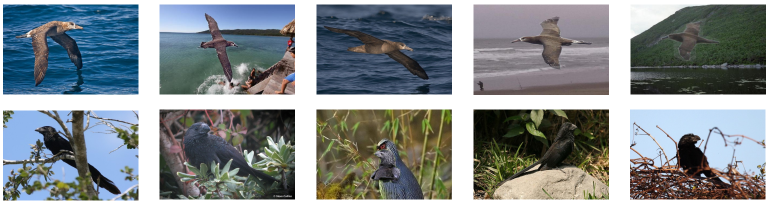


Figure A1: Examples of training data in Waterbirds dataset. Waterbirds (Top) are 95% biased towards water backgrounds, while Landbirds (Bottom) are 95% biased towards land backgrounds.

B.2 UrbanCars

In Fig A2 we present examples from the UrbanCars training data. The classes are heavily biased to multiple shortcuts - Background and Co-Occurring objects.

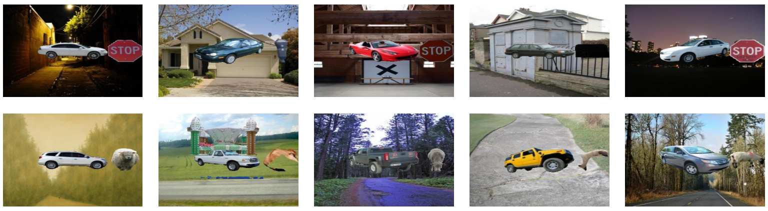


Figure A2: Examples of training data in UrbanCars dataset. Urban cars (Top) are 95% biased towards urban backgrounds and urban co-occurring objects. Country cars (Bottom) are 95% biased towards country backgrounds and country co-occurring objects.

B.3 COCO-GB

In Fig A3 we present examples from the COCO-GB training data. The "Man" class is known to be heavily biased in MS-COCO to everyday objects.

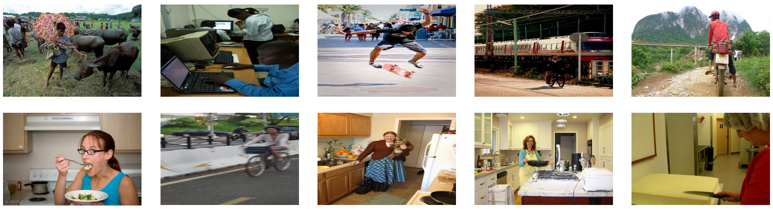


Figure A3: Examples of training data in MS-COCO dataset. Images of men are heavily biased towards common, everyday objects, as opposed to women. Authors of [9] find over a 90% in all object correlations towards men.

B.4 Splits

In Table A1 we present the train, validation, and test splits for our three datasets.

Dataset	Train	Test	Validation
Waterbirds	4795	1199	5794
UrbanCars	8000	1000	1000
COCO-GB	32582	1331	1000

Table A1: Dataset sizes for Train, Test, and Validation sets

C Concept Sets

In Table A2 we present the full concept sets for each dataset. The Waterbirds dataset has 64 unique concepts, the UrbanCars dataset has 17 unique concepts, and COCO-GB has 81 unique concepts, all from the MS-COCO dataset. Note that both MS-COCO and UrbanCars have ground truth concepts, while for Waterbirds, we use the extracted captions here.

Dataset	Concepts
Waterbirds	`duck, pond, tree, grass, post, ocean, bridge, surfer, surfboard, beach, people, forest, beak, sailboat, bamboo, sunlight, boy, foot, boat, mountain, seagull, field, rock, crab, wall, woman, cell phone, man, wing, deer, leaf, backpack, hillside, statue, display, wave, lake, pen, palm tree, shirt, sign, bamboo forest, grass plant, tree branch, bushes, horse, sidewalk, parrot, sun, cup, town, snowy forest, red eye, twig, wooden fence, path, penguin, fishing rod, pelican, kayak, wine glass, lighthouse, mountain landscape, wooden path`
UrbanCars	`alley, crosswalk, downtown, gas station, garage-outdoor, driveway, forest road, field road, desert road, fireplug, stop sign, street sign, parking meter, traffic light, cow, horse, sheep`
COCO-GB	`stop sign, tie, knife, car, bicycle, fire hydrant, cow, motorcycle, umbrella, sports ball, cat, surfboard, elephant, skateboard, skis, backpack, couch, bed, wine glass, carrot, cup, airplane, handbag, cake, cell phone, woman, refrigerator, potted plant, sandwich, vase, chair, bus, frisbee, parking meter, bench, horse, truck, snowboard, train, clock, keyboard, scissors, man, bottle, kite, traffic light, book, dining table, sheep, fork, spoon, tennis racket, dog, bowl, suitcase, boat, donut, baseball bat, orange, toothbrush, banana, oven, laptop, toilet, sink, pizza, mouse, baseball glove, tv, teddy bear, hot dog, broccoli, remote, bird, microwave, apple, zebra, bear, toaster, giraffe, hair drier`

Table A2: Concepts for Waterbirds, UrbanCars, and COCO-GB datasets

D Dataset Imbalances

In this section we shed more insight into what sort of concept imbalances ConBias discovers. These object level insights are also, to the best of our knowledge, the first of its kind, shedding more light on the secret co-ocurrence biases hidden in data.

D.1 Waterbirds

In addition to the main paper, we list some other category imbalances in Waterbirds in Figure A4. Some of these extreme imbalances appear in diverse 2-clique/3-clique combinations. For example, we see that concepts like `forest, man, woman` are significantly biased towards the `Landbird` class, while concepts like beach, man, sun, lake, mountain are biased towards the Waterbird class. This is the background bias that is known in the Waterbirds dataset, that ConBias successfully uncovers.

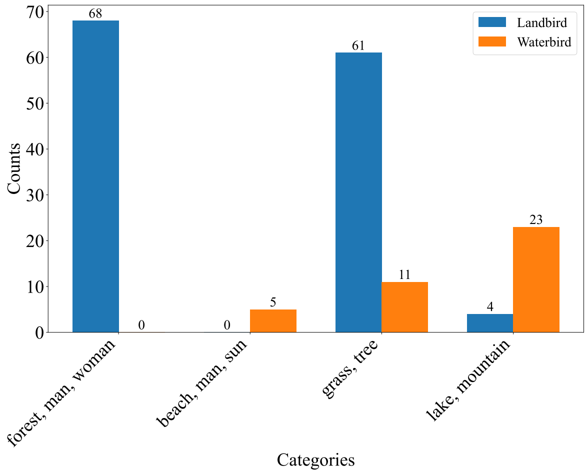


Figure A4: Extreme imbalance of particular concepts in Waterbirds dataset, as discovered by ConBias.

D.2 UrbanCars

In UrbanCars, the class labels (country car, urban car) are intentionally biased towards background and co-occurring objects. In Figure A5, we see that there exists an extreme imbalance betwee urban concepts such as driveway, traffic light towards urban cars, and country concepts such as forest road, field road, cow, horse towards country cars. These are exactly the background and co-occurring biases in the construction of the data, that ConBias successfully uncovers.

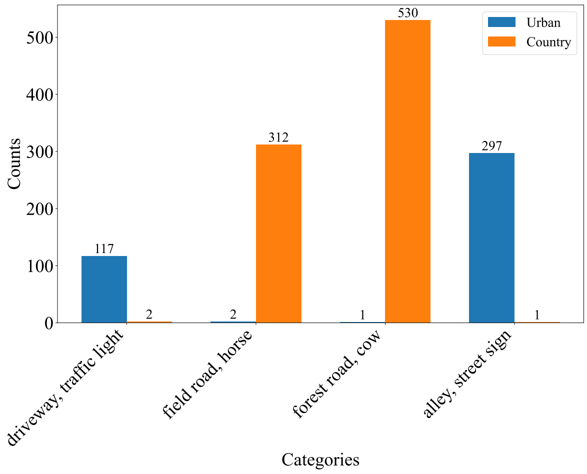


Figure A5: Extreme imbalance of particular concepts in UrbanCars dataset, as discovered by ConBias.

D.3 COCO-GB

The gender bias in COCO-GB has been extensively studied in [9]. In Figure A6, we show the extreme imbalance towards specific concepts in the MS-COCO dataset. Concepts such as baseball bat, sports ball, motorcycle, truck overwhelmingly correlate with images of men, which may be problematic for the classifier to learn.

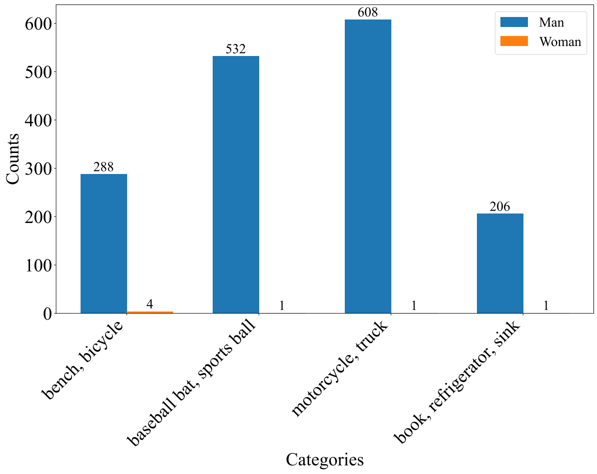


Figure A6: Extreme imbalance of particular concepts in MS-COCO dataset, as discovered by ConBias.

E Generative Model

Here we present more details of our generative model. We use Stable Diffusion based inpainting, as illustrated in Figure A7. Given the prompt, we first generate an image using Stable Diffusion [7]. Next, using ground truth masks of the object, we paste the object at the foreground of the generated image. In this way, we preserve the original object in the image, which is a challenge for traditional image editing methods such as InstructPix2Pix. We believe the inpainting method is a more principled approach to synthetic image generation, particularly if the downstream task is classification in nature.

