ABSTRACT

Title of Dissertation: RELIABLE MACHINE LEARNING: ROBUSTNESS, CALIBRATION, AND REPRODUCIBILITY

Chihuang Liu

Dissertation Directed by: Professor Joseph JaJa
Department of Electrical and Computer Engineering

Modern machine learning (ML) algorithms are being applied today to a rapidly increasing number of tasks that affect the lives and well-being of people across the globe. Despite all the successes of artificial intelligence (AI), these methods are not always reliable and in fact are often quite brittle. It has been shown that a wide range of recent ML algorithms are vulnerable to adversarial attacks, and are over-confident even when they are not accurate. In this dissertation, we focus on the overall goal of making machine learning algorithms more reliable in terms of adversarial robustness, confidence calibration, and reproducibility.

In the first part of the thesis, we explore novel approaches to improve the adversarial robustness of a deep neural network. We present a method that involves feature regularization and attention-based feature prioritization to motivate the model to only learn and rely on robust features that are not manipulated by the adversarial perturbation. We show that the resulting model is significantly more robust than other existing methods.

In the second part of the thesis, we discover that the current training scheme of using one-hot labels under cross-entropy loss is a major cause of the over-confident behavior of deep neural networks. We propose a generalized definition of confidence calibration that requires the entire output to be calibrated. This approach leads to a novel form of the smooth labeling algorithm, called *class-similarity based label smoothing*, which tries to approximate a distribution that is optimal for generalized confidence calibration. We show that a model trained with the proposed smooth labels is significantly better calibrated than all existing methods.

In the third part of the thesis, we propose an approach that can improve the calibration performance of robust models. We first learn a representation space using prototypical learning which bases its classification on the distances between the representation of a sample and the representations of each class prototype. We then use the distance information to train a confidence prediction network to encourage the model to make calibrated predictions. We demonstrate through extensive experiments that our method can improve the calibration performance of a model while maintaining comparable accuracy and adversarial robustness levels.

In the fourth part of the thesis, we tackle the problem of determining reproducible, large-scale functional patterns for the whole brain from a group of fMRI subjects. Because of the non-linear nature of the signals and significant inter-subject variability, how to reliably extract patterns that are reproducible across subjects has been a challenging task. We propose a group-level model, called LEICA, that uses Laplacian eigenmaps as the main data reduction step to preserve the correlation information in the original data as best as possible in a certain rigorous sense. The

nonlinear map is robust relative to noise in the data and inter-subject variability. We show that LEICA detects functionally cohesive maps that are much more reproducible than the state-of-the-art methods.

RELIABLE MACHINE LEARNING: ROBUSTNESS, CALIBRATION, AND REPRODUCIBILITY

by

Chihuang Liu

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2021

Advisory Committee:
Professor Joseph JaJa, Chair/Advisor
Professor Min Wu
Professor Furong Huang
Professor Shuvra S. Bhattacharyya
Professor Dinesh Manocha

© Copyright by
Chihuang Liu
2021

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my advisor Joseph JaJa for his guidance, vision and continuing support. He always gave me the freedom to pursue research directions that I am mostly excited about and I am extremely grateful that along the way, he was always interested in and open to what I wanted to work on. It is hard not to be inspired by his genuine passion for research. I am fortunate to have him as my advisor and have learned extensively about research, writing, presentation, and much more from him over the years.

I am grateful to have Yu Jin, Manasij Venkatesh, Amit Kumar Kundu, and Vaishnavi Sunil Patil in our group. Thanks for many creative discussions and insightful comments. They are always great resources for inspirations.

I count myself lucky to have many best friends for a more than a decade, indcluding Yuan Gao, Hao Wu, Yi Feng, and Lingxiang Zhang. Thanks for the countless times of chatting when I found myself in difficult situations. I wouldn't have made it through without your support and comfort.

I am thankful for my mentors, Steve Xi and Russell Howes during my internship at Facebook. They helped me tremendously.

I also would like to thank my other thesis committee members, Min Wu, Furong Huang, Shuvra S. Bhattacharyya, and Dinesh Manocha.

Last but not least, I want to deeply thank my family for their unconditional love. It goes without saying that I would not be where I am today without them. My parents did so much for my education and have encouraged me to pursue my

interests.

Table of Contents

Acknowledgements ii

Table of Contents iv

List of Tables vii

List of Figures viii

1 Introduction 1
1.1 Background . . . 1
1.1.1 Adversarial Robustness . . . 3
1.1.1.1 Adversarial Attacks . . . 3
1.1.1.2 Adversarial Defenses . . . 7
1.1.2 Confidence Calibration . . . 10
1.1.2.1 Measuring Calibration . . . 12
1.1.2.2 Improving Calibration . . . 13
1.1.3 Reproducibility . . . 16
1.2 Overview of Thesis . . . 17

2 Feature Prioritization and Regularization Improve Standard Accuracy and Adversarial Robustness 22
2.1 Introduction . . . 22
2.2 Related Work . . . 25
2.3 Approach . . . 26
2.3.1 Adversarial Training . . . 27
2.3.2 Attention Model . . . 28
2.3.3 Feature Regularization . . . 30
2.3.4 Model Loss . . . 31
2.4 Experiments and Results . . . 31
2.4.1 Robustness on MNIST . . . 31
2.4.2 Robustness on CIFAR-10 . . . 32
2.4.3 Empirical Justification for Attention . . . 35
2.4.4 Gradient Map . . . 37
2.4.5 Results on CIFAR-100 . . . 39
2.4.6 Comparison of Feature Regularization with Adversarial Logit Pairing . . . 39

2.5 Conclusion . . . 40

3 Class-Similarity Based Label Smoothing for Generalized Confidence Calibration 42
3.1 Introduction . . . 42
3.2 Confidence Calibration . . . 46
3.2.1 Prediction Calibration . . . 47
3.2.2 Output Calibration . . . 48
3.3 Learning with Different Labels . . . 49
3.4 Approach . . . 51
3.4.1 Image Space . . . 51
3.4.2 Representation Space . . . 52
3.4.3 Semantic Space . . . 53
3.4.4 Class-Similarity Based Smooth Label . . . 54
3.5 Experiments . . . 55
3.5.1 Setup . . . 55
3.5.2 Comparison with other Methods . . . 56
3.5.3 Confidence Calibration Results . . . 57
3.5.4 Effects of Parameters . . . 59
3.5.5 Evaluation on Out-of-Distribution Data . . . 60
3.6 Conclusion . . . 61

4 Improving Calibration of Robust Models via Prototypical Learning 63
4.1 Introduction . . . 63
4.2 Related Work . . . 64
4.3 Approach . . . 66
4.3.1 Notation . . . 66
4.3.2 Prototypical Learning . . . 67
4.3.3 Confidence Prediction . . . 69
4.3.4 Adversarial Training . . . 71
4.4 Experiments . . . 71
4.4.1 Setup . . . 71
4.4.2 Evaluation Metrics . . . 72
4.4.3 Comparing Methods . . . 73
4.4.4 Confidence Calibrated Robust Models . . . 73
4.4.5 Calibration on Multiple Types of Adversarial Examples . . . 75
4.4.6 Effectiveness of Confidence Prediction Module . . . 76
4.4.7 Effect of Parameter β . . . 78
4.5 Conclusion . . . 79

5 LEICA: Laplacian Eigenmaps for Group ICA Decomposition of fMRI Data 80
5.1 Introduction . . . 80
5.2 Materials and methods . . . 84
5.2.1 Group ICA model . . . 84
5.2.2 Dimensionality reduction using Laplacian eigenmaps . . . 88

5.2.3 Group-level spatial maps . . . 90
5.2.4 Evaluation strategies . . . 91
5.2.4.1 Similarity with other methods . . . 92
5.2.4.2 Functional connectivity structure . . . 93
5.2.4.3 Reproducibility and consistency . . . 94
5.2.5 fMRI datasets . . . 96
5.2.5.1 Resting state fMRI dataset . . . 96
5.2.5.2 Working memory task fMRI dataset . . . 97
5.2.6 Model parameters choice . . . 98
5.3 Results . . . 99
5.3.1 Resting state dataset . . . 99
5.3.1.1 Group-level spatial maps . . . 99
5.3.1.2 Similarity score . . . 100
5.3.1.3 Recovery of functional connectivity structure . . . 102
5.3.1.4 Reproducibility results . . . 105
5.3.2 Working memory dataset . . . 106
5.3.2.1 Group-level spatial maps . . . 106
5.3.2.2 Similarity score . . . 107
5.3.2.3 Recovery of functional connectivity structure . . . 108
5.3.2.4 Reproducibility results . . . 109
5.4 Discussion . . . 109
5.4.1 Model flexibility . . . 111
5.4.2 Number of nearest neighbors k . . . 111
5.4.3 Model order d . . . 112
5.4.4 Computational complexity and execution time . . . 113
5.5 Conclusion . . . 114

6 Conclusion and Future Work 116
6.1 Conclusion . . . 116
6.2 Future Work . . . 117

Bibliography 119

List of Tables

2.1 Performance comparison on MNIST . . . 32
2.2 Performance comparison on CIFAR-10 . . . 33
2.3 Performance comparison on CIFAR-10 using the 3-times wide ResNet 34
2.4 Classification accuracy on the gradient maps . . . 38
2.5 Performance comparison on CIFAR-100 . . . 40
2.6 Performance comparison of ALP and feature regularization . . . 40

3.1 Calibration ECE results . . . 57

4.1 Accuracy, robustness, and calibration results . . . 74
4.2 Accuracy, robustness, and calibration results on multiple types of adversarial examples . . . 75
4.3 Effectiveness of confidence prediction network results . . . 77
4.4 Average prediction confidence . . . 77

5.1 Cosine similarity scores for spatial maps on the resting state dataset . 100
5.2 Cosine similarity for the resting state dataset . . . 103
5.3 Reproducibility results for the resting state dataset . . . 104
5.4 Cosine similarity scores for spatial maps on the working memory dataset 107
5.5 Cosine similarity on the working memory dataset . . . 108
5.6 Reproducibility results for the working memory dataset . . . 110
5.7 Execution times . . . 114

List of Figures

1.1 Confidence histograms 11

2.1 Overview of the proposed model 27
2.2 The relationship between attention weights and feature robustness . . 36
2.3 The learned attention maps 36
2.4 CIFAR-10 gradient maps 38

3.1 Reliability diagrams 58
3.2 Effects of parameters 59
3.3 Confidences on out-of-distribution data 61

4.1 Overview of the proposed model 66
4.2 Effects of parameters 79

5.1 Processing steps used by LEICA 88
5.2 Spatial maps generated by LEICA on the resting state dataset 99
5.3 Sample pairs of spatial maps with different cosine similarity scores . . 101
5.4 Two spatial maps and their correlation maps 101
5.5 Spatial maps generated by LEICA on the working memory dataset . 106
5.6 Motor networks extracted by LEICA 109

Chapter 1: Introduction

1.1 Background

Machine learning algorithms fall into two primary categories: supervised and unsupervised. Supervised learning is defined by its use of labeled datasets to train algorithms to classify data or make predictions. By feeding input data into the model, the parameters are adjusted until the model achieves the desired performance. Some widely used methods in supervised learning include neural networks, naive bayes [1], linear regression [2], logistic regression [3], random forest [4], and support vector machines [5].

Unsupervised learning extracts information and patterns from unlabeled datasets. These algorithms discover hidden patterns or groupings that provide insights into the datasets and are often used in exploratory data analysis and pattern recognition. Dimensionality reduction techniques play an important role in unsupervised learning as they reduce the number of features in the data subject to some objective functions. Unsupervised learning algorithms include various clustering algorithms such as, k-means clustering [6], probabilistic clustering methods [7], various dimensionality reduction strategies such as principal component analysis [8] and independent component analysis [9] and various deep learning methods such as Variataional

Autoencoders and Deep Belief Networks..

Machine learning algorithms, especially those that employ deep learning models, have achieved remarkable performance in various widely studied applications, including image classification [10], speech recognition [11], natural language processing [12], and so on. They are also increasingly used to make critical decisions in medical diagnosis [13], autonomous driving [14], financial services [15] and even to make new scientific discoveries [16].

In addition to the vast amount of effort that aims to improve the standard performance of machine learning algorithms, there is considerable interest in targeting the reliability of these algorithms. There are many aspects related to the reliability of machine learning: robustness, fairness, generalization, explainability, calibration, and reproducibility. Depending on the application, the focus on reliability may differ. For example, if the input data to a classifier can be very noisy or even may contain malicious samples, we would like the model to be *robust* so that the classification is more likely to be correct; in high-stakes applications, the model needs to be *calibrated*, that is, it associates a likelihood of correctness with its prediction, so that we could stop the decision making or pass it to human experts when it is not confident enough to be accurate. In many medical diagnosis and analysis applications, where the sample size is usually small, it is important that the model be *reproducible* across different groups so that the results are stable; in applications where there are sensitive attributes such as gender, race, and wealth, we need to have a model that is *fair* so that the outcomes do not depend on biased factors.

In this thesis, we focus on robustness, calibration, and reproducibility of ma-

chine learning algorithms. We will first introduce the problems in these aspects, then provide an overview of our major contributions in this thesis.

1.1.1 Adversarial Robustness

1.1.1.1 Adversarial Attacks

Adversarial Examples Adversarial examples [17] are data points that are visually indistinguishable from "normal" examples but specifically tuned so as to fool or mislead the machine learning system [18, 19]. In creating adversarial examples, the attacker is often restrained to select within some uncertainty set $\epsilon(x)$, usually a l_p norm constrained region around the original input examples. Then finding adversarial examples can be framed as a constrained optimization problem

$$x' = \arg \max_{x' \in \epsilon(x)} l\left(h(x'), y\right) \tag{1.1}$$

where h is the neural network, l is the loss function, and x, y are the input data and label. Depending on the level of knowledge of the target model that an attacker has, we can distinguish between white-box attacks and black-box attacks.

White-box Attacks In white-box attacks, the attacker has full knowledge of the model, i.e., model structure, parameters, optimization function, and evaluation criterion. With the full knowledge of the model, algorithms generating white-box adversarial examples optimize some pre-defined objective function using line search or gradient-based methods.

L-BFGS [17] uses Limited-memory Broyden-Fletcher-Goldfarb-Shanno method to optimize

$$\min_r c|r| + J(x + r, l) \tag{1.2}$$

where l is the target class. Fast Gradient Sign Method (FGSM) [20] uses linear approximation to find adversarial examples

$$x' = x + \epsilon \cdot \text{sign}(\nabla_x J(x, y)) \tag{1.3}$$

Jacobian-based Saliency Map Attack (JSMA) [21] finds adversarial perturbation to input examples by adding saliency value to the pixels with the largest values until the resulting example is misclassified. Random FGSM (R+FGSM) [22] extends FGSM by adding random perturbations sampled from a Gaussian distribution before calculating the first derivative of the loss with respect to the input. Deepfool [23] considers the linearization of the class decision boundaries as hyperplanes and adds perturbation that projects input images onto the closest hyperplane. CW attack [24] is based on logit margin and optimizes the distance between the target class t and the most-likely class

$$\max(\max\{f(x')_i : i \neq t\} - f(x')_t, -\gamma) \tag{1.4}$$

where $f(x)$ denotes the model output logits for input x, and γ is a constant that controls the desired confidence score. Iterative FGSM (I-FGSM) [19], more commonly referred to as a Projected Gradient Descent (PGD) [25], extends FGSM to

multiple iterations by taking a fixed step at each iteration towards the steepest gradient direction

$$x' = \text{Proj}_{x' \in \epsilon(x)} \left(x' + \alpha \cdot \text{sign}(\nabla_{x'} l(h(x'), y)) \right) \tag{1.5}$$

where α is the step size.

Among these adversaries, PGD is the most widely used and studied in the literature, and has become a standard and consistent benchmark when evaluating robustness in the l_∞ setting. Numerous improvements have been proposed for the PGD adversary, for example, momentum were incorporated into the PGD adversary [26]; multiple restarts and more iterations can improve the effectiveness of the attack [27]; replace non-differentiable model components with differentiable approximations [28]. However, they are of varying degrees of success and PGD is still the benchmark adversary.

Black-box Attacks In black-box attacks, the attacker has no access to the target model but can only observe the output of the model. There are two lines of techniques for generating adversarial examples in the black-box setting: using substitute models and the transferability of adversarial examples, and iterative optimization schemes.