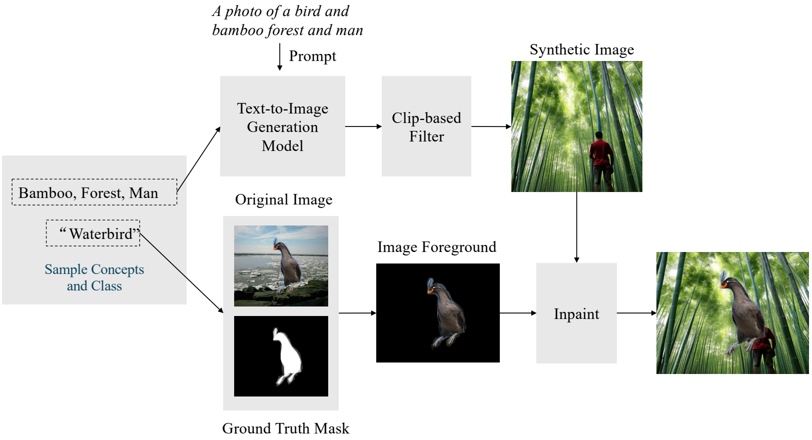


Figure A7: Image generation Pipeline: Given concepts to be upsampled as discovered ConBias, we sample the concept combinations and images from the class to be upsampled. We prompt Stable Diffusion for an image containing such concepts. We extract the object of interest using ground truth masks, and inpaint the object over the generated image. This ensures that the object features are not harmed during generation. We use a CLIP-based scoring filter to make sure the generated image contains the concepts requested in the prompt. We have found a score of 0.6 to be satisfactory as a threshold.

The generation process of a single image takes the followings as input: The sampled concept combination and the class for which this concept combination needs to be generated. The output of the generative model is the final image with the specified concepts in the background and an instance of the specified category in the foreground.

The process will first transform the concept list [concept 1, concept 2, . . . , concept N] into a prompt: "a photo of concept 1, concept 2, . . . , and concept N." The prompt is then passed into the text-to-image generation model (stable diffusion) to get the generated image as background. We apply a clip-score filtering after the generation process to only keep the images with a CLIP-score over 0.6 to make sure that the generated images can accurately represent the concept list. Next, the process will sample an image of the specified category from the original dataset, and use the mask to segment out the desired object. Finally, inpainting is performed to clip the desired object as foreground onto the generated image to obtain the final image.

F Generated Images by ConBias

In this section we present examples of synthetic data generated by ConBias for Waterbirds, UrbanCars, and COCO-GB.

F.1 Waterbirds

In Figure A8 we present diverse images generated by ConBias for the two classes of Landbird and Waterbird. Due to the bias diagnosis stage where we found the overwhelming correlation between landbirds with land based backgrounds such as tree, forest, field, grass, etc, and waterbirds with water based backgrounds such as beach, ocean, boat, etc, ConBias was automatically able to decide which concept combinations to use to generate new, debiased images.

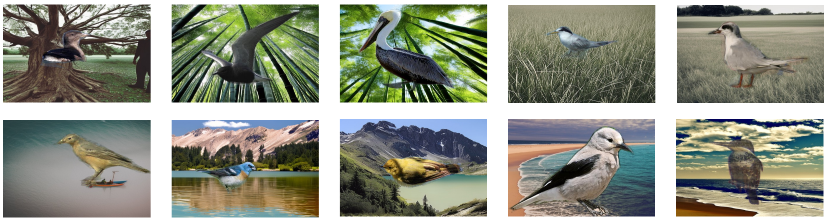


Figure A8: (Top) Generated images of waterbirds with land-based backgrounds. (Bottom) Generated images of landbirds with water-based backgrounds, as discovered by ConBias. Note the consistency in object preservation.

F.2 UrbanCars

In Figure A9 we present diverse images generated by ConBias for the two classes of Urban and Country cars. Due to the bias diagnosis stage, we were able to discover the overwhelming correlation between urban cars with urban based backgrounds such as gas station, driveway, alley, etc and urban co-occurring objects such as fireplug, stop sign, etc. Similarly, for country cars, we discovered bias towards country backgrounds such as desert road, field road, forest road, and, and country co-occurring objects such as cow, sheep, horse. As a result, ConBias helps generate urban cars with country based backgrounds and co-occurring objects, and vice versa.

F.3 COCO-GB

In Figure A10 we present diverse images generated by ConBias for the two classes of Man and Woman in COCO-GB. In this dataset, we were able to discover significant under-representation of women with respect to common, everyday objects in the MS-COCO dataset. Some examples include `skateboard, motorcycle, car, truck`, etc. These objects could have gendered assumptions and it is imperative for debiased datasets to have uniform representation across classes for such concepts.

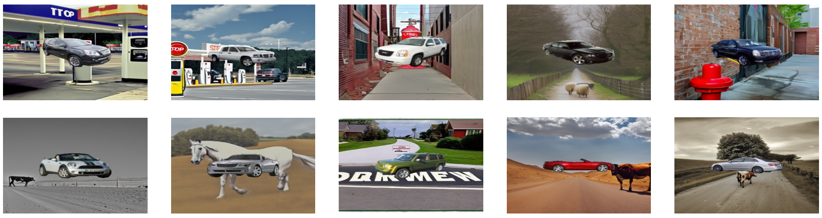


Figure A9: (Top) Generated images of country cars with urban-based backgrounds and co-ocurring objects. (Bottom) Generated images of urban cars with urban-based backgrounds and co-occurring objects, as discovered by ConBias. Note the consistency in object preservation.

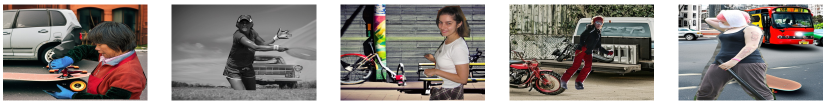


Figure A10: Generated images of COCO-GB using everyday, common objects that are discovered to be biased towards men by ConBias. Example concepts include `skateboard, motorcycle, truck, sports ball`, etc. Note the consistency in object preservation.

We would also like to bring to the attention of our readers the successful nature of the inpainting procedure. We are able to consistently preserve the *class label* of interest in the synthetic images. This is imperative to ensure that the generative pipeline does not create unreasonable objects that make it infeasible for the classifier to learn.

G Confidence Intervals

In Table A3 we present the averaged results with standard deviations over three training runs. For both Waterbirds and UrbanCars, our improvements are large and significant. For COCO-GB, while originally did not observe statistically significant results, in the main paper we showed that increasing the number of images in D_{aug} leads to significant improvements over the baselines.

Table A3: **State-of-the-art comparison on different datasets.** Results are averaged over three training runs. **CB**: class balanced split. **OOD**: out-of-distribution split. Binary class classification accuracy is used as the metric. ConBias outperforms previous approaches across multiple datasets. Standard deviations included.

Method	**Waterbirds** [8]		**UrbanCars** [4]		**COCO-GB** [9]	
	CB	OOD	CB	OOD	CB	OOD
Baseline [3]	67.1 ± 0.5	44.9 ± 0.8	73.5 ± 0.6	40.5 ± 0.8	58.5± 0.7	51.9± 0.7
+ RandAug [1]	73.7 ± 0.8	60.2± 0.7	76.3 ± 0.8	46.1 ± 0.9	55.8± 0.4	50.2 ± 0.6
+ CutMix [10]	67.9 ± 0.7	45.6 ± 0.7	74.4± 0.7	39.3 ± 0.9	57.4± 0.5	51.2± 0.6
+ ALIA [2]	69.6 ± 1.2	48.2 ± 1.0	74.0 ± 0.9	42.5 ± 0.9	58.7± 0.4	**52.4**± 0.6
+ ConBias (ours)	**77.9** ± 0.9	**69.3** ± 0.8	**78.3** ± 0.7	**52.9**± 0.7	**58.8** ± 0.6	51.4 ± 0.4

H Compute Details

We trained all models on a single NVIDIA RTX A4000 and used PyTorch [6] for all experiments. With the early stopping cosine learning scheduled described in the main paper, we observed fast training times, with 90 minutes for three runs on Waterbirds and UrbanCars, and 180 minutes for three runs on COCO-GB.

References

[1] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.

[2] Lisa Dunlap, Alyssa Umino, Han Zhang, Jiezhi Yang, Joseph E Gonzalez, and Trevor Darrell. Diversify your vision datasets with automatic diffusion-based augmentation. *Advances in Neural Information Processing Systems*, 36, 2024.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[4] Zhiheng Li, Ivan Evtimov, Albert Gordo, Caner Hazirbas, Tal Hassner, Cristian Canton Ferrer, Chenliang Xu, and Mark Ibrahim. A whac-a-mole dilemma: Shortcuts come in multiples where mitigating one amplifies others. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20071–20082, 2023.

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

[6] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[7] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[8] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks. In *International Conference on Learning Representations*, 2019.

[9] Ruixiang Tang, Mengnan Du, Yuening Li, Zirui Liu, Na Zou, and Xia Hu. Mitigating gender bias in captioning systems. In *Proceedings of the Web Conference 2021*, pages 633–645, 2021.

[10] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.

/13

Paper IV

On Disentangled Representations and the Oversmoothing Problem in Graph Convolutional Networks

Rwiddhi Chakraborty, Benjamin Ricaud, Robert Jenssen, Michael Kampffmeyer

[1]Department of Physics and Technology, UiT The Arctic University of Norway
firstname[.middle initial].lastname@uit.no

Abstract

Disentangled graph convolutional models have recently demonstrated superior performance on a variety of datasets. It has been hypothesized that the improved performance emanates from a mitigation of the oversmoothing problem, a fundamental problem in graph based learning. However, no systematic study exists to verify this hypothesis. In this work, we show that this hypothesis is dependent on a tradeoff between the graph structure and feature representations in the data. We propose a Dirichlet Energy-based estimator that effectively quantifies this tradeoff, and demonstrate that disentanglement mitigates oversmoothing on datasets with informative graph structure, as opposed to datasets with informative node features. Finally, we leverage an intuitive experimental framework to illustrate the importance of explicitly taking into account this tradeoff to mitigate the oversmoothing problem in graph convolutional networks. [1]

Introduction

Graph Convolutional Networks(GCNs) have emerged as the popular choice of learning algorithms for graph-structured data (Kipf and Welling 2016; Velickovic et al. 2017; Fan et al. 2019; Gao et al. 2023). The graph propagation typical in GCNs is a weighted aggregation of features over node neighborhoods, defined over the (usually) symmetric normalized adjacency matrix. This inductive bias has successfully led to impressive performance on learning tasks such as graph classification (Lee, Rossi, and Kong 2018; Liang et al. 2021; Chen et al. 2022), node classification (Zhang et al. 2022; Zhao, Zhang, and Wang 2021; Ji et al. 2023), and even in multi-modal learning, where graph and image- (or text) based data have been used in conjunction for diverse embeddings (Chen et al. 2019b; Chen, Zou, and Chen 2022; Zhou et al. 2022; Yuan et al. 2023).