In the first line of work, attacking models with substitute models was introduced [29]. The attacker first trains a substitute model to mimic the target model's decision boundaries, then it generates adversarial examples on the substitute model

and uses them to attack the original target model. Ensemble-based approach generates adversarial examples that transfer among multiple models [30]. Input transformation was applied at each iteration when generating adversarial examples to improve transferability [31]. Both gradient ascent and descent directions of the substitute model were explored to enhance the diversity as well as transferability of adversarial examples [32].

In the other line of work, Narodytska et al. [33] use a greedy search algorithm that selectively changes individual pixel values to generate adversarial examples. Chen et al. [34] design black-box adversarials based on finite differences and gradient-based optimization. Their method uses coordinate descent to attack black-box networks and also introduces various optimization techniques to decrease sample complexity. Ilyas et al. [35] propose a black-box attack strategy that uses natural evolution strategies to estimate the gradients. They then use their algorithm as primitive in attacks under more restricted threat settings. Brendel et al. [36] propose boundary attack that starts from an image in the target class and then alternates between taking steps on the decision boundary to maintain the adversarial classification and taking steps towards the original image. Ilyas et al. [37] estimate the gradients by exploiting the spatial and temporal correlations in gradients. They use gradient priors and propose a bandit optimization based method to update these priors.

Furthermore, a recent study [38] shows that even real-world, unmodified, and naturally occurring examples can cause the classifier accuracy to significantly degrade.

1.1.1.2 Adversarial Defenses

In response to the susceptibility of neural networks to adversarial examples, a significant amount of methods are proposed to construct defenses to increase the robustness of neural networks. Despite various attempts, many techniques are subsequently shown to be ineffective [27, 28, 39, 40].

Adversarial Training Adversarial training is a form of data augmentation where it injects adversarial examples during training. Goodfellow et al. [20] first included adversarial examples in the training stage. They generated adversarial examples in every step of training and inject them into the training set, and showed that adversarial training improved the robustness of deep neural networks. In [19], a mixture of clean images and adversarial images generated by one step Fast Gradient Sign Method (FGSM) are used to update the network's parameters for every training mini batch. They also extend the comprehensive analysis of adversarial training methods to the ImageNet dataset. However, since they used FGSM as the adversarial generating method during training, the improved robustness is not able to hold up against iterative attacks. Na et al. [41] improved adversarial training by adding adversarial examples generated by iterative methods. Madry et al. [25] proposed to replace all clean images with adversarial images which is a direct result of optimizing a saddle point (min-max) formulation. They suggest that PGD is a universal first-order adversary which is then used in their adversary generating process.

Regularization Various regularization methods have been proposed as a standalone method to improve robustness or as a complement to adversarial training. Yin et al. [42] study the adversarially robust generalization problem through the lens of Rademacher complexity and theoretically show that having l_1 norm constraints on the weight matrices can improve generalization in the adversarial setting. Sinha et al. [43] take the perspective of distributionally robust optimization by formulating the threat to be distributions within a Wasserstein ball of the original distribution. Their theoretical analysis results in a Lagrangian which penalizes the l_2 distance between the clean and adversarial examples. Ross et al. [44] add l_2 regularizer on input gradients to defend against transferred adversarial examples. Adversarial Logit Pairing [45] improves robustness by adding a l_2 regularizer for the logits between the clean and adversarial examples. Cisse et al. [46] introduce Parseval networks which is a form of deep neural networks in which the Lipschitz constant of individual layers is constrained to be smaller than 1. Farnia et al. [47] show that the upper bound of generalization error depends on the Lipschitz constant, which motivates their approach for spectral regularization to make sure the spectral norm of the model weights is smaller than 1. TRADES [48] added a regularizer which is the cross-entropy loss of the logits of the clean example and its adversarial counterpart, and optimized their adversarial example by maximizing this regularizer loss instead of the model classification cross-entropy loss.

Certified Defense Certified defenses provide guarantees of robustness, which is the worst-case robustness against all possible attacks under some threat level. There

is a line of defense works that offer proven guarantee against any attack methods. Raghunathan et al. [49] develop a semidefinite programming-based relaxation of the adversarial polytope and employ it for training a robust classifier. Wong and Kolter [50] provide provably robust networks against l_∞ perturbations by using convex relaxations. Both of these methods apply only to small-scale networks. Wong et al. [51] extend the approach to substantially larger models and more general networks with skip connections and general nonlinearities.

Verification There is also a large body of works to verify the robustness of a model, in terms of providing a minimum adversarial distortion. For ReLU networks, the problem of solving the minimum adversarial distortion can be transformed into a Mixed Integer Linear Programming (MILP) problem by using binary variables to encode the states of ReLU activation in each neuron [52, 53]. However, due to the NP-completeness for solving such a problem, these methods can solve for very small networks. An alternative is to find a non-trivial and certified lower bound of the minimum distortion. Hein and Andriushchenko [54] provides a lower bound for a continuously-differentiable network with one hidden layer using local Lipschitz constant. Weng et al. [55] further provide the lower bound guarantee to non-differentiable functions by Lipschitz continuity assumption and propose the first robustness score. Weng et al. [56] find the certified lower bound of minimum adversarial distortion by computing lower and upper bound for each ReLU unit at every network layer. Zhang et al. [57] later extend the lower bound to more general activation functions.

Other Defenses Feature denoising approaches aim to improve robustness of a model by actively remove added adversarial perturbation from the input, so that the final output of the model for an adversarial example will be the same as the clean example. Liao et al. [58] introduce a denoising U-net which is guided by the noise at the feature level, which is the l_1 difference between features extracted from the clean and adversarial examples. Xie et al. [59] add residual feature denoising blocks to their network.

Leveraging the power of generative models has also proven to be useful for improving adversarial robustness. Schott et al. [60] use a variational autoencoder for each class to model the class-conditional sample distributions $p(x|y)$. A probability distribution of the input is estimated for every class at test time and the final prediction is the class with the highest likelihood. Rob-GAN [61] combines adversarial training with GAN learning to model the adversarial example distribution.

1.1.2 Confidence Calibration

Machine learning methods inevitably make mistakes, and they should indicate when they are likely to be incorrect, which allows the decision-making to be stopped or passed to human experts when the models are not sufficiently confident to produce a correct prediction. In order to improve reliability, it is therefore strongly desirable that a method provides a calibrated confidence measure in addition to its prediction; that is, the confidence should reflect its ground truth likelihood of correctness [62].

The topic of confidence calibration for deep learning was first discussed in the

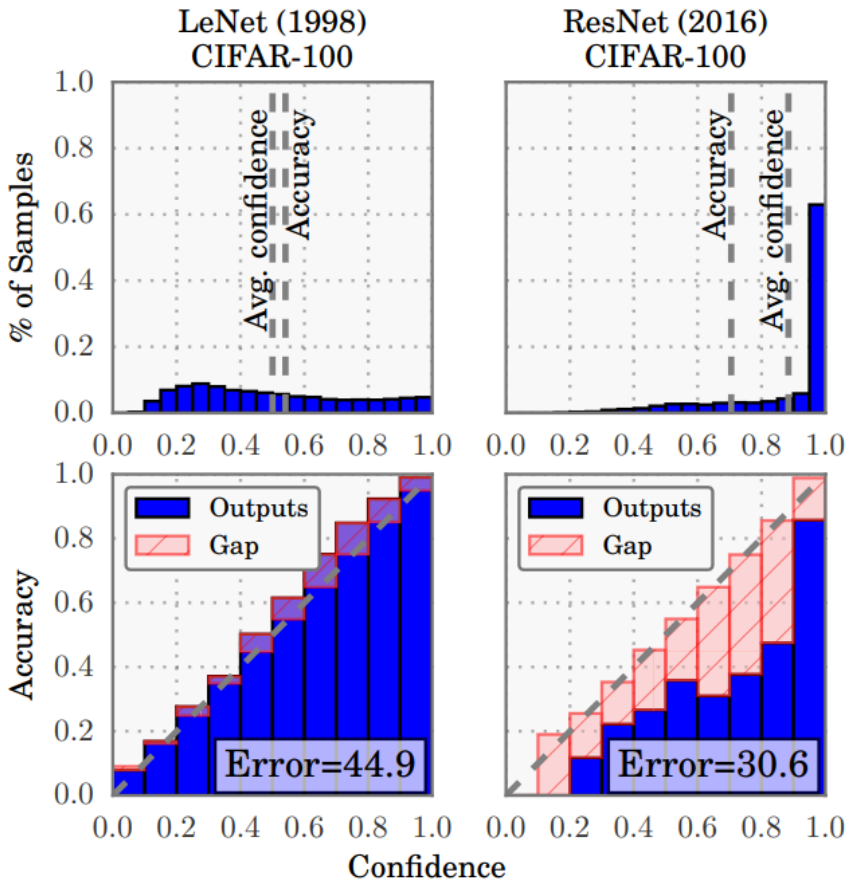


Figure 1.1: Example confidence histograms (top) and reliability diagrams (bottom) for a 5-layer LeNet (left) and a 110-layer ResNet (right) on CIFAR-100 [63].

pioneering work [63]. Shown through extensive empirical studies that even though impressively accurate, modern deep neural networks are poorly calibrated [64].

Ideally, for perfect confidence calibration, we would like the confidence estimate $\hat{P}$ to represent a true probability that indicates the likelihood of correctness, which is defined as

$$\mathbb{P}(\hat{Y} = Y \mid \hat{P} = p) = p, \quad \forall p \in [0, 1] \tag{1.6}$$

But it turns out that modern DNNs are overconfident - the actual prediction accuracy is likely to be lower than what is indicated by the associated confidence.

1.1.2.1 Measuring Calibration

A model's performance on confidence calibration is measured by expected calibration error (ECE).

Histogram-based ECE Histogram-based ECE is the most widely adopted estimator for ECE [62,63]. Let h be a neural network. The model's output probabilities $p_i^k = h(k|x_i)$ are grouped into M interval bins, where p_i^k is the model's output confidence for sample i belonging to class k. Let B_m be the set of indices of which the confidence falls into bin m, i.e. $p_i^k \in (\frac{m-1}{M}, \frac{m}{M}]$. The likelihood and confidence of B_m are defined as

$$\begin{aligned} \text{lik}(B_m) &= \frac{1}{|B_m|} \sum_{(i,k)\in B_m} \mathbb{1}(y_i = k) \\ \text{conf}(B_m) &= \frac{1}{|B_m|} \sum_{(i,k)\in B_m} p_i^k \end{aligned} \tag{1.7}$$

where y_i is true class label for sample i. The histogram-based ECE is defined as

$$\text{Histogram ECE} = \sum_{m=1}^{M} \frac{|B_m|}{nK} |\text{lik}(B_m) - \text{conf}(B_m)| \tag{1.8}$$

A smaller ECE indicates a better calibrated model over all the possible output labels.

KDE-based ECE While a histogram-based estimator is easy to implement, it inevitably inherits drawbacks from histograms, for example being sensitive to the binning schemes and data-inefficient [65]. [66] propose a KDE-based estimator by

replacing histograms with density estimation using a continuous kernel.

Let $\phi : \mathbb{R} \to \mathbb{R}_{\geq 0}$ denote a kernel function and h denote the bandwidth. The density function and canonical calibration function are given by

$$\begin{aligned} f(p) &= \frac{h^{-K}}{n} \sum_{i=1}^{n} \prod_{k=1}^{K} \phi(\frac{p - p_i^k}{h}) \\ c(p) &= \frac{\sum_{i=1}^{n} y_i \prod_{k=1}^{K} \phi(\frac{p-p_i^k}{h})}{\sum_{i=1}^{n} \prod_{k=1}^{K} \phi(\frac{p-p_i^k}{h})} \end{aligned} \tag{1.9}$$

and the KDE-based ECE is computed as

$$\text{KDE ECE} = \int \|p - c(p)\| f(p) dp \tag{1.10}$$

While KDE alleviates the dependence on histogram binning, it heavily depends on the kernel choice and may induce error from the integral approximation procedure. Therefore, we use both metrics to report the performance on confidence calibration.

1.1.2.2 Improving Calibration

Confidence calibration has since raised a great interest in the machine learning community resulting in a great deal of recent work trying to address this issue. Existing calibration techniques can be broadly categorized based on whether or not they are post-hoc methods.

Post-hoc Calibration Post-hoc calibration methods use parametric or non-parametric models to transform the network's predictions based on a held-out validation set to

improve calibration. Traditional techniques include Platt scaling [67] where it learns scalar parameters $a, b \in \mathbb{R}$ and outputs $\hat{q}_i = \sigma(az_i + b)$, where z_i is the model output logit, $\hat{q}_i$ is the calibrated probabilities, and σ is the activation function; Isotonic Regression [68] is a non-parametric calibration method that learns a piecewise constant function to transform uncalibrated outputs; Histogram binning [68] divides uncalibrated predictions p^i into mutually exclusive bins and assigns each bin a calibrated score that corresponds to the average number of positive-class samples in the bin; Bayesian binning [62,68] is an extension of histogram binning using Bayesian model averaging; matrix and vector scaling [63] are two multi-class extensions of Platt scaling; Beta calibration [69] introduces a family of calibration maps that includes both sigmoids and inverse sigmoids to adjust prediction probabilities; and temperature scaling [63] uses a temperature to soften the confidence scores.

Among these methods, temperature scaling consistently outperforms other methods [63]. As an abstraction of Platt scaling, a single parameter T, called the temperature is used to scale all classes. The calibrated confidence prediction is calculated as the function:

$$\hat{q}_i = \max_k \sigma_{SM}(z_i/T)^{(k)} \tag{1.11}$$

where T acts as softening the confidence because, with larger T values, the calibrated probability $\hat{q}_i$ will reach $1/K$ representing uncertainty. However, the strict inequality will remain consistent and thus won't affect the accuracy of the model.

More recent advances include Dirichlet calibration [70] which generalizes the

beta calibration method with Dirichlet distributions; ensemble temperature scaling [66] employs an ensemble of models in temperature scaling to achieve better calibration; mutual information based binning [71] uses a single shared calibrator among similar classes with similar class priors so that the training sets of their class-wise calibration problems can be merged to train the single calibrator; and spline recalibration [72] obtains a recalibration function by approximating the empirical cumulative distribution using a differentiable function via splines, which maps the network outputs to calibrated class assignment probabilities.

Non-post-hoc Calibration Non-post-hoc methods are mostly based on adapting the training procedure. [73] replaces the traditional cross-entropy loss with focal loss to improve calibration. In label smoothing [74, 75], instead of using traditional one-hot labels, uniform smooth labels are used to train the model. The smooth label is defined as

$$y_k^{LS} = y_k(1-\alpha) + \alpha/K \tag{1.12}$$

where α is the smoothing factor and K is the number of classes. Different variants of data augmentation have also shown to improve calibration [76, 77]. They perform linear interpolation on both the input features and the target labels before feeding them to the model.

Bayesian Neural Networks Another line of related work is Bayesian Neural Networks [78–82]. Bayesian methods provide a natural probabilistic representation of uncertainty in deep learning and are well-suited for providing calibrated uncer-

tainty estimation. However, these methods are resource-demanding and hard to scale to modern datasets and architectures [80].

1.1.3 Reproducibility

In the machine learning community, reproducibility normally means the ability to replicate the results using the exact same setup. However, in this thesis, we focus on the concept which is often crucial in the applications in neuroscience, that is a machine learning algorithm should produce stable results across different groups of subjects. In recent years, challenges related to reproducibility of scientific findings have been heavily debated in the literature [83–85]. Statistical power of neuroimaging studies has increased over time but remains relatively low, especially for group comparison studies. Neuroimaging analysis has a high degree of flexibility in analysis methods, which can lead to inflated false-positive rates unless controlled for [86].

Therefore, the reproducibility of an algorithm significantly impacts the reliability of its findings. Unlike some applications where there are ample data available, in neuroscience the data is usually very limited (less than 1000 subjects), and is often corrupted with various source of noise [87, 88], which makes reproducibility a challenging task. When conducting group-level analysis, it is further complicated by the inter-subject variability.

In this thesis, we aim to mitigate the reproducibility issue in extracting large-scale group-level functional networks from fMRI data. The key step in this process

is based on Independent Component Analysis (ICA) [9] to extract independent networks. But because of the high-dimensionality nature of the data, the dimensionality reduction step is necessary to extract meaningful features from group-level data which then determines the reproducibility of the algorithm.