However, the inductive bias of GCNs renders them susceptible to the *oversmoothing problem*(Li, Han, and Wu 2018), where features from distant neighborhoods get more similar to each other as the number of layers increases. While this is useful for learning distinct class representations, an excess of this effect has been shown to have adverse consequences on classification tasks, as features of nodes from different classes in distant neighborhoods may mix together, leading to sub-optimal representations. This has led to a challenge in building deep graph convolutional models, and over the years, many works have aimed at mitigating this issue, leading to deeper models and lesser oversmoothing (Zhao and Akoglu 2019; Wu et al. 2022; Zheng, Fu, and He 2021; Rusch, Bronstein, and Mishra 2023). In addition, measures to calculate the degree of oversmoothing have also been proposed, allowing for better interpretability of the effect (Chen et al. 2019a; Zhou et al. 2021).

Recently, the introduction of a family of disentangled graph convolutional models (Ma et al. 2019; Li et al. 2021, 2023) has led to impressive performance on tasks in semi-supervised learning. Such models leverage mechanisms to separate the GCN feature space into distinct chunks, based on the assumption that a node is connected to its neighborhood due to distinct factors. While it has been hypothesized that the performance improvement and oversmoothing mitigation stems from the use of disentangled representations (Ma et al. 2019; Guo et al. 2022), no systematic study exists to verify this hypothesis as such.

In this work, we shed more insight on the effect of disentangled representations on oversmoothing in GCNs. We find that, in addition to the process of disentanglement, the datasets being used are also of immense importance when studying the effect of oversmoothing - particularly with respect to the graph structure and node feature representations, both of which are inherent properties of the data. To study the effect of disentanglement on graph data, we design a novel estimator using the Dirichlet Energy, which allows us to quantify the relative importance of graph structure with respect to feature representations in graph propagation. This estimator, the *Latent Dirichlet Energy* (LDE) ratio, is an intuitive and useful metric for two reasons:

- It allows us to demonstrate that in datasets with low LDE ratios, where the graph structure is relatively more important, disentangled models are successful in mitigating the oversmoothing effect. However, in datasets with high LDE ratios where the graph structure does not hold useful information, disentanglement proves to be unsuccessful in mitigating oversmoothing.
- Our estimator allows us to observe and quantify a tradeoff between graph structure and feature representations in the dataset. Based on these observations, we showcase a learning framework that explicitly leverages this tradeoff, and helps alleviate the oversmoothing problem

[1] We will open source our code for better reproducibility.

in datasets with high LDE ratios, leading to more stable results when increasing the number of layers.

Our work is similar in spirit to (Abel, Benami, and Louzoun 2019), but crucial differences exist, particularly that this work does not involve the the use of disentangled models, and it does not study the use of disentangling the feature representations and graph topology with respect to the oversmoothing problem. Our proposed framework provides insights into the relationship between datasets, disentangled models, and the oversmoothing problem, in addition to how more stable aggregated results on increasing layers can be achieved by explicitly learning the graph structure and feature representation tradeoff.

In summary, our contributions include:

1. The first systematic study of the effect of disentangled representations on oversmoothing in GCNs.
2. The design of a novel, Dirichlet-Energy based estimator that quantifies the tradeoff between graph structure informativeness and node feature informativeness in graph datasets.
3. Showcasing a novel experimental framework that explicitly leverages such a tradeoff, leading to a learning mechanism that alleviates oversmoothing on challenging datasets.

Related Work

Oversmoothing in GCNs

Oversmoothing in graph neural networks was first demonstrated in (Li, Han, and Wu 2018), where it was shown that the propagation rule in a standard GCN was a smoothing (weighted mean) operation equivalent to damping the symmetric normalized laplacian of the signal. As the number of layers in a GCN network increases, the weighted aggregation of k-hop nodes rendered more and more features from nodes with different classes to be similar to each other, adversely affecting classification performance. Over the years, many mitigation techniques have been proposed (Oono and Suzuki 2019; Zhao and Akoglu 2019; Liu, Gao, and Ji 2020), that aim to build deeper GCNs without adversely affecting performance. In addition to mitigating techniques, proxy metrics to directly measure the effect of oversmoothing on graph networks have also been proposed (Chen et al. 2019a; Zhou et al. 2021). We use the Mean Average Distance (MAD) in our work, which is a measure that indicates how close the features from different nodes are.

Disentangled Representations

The DisenGCN (Ma et al. 2019) proposed a neighborhood routing mechanism to disentangle node representations into k latent factors. The assumptions were that nodes are connected to their neighborhoods as a result of distinct, disentangled relations, and the neighborhood routing mechanism aimed at uncovering these relations. The neighborhood mechanism outputs node representations that are disentangled into k-distinct factors. Using the DisenGCN as a backbone, IPGDN (Liu et al. 2019) adds an additional HSIC based pairwise independence criterion on the latent factors. The independence constraint further forces disentangled factors to assume orthogonality, leading to better separation between features, and discriminative quality. The LGD-GCN (Guo et al. 2022) builds a knn graph out of the features and propagates as an additional step on the disentangled representations. This is done to reinforce the feature representations over the graph structure. All these baselines improve the richness of feature representations, and therefore are useful to study in relation to how they affect oversmoothing in different datasets. We use these three disentangled models in our systematic study.

Table 1: Dataset Statistics

Datasets	Cora	Citeseer	Pubmed	Flickr	Blogcatalog
Nodes	2708	3327	19717	7575	5196
Edges	5429	4732	44338	239738	171743
Features	1433	3703	500	12047	8189
Classes	7	6	3	9	6
Train	140	120	60	757	519
Validation	500	500	500	1515	1039
Test	1000	1000	1000	5303	3638

Table 2: The different Dirichlet energy ratios computed for each dataset. The social networks have systematically higher values and give evidence for a stronger oversmoothing problem.

	Cora	Citeseer	Pubmed	Flickr	Blogcatalog
s	0.95	1.84	0.04	3.23	3.17
s_{ER}	0.92	0.83	0.84	1.01	1
s_{BA}	0.94	0.83	0.84	0.99	0.98

Preliminaries

We briefly introduce some notation to ensure consistency across the work. Next, we discuss the basics of disentangled models, and the metric for calculating the degree of oversmoothing in node representations.

Notation

A Graph $G = (V, E)$ is defined over two sets, V, and E, the vertex set and edge set respectively. The edge set E defines the adjacency matrix $A \in \mathbb{R}^{n \times n}$, where $n = |V|$. Each node $v \in V$ has a feature representation $x_v \in \mathbb{R}^d$. For convenience, we denote the feature *matrix* $X \in \mathbb{R}^{n \times d}$ that collects the feature representations for all nodes. For G, we denote the degree matrix as D, the normalized adjacency matrix $\hat{A} = A + I$ and normalized degree matrix $\hat{D} = D + I$. The graph laplacian is $L = D - A$.

Neighborhood Routing in Disentangled GCNs

The disentanglement mechanism proceeds first by a k-subspace projection, and second, an expectation-maximisation (EM) framework to update and aggregate the node representations in the k-disentangled spaces. Given each node x_i and the latent factor z_{ik}, the subspace projection is defined:

$$z_{ik} = \frac{\sigma(w_k^T x_i + b_k)}{\|z_{ik}\|_2} \tag{1}$$

where σ is a non-linearity, and w_k the weights of each latent factor. Next the EM framework updates the probabilities that nodes are connected *due* to latent factor k. For a particular node i, a neighboring node j, and the edge list of the graph E:

$$p_{jk} = \text{softmax}\left(\frac{z_{jk} \cdot c_k}{\tau}\right) \tag{2}$$

where c_k is initialized with z_{ik} and updated iteratively:

$$c_k = z_{ik} + \sum_{j \in E} p_{jk} z_{jk} \tag{3}$$

Eq (2) and (3) define the EM framework and the neighborhood routing mechanism, the output of which is a feature matrix of nodes with k-disentangled latent factors.

Mean Average Distance

In addition to mitigating techniques, proxy metrics to directly measure the effect of oversmoothing on graph networks have also been proposed. One well used metric is the Mean Average Distance (MAD), which computes how similar features get to each other as graph propagation continues.

Given a layer representation $X \in \mathbb{R}^{n \times d}$, where n is the number of nodes, d the number of features, the distance matrix $\bar{D} \in \mathbb{R}^{n \times n}$ is calculated using cosine similarity

$$\bar{D}_{ij} = 1 - \frac{X_{i,:} \cdot X_{j,:}}{|X_{i,:}| \cdot |X_{j,:}|} \tag{4}$$

Then, the MAD is computed as the global mean of the matrix $\bar{D}$ in Eq 4. We use the MAD in our work to calculate the degree of oversmoothing in the different models.

Disentangled Graph Convolutional Models and Oversmoothing

While previous works have hypothesized that disentangled models mitigate the oversmoothing problem (Ma et al. 2019; Liu et al. 2019; Guo et al. 2022), there has been no systematic study to verify the claim. In this section, we begin our investigation into the relationship between disentangled models and the oversmoothing phenomenon. We introduce a novel Dirichlet energy-based estimator, the LDE ratio, that allows us to quantify the oversmoothing potential in a graph dataset.