One existing method is to concatenate the data from multiple subjects along the temporal dimension [89] followed by the use of Principal Component Analysis (PCA) [8] to reduce the dimensionality of the data. The tensorial extension to ICA [90] uses a three-dimensional tensor to estimate shared spatial patterns and time courses between subjects, and subject-specific loading of the components to capture the multidimensional structure of the data. In [91], a general framework is proposed for using the expectation maximization algorithm to estimate an unconstrained mixing matrix for the group-level independent components. Canonical ICA [87] uses Canonical Correlation Analysis (CCA) to find components that maximize cross-correlation.

1.2 Overview of Thesis

This thesis consists of four parts.

Feature Prioritization and Regularization Improve Standard Accuracy and Adversarial Robustness In the first part, we focus on the problem of improving the robustness of a deep neural network [92]. The existence of adversarial examples poses a real threat to applying a new method to any safety-critical domains, and the reliability on a non-robust model is very shaky. Although adversarial

training methods have proven to improve the robustness of a deep learning model, there are no explicit design components to encourage it.

In Chapter 2, we propose an approach that enhances adversarial training with feature prioritization and regularization to improve the robustness of a model. We use an attention module that focuses on the area of an image which contains the actual object and demonstrate through extensive experiments that it helps the classifier to only rely on features extracted from those areas. The background clutter and irrelevant features are effectively suppressed. We also employ feature regularization to further encourage the model to extract robust features that are not manipulated by the adversarial perturbations. The resulting model is significantly more robust. It also has a highly interpretable gradient map that aligns very well with salient data characteristics.

Class-Similarity Based Label Smoothing for Generalized Confidence Calibration In the second part, we study the problem of confidence calibration [93]. Deep neural networks are trained with one-hot labels under cross-entropy loss. This combination encourages the model to associate its prediction with the highest possible confidence, therefore the model has a significantly higher confidence about its prediction than the actual accuracy, i.e., the model becomes over-confident. An over-confident model is not reliable in applications because we do not know whether its decisions could be trusted.

In Chapter 3, we first introduce a generalized definition of confidence calibration that requires the entire output to be calibrated. Inspired by directly optimizing

the generalized calibration, we propose a novel form of the smooth labeling algorithm, called class-similarity based label smoothing, which tries to approximate a distribution that is optimal for generalized confidence calibration. In the proposed smooth label of a reference class, the score for another class is based on its similarity with the reference class, and hence more similar classes result in higher values. The similarity between two classes is measured by computing the average distance between all pairs of samples from the two classes, which is then converted into a similarity measure using the softmax function. We show that a model trained with the proposed smooth labels is significantly better calibrated than all existing methods.

Improving Calibration of Robust Models via Prototypical Learning Adversarial robustness and confidence calibration are two important aspects of reliability. However, in practice, they are studied relatively independent of each other. Even if a model is decently robust, it could still make errors with high confidence. Therefore, having a calibrated robust model could significantly decrease the harmfulness in case of misclassifications. In this chapter, we make the first attempt to bridge this gap and study how to improve confidence calibration while maintaining adversarial robustness.

In Chapter 4, we propose a novel approach that is able to improve confidence calibration while maintaining comparable adversarial robustness performance with state-of-the-art methods. We first employ a prototypical network with prototypical learning to learn a representation space in which the distance from a sample representation and the representations class prototypes is used for classification. We then

use this distance to train a confidence prediction network to provide a calibrated confidence output. We show that our model is significantly more calibrated than existing robust models while the robustness is still very comparable.

LEICA: Laplacian Eigenmaps for Group ICA Decomposition of fMRI Data In the area of neuroscience, it has been established that functional magnetic resonance imaging (fMRI) signals provide fundamental insights about functional organization of the brain, and determining large-scale patterns of brain activity from fMRI is an important problem which also serves as the basis of downstream research. Classical linear machine algorithms such as principal component analysis (PCA) and independent component analysis (ICA) have been applied to this problem with some success. However, because the underlying signal dynamics involve multiple nonlinear sources, linear methods are not well suited for the problem. Also, the analysis is often carried out at the group level containing a number of subjects with a similar demographic, and it is crucial to eliminate the inter-subject variability in order to get intrinsic brain patterns. The existing methods suffer from the problem of reproducibility, i.e., the results are not reproducible across different subjects in the same group. If the patterns are not reproducible, it means that they are not closely related with the intrinsic brain dynamics, and significantly vary across different subjects, and therefore the results are not reliable for downstream studies. In the third part of this thesis, we focus on extracting more reproducible group-level functional patterns for fMRI data [94].

In Chapter 5, we propose a group-level model, called LEICA (Laplacian Eigen-

maps for group ICA decomposition), that uses Laplacian eigenmaps as the main data reduction step, which preserves the correlation information in the original data as best as possible in a certain rigorous sense. The nonlinear map is robust relative to noise in the data and to inter-subject variability. After the Laplacian eigenmaps transformation, ICA is applied to the reduced data to estimate spatial features, followed by thresholding to extract group-level independent spatial maps. We show that LEICA detects functionally cohesive maps that are much more reproducible than the state-of-the-art methods.

Conclusion and Future Work Finally, in Chapter 6, we conclude this thesis and discuss the future work.

Chapter 2: Feature Prioritization and Regularization Improve Standard Accuracy and Adversarial Robustness

2.1 Introduction

Deep learning models have demonstrated impressive performance in a wide variety of applications [95, 96]. However, recent works have shown that these models are susceptible to adversarial attacks: imperceptible but carefully chosen perturbation added to the input can cause the model to make highly confident but incorrect predictions [17–19].

Exploring the adversarial robustness of neural networks has recently gained significant attention and there is a rapidly growing body of work related to this topic [19, 22, 24, 25, 50, 97–99]. A wide variety of methods are proposed to defend a model against adversarial attacks [58, 100–102]. Despite these advances, many techniques are subsequently shown to be ineffective [28, 39], and adversarial training which uses adversarial samples in addition to clean images during the training process has been shown to be able to build relatively robust neural networks [25, 28, 103]. With strong adversaries such as the Projected Gradient Descent (PGD) [25] or the Iterative Fast Gradient Sign Method (I-FGSM) [19] adversarially trained models are

able to achieve state-of-the-art performance against a wide range of attacks.

Recent advances in the understanding of adversarial training provide insights of its effectiveness. It is shown that standard and robust models depend on very different sets of features [104, 105]. While standard models utilize features including non-robust ones that are weakly correlated with class labels and easily manipulated by small input perturbations, robust models only use robust features that are highly correlated with class labels and invariant to those perturbations. Although adversarial training learns robust features, there are no explicit design components to encourage a model to depend solely on robust features. Therefore, to further improve the robustness of a model, we propose feature regularization and prioritization schemes.

We first propose to use an attention mechanism that introduces to the model the flexibility of prioritizing features and bias a model towards robust features. We call the learned features at the final layer of a network the global features, and the ones at lower level layers the local features. In our attention module, we use the global features as a way to assign weights to the local features by a non-linear compatibility function. Since global features are directly used to produce class label prediction, we are effectively assigning weights to local features depending on their correlation with the labels. Robust features have higher correlation and therefore will be assigned larger weights which in turn contribute to the model's robustness.

Next, we propose to use feature regularization to learn robust features that are invariant to input perturbations. We add an L_2 regularization term that penalizes the distance between the learned features of a clean sample x and that of

its perturbed adversarial counterpart x' to the training objective. By optimizing this regularizer, we are pushing the model to extract very similar features from the original image and the adversarial image, and thus only features that are invariant to the perturbations are learned and the added noise is effectively ignored. From another point of view, a model with small L_2 feature distance maps the two nearby points in the image space to nearby points in the learned high dimensional manifold, which is a desirable behavior.

In this chapter, we describe our approach that enhances adversarial training with feature prioritization and regularization to improve the robustness of a model. We use extensive experiments to demonstrate that the attention module focuses on the area of an image which contains the actual object and helps the classifier to only rely on features extracted from those areas. The background clutter and irrelevant features which could be misleading are effectively suppressed. The feature regularization further encourages the model to extract robust features that are not manipulated by the adversarial perturbations. The resulting model has a highly interpretable gradient map that aligns very well with salient data characteristics.

The main contributions of this chapter include:

- A method based on feature prioritization and regularization, which significantly outperforms adversarial training. Our model is evaluated on the MNIST, CIFAR-10, and CIFAR-100 datasets, and demonstrates superior performance relative to both standard classification accuracy and adversarial robustness.

- We provide empirical justification for attention and show that it helps the

model to rely on robust features by assigning larger weights to them. Through qualitative inspection, we show that the attention maps generated by our non-linear attention estimator focus sharply on the regions of interest while suppressing irrelevant background clutter.

- In addition to qualitative evaluation of the gradient maps, we propose a novel experimental strategy that quantitatively demonstrates better alignment of the gradient maps generated by our model with salient data characteristics.

2.2 Related Work

Due to the extensive amount of literature in this area, we only review some of the most related works in this section. For a more comprehensive survey, please refer to Akhtar and Mian [106].

Adversarial training. Kurakin et al. [19] use adversarial training as a form of data augmentation where it injects adversarial examples during training. In every training mini batch, a mixture of clean images and adversarial images generated by one step Fast Gradient Sign Method (FGSM) are used to update the network's parameters. It was later improved by Na et al. [41] by adding adversarial examples generated by iterative methods. Madry et al. [25] proposed to replace all clean images with adversarial images which is a direct result of optimizing a saddle point (min-max) formulation. They suggest that PGD is a universal first-order adversary which is then used in their adversary generating process.

L_2 regularization. A similar idea with feature regularization is proposed in

Kannan et al. [45] which they call adversarial logit pairing (ALP), to prevent a model from being over-confident when making predictions. Compared with ALP, feature regularization is more intuitive as it motivates a model to learn very robust features that are invariant to input perturbations, which leads to a robust model. In addition, we propose to also use attention module to further encourage the model to favor robust features which will improve the robustness.

Attention Models. Attention in CNN is most commonly deployed for query-based tasks [107, 108]. Jetley et al. [108] presented a method to use a learned representation of the global image as a query to leverage multiple attention maps at different scales, which allows the expression of a complementary focus on different parts of the image. However, the application of attention to the adversarial robustness aspects has not been seriously explored. To the best of our knowledge, we are the first to employ an attention mechanism in training a robust deep neural network. In our application, we use a ReLU activated neural network instead of the linear-based method as the attention estimator. It allows highly non-linear compatibility between the learned global features and the lower-level local features.

2.3 Approach

We now present our model that combines the attention module and L_2 feature regularization, and show how it can be applied to enhance the adversarial training to improve the adversarial robustness of a model and its accuracy. Figure 2.1 provides an overview of our method. We start by forwarding each of the clean and

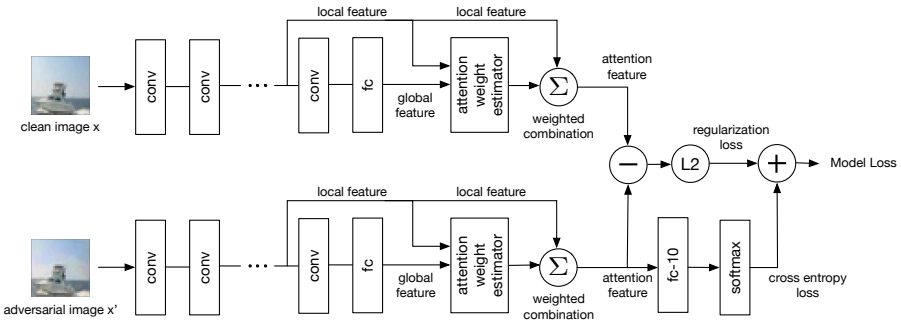


Figure 2.1: Overview of the proposed model. The top and bottom networks are the same copy that share all network parameters. Both the clean and adversarial images are forwarded through the network to produce the corresponding attention features. The L_2 regularization loss is defined as the Euclidean distance between the two sets of attention features. The final model loss is a combination of the L_2 regularization loss and the cross-entropy loss for only the adversarial input.

adversarial images and computing the attention weights by a non-linear estimator. Then the individual attention feature is defined to be the weighted combination of the corresponding local features. Next, we define an L_2 regularization loss to be the Euclidean distance between the two sets of learned attention features. The attention features of the adversarial image are then used to produce the logits, which is followed by softmax layer to produce the cross-entropy loss. The final loss function of our model is a combination of cross-entropy loss and the regularization loss.

2.3.1 Adversarial Training

We adopt the adversarial training described in Madry et al. [25] as the basic training approach. It replaces natural training examples by PGD examples, which is suggested to represent a universal first-order adversary. So far PGD has been shown to represent the strongest attack method [28, 39]. A model that is trained with PGD adversaries is also robust against a wide range of other attacks and not

yet outperformed by any other approach. The adversarial training has a saddle point formulation:

$$\min_{\theta} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[\max_{\boldsymbol{\delta}\in S} L(\theta, \boldsymbol{x}+\boldsymbol{\delta}, y)] \tag{2.1}$$

where $\mathcal{D}$ is the distribution of data $\boldsymbol{x}$ and class labels y, L is the cross-entropy loss function for a model with parameters θ, $\boldsymbol{\delta}$ is the additive adversarial perturbation with bound S. In this paper we consider l_∞ bound as in Madry et al. [25]. Our adversarial samples $\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{\delta}$ are created by PGD:

$$\boldsymbol{x}^{t+1} = \Pi_{\boldsymbol{x}+S}\left(\boldsymbol{x}^t + \alpha\,\mathrm{sgn}(\nabla_{\boldsymbol{x}} L(\theta, \boldsymbol{x}, y))\right) \tag{2.2}$$

PGD adversaries are computed at each iteration as an approximated optimum of the inner maximization in equation (2.1) and an update of the parameters θ is made according to the outer minimization formulation.

2.3.2 Attention Model

As we discussed in Section 2.1, our goal of attention model is to favor robust features in making predictions. We propose a non-linear attention model that acts as a feature prioritizing scheme, which is able to put more weight on robust features and less weight on non-robust features to increase the robustness of a classifier.

Let $\boldsymbol{l}_n^i$ be the learned feature vector at layer $i \in \{1, 2, ..., I\}$ of a neural network at spatial location $n \in \{1, 2, ..., N\}$, and let $\boldsymbol{g}$ be the feature vector of the layer just

before the final fully connected layer which produces the class label prediction scores (logits). We use a one-hidden-layer ReLU network to generate compatibility scores between the global feature $\boldsymbol{g}$ and local features $\boldsymbol{l}_n^i$:

$$c_n^i = f(\boldsymbol{l}_n^i, \boldsymbol{g}) \tag{2.3}$$

where f is the neural network and the concatenation of $\boldsymbol{g}$ and $\boldsymbol{l}_n^i$ is fed to the network to produce the compatibility scores c_n^i. We then normalize the scores with a softmax operation to get the attention weights:

$$w_n^i = \frac{\exp c_n^i}{\sum_m \exp c_m^i} \tag{2.4}$$

Next, we compute the weighted sum of local feature vectors which is the attention feature vector at layer i:

$$\boldsymbol{h}^i = \sum_n w_n^i \boldsymbol{l}_n^i \tag{2.5}$$

We use the outputs of the last residual block as the local feature for computing attention, and replace the global feature $\boldsymbol{g}$ with the corresponding attention descriptor $\boldsymbol{h}^i$ for final classification.

By using a neural network instead of the linear alignment models as in Jetley et al. [108], we are able to capture non-linear compatibility between the local and global features when producing the attention weights, which is beneficial considering the multiple non-linear function activated layers between the local and global features.