Next, we conduct a preliminary empirical study[2], where we produce aggregated results (of ten runs) on commonly used datasets in the graph learning literature. We observe a clear pattern that emerges - Disentanglement helps mitigate oversmoothing on datasets with low LDE ratios, while disentanglement suffers from strong oversmoothing on datasets with high LDE ratios (Figure 1). The insights from our novel estimator provide clarity on when disentanglement is useful, and when it may not be enough.

Estimating the Oversmoothing Potential

We propose using the Dirichlet energy as a natural estimator to capture the smoothing tendency in a graph. Our estimator, the LDE ratio, reports the Dirichlet energy of a dataset relative to a random Erdos-Renyi graph. The intuition for doing this is two-fold: First, the Dirichlet energy is a well used estimator of feature smoothness over a graph (Smith et al. 2017) and captures the effect of Laplacian smoothing in general (Zhou et al. 2021; Cai and Wang 2020), and second, we compare the result to a baseline containing no structural information, a random graph with the same number of node and edges as in the tested graph. The ratio thus allows us to capture how much the graph topology plays a role in the evaluation, as opposed to the feature representations, since a random graph has no information in its topology at all. We use common datasets in the graph learning literature, whose details are in Table 1.

Given the graph Laplacian L, and a single feature f on the nodes of the graph, the Dirichlet energy is defined as $s_f = (f^T L f)/\|f\|^2$, where $\|f\|$ is the l2 norm of f . This quantity captures the variations of the feature values over the graph. A high value of the Dirichlet energy for a given graph means large variations of its features between connected nodes. This signals that node neighborhoods may not be connected due to similar features, which is an implicit assumption of the disentangled models. We compare different graphs with multiple features per node and different number of edges. We generalize the definition in our context. Given $\bar{X}$ the feature matrix X where each column has been normalized (l2-norm), the mean Dirichlet energy per edge of the feature data as

$$s = \frac{\text{Tr}(\bar{X}^T L \bar{X})}{m},$$

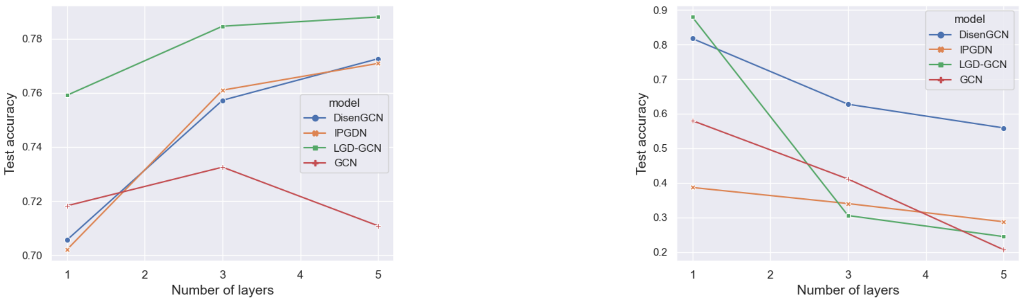
where m is the number of edges , and $\text{Tr}(.)$ is the trace function. Further, we want to evaluate the impact of the graph structure to feature variations by comparing to a random model. For each graph, we compute the ratio s_r relative to the Dirichlet energy of a random graph with the same number of nodes but randomly assigned edges. Therefore, the Dirichlet energy ratio s_r is defined as

$$s_r = \frac{\text{Tr}(\bar{X}^T L \bar{X})}{\text{Tr}(\bar{X}^T L_\text{r} \bar{X})}, \tag{5}$$

and L_r is the Laplacian of the random graph. We use 2 random graphs in the experiments, an Erdos-Renyi graph and a Barabasi-Albert graph giving s_ER and s_BA respectively.

Intuitively, a small s_r suggests that the graph structure connects neighbors with similar feature values more than a *random graph*, meaning that the graph topology is potentially useful in representation learning and less susceptible to oversmoothing. On the contrary, the case where s_r is close

[2] For details, see the Experiments section.



(a) Low LDE

(b) High LDE

Figure 1: Aggregated performance of 10 runs on the low LDE and high LDE datasets

to one shows that the graph does not contain more information than a random graph, which would make it susceptible to oversmoothing. We report the complete LDE ratios in Table 2. Flickr and Blogcatalog have a higher value for the LDE ratios, some close or even surpassing one, suggesting less informative graph structure. On the contrary, Cora, Citeseer, and Pubmed have relatively lower LDE ratios, all less than one, suggesting more informative graph structure.

Analysis of Oversmoothing in Disentangled Models

The LDE ratios in the previous section suggest that disentangled models on datasets with high LDE ratios may be prone to stronger oversmoothing. In this first analysis section, we tackle the following research question - When does disentanglement mitigate oversmoothing? In Figure 1, we present aggregated results separately on the low LDE ratio data - Cora, Citeseer, Pubmed(Sen et al. 2008), and the high LDE data - Flickr, BlogCatalog(Huang, Li, and Hu 2017). Individual results for the interested reader can be found in the supplementary material. Typically, oversmoothing is indicated by a sharp drop in accuracy as the number of layers increases. In Figure 1a, for the low LDE ratio data, we do not observe an adverse drop in performance for the disentangled models, suggesting a mitigation against the oversmoothing issue.

However, for the high LDE data in Figure 1b, we observe a strong oversmoothing effect for all the models. Despite having the best overall performance with a single layer, the drop is significant for LGD-GCN with 3 or more layers. While the DisenGCN is relatively more stable than the other models, it still cannot avoid the oversmoothing issue. These results are interesting as Cora, Citeseer, Pubmed belong to the so called *citation datasets*, while Flickr and BlogCatalog belong to the so called *social media datasets*. The poor performance of the disentangled models when increasing layers on social media data indicates that the graph based aggregation leads to strong oversmoothing. We recall that disentangled models operate on the assumption that each node in a graph is connected to its neighborhood as a result of distinct factors, and that similar nodes would display similar distinct factors. Intuitively, then, it would seem that the performance of disentangled models would be adversely affected if node neighborhoods did not hold useful information, i.e. the graph structure was not sufficiently informative. A citation dataset contains links between academic publications - it would seem natural that papers on similar topics would be connected to each other in the graph. For social media data (eg: Flickr), users may be connected to each other and yet belong to multiple groups at the same time, making the graph structure and label relationship more complex. These results help answer our first research question - Disentanglement helps mitigate oversmoothing in datasets with low LDE ratio, but is adversely affected by oversmoothing in datasets with high LDE ratio, i.e. datasets with insufficient structural information.

The Tradeoff between Graph Structure and Feature Representations

The results for our first analysis indicate a tradeoff between the information held in node feature representations, and the information held in the graph structure. Since the oversmoothing effect was considerably strong on the social media data, such an observation immediately leads to our second analysis - Can a learning mechanism explicitly leverage this tradeoff to mitigate oversmoothing?

Learning the Tradeoff between Graph Structure and Feature Representations

The results from the previous section indicate that disentangled models are susceptible to stronger oversmoothing in graph datasets where the graph structure is not sufficiently meaningful, as opposed to node feature representations. The results also point to the possible benefit of leveraging a tradeoff between the graph structure and feature representations during the learning process. To investigate this hypothesis, we propose an experimental framework that uses two branches in its training process, followed by a fusion operation. To encode the information in the graph structure, we can simply use the original adjacency matrix A_g (the *topology branch*) of the graph dataset. To encode feature affinity, we construct a k-nn graph A_f of the feature representations in the original graph (the *feature branch*), similar to LGD-GCN, but a crucial difference exists - The feature aggregation on the knn graph occurs *in parallel*, as opposed to *in sequence* for the LGD-GCN, which does not have this explicit tradeoff in its learning process.

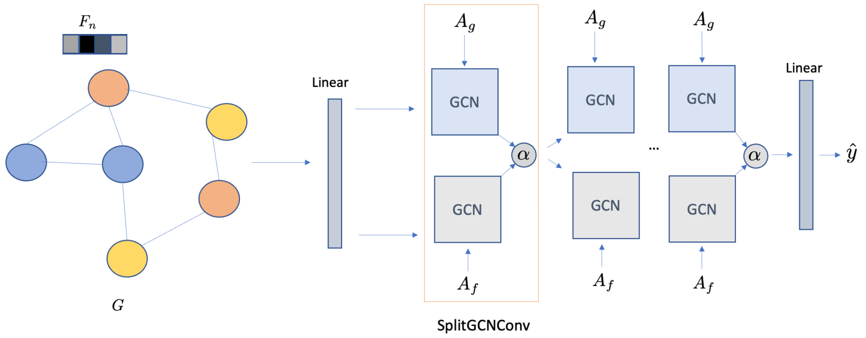


Figure 2: SplitGCN Architecture: Learning the tradeoff between feature representations and graph topology. Given the graph G, and the feature matrix X, the layers marked with A_f receive the k-nn graph as the input adjacency matrix, which we call the *feature branch*. The layers marked with A_g receive the original adjacency matrix as input, which we call the *topology branch*. The outputs from each layer are fused with the learnable parameter α, and passed as input to the next layer. The region marked in orange is the generic SplitGCNConv layer that we propose.

We have two model variants: The SplitDisenGCN, where the topology branch propagates on a DisenGCN layer, and the SplitGCN, where the topology branch operates on a GCN layer. In addition to learning the tradeoff between the topology (structure) and feature representations, these variants allow us to isolate the effects of *disentanglement* and the *knn graph*, shedding insight on which factor plays the key role in oversmoothing. To implement the tradeoff, we introduce the fusion parameter α that is learnable. This allows the network to optimally learn the weights needed to assign to each branch at each layer. The SplitGCN, illustrated in Figure 2, helps split the feature learning and graph learning through two separate paths. We show that a simple weighted combination of the feature and topology branches, leads to more stable results. Note that this framework is not meant to be a state-of-the-art model. It is designed to demonstrate that oversmoothing can be mitigated if the tradeoff between graph structure and node feature representations is explicitly accounted for during the learning process. In this way, we provide a synergistic discourse on the relationships between datasets, disentangled graph convolutional models, and the oversmoothing problem.

The SplitGCNConv Layer

Given the adjacency matrix $A_g \in \mathbb{R}^{n\times n}$, the input feature matrix $X \in \mathbb{R}^{n\times d}$, the output at a particular GCN layer l, parameterized by Θ^l is

$$z_g^{l+1} = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}_g\hat{D}^{-\frac{1}{2}}X^l\Theta^l). \tag{6}$$

Similarly, given the knn feature matrix $A_f \in \mathbb{R}^{n\times n}$, the input feature matrix $X \in \mathbb{R}^{n\times d}$, the output at a particular GCN layer l, parameterized by $\tilde{\Theta}^l$ is

$$z_f^{l+1} = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}_f\hat{D}^{-\frac{1}{2}}X^l\tilde{\Theta}^l). \tag{7}$$

The generalized SplitGCNConv layer is a weighted combination of the outputs z_g and z_f in Equations (6) and (7)

$$\hat{z}^{l+1} = \alpha_l \cdot z_g^{l+1} + (1-\alpha_l)\cdot z_f^{l+1}. \tag{8}$$

Equation (8) represents the output of a single SplitGCNConv layer. For the SplitDisenGCN variant, Eq. (6) is replaced with the disentanglement process described in Eq. (1), (2), and (3). This operation is repeatable over many layers. Over the course of training, the parameter α *learns* the optimal weights to be assigned to each branch, reflecting the differing importance of the features and graph topology. It is important to note the difference between such a joint-learning based, dynamic architecture, over a disentangled framework. For example, in the LGD-GCN, the current state-of-the-art disentangled model, at each node the neighbors are assigned and contribute to different latent factors in the node representation depending on their feature values. For SplitGCN, nodes are connected if they have similar feature vectors wherever they are on the graph, they do not need to be neighbors on the original graph. Moreover, in SplitGCN, the parameter α allowing to combine the graph and feature point of views is a learned real valued number. As the results will show, such modifications significantly stabilize overall model performance.

Experiments

Setup

Our baselines are the DisenGCN(Ma et al. 2019), IPGDN (Liu et al. 2019), and LGD-GCN(Guo et al. 2022). For all the models, we use publicly available implementations, and the best hyperparameters as shared by the original papers. Where hyperparameters were not available, we used a Bayesian hyperparameter search and then evaluated on the best values on the validation sets. We use the same dataset splits and hyperparameters as used in (Guo et al. 2022) to

Table 3: Aggregate performance on all datasets (Rounded to 2 decimal places)

	Blogcatalog			Flickr		
Model	1 layer	3 layers	5 layers	1 layer	3 layers	5 layers
GCN	0.66 ± 0.00	0.61 ± 0.02	0.32 ± 0.02	0.50 ± 0.00	0.21 ± 0.00	0.10 ± 0.01
DisenGCN	0.86 ± 0.01	0.68 ± 0.01	0.62 ± 0.02	0.77 ± 0.01	0.57 ± 0.01	0.49 ± 0.01
IPGDN	0.43 ± 0.02	0.38 ± 0.01	0.36 ± 0.03	0.34 ± 0.02	0.30 ± 0.03	0.21 ± 0.02
LGD-GCN	0.91 ± 0.01	0.38 ± 0.06	0.31 ± 0.05	0.85 ± 0.01	0.23 ± 0.04	0.18 ± 0.01
SplitGCN (ours)	0.91 ± 0.00	0.84 ± 0.01	0.66 ± 0.03	0.78 ± 0.01	0.75 ± 0.01	0.73 ± 0.01
Split-Disgcn (ours)	0.87 ± 0.00	0.89 ± 0.01	0.85 ± 0.01	0.79 ± 0.00	0.75 ± 0.01	0.65 ± 0.01
	Cora			Citeseer		
Model	1 layer	3 layers	5 layers	1 layer	3 layers	5 layers
GCN	0.77 ± 0.00	0.79 ± 0.02	0.76 ± 0.02	0.66 ± 0.01	0.66 ± 0.01	0.65 ± 0.02
DisenGCN	0.71 ± 0.01	0.80 ± 0.01	0.82 ± 0.02	0.65 ± 0.01	0.69 ± 0.01	0.71 ± 0.01
IPGDN	0.71 ± 0.02	0.81 ± 0.01	0.82 ± 0.03	0.63 ± 0.02	0.69 ± 0.03	0.70 ± 0.02
LGD-GCN	0.79 ± 0.01	0.83 ± 0.06	0.84 ± 0.05	0.71 ± 0.01	0.73 ± 0.04	0.70 ± 0.01
SplitGCN (ours)	0.78 ± 0.01	0.78 ± 0.01	0.71 ± 0.03	0.68 ± 0.01	0.67 ± 0.01	0.62 ± 0.02
Split-Disgcn (ours)	0.72 ± 0.01	0.78 ± 0.01	0.80 ± 0.00	0.67 ± 0.01	0.70 ± 0.01	0.71 ± 0.01

calculate the aggregate performance. For the SplitGCN, we run a Bayesian hyperparameter sweep on the the number of neighbors to be used in the k-nn graph to build A_f, the learning rate, dropout, and weight decay. We train for 500 epochs, but use early stopping wherever appropriate. For exact values, see supplementary material. To ensure fair comparisons, we use the best overall hyperparameters when evaluating the results over the layers. The implementation is in PyTorch (Paszke et al. 2019) and PyG(Fey and Lenssen 2019).

Results

We present the results on the low LDE and high LDE datasets in Table 3 (see supplementary for Pubmed results). For the low LDE data, we observe that disentanglement does indeed mitigate the oversmoothing, with improving model accuracy over increasing layers. This is expected, since the graph structure is informative in this case. It is interesting to note the benefit of disentanglement between SplitDisGCN and SplitGCN. Without disentanglement, performance is adversely affected for all three datasets for five layers in the case of SplitGCN, while for the SplitDisGCN, this is not the case. For the high LDE data, where we originally observed the strong oversmoothing by exclusively using the disentangled models, we observe significantly improved performance on increasing the number of layers (when comparing across models comprising the same number of layers), *both* for the SplitGCN and SplitDisGCN. This demonstrates the importance of reinforcing the knn feature graph based propagation, as opposed to using only the graph structure. These results also validate the intuition of our framework. While all the models drop in accuracy on increasing the number of layers, we demonstrate that both the SplitDisGCN and SplitGCN are the most robust to these drops.

Ablations

The benefit of the feature-topology tradeoff To verify that using a combination of both the feature and topology branches are important, instead of using either of the two branches exclusively, we present the results on BlogCatalog and Flickr datasets using just the feature branch (A_f only), and the topology branch (A_g only) separately. We compare against the SplitGCN which uses both the branches, and also add the SplitDisGCN variant as well for completeness, in Table 4. We are able to demonstrate that both the variants benefit from learning an optimal combination of the feature and topology representations, as the accuracies and stability is considerably better than using each branch in isolation.

The evolution of α

We are also interested in how the fusion parameter α in SplitGCN evolves over training, since it is this parameter that quantifies the tradeoff during the learning process. Intuitively, we expect that for the social media datasets, the network should learn to emphasize the feature branch more than the topology branch, since for these datasets, the graph topology is not as important as the feature representations. We present the BlogCatalog results in Figure 3 (For Flickr, see supplementary material). Over the epochs, we find that

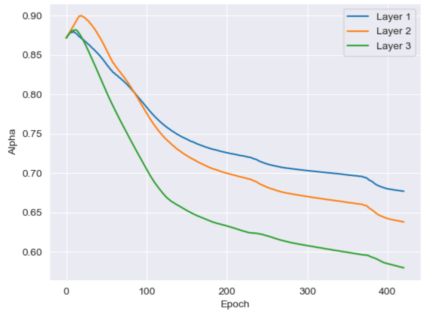
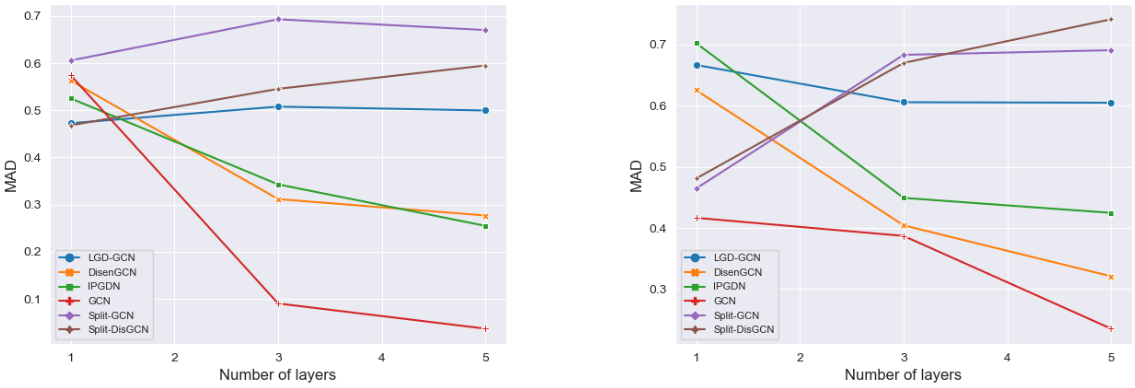


Figure 3: The evolution of α: BlogCatalog

Table 4: Ablating against topology branch only and feature branch only variants. (Results rounded to 2 decimal places)

			Blogcatalog			Flickr		
Model	A_g	A_f	1 layer	3 layers	5 layers	1 layer	3 layers	5 layers
SplitGCN	✗	✗	0.86 ± 0.01	0.68 ± 0.01	0.62 ± 0.02	0.50 ± 0.00	0.21 ± 0.00	0.10 ± 0.01
SplitGCN	✗	✓	0.80 ± 0.01	0.76 ± 0.01	0.58 ± 0.02	0.66 ± 0.01	0.60 ± 0.02	0.33 ± 0.04
SplitGCN	✓	✓	$\mathbf{0.91 \pm 0.0}$	0.84 ± 0.01	0.66 ± 0.03	0.78 ± 0.01	0.75 ± 0.01	$\mathbf{0.73 \pm 0.01}$
SplitDisGCN	✓	✓	0.87 ± 0.0	$\mathbf{0.89 \pm 0.01}$	$\mathbf{0.85 \pm 0.01}$	$\mathbf{0.79 \pm 0.0}$	$\mathbf{0.75 \pm 0.01}$	0.65 ± 0.0



(a) BlogCatalog

(b) Flickr

Figure 4: MAD Analysis on social media datasets

the network eventually learns to weight the feature branch more than the topology branch, validating our intuition.