2.3.3 Feature Regularization

In addition to the attention mechanism, we also propose to use an L_2 regularization term to encourage the model to extract similar features for the clean image and the corresponding adversarial image. Denote by $\mathcal{G}_\theta$ the deep neural network, $\boldsymbol{x}$ and $\boldsymbol{x}'$ the natural image and adversarial image. Denote by $\mathcal{G}_\theta(\boldsymbol{x})$, $\mathcal{G}_\theta(\boldsymbol{x}')$ the learned features of the layer just before the final fully connected layer (in our case this is the attention weighted global descriptor) which produces the class label prediction scores. The L_2 regularizer has the following form:

$$L_r(\boldsymbol{x}, \boldsymbol{x}') = \|\mathcal{G}_\theta(\boldsymbol{x}) - \mathcal{G}_\theta(\boldsymbol{x}')\|_2 \tag{2.6}$$

By minimizing the regularization function, the model effectively learns very similar features for the clean sample and the adversarial sample, which are robust features since they are invariant to the adversarial perturbation. From another perspective, the learned features of a neural network lie on a high dimensional manifold that is linearly separable for different classes because the classification layer is a linear classifier followed by a softmax function. With adversarial training alone, a model only tries to map $\boldsymbol{x}$ and $\boldsymbol{x}'$ to the same side of the decision boundary, while with the additional regularization, they are not only on the same side but also mapped to nearby points in the space. This mapping is a desired behavior considering that, in the original image space, they are very close points representing essentially the same image.

2.3.4 Model Loss

Equipped with the presented methods, the total loss of our model is:

$$\text{Loss} = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[L(\theta, \boldsymbol{x}', y) + \lambda\|\mathcal{G}_\theta(\boldsymbol{x}) - \mathcal{G}_\theta(\boldsymbol{x}')\|_2] \tag{2.7}$$

where λ is a hyperparameter that controls the relative weight of the L_2 regularization loss.

2.4 Experiments and Results

In this section, we evaluate our model on the MNIST, CIFAR-10 and CIFAR-100 datasets, and present empirical justification to attention module and some quantitative and qualitative results.

2.4.1 Robustness on MNIST

We use a CNN with two convolutional layers with 32 and 64 filters respectively, followed by two fully connected layers of size 1024 and 10. The network is trained with 40-step PGD adversary with a step size of 0.01 and l_∞ bound of $\epsilon = 0.3$. The settings are the same as in Madry et al. [25]. Since MNIST is a very small scale dataset and the model is very robust with just adversarial training, we do not employ the attention mechanism, but only study the effectiveness of the proposed feature regularization method.

The evaluation results are presented in Table 2.1. Regarding the value of

Method	Natural	White box	Transfer
Madry et al.	98.72%	92.86%	95.97%
AT-reg	**98.97%**	**95.95%**	**96.90%**

Table 2.1: Performance comparison of the adversarial training and our proposed adversarial training with feature regularization (AT-reg) on MNIST against PGD 5 adversaries. Transfer attack accuracies are evaluated against adversaries generated from an independently trained copy of the same method with identical configurations.

weight λ of the L_2 regularizer, we find that roughly any $\lambda \in [0.001, 0.1]$ works well. The reported results are obtained with $\lambda = 0.1$. Table 2.1 shows that a model trained with the proposed feature regularization method is significantly more robust against PGD adversary than the baseline model with adversarial training alone. The improvement is more than 3% for white box attack and 1% for transfer attack.

2.4.2 Robustness on CIFAR-10

We use the same wide residual network as Madry et al. [25] with [16, 16, 32, 64] filters and its 3-times wider variant with [16, 48, 96, 192] filters respectively. For our attention model, we modify the ResNet by replacing the spatial global average pooling layer after the residual block 4 with a convolutional layer sandwiched between two max-pooling layers to obtain the global feature $\boldsymbol{g}$. We use a one-hidden-layer ReLU neural network with 64 hidden units as the non-linear attention weight estimator. The model is trained with 5-step PGD adversary and a step size of 2 and l_∞ bound of $\epsilon = 8/255$. In order to isolate and analyze the effectiveness of attention module and feature regularization independently, we train three models with the following configurations: adversarial training with feature regularization (AT-reg),

Method	Madry et al.	AT-reg	AT-att	AT-att-reg
Natural	80.79%	79.52%	**82.43%**	81.20%
White, PGD 5	49.89%	52.35%	51.22%	**53.38%**
White, PGD 20	39.72%	44.25%	41.40%	**45.20%**
White, PGD 100	38.76%	43.73%	40.85%	**44.60%**
White, PGD 200	38.64%	43.70%	40.72%	**44.54%**
White, CW 30	40.27%	42.96%	40.60%	**44.37%**
White, CW 100	39.98%	42.87%	40.31%	**44.26%**
Transfer, PGD 5	60.13%	61.82%	**63.26%**	62.32%
Transfer, PGD 20	56.60%	56.40%	**58.26%**	57.52%
Transfer, PGD 100	56.44%	56.28%	**58.08%**	57.53%
Transfer, PGD 200	56.49%	56.25%	**58.04%**	57.53%
Transfer, CW 30	57.11%	56.86%	**58.07%**	57.57%
Transfer, CW 100	57.10%	56.79%	**58.13%**	57.41%

Table 2.2: Performance comparison of the adversarial training, adversarial training with feature regularization (AT-reg), adversarial training with attention (AT-att), and adversarial training with both (AT-att-reg) on CIFAR-10 against PGD and CW adversaries.

adversarial training with attention (AT-att), and adversarial training with both attention and feature regularization (AT-att-reg). We then evaluate the models using PGD and CW [24] adversaries optimized with various number of steps under white box and transfer attack settings. Transfer attack accuracies are evaluated against adversaries generated from an independently trained copy of the same method with identical configurations.

The evaluation results of aforementioned narrow and 3-times wide models on CIFAR10 are presented in Table 2.2 and Tabel 2.3. We find that roughly any $\lambda \in [0.01, 1]$ works well for feature regularization. The reported results are obtained with $\lambda = 1$ for AT-reg and $\lambda = 0.1$ for AT-att-reg. From the table, we see that all of the three proposed models have better adversarial robustness over the baseline

Method	Madry et al.	AT-reg	AT-att	AT-att-reg
Natural	85.41%	84.65%	**86.48%**	85.98%
White, PGD 5	49.15%	52.21%	50.91%	**53.23%**
White, PGD 20	38.19%	41.00%	39.52%	**41.55%**
White, PGD 100	37.39%	40.28%	38.98%	**40.78%**
White, PGD 200	37.20%	40.24%	38.89%	**40.67%**
White, CW 30	38.92%	**42.20%**	40.75%	42.12%
White, CW 100	38.71%	41.88%	40.32%	**42.06%**
Transfer, PGD 5	68.03%	68.00%	**69.78%**	69.00%
Transfer, PGD 20	62.77%	63.14%	**64.70%**	64.01%
Transfer, PGD 100	62.65%	63.12%	**64.78%**	64.15%
Transfer, PGD 200	62.73%	63.12%	**64.69%**	64.11%
Transfer, CW 30	63.89%	64.23%	**65.65%**	64.50%
Transfer, CW 100	63.77%	64.04%	**65.43%**	64.35%

Table 2.3: Performance comparison of the adversarial training, adversarial training with feature regularization (AT-reg), adversarial training with attention (AT-att), and adversarial training with both (AT-att-reg) on CIFAR-10 using the 3-times wide ResNet network against PGD and CW adversaries.

model that only uses adversarial training, and both models with attention show improvement on the classification accuracy on natural examples as well. Note that the white box accuracy of the 3-times wide models are lower than the narrow models, which is due to overfitting of adversarial training [109], and could be solved by early stopping.

We assess the effectiveness of feature regularization by comparing baseline to AT-reg in both Tables. The robustness is significantly improved especially under stronger attack with more steps, more than 5% in some cases.

Next, by comparing the results of models with and without the attention module, we can see that attention contributes to both standard and adversarial accuracy such that the robustness improves by 1.5% and at the same time standard accuracy

by a similar margin. The attention structure not only favors robust features, it also relies heavily on features extracted from the spatial area that contains the actual object of concern. By suppressing the background clutter and misleading perturbations in irrelevant areas, the model with attention module more precisely learns the underlying distribution of the data which leads to better accuracy.

AT-att-reg takes advantage of both techniques and offers the most improvement in robustness and also better standard accuracy. We think attention and feature regularization complements each other. While regularization encourages the model to learn robust features from the input, attention assigns larger weights to robust features to make the model rely even more on them.

2.4.3 Empirical Justification for Attention

First of all, in order to show the advantage of our proposed non-linear attention model over the linear alignment model in Jetley et al. [108], we train two narrow ResNet on natural CIFAR-10 samples with the two attention schemes respectively and the accuracy is 92.87% for our attention and 91.34% for the linear attention model.

Next, we empirically demonstrate that the attention module assigns larger weights to more robust features by examining the attention weights relative to the feature robustness. Figure 2.2 shows the relationship between the robustness of a feature and the magnitude of its assigned attention weight. The robustness measure we use is the L_2 distance between the learned features of a clean and an adversarial

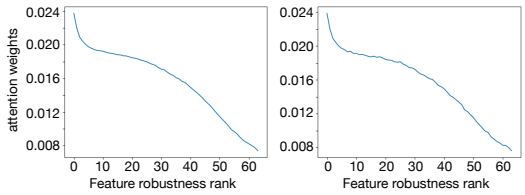


Figure 2.2: The relationship between attention weights and feature robustness. The horizontal axis is the robustness rank, with 0 being the most robust and 63 the least robust, and the vertical axis is the corresponding attention weights. Left plot shows the results for training set and right plot is for test set of CIFAR-10.

image, i.e. the smaller the distance between the features, the more invariant the feature is against input perturbations, therefore the feature is more robust. We rank the features accordingly and compute the average attention weights for each feature across all images. As shown in Figure 2.2, our proposed attention mechanism indeed assigns larger weights to more robust features and less weights to non-robust features, so the model is more invariant to adversarial perturbation.

Figure 2.3: The learned attention maps of our model. The first row are the input images and the second are the attention maps learned at residual block 4.

Finally, we show the attention maps of our model in Figure 2.3 to visualize the attention weights. The attention maps focus sharply on the objects in the images, and the most relevant features like the head and legs of an animal and the wings of an airplane contribute more to the model's prediction and ignore the irrelevant background clutter.

2.4.4 Gradient Map

In this section, we study the gradient maps, which are the gradients of the cross-entropy loss with respect to input image pixels. The gradient maps directly indicate how the input features are utilized by a model for prediction and highlight the features which affect the loss most strongly. A large gradient on an input feature signifies a heavy dependence from the model. Human vision is robust against small input perturbations and the perception of an image reflects the reliance on input features. Therefore, models that depend on robust features will be better aligned with human vision, and the alignment can be used to evaluate the robustness of a model. Next, we show that the gradient maps generated from our model align better with the salient data characteristics.

First we present the qualitative result. Figure 2.4 shows the gradient maps from Madry et al. [25] and our model. Overall, we note that gradient maps from both models are highly interpretable and align very well with the image features. Upon careful inspection, it is evident that the gradient maps generated from our model are better than the baseline. To point out a few, note that, in columns 2, 3, 6 and 8, our gradient maps have cleaner backgrounds and the gradients only focus on the objects; especially in column 6, the baseline model has large gradients on the text field in the background which is irrelevant to the class label (automobile), while in our model gradients in that area are much more suppressed. In columns 1, 5, and 9 the edges of the faces and heads of the animals are depicted clearer in our model. Human inspection which could be very subjective, therefore, we also

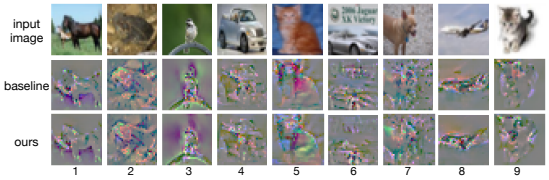


Figure 2.4: Original CIFAR-10 images (top row) and corresponding gradient maps from Madry et al. [2017] model (mid row) and our model (bottom row). The raw gradients are clipped to within ± 3 standard deviation and rescaled to lie in the [0, 1] range for visualization.

introduce a quantitative evaluation method for gradient maps.

	With clipping		**Without clipping**	
Method	**Train data**	**Test data**	**Train data**	**Test data**
Madry et al.	27.10%	26.78%	28.60%	28.72%
Ours	**30.11%**	**30.32%**	**31.46%**	**31.59%**

Table 2.4: Classification accuracy on the gradient maps from baseline and our model on both the training and test set of CIFAR-10. We run the experiment on gradient maps both with and without clipping to avoid the influence of gradient clipping.

The problem we consider here is to decide how well the gradient maps align with the original images. The better they align, the more recognizable the gradient images are. A standard neural network extracts relevant features and make predictions based on them. When a gradient map is highly aligned with the original image, the neural net is able to identify more relevant features and thus the classification accuracy will be higher. Therefore, we can quantitatively compare the alignment by the classification accuracy of gradient maps. We use a pretrained network to classify the gradient maps for images in both the training set and the test set.

We pretrained the same ResNet model as in Section 2.4.2 with only natural training data of CIFAR-10. It achieves 88.79% accuracy on the test set. The

classification results are presented in Table 2.4. To avoid the possible influence of gradient clipping we also run the evaluation on raw gradients. As demonstrated by the classification accuracy, the gradient maps from our model express significantly better alignment with the original images.

To summarize, both the qualitative and quantitative results show that the gradient maps from our model have better interpretability and alignment with the original images. It suggests that our model depends on robust features of the input images which explains the improved performance on both standard accuracy and adversarial robustness.

2.4.5 Results on CIFAR-100

Here we present our results on the CIFAR-100 dataset. The experiment setup is the same as CIFAR-10 in Section 2.4.2.

From Table 2.5, similar with CIFAR-10, our model outperforms baseline by as much as 4% on CIFAR-100 dataset.

2.4.6 Comparison of Feature Regularization with Adversarial Logit Pairing

Though feature regularization is more intuitive than adversarial logit pairing (ALP) as explained in Section 2.2, their performance under PGD attack is similar. However, because ALP is based on logits, it is not as robust as feature regularization under CW attack [24], which is based on logit margin.

Method	Madry et al.	AT-reg	AT-att	AT-att-reg
Natural	52.70%	49.53%	**53.67%**	50.66%
White, PGD 5	25.14%	26.99%	26.33%	**27.76%**
White, PGD 20	19.65%	23.16%	20.82%	**23.80%**
White, PGD 100	19.47%	23.07%	20.59%	**23.62%**
White, PGD 200	19.41%	22.96%	20.53%	**23.62%**
White, CW 30	18.64%	20.78%	19.39%	**20.88%**
White, CW 100	18.61%	20.63%	19.26%	**20.76%**
Transfer, PGD 5	35.37%	35.10%	**35.95%**	35.17%
Transfer, PGD 20	31.99%	31.88%	**32.48%**	32.04%
Transfer, PGD 100	32.03%	31.84%	**32.38%**	32.00%
Transfer, PGD 200	32.00%	31.80%	**32.37%**	32.06%
Transfer, CW 30	32.50%	31.96%	**32.75%**	29.81%
Transfer, CW 100	32.46%	31.86%	**32.74%**	29.71%

Table 2.5: Performance comparison of the adversarial training, adversarial training with feature regularization (AT-reg), adversarial training with attention (AT-att), and adversarial training with both (AT-att-reg) on CIFAR-100 against PGD and CW adversaries.

Method	ALP		AT-reg	
Adversary	CW 30	CW 100	CW 30	CW 100
Accuracy	39.17%	39.15%	**42.96%**	**42.87%**

Table 2.6: Performance comparison of ALP and feature regularization on CIFAR-10 against CW adversaries.

As shown in Table 2.6, AT-reg outperforms ALP by around 3.8% against CW attacks.

2.5 Conclusion

In this chapter, we propose feature prioritization and regularization to enhance both standard classification accuracy and adversarial robustness of a model over the baseline adversarial training approach. We provide empirical justifications for

attention to show that it effectively favors robust features and focuses sharply on the region of interest. We then conduct quantitative and qualitative evaluation on gradient maps to show that they align perfectly with salient data characteristics, which further proves that our model heavily relies on the robust features.

Chapter 3: Class-Similarity Based Label Smoothing for Generalized Confidence Calibration

3.1 Introduction

Machine learning algorithms have progressed rapidly in recent years and are becoming the critical component in a wide variety of technologies [95] such as object detection [110], machine translation [111], medical diagnosis [13, 94], and autonomous driving vehicles [112]. In most of these applications, making wrong decisions could lead to very high costs, including significant business losses or even severe human injuries [113]. As a result, in real-world decision-making systems, machine learning models should not only try to be as accurate as possible, but also should indicate when they are likely to be incorrect, which allows the decision-making to be stopped or passed to human experts when the models are not sufficiently confident to produce a correct prediction. It is therefore strongly desirable that a network provides a calibrated confidence measure in addition to its prediction; that is, the probability associated with the predicted class label should reflect its ground truth likelihood of correctness [62].

However, recent works [63, 64] have shown through extensive empirical studies

that even though impressively accurate, modern deep neural networks are poorly calibrated. It turns out that modern DNNs are overconfident - the prediction accuracy is likely to be lower than what is indicated by the associated confidence. Since the discovery of this challenging problem, several methods have been explored and empirically shown to improve confidence calibration. Temperature scaling [63] is a simple post-processing step that consists of multiplying the logits by a scalar before applying the softmax operator. Uniform label smoothing [74, 75] smooths one-hot labels by distributing a small amount of mass over the other classes uniformly. Mixup [76] trains a classifier not only on the training data but also on linear interpolations of random pairs of samples and their labels. While these techniques have shown some improvements empirically, they are not specifically designed for confidence calibration, i.e., confidence calibration is not the objective function they optimize.

In this chapter, we identify a major component that is missing from the existing confidence calibration methods and introduce a new generalized definition of confidence calibration. We then describe our method that is designed to directly optimize the calibration objective. We refer to the existing notion of confidence calibration as *prediction calibration*, for which only the model's prediction (the winning class) and its associated confidence (the maximum softmax score) are considered. However, only considering the prediction is an oversimplification of the confidence calibration problem. In the output of a neural network, each of the scores represents the model's predicted probability of the current input belonging to that corresponding class. Therefore, for a well-calibrated model, each probability score should be

indicative of the actual likelihood of the input belonging to each class, not just the one with the maximum score. We refer to this generalized confidence calibration as *output calibration* where the entire output is taken into consideration.

The output scores define a probability distribution over all classes, and we denote the perfectly calibrated distribution as the optimal output distribution. Under the optimization scheme with cross-entropy loss that most modern DNNs adopt, a model would achieve perfect confidence calibration if directly trained using the optimal distribution as input label. Unfortunately, as one would expect, the optimal distribution depends on various factors and cannot be directly computed in general. However, under certain reasonable assumptions, we can develop good approximations. Different classes are not equally distinct, and a class can be inherently more similar with some classes than others. For example, in the CIFAR-100 dataset, we could generally agree that the class dolphin is much more similar to the class seal than to the class rose, and the probability of a seal should be higher than a rose in the output distribution of a dolphin input. Therefore, we make the following two assumptions:

1. The distribution of other (i.e. non ground-truth) classes is non-uniform in general.

2. The probability value should correlate positively with the similarity between the true class and any other class, i.e. the more similar they are the higher the value.

In this paper, we propose a novel form of the smooth labeling, called *class-similarity*

based label smoothing, which uses class similarities to approximate the optimal distribution. In the proposed smooth label of a reference class, the score for another class is based on its similarity with the reference class, and hence more similar classes result in higher values.

From the label smoothing perspective, our proposed smooth label is more intuitive than the traditional smooth label which we refer to as the uniform smooth label. One-hot labels have all probability mass in one class, which are zero-entropy signals that admit no uncertainty about the input. When a neural network is trained using such labels it inevitably becomes overconfident. With uniform smooth labels, the output of a network is trained to be a mixture of a Dirac distribution and a uniform distribution. This implies that the predicted probabilities for other classes are encouraged to be equal [114]. As we discussed above, the label value should be based on the similarity between classes, which is not accounted for in either the one-hot or the uniform smooth labels. Relational information can be crucial and provides high-order properties that can improve performance of a model in various tasks [115].

In our proposed method, we first measure the similarity between classes which is then mixed with the one-hot labels to serve as the final smooth label. Since measuring similarity is still a broadly open problem [116, 117], we adopt different metrics, including a new notion of semantic similarity, and evaluate their effectiveness in confidence calibration on various data benchmarks and architectures.

The main contributions of this chapter include:

- We identify a major missing component from the existing confidence calibration and introduce a new generalized definition of confidence calibration, where not only the prediction but the entire output of a neural network are taken into consideration.

- We propose a novel form of label smoothing which we call class-similarity based label smoothing, a technique based on the inherent class similarities estimated by a number of measures, including a novel one that captures semantic similarity. Training a model using the proposed smooth labels directly optimizes the confidence calibration objective, and as a result, the model is significantly better calibrated compared with other techniques.

- We perform extensive experiments to demonstrate the effectiveness of our proposed method in confidence calibration, along with evaluating important hyper-parameters and predictive uncertainty on out-of-distribution data.

3.2 Confidence Calibration

In this section, we formally introduce the definition of confidence calibration and the measurements. We denote the input as $X \in \mathcal{X}$ and label as $Y \in \mathcal{Y} = \{1, 2, ..., K\}$. Let h be a neural network with $h(Y|X)$ being the output, where $h(k|x)$ is the confidence estimate of sample x belonging to class k. The prediction of the network is the winning class $\hat{Y} = \arg\max h(Y|X)$ and its associated confidence is the maximum confidence $\hat{P} = \max h(Y|X)$. The number of samples is n.

3.2.1 Prediction Calibration

We refer to the traditional confidence calibration as prediction calibration where only the winning class and its associated confidence are considered. For prediction calibration, we would like the confidence estimate $\hat{P}$ to represent a true probability that indicates the likelihood of correctness. Perfect prediction calibration is defined as

$$\mathbb{P}(\hat{Y} = Y \mid \hat{P} = p) = p, \;\; \forall p \in [0, 1] \tag{3.1}$$

Since the probability above cannot be computed using finitely many samples, prediction calibration is empirically measured as follows. The model's predictions $\hat{y}_i$ are grouped into M interval bins each of size $1/M$ according to the associated confidences $\hat{p}_i$. Let B_m be the set of indices of samples whose prediction confidence falls into bin m, i.e. $\hat{p}_i \in (\frac{m-1}{M}, \frac{m}{M}]$. The accuracy and confidence of B_m are defined as

$$\begin{aligned} \text{acc}(B_m) &= \frac{1}{|B_m|} \sum_{i \in B_m} \mathbb{1}(y_i = \hat{y}_i) \\ \text{conf}(B_m) &= \frac{1}{|B_m|} \sum_{i \in B_m} \hat{p}_i \end{aligned} \tag{3.2}$$

where y_i is true class label for sample i and $|B_m|$ denotes its size. The prediction calibration is measured by the Expected Calibration Error (ECE) which is defined as

$$\text{Prediction ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} |\text{acc}(B_m) - \text{conf}(B_m)| \tag{3.3}$$

A smaller ECE indicates a better calibrated model.

3.2.2 Output Calibration

In this section we introduce a generalized confidence calibration which we refer to as output calibration. The output of a neural network $h(Y|X)$ is the confidence estimate for all classes. For a calibrated model, we would like not only the maximum but all confidences to be calibrated, which means that $h(k|X)$ represents the actual likelihood of $Y = k$ for all classes k. Perfect output calibration is defined as

$$\mathbb{P}\left(Y = k \,|\, h(k|X) = p\right) = p, \;\; \forall p \in [0, 1] \text{ and } \forall k \in \{1, 2, ..., K\} \tag{3.4}$$

Note that output calibration infers prediction calibration but not vice versa. We measure the output calibration of a neural network as follows. The model's output confidences $p_i^k = h(k|x_i)$ are grouped into M interval bins, where p_i^k is the model's output confidence for sample i belonging to class k. Let B_m be the set of indices pairs of which the confidence falls into bin m, i.e. $p_i^k \in (\frac{m-1}{M}, \frac{m}{M}]$. The likelihood and confidence of B_m are defined as

$$\begin{aligned} \text{lik}(B_m) &= \frac{1}{|B_m|} \sum_{(i,k) \in B_m} \mathbb{1}(y_i = k) \\ \text{conf}(B_m) &= \frac{1}{|B_m|} \sum_{(i,k) \in B_m} p_i^k \end{aligned} \tag{3.5}$$

where y_i is true class label for sample i. The ECE for output calibration is defined as

$$\text{Output ECE} = \sum_{m=1}^{M} \frac{|B_m|}{nK} |\text{lik}(B_m) - \text{conf}(B_m)| \tag{3.6}$$

Again the smaller the output ECE value the more calibrated the model over all the possible output labels.

3.3 Learning with Different Labels

In this section, we mathematically discuss the learning objective with respect to different labels and present the intuition behind our method. For each input example x with ground-truth class y, the output of a neural network h is a probability distribution over all classes normalized by the softmax function

$$h(k|x) = \frac{\exp(z_k)}{\sum_{i=1}^{K} \exp(z_i)} \tag{3.7}$$

where $k \in \{1, 2, ..., K\}$ is a class, and z_k is the activation of output in the last layer of the model. Let $\pi(k|x)$ be the label corresponding to input x. The model is trained by minimizing the cross-entropy loss defined as

$$l(x) = -\sum_{i=1}^{K} \pi(k|x) \log(h(k|x)) \tag{3.8}$$

For one-hot label, $\pi(y|x) = 1$ for the ground-truth class y and $\pi(k|x) = 0$ for all other class $k \neq y$. The cross entropy loss with one-hot label can be simplified to

$$l(x) = -\log(h(y|x)) \tag{3.9}$$

Under this training loss, the model is not only trained to make a correct prediction but also with the highest confidence z_y possible to reduce the loss, which causes the model to become overconfident. Uniform smooth labels are defined by

$$\pi(k|x) = (1-\alpha)e_y + \alpha * u(k)/K \tag{3.10}$$

where e_y is the coordinate vector and u is a uniform distribution. The cross-entropy loss with uniform smooth label can be written as

$$l(x) = -(1-\alpha)\log(h(y|x)) + \alpha H(u,h) \tag{3.11}$$

When the model is trained using this loss function, it tries to match its output with a uniform distribution in addition to maximizing the output for the correct class. Such a scheme implies that the example is regarded to be equally probable to be in any other class. This is clearly not the case in general as we discussed in Section 3.1 as some classes could be inherently more similar than other classes. The intuition behind our method is that we assume there is an unknown optimal distribution for perfect calibration $q^*(k|x)$ that satisfies the two assumptions we discussed in Section 3.1 and that can be approximated by estimating the class similarities from the data. Let $\hat{q} \approx q^*$ be the approximated optimal distribution, then our proposed smooth label is defined as

$$\pi(k|x) = (1-\alpha)e_y + \alpha * \hat{q}(k|x) \tag{3.12}$$

With this label, the cross entropy loss can be written as

$$l(x) = -(1-\alpha)\log(h(y|x)) + \alpha H(\hat{q}, h) \tag{3.13}$$

and the model is trained to make a correct classification while matching it output distribution with the approximated optimal probability distribution. Hence, *minimizing $H(\hat{q}, h)$ is a direct optimization for both prediction and output calibration.*

3.4 Approach

In this section, we describe our approach to capture the inter-class similarities in order to compute the approximation $\hat{q}$ and generate our proposed smooth labels. Capturing semantic similarity has been a longstanding and still is a wide-open problem [116, 117]. Different metrics quantify the similarity from different perspectives, therefore we propose to use several simple distance metrics, including a novel one based on word2vec mapping, and evaluate how well the proposed smooth label performs under varying notions of the captured similarities.

3.4.1 Image Space

For directly computing the distance in the image space, we use L_p norms and choose $p = 1$ and $p = 2$. For two inputs x in class k and x' in class k', the pairwise distances are given by

L_p Distance

$$d(x, x') = \|x - x'\|_p \tag{3.14}$$

3.4.2 Representation Space

Studies have shown that features learned by neural networks are often surprisingly useful as a representational space for a much wider variety of tasks and match with human perception [117, 118]. Autoencoders are among the most widely used unsupervised deep learning models [119, 120], which provide a powerful framework for learning latent representations by encoding as much of as the information needed to reconstruct the original data points. The latent representations have been shown to be useful for various downstream tasks [121]. Therefore, we also propose to use an autoencoder to map the data to the representation space and compute distances between their encodings. We first train an autoencoder in an unsupervised fashion on the training set and then compute the pairwise Euclidean distances of the latent encodings.

Autoencoder Distance

$$d(x, x') = \|f(x) - f(x')\|_2 \tag{3.15}$$

where $f(x)$ is the latent encoding of input x.

The inter-class distance between a pair of classes is determined by averaging the distances between all pairs of inputs that belong to them. The distance between class k and k' is

$$d_k(k') = \frac{1}{|C(k)||C(k')|} \sum_{x \in C(k), x' \in C(k')} d(x, x') \tag{3.16}$$

where $C(k)$ is the set of data points in class k, and d_k is a vector that contains the distances between class k and all other classes.

3.4.3 Semantic Space

The previous distance metrics were defined using either the original features of the objects or those generated by a deep neural network. In general, samples in the same class are expected to share a common semantic meaning, which may not be captured in the feature space. It was observed for example that image visual similarity is not necessarily the same as semantic similarity [122, 123]. The problem of defining semantic similarity has been studied extensively in the NLP literature as well as in various disciplines for which knowledge can be captured through an ontology or some type of a hierarchy of classes. For most datasets, the words used to label each class capture significant semantics of the class [124]. We make use of the advances in NLP based on the labels to define the semantic similarity between classes as follows. We first note that vector representation of words has been shown to successfully capture semantic similarities such that words with similar meaning are mapped to a similar points in the vector space [125, 126]. Therefore, we propose to use a word2vec model to map the words defining the labels of the classes into Euclidean space, and then compute the distances between the vectors to define the class similarities.

Word Embedding Distance

$$d_k(k') = \|\mathcal{V}(w(k)) - \mathcal{V}(w(k'))\|_2 \tag{3.17}$$

where $\mathcal{V}$ is a word2vec model and $w(k)$ be the natural language word associated with class k. This notion can be generalized to the case in which each class is defined by a set of words or a sentence.

3.4.4 Class-Similarity Based Smooth Label

After the class distances are computed, they are converted into class similarities using the softmax function

$$s_k(k') = \frac{\exp(-\beta d_k(k'))}{\sum_{i=1}^{K} \exp(-\beta d_k(i))} \tag{3.18}$$

where $\beta \geq 0$ is a hyperparameter that controls how "uniform" the similarity distribution is. The lower the value of β, the more uniform the distribution is, and vice versa. When $\beta = 0$ the similarity distribution s_k reduces to uniform distribution. In order to ensure the consistency of the parameter β for different distance metrics, we normalize the distances d_k to zero mean and standard deviation of one before applying the softmax function. Note that this normalization is equivalent to scaling β and does not affect the relative relationship between classes.

Finally, the class-similarity based smooth label is defined as follows. Let e_k be the one-hot label vector, the smooth label for class k is

$$y_k = (1 - \alpha)e_k + \alpha * s_k \tag{3.19}$$

where α is the label smoothing factor. Note that in practice we set $s_k(k) = 0$ and

scale $\sum_{k' \neq k} s_k(k') = 1$ to make sure that α consistently represents the total mass in the label over the other (non ground-truth) classes. Otherwise the total mass will be different because of different values of $s_k(k)$.

3.5 Experiments

In this section, we perform numerous experiments to demonstrate the effectiveness of the proposed class-similarity based label smoothing and show the advantage of our method over other recent works on confidence calibration of neural networks. We also explore the effect of some important hyper-parameters and evaluate our methods on out-of-distribution data.

3.5.1 Setup

We perform experiments on CIFAR-100 [127] dataset and Tiny-ImageNet [128] dataset. For all experiments, the network architectures and all parameters are identical for all methods so as to enable a fair comparison. For CIFAR-100, we use a wide residual network [129] with $\{16, 16, 32, 64\}$ filters respectively. For Tiny-ImageNet, we use a ResNet-34 [10] model. All models are trained with SGD with momentum for 100 epochs. The initial learning rate is 0.1 and is multiplied by 0.1 at 50 and 75 epochs. We use batch normalization and weight decay of 2×10^{-4}. We set the number of interval bins $M = 15$ following [63, 75]. For the autoencoder architecture [119], the encoder consists of blocks of two consecutive 3×3 convolutional layers followed by 2×2 average pooling. All convolutions (in the encoder

and decoder) are zero-padded so that the input and output height and width are equal. The number of channels is doubled before each average pooling layer. Two more 3×3 convolutions are then performed, the last one without activation and the final output is used as the latent representation. All convolutional layers except for the final use a leaky ReLU nonlinearity [130]. The dimensionality of the latent space is set to 256 for CIFAR-100 and 1024 for Tiny-ImageNet. We use a pretrained Wikipedia2Vec [131] as our word2vec model $\mathcal{V}$. Each label word is first converted to the corresponding Wikipedia entry and then mapped to a vector of length 100 in the Euclidean space.