Mitigating Oversmoothing

The MAD scores serve as proxies to measure the effect of oversmoothing. To capture the stabilising effect of the SplitGCN and SplitDisGCN variants, we calculate the MAD scores on the social network datasets, shown on Fig. 4. For both variants, we observe progressively higher values of MAD as the number of layers increases. These results, coupled with the accuracy results in Table 3, indicate a stronger mitigation in oversmoothing as opposed to the disentangled-only models.

Conclusion

The purported benefits of disentanglement in mitigating oversmoothing had been hypothesized in multiple earlier works. In this work, we presented the first systematic analysis of the effect of disentangled representations on the oversmoothing problem in GCNs. Our proposed Dirichlet Energy-based estimator, the LDE ratio, allowed a natural quantification of the inherent tradeoff between graph structure and feature representations in the data. We then demonstrated the usefulness of disentangled representations on datasets with informative graph structure (low LDE data), and the susceptibility of disentangled models to the oversmoothing phenomenon on datasets with less informative graph structure (high LDE data). The usefulness of the disentangled models lay primarily in the neighborhood routing procedure - processing node feature affinities in a better way over vanilla graph convolutional models (similar features suggest similar node neighborhoods). This is the reason why disentangled representations were useful on low LDE data. On the contrary, the same reliance on node neighborhoods rendered these models susceptible to oversmoothing in high LDE data, owing to no discernible information in the graph structure. These insights allowed us to showcase an intuitive experimental framework, that explicitly leveraged the tradeoff between the topology and the feature representations in the learning mechanism, thereby mitigating the oversmoothing phenomenon. We achieved this using a linear combination of the original adjacency matrix based feature aggregation, and the k-nn feature affinity graph based feature aggregation, learning the importance of each branch during the training process for each dataset. Further, we were able to demonstrate the effectiveness of using *both* branches during training, as opposed to a single branch. In addition to the more overall stable results across layers for the datasets, mitigating oversmoothing in high LDE data, the framework allowed us to shed insights on when disentanglement is useful for learning, and when disentanglement may not be enough. We hope our investigations lead to more promising future research in the interesting connections between disentanglement, oversmoothing, dataset properties, and the development of new models.

References

Abel, R.; Benami, I.; and Louzoun, Y. 2019. Topological based classification using graph convolutional networks. *ArXiv*, abs/1911.06892.

Cai, C.; and Wang, Y. 2020. A Note on Over-Smoothing for Graph Neural Networks. *ArXiv*, abs/2006.13318.

Chen, D.; Lin, Y.; Li, W.; Li, P.; Zhou, J.; and Sun, X. 2019a. Measuring and Relieving the Over-smoothing Problem for Graph Neural Networks from the Topological View. In *AAAI Conference on Artificial Intelligence*.

Chen, T.; Xu, M.; Hui, X.; Wu, H.; and Lin, L. 2019b. Learning Semantic-Specific Graph Representation for Multi-Label Image Recognition. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 522–531.

Chen, X.; Liu, M.; Peng, Y.; and Shi, B. 2022. Can higher-order structural features improve the performance of graph neural networks for graph classification? *2022 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, 788–795.

Chen, Y.; Zou, C.; and Chen, J. 2022. Label-aware graph representation learning for multi-label image classification. *Neurocomputing*, 492: 50–61.

Fan, W.; Ma, Y.; Li, Q.; He, Y.; Zhao, E.; Tang, J.; and Yin, D. 2019. Graph neural networks for social recommendation. In *The world wide web conference*, 417–426.

Fey, M.; and Lenssen, J. E. 2019. Fast Graph Representation Learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

Gao, C.; Zheng, Y.; Li, N.; Li, Y.; Qin, Y.; Piao, J.; Quan, Y.; Chang, J.; Jin, D.; He, X.; et al. 2023. A survey of graph neural networks for recommender systems: Challenges, methods, and directions. *ACM Transactions on Recommender Systems*, 1(1): 1–51.

Guo, J.; Huang, K.; Yi, X.; and Zhang, R. 2022. Learning Disentangled Graph Convolutional Networks Locally and Globally. *IEEE Transactions on Neural Networks and Learning Systems*.

Huang, X.; Li, J.; and Hu, X. 2017. Label Informed Attributed Network Embedding. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, WSDM '17, 731–739. New York, NY, USA: Association for Computing Machinery. ISBN 9781450346757.

Ji, F.; Lee, S. H.; Meng, H.; Zhao, K.; Yang, J.; and Tay, W. P. 2023. Leveraging Label Non-Uniformity for Node Classification in Graph Neural Networks. *ArXiv*, abs/2305.00139.

Kipf, T.; and Welling, M. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *ArXiv*, abs/1609.02907.

Lee, J. B.; Rossi, R. A.; and Kong, X. 2018. Graph Classification using Structural Attention. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Li, H.; Wang, X.; Zhang, Z.; Yuan, Z.; Li, H.; and Zhu, W. 2021. Disentangled Contrastive Learning on Graphs. In *Neural Information Processing Systems*.

Li, H.; Zhang, Z.; Wang, X.; and Zhu, W. 2023. Disentangled Graph Contrastive Learning With Independence Promotion. *IEEE Transactions on Knowledge and Data Engineering*, 35: 7856–7869.

Li, Q.; Han, Z.; and Wu, X.-M. 2018. Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning. *ArXiv*, abs/1801.07606.

Liang, Y.; Zhang, Y.; Gao, D.; and Xu, Q. 2021. An End-to-End Multiplex Graph Neural Network for Graph Representation Learning. *IEEE Access*, 9: 58861–58869.

Liu, M.; Gao, H.; and Ji, S. 2020. Towards Deeper Graph Neural Networks. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Liu, Y.; Wang, X.; Wu, S.; and Xiao, Z. 2019. Independence Promoted Graph Disentangled Networks. *ArXiv*, abs/1911.11430.

Ma, J.; Cui, P.; Kuang, K.; Wang, X.; and Zhu, W. 2019. Disentangled Graph Convolutional Networks. In *International Conference on Machine Learning*.

Oono, K.; and Suzuki, T. 2019. Graph Neural Networks Exponentially Lose Expressive Power for Node Classification. *arXiv: Learning*.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc.

Rusch, T. K.; Bronstein, M. M.; and Mishra, S. 2023. A Survey on Oversmoothing in Graph Neural Networks. *ArXiv*, abs/2303.10993.

Sen, P.; Namata, G.; Bilgic, M.; Getoor, L.; Gallagher, B.; and Eliassi-Rad, T. 2008. Collective Classification in Network Data Articles. *AI Magazine*, 29: 93–106.

Smith, K.; Ricaud, B.; Shahid, N.; Rhodes, S.; Starr, J. M.; Ibáñez, A.; Parra, M. A.; Escudero, J.; and Vandergheynst, P. 2017. Locating temporal functional dynamics of visual short-term memory binding using graph modular dirichlet energy. *Scientific reports*, 7(1): 42013.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio', P.; and Bengio, Y. 2017. Graph Attention Networks. *ArXiv*, abs/1710.10903.

Wu, X. S.; Chen, Z.; Wang, W. W.; and Jadbabaie, A. 2022. A Non-Asymptotic Analysis of Oversmoothing in Graph Neural Networks. *ArXiv*, abs/2212.10701.

Yuan, J.; Chen, S.; Zhang, Y.; Shi, Z.; Geng, X.; Fan, J.; and Rui, Y. 2023. Graph Attention Transformer Network for Multi-Label Image Classification. *ACM Trans. Multimedia Comput. Commun. Appl.*, 19(4).

Zhang, B.; Guo, X.; Tu, Z.; and Zhang, J. 2022. Graph alternate learning for robust graph neural networks in node classification. *Neural Computing and Applications*, 34: 8723 – 8735.

Zhao, L.; and Akoglu, L. 2019. PairNorm: Tackling Oversmoothing in GNNs. *ArXiv*, abs/1909.12223.

Zhao, T.; Zhang, X.; and Wang, S. 2021. GraphSMOTE: Imbalanced Node Classification on Graphs with Graph Neural Networks. *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*.

Zheng, L.; Fu, D.; and He, J. 2021. Tackling Oversmoothing of GNNs with Contrastive Learning. *ArXiv*, abs/2110.13798.

Zhou, K.; Huang, X.; Zha, D.; Chen, R.; Li, L.; Choi, S.-H.; and Hu, X. 2021. Dirichlet Energy Constrained Learning for Deep Graph Neural Networks. In *Neural Information Processing Systems*.

Zhou, W.; Xia, Z.; Dou, P.; Su, T.; and Hu, H. 2022. Double Attention Based on Graph Attention Network for Image Multi-Label Classification. *ACM Transactions on Multimedia Computing, Communications and Applications*, 19: 1 – 23.

Supplementary material

In this supplementary, we present additional results as well as experimental details. Specifically, we present the individual aggregate results of the disentangled models on each dataset, the detailed tables for hyperparameters, and the evolution of α in SplitGCN for the Flickr dataset.

Aggregate Results for all datasets

In Figure 1 in the main paper, we computed the aggregated results for low LDE and high LDE datasets for each model. In Table 1 here, we present the complete results on each dataset. Overall, we confirm the benefit of disentangled representations on low LDE datasets (Cora, Citeseer, Pubmed), and the susceptibility of these models on the high LDE datasets (BlogCatalog, Flickr).