3.5.2 Comparison with other Methods

We compare our proposed method to the vanilla training using one-hot labels, as well as three other techniques that improve confidence calibration:

- Temperature Scaling [63], which is a post-processing step and consists of multiplying the logits by a scalar before applying the softmax operator.
- Uniform Label Smoothing [74, 75], where the one-hot label is smoothed by distributing a small amount of mass over the other class uniformly.
- Mixup Training [76] which trains a classifier not only on the training data but also on the linear interpolation of two random sampled input points and their labels.

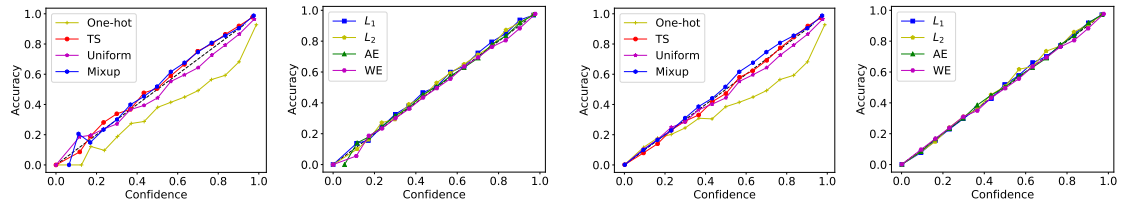
Dataset	**CIFAR-100**			**Tiny-ImageNet**		
Method	Accuracy	Pred ECE	Out ECE	Accuracy	Pred ECE	Out ECE
One-hot	68.47%	12.17%	25.71%	44.31%	24.95%	58.01%
Temp. Scaling	68.27%	2.34%	5.11%	43.53%	4.83%	9.94%
Uniform LS	69.32%	2.77%	5.41%	44.79%	5.24%	10.92%
Mixup	**69.98%**	2.29%	5.02%	**45.43%**	5.87%	11.76%
L_1 (Ours)	69.25%	1.68%	2.91%	44.77%	2.95%	5.78%
L_2 (Ours)	69.65%	1.56%	2.60%	44.86%	2.86%	5.59%
AE (Ours)	69.16%	0.77%	1.64%	44.81%	**2.49%**	**5.05%**
WE (Ours)	69.35%	**0.72%**	**1.55%**	44.55%	**1.23%**	**2.12%**

Table 3.1: Test accuracy, prediction ECE and output ECE results of different methods on CIFAR-100 and Tiny-ImageNet dataset.

3.5.3 Confidence Calibration Results

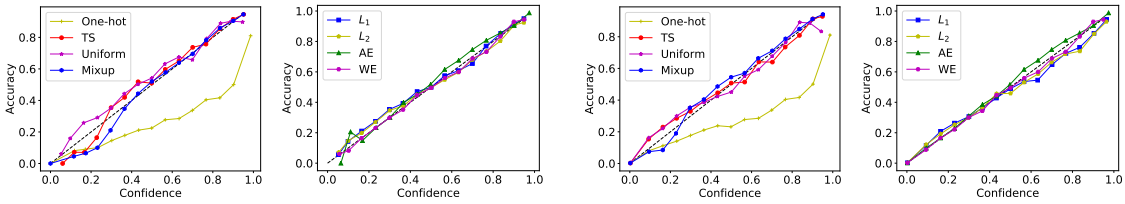
We show the results of test accuracy and confidence calibration ECE on CIFAR-100 and Tiny-ImageNet in Table 3.1. For CIFAR-100, $\alpha = 0.05$ for uniform label smoothing (LS) and word embedding (WE), and $\alpha = 0.1$ for L_1, L_2, and autoencoder (AE). For Tiny-ImageNet $\alpha = 0.1$ for all techniques. Best results for $\beta \in [0.5, 6]$ are reported. Parameter effects are discussed in detail in Section 3.5.4.

From the results, our proposed class-similarity based label smoothing, with all four distance metrics, significantly outperforms other methods on both datasets and on both prediction calibration and output calibration. Word embedding distance performs the best on both datasets, which is expected considering that the distance in the vector space more faithfully reflects class semantic relationships than pixel-wise distances in the image space and the encoding distance in the representation space. Note that while autoencoder and word embedding distance perform similarily on CIFAR-100, the latter has a significant advantage on Tiny-ImageNet. As the



(a) Prediction on CIFAR-100

(b) Output on CIFAR-100



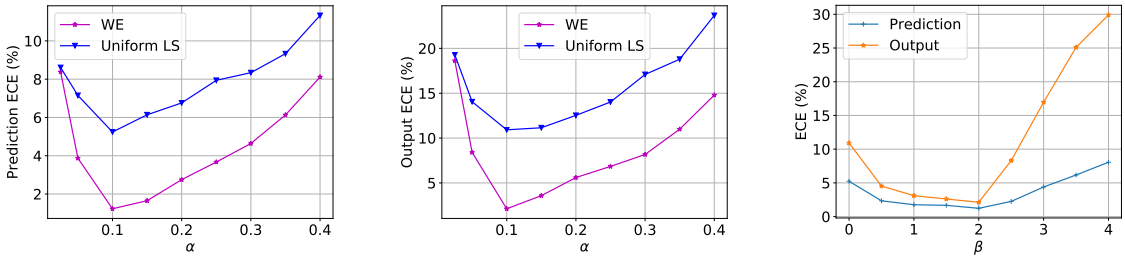
(c) Prediction on Tiny-ImageNet

(d) Output on Tiny-ImageNet

Figure 3.1: Reliability diagrams of different methods on CIFAR-100 and Tiny-ImageNet dataset. The plots for the eight techniques are divided into two groups for better readability.

complexity of a dataset increases, it becomes more difficult for an autoencoder to learn good representations. Mixup performs best in test accuracy because of the augmented training data. As a label smoothing technique, our method performs comparably with uniform label smoothing in terms of test accuracy and both are better than the vanilla training using one-hot labels.

The reliability diagrams in Figure 3.1 confirm the above findings. For better readability, we divide the diagrams of the eight techniques into two figures. In all the plots, the dashed black diagonal line represents perfect calibration for which the confidence matches perfectly the accuracy. The model trained using one-hot labels is clearly overconfident since the accuracy is always below the confidence. All other methods have a better performance, but the models trained with our smooth labels are the best as their diagrams almost identically match the diagonal line.



(a) (b) (c)

Figure 3.2: Effects of parameters. (a): Prediction ECE relative to α for WE and uniform LS. (b): Output ECE relative to α for WE and uniform LS. (c): Prediction and output ECE relative to β.

3.5.4 Effects of Parameters

In this section, we perform a series of experiments on Tiny-ImageNet to explore the effects of two important hyper-parameters α and β.

First, we test different α values to determine the strength of label smoothing, i.e. the total mass distributed to other classes in the label. We compare WE to uniform LS using different α values. β is set to 2. The results are presented in Figure 3.2 (a) and 3.2 (b). By inspecting the plots we observe that our proposed smooth label generally outperforms the uniform smooth label for all α values, and the best results for both methods are achieved at $\alpha = 0.1$ which is the commonly value used in practice. When $\alpha = 0.025$ is very small, the labels are only weakly smoothed and the model is still over-confident. When $\alpha = 0.4$ becomes large, the labels are too noisy and the model is not well-calibrated because of the excessive smoothing.

Next, we perform experiments on different β, a hyperparameter that determines how uniform the similarity distribution is. We choose the word embedding

distance model and set $\alpha = 0.1$. The results of prediction and output ECE with different β are shown in Figure 3.2 (c). We can see initially as β is increasing both the prediction and output ECE decrease. When β is too small, the similarity distribution is close to uniform and the class relation is not well represented, therefore the model is not well calibrated. As β keeps increasing when it becomes too large, the softmax function will produce extreme similarity values that are concentrated in only a few classes that do not represent the optimal distribution either.

3.5.5 Evaluation on Out-of-Distribution Data

Studies have shown that DNNs are not only overconfident on the data they are trained on but also on unseen out-of distribution data [76, 132]. In this section, we explore this aspect by evaluating the methods on two types of out-of-distribution data: test data from another unseen dataset and random noise. We test the models that are trained on CIFAR-100 in Section 3.5.3. We use the validation set of Tiny-ImageNet as the unseen dataset and generate uniformly distributed random samples as the second type of out-of-distribution data. We show the distributions of the prediction confidences on both types of data in Figure 3.3.

From the left plot, we can see that our method and mixup perform the best at refraining to produce high confidence on unseen out-of-distribution data. On the random noise samples, all three methods significantly outperform the one-hot training, although the mixup training offers more improvement. Note that our method is based on class similarities in order to train a more calibrated model,

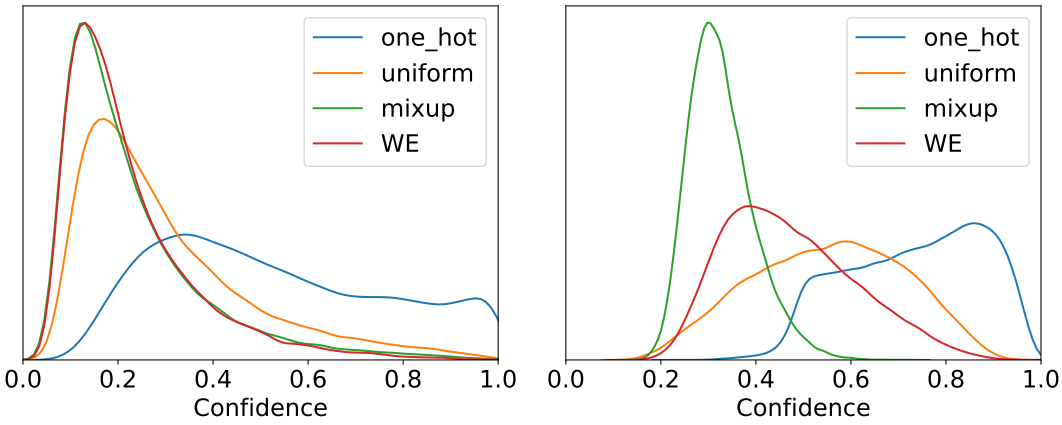


Figure 3.3: Distribution of the prediction confidences on the validation set of Tiny-ImageNet (left) and uniform random noise (right). Models are trained on CIFAR-100.

which is not well-suited for random noise samples where the notion of similarity does not exist. Although this application is not the main focus of our proposed method, we note that it still significantly outperforms the uniform label smoothing in both scenarios.

3.6 Conclusion

In this chapter, we address the confidence calibration problem in a more holistic framework. We first identify a major missing component in confidence calibration, that is, the lack of consideration of the entire predictions of a neural network. We introduce a new generalized definition of calibration that captures how well all scores in the output of a neural network represent the actual likelihood of the input belonging to the corresponding classes. Motivated by directly optimizing the objective of confidence calibration, we propose class-similarity based label smoothing. The similarity between classes is computed, which is then used to approximate

the optimal distribution that achieves the perfect calibration. We adopt several similarity metrics, including one that is based on the semantics of the classes, and demonstrate that our method significantly outperforms other techniques. We also show that the proposed label smoothing outperforms the uniform label smoothing in terms of predictive uncertainty on out-of-distribution and random data.

Chapter 4: Improving Calibration of Robust Models via Prototypical Learning

4.1 Introduction

Deep Neural Networks have progressed rapidly in recent years and this has led to their widespread deployment in a wide variety of real-world systems [13, 94, 95, 110–112]. However, many current multi-class classification networks, in particular, are not always reliable. They were shown to be susceptible to adversarial attacks: imperceptible but carefully chosen perturbation added to the input can cause misclassification [17–19]. Recent works have also shown that modern deep neural networks are poorly calibrated: they are overconfident in the sense that the probability values that they associate with the predicted class overestimate the likelihoods of those class predictions being actually correct [63, 64].

Both of these are major problems, especially in real-world safety-critical applications. Adversarial vulnerability leaves an opportunity for a malicious attacker to cheat the system, and the decision-maker or downstream components cannot trust the predictions if a model is poorly calibrated. These issues have gained significant attention and there are numerous works trying to improve the adversarial robust-

ness [19, 24, 25, 50, 92, 99] or the confidence calibration of a network [62, 63, 73, 75, 76, 133]. However, a network remains unreliable with only one issue mitigated.

In this chapter, we try to bridge this gap by proposing a technique that improves both the adversarial robustness and confidence calibration performance of a model. In our approach, we first learn a representation space using prototypical learning which bases its classification on the distances between the representation of a test sample and the prototype representations of each class. Then we use the distances as the confidence target and train a confidence prediction module jointly. The intuition is that the distance between a sample representation and a class prototype representation contains important information that a model uses for classification, we can utilize this information to guide our confidence probability estimation. The further away a sample is from a prototype, the less likely it belongs to that class therefore a smaller estimated probability. For multiple types of inputs, e.g. clean samples, adversarial examples with different types of constraints, we use a dedicated prototype for each type. Every type of input lies in a local cluster in the representation space, and using separate class centers provides the most accurate confidence estimations. We demonstrate the effectiveness of our approach through experiments on various datasets and network architectures.

4.2 Related Work

Improving adversarial robustness has been explored extensively [19, 24, 25, 48, 50, 92, 99, 134]. Adversarial training [25] is the most widely adopted approach. It uses

iterative gradient descent with random restarts to optimize adversarial examples and injects them into training samples to strengthen a network. TRADES [48] optimizes a theoretical upper bound to trade adversarial robustness off against accuracy. Free adversarial training [134] significantly accelerates adversarial training yet achieves comparable robustness performance.

There is also a great deal of recent works trying to improve confidence calibration [62, 63, 68, 69, 73, 75, 76, 133]. Post-hoc calibration methods use parametric or non-parametric models to transform the network's predictions based on a held-out validation set to improve calibration. More recent advances include Dirichlet calibration [70], ensemble temperature scaling [66], mutual information based binning [71], spline recalibration [72], focal loss [73], and accuracy versus uncertainty calibration [135]. Non-post-hoc methods are mostly based on adapting the training procedure, including modifying the training loss [73], label smoothing [74, 75], and data augmentation [76, 77].

Confidence-Calibrated Adversarial Training [136] combines adversarial training with label smoothing. They adjust the label smoothing strength according to the perturbation constraint such that stronger adversarial examples are assigned lower confidences during training in order to improve the model's generalization to unseen attacks. Adversarial Robustness based Adaptive Label Smoothing [137] set the label smoothing factor based on the model's robustness on a set of adversarial examples in order to improve the standard confidence calibration performance. These two works are mostly related to this work in the sense of making a connection between adversarial robustness and confidence calibration, however, their main

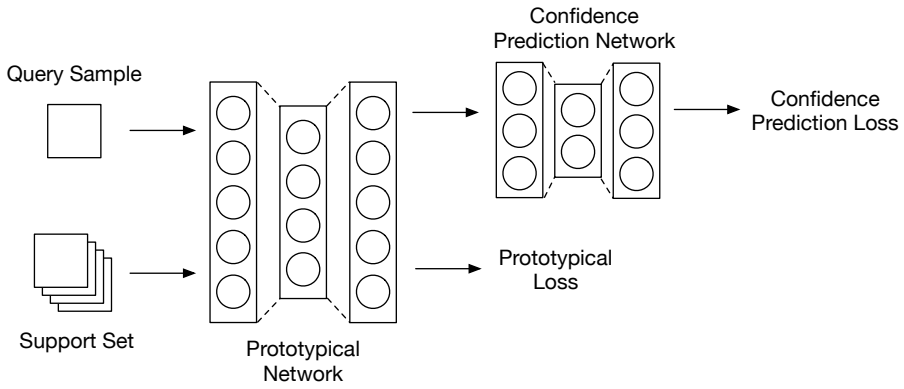


Figure 4.1: Overview of the proposed model. A prototypical network and a confidence prediction network are jointly trained.

goals and approaches differ significantly from ours.

4.3 Approach

In this section we describe our approach based on prototypical learning. Figure 4.1 provides an overview of our method.

4.3.1 Notation

We denote the set of training samples $S = \{(x_1, y_1), ..., (x_n, y_n)\}$ where each x_i is the input example and $y_i \in \{1, ..., K\}$ is the corresponding label. We denote $S^p = \{(x_1^p, y_1), ..., (x_n^p, y_n)\}$ the set of adversarial examples generated using the standard Projected Gradient Descent (PGD) approach under l_p constraints from samples in S. f_θ is the prototypical network, and g_ϕ is the confidence prediction network.

4.3.2 Prototypical Learning

We follow the standard episodic training approach in our model. At every training update, we randomly sample a support set $S_m = \{(x_1, y_1), ..., (x_{mK}, y_{mK})\}$ that contains m examples for each of the K classes and a query set $Q = \{(x_1, y_1), ..., (x_q, y_q)\}$ containing different examples from the K classes.

Prototypes, or class centers, of each class is the mean vector of the representations of the samples in the support set S_m belonging to its class:

$$c_k = \frac{1}{m} \sum_{y_i=k, (x_i,y_i) \in S_m} f_\theta(x_i) \tag{4.1}$$

where $f_\theta(x_i)$ is the encoded inputs x_i in the representation space.

Given a distance function d, prototypical networks produce a distribution over class for a query point x based on a softmax over distances to the prototypes in the representation space:

$$p_\theta(y = k|x) = \frac{\exp(-d(f_\theta(x), c_k))}{\sum_l \exp(-d(f_\theta(x), c_l))} \tag{4.2}$$

The query samples are classified based on their distance to each class center in the representation space. We use l_2 distance for metric d.

Learning proceeds by minimizing the negative log-probability $J(\theta) = -\log p_\theta(y = k|x)$ of the true class k via SGD. Training episodes are formed by randomly selecting a subset of examples within each class to act as the support set and a subset of the

remainder to serve as query points.

This training scheme minimizes intra-class distances while maximizing inter-class distances. As a result, samples within the same class are encouraged to be mapped close in the representation space while far from samples of other classes.

At inference, the class centers are calculated by by averaging the representations of all corresponding samples in the training set for all classes:

$$\hat{c}_k = \frac{1}{|k|} \sum_{y_i=k} f_\theta(x_i) \tag{4.3}$$

where $|k|$ denotes the number of training samples in class k. Then, for any given test sample x_j, the prediction of the label of x_j is the class that x_j is closest to in the representation space:

$$\hat{y}_j = \arg\min_k d\left(f_\theta(x_i), \hat{c}_k\right) \tag{4.4}$$

We can see that the closer an input is to its ground-truth class center, the more likely it is correctly classified, therefore should have a higher predicted confidence; while the further away it is from its ground-truth class center the less likely and a lower predicted confidence. This indicates that the this distance is strongly correlated with the desired confidence output, which is our intuition for using the distance to calibrate the model's prediction output. In the next section we will introduce the details of our confidence prediction method.

4.3.3 Confidence Prediction

We rely on the distance between a sample representation and the corresponding class center to calibrate the confidence prediction. First, for a sample x for which the ground-truth class is $y = k$, we use the softmax normalized distances between $f_\theta(x)$ and c_k as our target output confidence:

$$s_k = \frac{\exp(-d(f_\theta(x), c_k))}{\sum_l \exp(-d(f_\theta(x), c_l))} \tag{4.5}$$

However, since the ground-truth class is unknown at inference, if we simply use the predicted class center for confidence prediction, a mis-classified sample would introduce a large amount of calibration error. For example, if this sample is incorrectly classified as class k', then we have $s_{k'} > s_k$, which results in high confidence prediction for mis-classified samples if we output $s_{k'}$ as the predicted confidence.

Therefore, we jointly train a confidence prediction network g_ϕ. The input to g_ϕ is the learned representations from the prototypical network $f_\theta(x)$, and output is the predicted probability distribution for all classes. The loss function of the confidence prediction network is

$$J(\phi) = -\sum_{k=1}^{K} t_k \log(g_\phi(f_\theta(x))) \tag{4.6}$$

which is the cross-entropy loss with the target output t. We set the target probability for the ground-truth class to be the desired confidence s_k, and randomly sampled

values for non-ground truth classes:

$$\begin{aligned} t_k &= s_k \\ t_l &= \frac{1 - s_k}{K - 1} \mathcal{N}(0, 1),\ l \neq k \end{aligned} \tag{4.7}$$

and t is normalized to sum to 1. Under this scheme, the confidence prediction network is trained to follow the guide of the learned distances and produce calibrated probabilities associated with its predictions.

Overall, our model loss is

$$L(\theta, \phi) = J(\theta) + \alpha J(\phi) \tag{4.8}$$

which is optimized by SGD jointly.

For the choice of the weight parameter α, instead of using a fixed value, we propose to use a confidence-based weight for each sample. As we have discussed, the reason for the confidence prediction module is to prevent poor calibration on misclassified samples. Therefore, we would like the module to focus more on samples that are more likely to be incorrect, i.e., the samples that are closer to the model's decision boundary. For input x, we set the weight parameter is

$$\alpha = \frac{\beta}{s_k} \tag{4.9}$$

where s_k is the target output probability for sample x and β is a scaling factor, which determines the relative weight of the confidence prediction loss. With this

definition of α, if an input is close to the decision boundary, s_k is small and it will contribute more to the training of the confidence prediction module.

4.3.4 Adversarial Training

Finally we introduce our adversarial training method. We follow the standard adversarial training scheme, where we first generate adversarial examples that maximize the model loss and then inject them in the model training procedure.

Specifically, we use PGD to find adversarial examples under l_p constraint corresponding to each sample in the query set by maximizing the model's total loss L

$$x' = \arg \max_{\|x'-x\|_p \leq \epsilon} L(\theta, \phi) \tag{4.10}$$

and replace the original query set with the generated adversarial samples to train the model.

4.4 Experiments

In this section, we perform numerous experiments to demonstrate the effectiveness of the proposed approach and show that our model is able to achieve both adversarial robustness and confidence calibration.

4.4.1 Setup

We perform experiments on CIFAR-100 [127] dataset and Tiny-ImageNet [128] dataset using various architectures: DenseNet (DN), ResNet (RN), and WideRes-

Net(WRN). For all experiments, the network architectures and all parameters are identical for all methods. For CIFAR-100, we use DenseNet-161, ResNet-18, and WRN with $\{16, 16, 32, 64\}$ filters respectively. For Tiny-ImageNet, we use DenseNet-161, ResNet-34, and WRN-50-2.

4.4.2 Evaluation Metrics

We evaluate our models in standard accuracy, adversarial robustness, and calibration error.

We follow the standard definition of accuracy. For robustness, at test time, we generate white-box adversarial examples using PGD under l_p constraint as shown in Equation 4.10, which is the same as during training.

We use the standard histogram-based ECE metric to measure calibration error. The model's output probabilities p_i are grouped into M interval bins, where p_i is the model's output confidence for sample i with class prediction k. Let B_m be the set of indices of which the confidence falls into bin m, i.e. $p_i \in (\frac{m-1}{M}, \frac{m}{M}]$. The likelihood and confidence of B_m are defined as

$$\begin{aligned} \text{acc}(B_m) &= \frac{1}{|B_m|} \sum_{i \in B_m} \mathbb{1}(y_i = k) \\ \text{conf}(B_m) &= \frac{1}{|B_m|} \sum_{i \in B_m} p_i \end{aligned} \tag{4.11}$$

where y_i is true class label for sample i. The histogram-based ECE is defined as

$$\text{Histogram ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} |\text{acc}(B_m) - \text{conf}(B_m)| \tag{4.12}$$

A smaller ECE indicates a better calibrated model over all the possible output labels. We use the standard number of interval bins $M = 15$ following [63, 75].

4.4.3 Comparing Methods

We compare our proposed method to:

- Projected Gradient Descent (PGD) [25]. PGD is the standard adversarial training approach by generating adversarial samples that maximize the model's cross-entropy loss and using them to train the model.

- TRADES [48]. Instead of adversarial training, TRADES adds a regularization term encourages the output to be smooth, that is, it pushes the decision boundary of classifier away from the sample instances via minimizing the difference between the prediction of natural example and that of adversarial example.

- Confidence-Calibrated Adversarial Training (CCAT) [136]. CCAT is an approach trying to make the model generalizing to unseen threats by biasing the model towards low confidence predictions on adversarial examples. It combines adversarial training with label smoothing where the smoothing factor is determined by the adversarial perturbation, such that stronger adversarial examples are encouraged to be predicted with lower confidence.

4.4.4 Confidence Calibrated Robust Models

In this experiment, we consider l_∞ adversarial examples train and evaluate the models. We use 7 PGD steps for CIFAR-100 and 5 steps for Tiny-ImageNet. We

show the results on the two datasets in Table 4.1. We report the standard accuracy on clean samples and robustness on adversarial inputs, as well as the corresponding standard histogram-based expected calibration errors (ECE).

Model	Method	CIFAR-100				Tiny-ImageNet			
		Clean		Adversarial		Clean		Adversarial	
		Acc	ECE	Rob	ECE	Acc	ECE	Rob	ECE
DN	PGD	62.87	15.36	27.20	5.42	44.39	12.75	21.37	4.90
	TRADES	60.14	14.81	28.59	6.01	42.82	12.33	23.58	5.01
	CCAT	59.59	13.09	26.37	4.82	42.31	11.41	20.85	4.52
	Ours	62.24	**2.28**	24.98	**1.17**	44.68	**2.43**	19.97	**1.59**
RN	PGD	58.84	13.75	24.09	5.07	47.02	14.13	20.26	4.32
	TRADES	56.53	13.29	25.99	5.51	44.97	13.87	22.10	4.27
	CCAT	57.05	12.74	23.87	4.16	43.58	11.05	19.46	3.34
	Ours	59.22	**1.85**	22.48	**0.89**	47.53	**2.57**	19.82	**1.20**
WRN	PGD	61.92	14.88	26.60	5.36	46.83	13.66	22.75	4.88
	TRADES	58.27	14.55	28.01	5.58	43.27	13.58	24.74	4.95
	CCAT	58.62	12.75	25.80	4.39	42.60	10.57	21.39	3.76
	Ours	62.07	**2.13**	24.46	**1.04**	45.90	**2.21**	21.02	**1.67**

Table 4.1: Accuracy (Acc), adversarial robustness (Rob), and expected calibration error (ECE) on CIFAR-100 and Tiny-ImageNet.

We can see from the results on both datasets, our proposed method significantly outperforms all comparing methods in terms of both accuracy and confidence prediction on clean input samples. For adversarial inputs, our method generates well-calibrated probabilities although the robustness is slightly lower but still comparable with PGD and TRADES. CCAT uses soft labels during training therefore the calibration is better than the other two methods but is significantly worse than our method.

Model	Method	Clean		l_∞		l_2		l_1	
		Acc	ECE	Rob	ECE	Rob	ECE	Rob	ECE
DN	PGD	71.32	16.75	35.10	11.93	31.68	11.59	36.54	12.37
	TRADES	69.47	15.45	37.80	12.62	32.87	12.05	37.23	11.40
	CCAT	67.92	12.25	33.98	10.15	29.76	9.18	34.85	10.02
	Ours (Single)	70.99	7.46	32.09	7.90	27.59	6.16	31.32	7.40
	Ours (Separate)	71.77	**4.45**	33.78	**3.37**	30.59	**3.52**	32.04	**3.16**
RN	PGD	67.15	14.14	31.98	10.43	27.52	10.17	32.67	11.08
	TRADES	65.78	13.11	33.83	11.07	29.31	10.94	33.61	11.37
	CCAT	64.76	11.12	29.56	8.73	25.61	8.25	30.07	9.22
	Ours (Single)	68.17	7.67	27.92	5.48	24.62	6.08	28.80	6.77
	Ours (Separate)	68.05	**4.22**	29.33	**2.41**	26.15	**3.28**	30.42	**3.04**
WRN	PGD	69.85	14.20	33.05	10.13	29.97	10.48	34.44	11.72
	TRADES	66.51	14.07	34.83	11.49	31.46	10.85	36.42	12.33
	CCAT	65.92	11.20	31.95	8.43	28.89	8.67	33.18	9.83
	Ours (Single)	69.43	7.32	28.07	5.19	25.33	7.30	28.89	7.51
	Ours (Separate)	69.92	**4.77**	31.05	**3.10**	28.19	**3.08**	32.87	**3.98**

Table 4.2: Accuracy (Acc), adversarial robustness (Rob), and expected calibration error (ECE) on CIFAR-10 for different types of adversarial examples.

4.4.5 Calibration on Multiple Types of Adversarial Examples

In this section, we generate 3 types of adversarial examples under l_∞, l_2 and l_1 constraint respectively. We use all three types of adversarial examples to train the models so that they are robust against all three types of adversarial attacks. Since the robustness on a single type of adversarial is already relatively low on CIFAR-100 and Tiny-ImageNet, we choose CIFAR-10 in this experiment. For this experiment, we propose another variant for our method. Instead of having a single prototype for each class, we use separate class centers for each types of adversarial examples and one for the clean input, that is, 4 prototypes per class. After the model is fully trained, we run the model for an extra epoch to generate the three types of

adversarial examples for all training samples, and then compute the average as the aforementioned 4 prototypes. We will compare our method with the single and separate class centers and show the advantage. The results are shown in Table 4.2.

We can see that both variants of our method offer significant improvements on calibration across all types of inputs. By comparing the results of the two variants, we can see that using separate centers has a noticeable improvement over using single centers in terms of both robustness and calibration. The model with separate centers has a more accurate distance information which results in the improvement. We note that similar with the results in Section 4.4.4, the excellent calibration performance of our models comes with a small cost of robustness, but they are still very comparable with other methods. This result also demonstrates the possibility for our model to generalize to even more types of inputs and achieving accurate calibration.

4.4.6 Effectiveness of Confidence Prediction Module

In this section we show that the confidence prediction module is essential in our model. We first show the results of simply using the softmax normalized distance between a sample representation and the nearest class center as the predicted confidence in Table 4.3. We choose WRN model and compare our models with baseline PGD.

We can see that without the confidence prediction module, the gain in calibration performance is mininal comparing to the baseline model.

Model	Method	CIFAR-100				Tiny-ImageNet			
		Clean		Adversarial		Clean		Adversarial	
		Acc	ECE	Rob	ECE	Acc	ECE	Rob	ECE
WRN	PGD	61.92	14.88	26.60	5.36	46.83	13.66	22.75	4.88
	Ours	62.07	**2.13**	24.46	**1.04**	45.90	**2.21**	21.02	**1.67**
	Ours (w/o)	61.38	13.75	24.72	4.55	45.25	11.46	21.16	4.15

Table 4.3: Accuracy (Acc), adversarial robustness (Rob), and expected calibration error (ECE) on CIFAR-100 and Tiny-ImageNet for PGD, our model, and our model without the confidence prediction module.

Next, we perform an experiment which shows that our method is able to lower the prediction confidence on mis-classified samples. We use the WRN model in Section 4.4.4 and compute the average prediction confidence on correctly classified samples and mis-classified samples. We show the results in Table 4.4.

Method	Correct Samples	Incorrect Samples
PGD	0.62	0.46
TRADES	0.61	0.46
CCAT	0.58	0.41
Ours	0.58	0.22

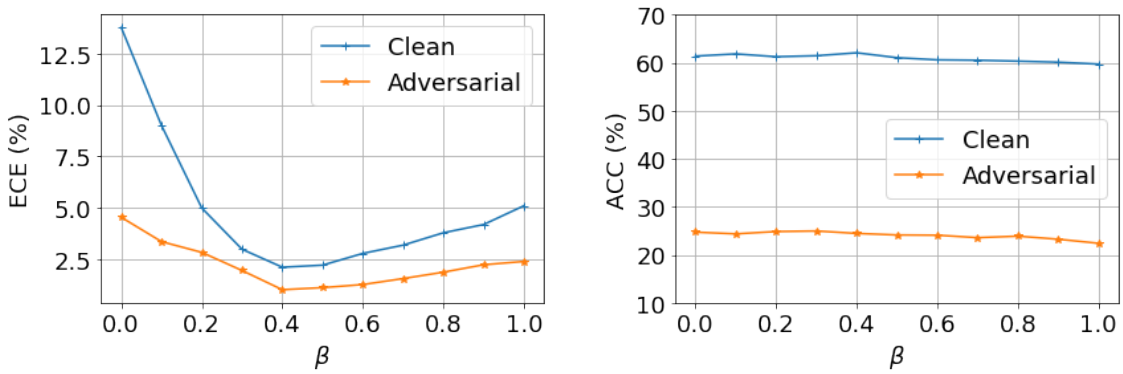
Table 4.4: Average prediction confidence on correctly and incorrectly classified samples.