Hyperparameters

We list the complete hyperparameters for SplitGCN and SplitDisGCN in Table 2 and Table 3 respectively. We ran a Bayesian hyperparameter search for the learning rate, dropout, number of layers, number of neighbors, and regularization strength. Given the best parameters, we fixed these for all the layers to maintain a fair comparison, and to ensure that the best hyperparemeters are not just tuned for a particular layer, but robust to results across layers. We refer to the learning rate as LR, regularization as $Reg.$, the feature dimensionality as dim, and the number of neighbors in the knn graph as k. For SplitDisGCN, we refer to the number of disentangled factors as n, and number of routing iterations as r

Table 2: Hyperparams: SplitGCN

Datasets	Cora	Citeseer	Pubmed	Flickr	Blogcatalog
LR	7e-4	0.01	0.003	0.008	0.002
Dropout	0.44	0.13	0.19	0.25	0.39
Reg.	0.002	0.002	4e-4	9e-6	2e-4
k	12	2	12	2	2
dim	112	112	112	112	112

Table 3: Hyperparams: SplitDisGCN

Datasets	Cora	Citeseer	Pubmed	Flickr	Blogcatalog
LR	0.002	0.007	0.02	0.01	3e-4
Dropout	0.43	0.13	0.35	0.22	0.23
Reg.	5e-6	0.05	0.03	0.001	2e-9
k	12	6	4	8	2
dim	112	112	112	112	112
n	7	7	7	7	7
r	7	7	7	7	7

Evolution of α

In addition to the BlogCatalog results for the behaviour of α during training, presented in the main paper for SplitGCN, we also present the Flickr results in Fig 1. Here, we also notice overall decrease in the emphasis on the topology branch, as was expected, since the high LDE do not have informative graph structures. SplitGCN learns to weigh heavily the feature branch over all the layers, especially in the first phases of training. It eventually stabilises to high emphasis on the feature branch (negative values indicate the emphasis on the topology branch).

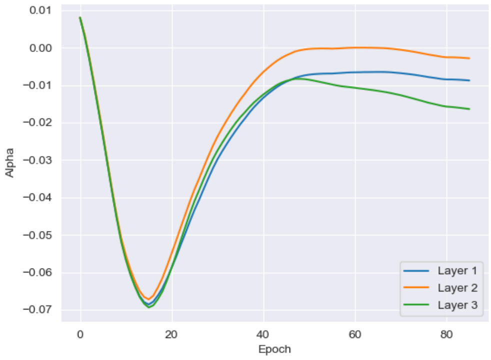


Figure 1: The evolution of α: Flickr

Connections to Homophily

The LDE ratio can be considered an implicit measure of node homophily. We can compute the homophily by using the pairwise cosine similarity of the vectors between connected nodes and taking the average over the number of edges. In Figure 2, we present these homophily values for all datasets. Low scores (close to zero) mean on average, connected nodes are slightly more similar than random pairs would be, but they are not highly similar. We notice the social media datasets have considerably lower homophily than the citation dataset. This supports the intuition from the LDE ratios that suggest that social network datasets do not have graph structure better than random connections. The LDE, unlike homophily, provides an *unsupervised* way to measure graph dataset quality.

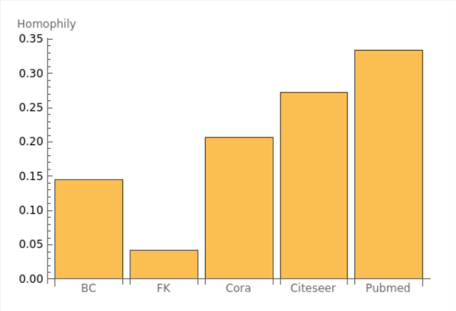


Figure 2: Homophily values for different datasets: Lower homophily indicates poorer quality of graph connectivity. We observe this in social media datasets BlogCatalog (BC) and Flickr (FK).

Table 1: Aggregate performance on all datasets (Rounded to 2 decimal places)

	Blogcatalog			Flickr		
Model	1 layer	3 layers	5 layers	1 layer	3 layers	5 layers
GCN	0.66 ± 0.00	0.61 ± 0.02	0.32 ± 0.02	0.50 ± 0.00	0.21 ± 0.00	0.10 ± 0.01
DisenGCN	0.86 ± 0.01	0.68 ± 0.01	0.62 ± 0.02	0.77 ± 0.01	0.57 ± 0.01	0.49 ± 0.01
IPGDN	0.43 ± 0.02	0.38 ± 0.01	0.36 ± 0.03	0.34 ± 0.02	0.30 ± 0.03	0.21 ± 0.02
LGD-GCN	0.91 ± 0.01	0.38 ± 0.06	0.31 ± 0.05	0.85 ± 0.01	0.23 ± 0.04	0.18 ± 0.01

	Cora			Citeseer		
Model	1 layer	3 layers	5 layers	1 layer	3 layers	5 layers
GCN	0.77 ± 0.00	0.79 ± 0.02	0.76 ± 0.02	0.66 ± 0.01	0.66 ± 0.01	0.65 ± 0.02
DisenGCN	0.71 ± 0.01	0.80 ± 0.01	0.82 ± 0.02	0.65 ± 0.01	0.69 ± 0.01	0.71 ± 0.01
IPGDN	0.71 ± 0.02	0.81 ± 0.01	0.82 ± 0.03	0.63 ± 0.02	0.69 ± 0.03	0.70 ± 0.02
LGD-GCN	0.79 ± 0.01	0.83 ± 0.06	0.84 ± 0.05	0.71 ± 0.01	0.73 ± 0.04	0.70 ± 0.01

	Pubmed		
Model	1 layer	3 layers	5 layers
GCN	0.74 ± 0.01	0.72 ± 0.01	0.62 ± 0.02
DisenGCN	0.75 ± 0.01	0.77 ± 0.01	0.78 ± 0.01
IPGDN	0.75 ± 0.02	0.77 ± 0.03	0.79 ± 0.02
LGD-GCN	0.78 ± 0.01	0.77 ± 0.01	0.78 ± 0.01

Bibliography

[1] Daron Acemoglu and Pascual Restrepo. Artificial intelligence, automation and work. *Alfred P. Sloan Foundation Economic Research Paper Series*, 2018.

[2] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[3] Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.2, knowledge manipulation. *arXiv preprint arXiv:2309.14402*, 2023.

[4] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

[5] Devansh Arpit, Stanislaw Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. *International Conference on Machine Learning*, pages 233–242, 2017.

[6] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE*, 10(7):e0130140, 2015.

[7] Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. In *The Journal of Machine Learning Research*, 2018.

[8] Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

[9] Abeba Birhane and Vinay Uday Prabhu. Multimodal datasets: Misogyny, pornography, and malignant stereotypes. *arXiv preprint arXiv:2110.01963*, 2021.

[10] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pages 92–100. ACM, 1998.

[11] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems*, pages 4349–4357, 2016.

[12] Rishi Bommasani et al. Opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[13] Nick Bostrom and Eliezer Yudkowsky. The ethics of artificial intelligence. *Artificial Intelligence Safety and Security*, 2014.

[14] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *International Conference on Computational Statistics*, 2010.

[15] Malik Boudiaf, Ziko Imtiaz Masud, Jérôme Rony, José Dolz, Pablo Piantanida, and Ismail Ben Ayed. Transductive Information Maximization For Few-Shot Learning. In *NeurIPS*, 2020.

[16] Tom B Brown et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[17] Miles Brundage et al. The malicious use of artificial intelligence: Forecasting, prevention, and mitigation. *arXiv preprint arXiv:1802.07228*, 2018.

[18] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 77–91, 2018.

[19] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

[20] L. Elisa Celis and Vijay Keswani. Implicit diversity in image summarization. *Proceedings of the ACM on Human-Computer Interaction*, 4:1 – 28, 2019.

[21] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. A discussion of semi-supervised learning and transduction. In *Semi-Supervised Learning*, 2006.

[22] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. *Semi-Supervised Learning*. MIT Press, 2006.

[23] Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *AAAI Conference on Artificial Intelligence*, 2019.

[24] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016.

[25] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, pages 1597–1607, 2020.

[26] Wentao Cui and Yuhong Guo. Parameterless transductive feature re-representation for few-shot learning. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 2212–2221. PMLR, 18–24 Jul 2021.

[27] Chris Cummins, Volker Seeker, Dejan Grubisic, Baptiste Roziere, Jonas Gehring, Gabriel Synnaeve, and Hugh Leather. Meta large language model compiler: Foundation models of compiler optimization. *arXiv preprint arXiv:2407.02524*, 2024.

[28] Kanjar De and Marius Pedersen. Impact of colour on robustness of deep neural networks. In *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 21–30, 2021.

[29] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 3844–3852, Red Hook, NY, USA, 2016. Curran Associates Inc.

[30] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[31] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.

[32] Sanchari Dhar and Lior Shamir. Evaluation of the benchmark datasets for testing the efficacy of deep convolutional neural networks. *Visual informatics*, 5(3).

[33] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[34] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[35] Nanyi Fei, Yizhao Gao, Zhiwu Lu, and Tao Xiang. Z-Score Normalization, Hubness, and Few-Shot Learning. In *ICCV*, 2021.

[36] Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

[37] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[38] Robert Geirhos, Patrick Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2018.

[39] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[40] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015.

[41] Charles A. E. Goodhart. Problems of monetary management: The uk experience. 1984.

[42] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284, 2020.

[43] Xiaojie Guo, Yu Li, and Haibin Ling. Lime: Low-light image enhancement via illumination map estimation. *IEEE Transactions on Image Processing*, 26:982–993, 2017.

[44] David Harrison and Daniel L Rubinfeld. *Hedonic prices and the demand for clean air*, volume 5. Elsevier, 1978.

[45] Simon Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

[46] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

[47] Kaiming He and Jian Sun. Convolutional neural networks at constrained time cost. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5353–5360, 2015.

[48] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[49] Aaron Hertzmann. Can computers create art? the dilemma of ai-generated content in entertainment. *ACM SIGGRAPH*, 2022.

[50] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851, 2020.

[51] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[52] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[53] Hossein Hosseini, Baicen Xiao, Mayoore Jaiswal, and Radha Poovendran. Assessing shape bias property of convolutional neural networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2004–20048, 2018.

[54] Md Amirul Islam, Matthew Kowal, Patrick Esser, Sen Jia, Bjorn Ommer, Konstantinos G Derpanis, and Neil Bruce. Shape or texture: Understanding discriminative features in cnns. *arXiv preprint arXiv:2101.11604*, 2021.

[55] Pavel Izmailov, Polina Kirichenko, Nate Gruver, and Andrew G Wilson. On feature learning in the presence of spurious correlations. *Advances in Neural Information Processing Systems*, 35:38516–38532, 2022.

[56] Zhang JingMao and Shen YanXia. Review on spectral methods for clustering. In *2015 34th Chinese Control Conference (CCC)*, pages 3791–3796, 2015.

[57] David Kampmann. Venture capital, the fetish of artificial intelligence, and the contradictions of making intangible assets. *Economy and Society*, 53:1–28, 01 2024.

[58] Jared Kaplan, Sam McCandlish, Tom Henighan, et al. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[59] Will Kay, Joao Carreira, Karen Simonyan, Alexey Zhavoronkov, Tom Duerig, and Andrew Zisserman. The kinetics human action video dataset. In *arXiv preprint arXiv:1705.06950*, 2017.

[60] Guolin Ke, Qiwei Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qi Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pages 3149–3157, 2017.

[61] Byungju Kim, Hyunwoo Kim, Kyungsu Kim, Sungjin Kim, and Junmo Kim. Learning not to learn: Training deep neural networks with biased data. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9004–9012, 2018.

[62] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[63] Diederik P Kingma and Max Welling. An introduction to variational autoencoders. *Foundations and Trends in Machine Learning*, 12(4):307–392, 2019.

[64] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[65] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2012.

[66] Solomon Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

[67] Thomas Kurbiel and Shahrzad Khaleghian. Training of deep neural networks based on distance measures using rmsprop. *arXiv preprint arXiv:1708.01911*, 2017.

[68] Sebastian Lapuschkin, Stephan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Unmasking clever hans predictors and assessing what machines really learn. *Nature communications*, 10(1):1096, 2019.

[69] Michalis Lazarou, Tania Stathaki, and Yannis Avrithis. Iterative label cleaning for transductive and semi-supervised few-shot learning. In *Proceedings of the ieee/cvf international conference on computer vision*, pages 8751–8760, 2021.

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

[71] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. *ArXiv*, abs/1801.07606, 2018.

[72] Tianqin Li, Ziqi Wen, Yangfan Li, and Tai Sing Lee. Emergence of shape bias in convolutional neural networks through activation sparsity. *Advances in Neural Information Processing Systems*, 36, 2024.

[73] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[74] Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *ACM Queue*, 16(3):31–57, 2018.

[75] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.

[76] Meng Liu, Hongyang Gao, and Shuiwang Ji. Towards deeper graph neural networks. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.

[77] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021.

[78] Zhuang Liu, Jia Ning, Yue Cao, Hanzi Wei, Zheng Zhang, Stephen Lin, and Han Hu. A convnet for the 2020s. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11976–11986, 2022.

[79] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. *Proceedings of the IEEE International Conference on Computer Vision*, pages 3730–3738, 2015.

[80] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774, 2017.

[81] Tom M Mitchell. *Machine learning*. McGraw Hill, 1997.

[82] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

[83] Junhyun Nam, Jaehyung Kim, Jaeho Lee, and Jinwoo Shin. Spread spurious attribute: Improving worst-group accuracy with spurious attribute estimation. In *10th International Conference on Learning Representations, ICLR 2022*, 2022.

[84] NVIDIA, Péter Vingelmann, and Frank H.P. Fitzek. Cuda, release: 10.2.89, 2020.

[85] Aude Oliva and Antonio Torralba. The role of context in object recognition. *Trends in cognitive sciences*, 11(12):520–527, 2007.

[86] Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. *arXiv: Learning*, 2019.

[87] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[88] Ryan Po, Wang Yifan, Vladislav Golyanik, Kfir Aberman, Jonathan T. Barron, Amit H. Bermano, Eric Ryan Chan, Tali Dekel, Aleksander Holynski, Angjoo Kanazawa, C. Karen Liu, Lingjie Liu, Ben Mildenhall, Matthias Nießner, Bjorn Ommer, Christian Theobalt, Peter Wonka, and Gordon Wetzstein. State of the art on diffusion models for visual computing. *Computer Graphics Forum*, 43, 2023.

[89] Guodong Qi, Huimin Yu, Zhaohui Lu, and Shuzhao Li. Transductive few-shot classification on the oblique manifold. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8412–8422, 2021.

[90] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[91] Miloš Radovanovic, Alexandros Nanopoulos, and Mirjana Ivanovic. Hubs in Space: Popular Nearest Neighbors in High-Dimensional Data. *JMLR*, 2010.

[92] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, 2016.

[93] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[94] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[95] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks. *International Conference on Learning Representations*, 2019.

[96] Juergen Schmidhuber. Annotated history of modern ai and deep learning. *arXiv preprint arXiv:2212.11279*, 2022.

[97] Christoph Schuhmann, Romain Beaumont, Radu Vencu, Cameron Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Anshul Katta, Chris Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022.

[98] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017.

[99] Muhammad Shafiq and Zhaoquan Gu. Deep residual learning for image recognition: A survey. *Applied Sciences*, 2022.

[100] Gagan Deep Sharma, Anshita Yadav, and Ritika Chopra. Artificial intelligence and effective governance: A review, critique and research agenda. 2020.

[101] Krishna Kumar Singh, Dhruv Mahajan, Kristen Grauman, Yong Jae Lee, Matt Feiszli, and Deepti Ghadiyaram. Don't judge an object by its context: Learning to overcome contextual bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11070–11078, 2020.

[102] Zixing Song, Xiangli Yang, Zenglin Xu, and Irwin King. Graph-based semi-supervised learning: A comprehensive review. *IEEE Transactions on Neural Networks and Learning Systems*, 34(11):8174–8194, 2022.

[103] Ivan Sosnovik, Michał Szmaja, and Arnold Smeulders. Scale-equivariant steerable networks. *arXiv preprint arXiv:1910.11093*, 2019.

[104] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

[105] Pierre Stock and Moustapha Cissé. Convnets and imagenet beyond accuracy: Understanding mistakes and uncovering biases. *arXiv preprint arXiv:1711.11443*, 2018.

[106] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7464–7473, 2019.

[107] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017.

[108] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. *International Conference on Machine Learning*, 2017.

[109] Kushal Tirumala, Aram Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. Memorization without overfitting: Analyzing the training dynamics of large language models. *Advances in Neural Information Processing Systems*, 35:38274–38290, 2022.

[110] Tatiana Tommasi, Novi Patricia, Barbara Caputo, and Tinne Tuytelaars. A deeper look at dataset bias. *Domain adaptation in computer vision applications*, pages 37–55, 2017.

[111] Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. *CVPR 2011*, pages 1521–1528, 2011.

[112] Vladimir N Vapnik. *Statistical Learning Theory*. Wiley, 1998.

[113] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[114] Jacintha Walters, Diptish Dey, Debarati Bhaumik, and Sophie Horsman. Complying with the eu ai act. In *European Conference on Artificial Intelligence*, pages 65–75. Springer, 2023.

[115] Angelina Wang, Alexander Liu, Ryan Zhang, Anat Kleiman, Leslie Kim, Dora Zhao, Iroha Shirai, Arvind Narayanan, and Olga Russakovsky. Revise: A tool for measuring and mitigating bias in visual datasets. *International Journal of Computer Vision*, 130(7):1790–1810, 2022.

[116] Yan Wang, Wei-Lun Chao, Kilian Q. Weinberger, and Laurens van der Maaten. SimpleShot: Revisiting Nearest-Neighbor Classification for Few-Shot Learning. *arXiv:1911.04623 [cs]*, 2019.

[117] Lynette Webb and Daniel Schönberger. Generative ai and the problem of existential risk. *arXiv preprint arXiv:2407.13365*, 2024.

[118] David H Wolpert. The lack of a priori distinctions between learning algorithms. In *Neural Computation*, volume 8, pages 1341–1390, 1996.

[119] Junjie Wu. *Advances in K-means clustering: a data mining thinking*. Springer Science & Business Media, 2012.

[120] Jing Xu, Xu Luo, Xinglin Pan, Yanan Li, Wenjie Pei, and Zenglin Xu. Alleviating the sample selection bias in few-shot learning by removing projection to the centroid. *Advances in neural information processing systems*, 35:21073–21086, 2022.

[121] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, 1995.

[122] Gokul Yenduri, M Ramalingam, G Chemmalar Selvi, Y Supriya, Gautam Srivastava, Praveen Kumar Reddy Maddikunta, G Deepti Raj, Rutvij H Jhaveri, B Prabadevi, Weizheng Wang, et al. Gpt (generative pre-trained transformer)–a comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions. *IEEE Access*, 2024.

[123] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. *arXiv preprint arXiv:2106.04560*, 2021.

[124] Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2979–2989, 2017.

[125] Lingxiao Zhao and Leman Akoglu. Pairnorm: Tackling oversmoothing in gnns. *ArXiv*, abs/1909.12223, 2019.

[126] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and Maosong Sun. Graph neural networks: A review of methods and applications. *ArXiv*, abs/1812.08434, 2018.

[127] Zhi-Hua Zhou and Ming Li. Tri-training: Exploiting unlabeled data using three classifiers. In *IEEE Transactions on Knowledge and Data Engineering*, volume 17, pages 1529–1541, 2005.

[128] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

[129] Hao Zhu and Piotr Koniusz. EASE: Unsupervised Discriminant Subspace Learning for Transductive Few-Shot Learning. In *CVPR*, 2022.

[130] Xiaojin Zhu. Semi-supervised learning literature survey. *Computer Sciences Technical Report 1530*, 2005.

[131] Zeyuan Allen Zhu and Yuanzhi Li. Physics of language models: Part 3.1, knowledge storage and extraction. *arXiv preprint arXiv:2309.14316*, 2023.