Table 4.4 shows why our method is better calibrated than other methods. It is able to generate high confidence on correctly classified samples, while the predicted confidence on mis-classified samples are significantly lower. In the case of a mis-classification, all existing methods will associate it with the highest output probability. While in our model, according to our design of the confidence prediction module, it will generate a low confidence since the distance between the sample representation and class center will be longer.

4.4.7 Effect of Parameter β

In this section we explore the effect of the parameter β, which controls the relative weight of the confidence prediction module loss. We train the WRN model on CIFAR-100 with $\beta \in [0, 1]$. For $\beta = 0$, this corresponds to a model without a confidence prediction module and we simply use the distance from a sample representation to the closest class center as the predicted confidence. We show the results in Figure 4.2.

From Figure **??** we can see that initially after adding the confidence prediction module and increasing β, ECE decreases rapidly. This again demonstrates that the effectiveness of the confidence prediction module. While *beta* keeps increasing the calibration performance slowly becomes worse, but still significantly better than small β values. From Figure **??** we can see that both the standard accuracy and adversarial robustness are not affected much by this parameter. This is shows that although our model is end-to-end trained, the confidence prediction loss mainly affects the module itself, while the class prediction is generated by the prototypical network and the resulting representation.



(a) (b)

Figure 4.2: Effects of parameter β. (a): Expected calibration error (ECE) relative to β (b): Accuracy relative to β.

4.5 Conclusion

In this paper we propose to a method to significantly improve the calibration performance of a adversarially robust model. We first learn a representation space using prototypical learning for which the distance from a sample representation to a class prototype are used in classification. We then use the distance information to train a confidence prediction network to encourage the model making a calibrated prediction. We demonstrate through extensive experiments that our method is able to improve calibration performance of a model while maintaining a comparable accuracy and robustness level.

Chapter 5: LEICA: Laplacian Eigenmaps for Group ICA Decomposition of fMRI Data

5.1 Introduction

Functional magnetic resonance imaging (fMRI) studies have shed light on the overall functional organization of the brain. For example, resting state studies have revealed the existence of a number of intrinsic functional networks, including the "default mode" and "salience" networks [138–140]. These studies have also led to the discovery of specific patterns related to brain disorders [141–143], age [144], and gender [145].

The determination of large-scale patterns of brain activity is a challenging problem due to several factors, including the high-dimensionality of the data, various noise sources, and inter-subject variability. The most widely used approach to capture group-level functional connectivity is based on Independent Component Analysis (ICA) [9], which is a data-driven approach that assumes the existence of statistically independent latent features, called sources, which can generate the data through linear mixtures. However, ICA does not naturally provide a suitable method for drawing inferences about groups of subjects [146], which has led to the devel-

opment of several multi-subject approaches that are partly based on ICA. These approaches revolve around a few strategies. One method is to concatenate the data from multiple subjects along the temporal dimension [89] followed by the use of Principal Component Analysis (PCA) [8] to reduce the dimensionality of the data, and ending with the application of ICA to extract spatial features common at the group level. The tensorial extension to ICA [90] uses a three-dimensional tensor to estimate shared spatial patterns and time courses between subjects, and subject-specific loadings of the components to capture the multidimensional structure of the data. In [91], a general framework is proposed for using the expectation maximization algorithm to estimate an unconstrained mixing matrix for the group-level independent components. Finally, canonical ICA [87] uses Canonical Correlation Analysis (CCA) to find components that maximize cross-correlation before applying the ICA step.

Two methods involving temporal concatenation of the data include GIFT [89] (`http://icatb.sourceforge.net/`) and MELODIC [147] (`http://fsl.fmrib.ox.ac.uk/fsl/fslwiki/MELODIC/`). GIFT approximates the PCA of the group temporally concatenated data by first reducing each subject's dataset to dimension m, and then running the PCA on the concatenation of the principal components across all subjects to further reduce temporal dimensionality. MELODIC, however, uses an incremental approach called MELODIC's Incremental Group PCA (MIGP) [88] to implement the approximation. Although MELODIC and GIFT have been broadly used for group level fMRI analysis, they are not without limitations as will be explained next.

These ICA-based strategies involve the use of PCA to reduce the dimensionality of the data. PCA is basically a linear projection of the original data onto a low-dimensional subspace in such a way as to retain most of the variance of the data. However, PCA does not distinguish between variance due to noise versus variance due to inherent underlying signal variations. Moreover, there is no principled justification for using the variance as the basis for data reduction based solely on a linear transformation. The BOLD signal is a complex function of neural activity, oxygen metabolism, cerebral blood volume, cerebral blood flow, and other physiological parameters [148]. The dynamics underlying neural activity and hemodynamic physiology involve multiple nonlinearities [149–151], a fact that has motivated the exploration of nonlinear approaches for fMRI analysis [152, 153]. To the best of our knowledge, nonlinear models have not been proposed to extract common spatial patterns for multi-subject fMRI analysis.

Several nonlinear dimensionality reduction approaches have been successfully used to address a wide range of applications [154]. Kernel PCA [155] maps data to a feature space using a nonlinear transformation (kernel) and performs linear PCA in the feature space. Isomap [156] is based on computing a low dimensional representation that tries to preserve the pairwise geodesic distances, which are approximated by the pairwise shortest paths between corresponding points. Maximum variance unfolding (MVU) [157] computes a low-dimensional manifold such that local distances and angles are preserved. Locally linear embedding (LLE) [158] is based on the geometric intuition of local linearity, which assumes that each point and its neighbors lie on an approximately linear patch of a low-dimensional manifold. The

mapping is computed by trying to preserve linearity locally, such that each point has the same neighborhood structure as in the high-dimensional space. A technique of a different flavor is the t-distributed stochastic neighbor embedding (t-SNE) [159], which captures similarities between pairs of high-dimensional points through a probability distribution, followed by determining a mapping to a lower-dimensional space that achieves a similar distribution. Laplacian eigenmaps [160] project data into a low dimensional Euclidean space such that local proximity relations are preserved as much as possible, mapping close points in the original space to close points in the low-dimensional space. Of these techniques, Laplacian eigenmaps appear to be the most suitable for fMRI data since its overall objective function can be expressed as minimizing

$$\sum_{ij} \|\boldsymbol{y_i} - \boldsymbol{y_j}\|^2 W_{ij} \tag{5.1}$$

where $\boldsymbol{y_i}, \boldsymbol{y_j}$ are the low-dimensional vectors corresponding to points i, j in the original space, and W_{ij} is a proximity measure of points i and j in the original space. If we choose W_{ij} to be the zero-thresholded correlation between voxels i and j in the original space, this nonlinear projection maps voxels with strong correlations to nearby points in the low-dimensional Euclidean space. Correlations between voxels in fMRI data reveal functional relationships and hence contain the information that can be used to construct functional networks.

We note that Laplacian eigenmaps have been successfully used in analyzing fMRI data [161,162]. Our approach differs from earlier studies in that they are ROI-based methods focusing on specific networks using prior brain-related knowledge,

while our framework makes use of ICA to extract intrinsic networks for general fMRI datasets.

In this chapter, we propose a group-level model that uses Laplacian eigenmaps as the main data reduction step, which preserves the correlation information in the original data as best as possible. The nonlinear map is robust relative to noise in the data and to inter-subject variability. After the Laplacian eigenmaps transformation, ICA is applied to the reduced data to estimate spatial features, followed by thresholding to extract group-level independent spatial maps. We call our algorithm LEICA (Laplacian Eigenmaps for group ICA decomposition).

The rest of the chapter is organized as follows. In section 5.2, we present the details of our algorithm and inference method, and describe several tests used to validate our method and compare the results with state-of-the-art methods. In section 5.3, we show the corresponding test results on both resting state and working memory task fMRI datasets for 100 subjects from the Human Connectome Project. Finally, in section 5.4, we discuss our results.

5.2 Materials and methods

5.2.1 Group ICA model

Spatial ICA assumes that the observed fMRI data are generated by a linear mixture of spatially independent components. The underlying model can be expressed as:

$$X = MA \tag{5.2}$$

where X is the observation matrix whose columns represent the time series corresponding to the voxels, A consists of the corresponding independent sources, and M is the mixing matrix. We note that (5.2) does not include a noise model, but fMRI data is inevitably confounded by noise. Moreover, as stated, the model is not suitable for multi-subject fMRI analysis.

Noisy group level ICA models have been proposed in the literature. Let $\{X_n; n = 1, 2, ..., N\}$ be the fMRI data of N subjects with n being the subject index. X_n has dimensionality $T \times V$, where V is the number of voxels and T is the number of time frames. To handle noise, group ICA models have the following form:

$$X = UA + E \tag{5.3}$$

where $X = [X_1^T X_2^T ... X_N^T]^T$ is the $NT \times V$ data matrix formed by temporally concatenating data across N individuals, A consists of the group-level independent sources, U is the group mixing matrix, and E is the noise. Existing methods differ in the assumptions and procedures used to estimate the group mixing matrix U and the noise E.

In temporal concatenation ICA [89, 163], the group mixing matrix has form $U = PM$ and the group model becomes

$$X = PMA + E \tag{5.4}$$

where P is estimated by PCA on the concatenated group data matrix and MA

consists of the group-level principal components, which are assumed to be a linear subspace of the sources. Low variance components are considered as noise. The classic noise-free ICA is then used to estimate the mixing matrix M and the independent sources A.

In the tensor ICA approach [90], the group mixing matrix is defined to have the Khatri-Rao product structure $U = M| \otimes |R$ where M is the subject-specific loading of the group components and R is the time course associated with the independent components. The group model can now be expressed as

$$X = (M| \otimes |R)A + E \tag{5.5}$$

where the data are modeled as a trilinear product of group-level independent sources A, associated with common time courses R and subject-specific loadings contained in M, corrupted with additional noise E.

In the unified framework [91], no assumption about the structure of U is made but instead it is estimated using the expectation maximization algorithm that maximizes the expected likelihood that the observed data is generated by this model. The noise is modeled by a zero-mean multivariate Gaussian distribution.

Canonical ICA [87] tries to handle noise in a hierarchical fashion in the sense that individual observation noise and subject variability noise are estimated succes-

sively. The group mixing matrix has the form:

$$U = \begin{bmatrix} P_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & P_N \end{bmatrix} \begin{bmatrix} Q_1 \\ \vdots \\ Q_N \end{bmatrix} M \tag{5.6}$$

where the P_i's are the loading matrices for individual principal components estimated by PCA, and the Q_i's are the loading matrices for the common group principal components estimated by CCA, which determines components that maximize cross-correlation.

We propose in this work a new model, called LEICA, based on the assumption that the linear mixture of independent sources lies in a low-dimensional manifold that cannot be captured by linear models. We do not make explicit assumptions about the group mixing matrix either, but address the noise problem through the construction of a similarity graph. Our overall model can be expressed as:

$$\begin{aligned} Y &= MA \\ X &= \phi^{-1}(Y) + E \end{aligned} \tag{5.7}$$

where ϕ is a nonlinear embedding. We propose to use the Laplacian eigenmaps algorithm to learn this embedding as explained next.

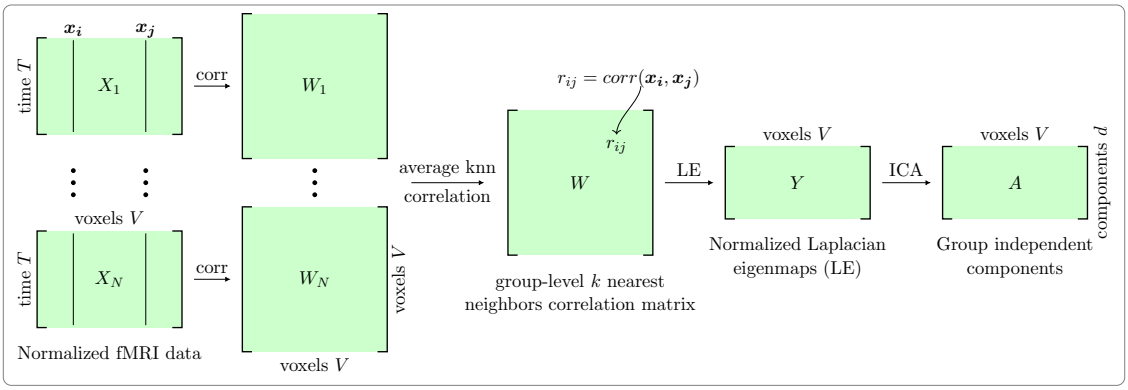


Figure 5.1: Processing steps used by LEICA. The correlation matrices corresponding to individual subjects (1 through N) are computed first, followed by averaging across subjects, and constructing the group-level k nearest neighbors (kNN) correlation matrix (the latter is always a sparse matrix). This matrix is used to compute normalized Laplacian eigenmaps followed by the application of the ICA algorithm.

5.2.2 Dimensionality reduction using Laplacian eigenmaps

Laplacian eigenmaps constitute a nonlinear dimensionality reduction technique that is based on determining a low dimensional representation of a high dimensional dataset, which preserves locality information as much as possible. It maps similar data points to nearby points in Euclidean space.

We start by normalizing the data of each subject. Then we let $X = [\boldsymbol{x_1}\, \boldsymbol{x_2} \ldots \boldsymbol{x_V}]$, where $\boldsymbol{x_v} \in \mathbb{R}^{NT}$ is the concatenated time series for voxel v across the subjects. The procedure of using Laplacian eigenmaps to derive a d-dimensional embedding of the voxels is as follows.

First we construct a group-level weighted similarity graph $G = (V, E)$ with the nodes V representing the voxels and the weights of the edges E are pairwise

similarities chosen to be the Pearson's correlation coefficient

$$r_{ij} = \frac{(\boldsymbol{x_i} - \bar{x}_i) \cdot (\boldsymbol{x_j} - \bar{x}_j)}{NT\sigma_i\sigma_j} \tag{5.8}$$

where $\bar{x}_i, \bar{x}_j$ are the sample means of $\boldsymbol{x_i}, \boldsymbol{x_j}$ and σ_i, σ_j are the sample standard deviations of $\boldsymbol{x_i}, \boldsymbol{x_j}$. To avoid explicitly considering the group-concatenated data, the group-level correlation matrix is computed by averaging the subject-level correlation matrices, as the data of each subject have already been normalized.

For each voxel $v \in [1, ..., V]$, we determine the k nearest neighbors set $\mathcal{NN}(v)$ of v consisting of the k voxels with the largest correlations with v. We then compute the following weight function

$$W_{ij} = \begin{cases} r_{ij} & \text{if } i \in \mathcal{NN}(j) \text{ or } j \in \mathcal{NN}(i) \\ 0 & \text{otherwise} \end{cases} \tag{5.9}$$

where r_{ij} is the Pearson's correlation coefficient of voxel i and j. By considering only positive correlations r_{ij}, negative correlations are not included in a voxel's k nearest graph, which also reflects the intuition that anti-correlated voxels are less likely to participate in the same network [164].

The choice of k needs to be carefully considered. If k is chosen to be too small, the graph will not contain sufficient local connectivity strengths for all voxels, and thus the low-dimensional embedding will not capture enough information. In contrast, if k is too large, many weak connections will be included in the graph and

the graph will be more subjective to noise and inter-subject variation. Picking a suitable k is a trade-off between connectivity information and noise, and the best value depends on the data. We will describe our approach later, which always leads to a sparse graph.

Once the graph G is constructed, we compute the normalized graph Laplacian

$$L = I - D^{-1/2} W D^{-1/2} \tag{5.10}$$

where D is diagonal degree matrix whose entries are defined by $D_{ii} = \sum_j W_{ij}$. The matrix L is also called the normalized Laplacian. Then we compute the smallest few eigenvalues and corresponding eigenvectors of L

$$L\boldsymbol{f} = \lambda \boldsymbol{f} \tag{5.11}$$

Let $\boldsymbol{f_0}, \boldsymbol{f_1}, ..., \boldsymbol{f_d}$ be the $d+1$ eigenvectors corresponding to the eigenvalues ordered as $0 = \lambda_0 \leq \lambda_1 \leq ... \leq \lambda_d$. The eigenvector $\boldsymbol{f_0}$ corresponding to eigenvalue 0 is a constant vector, so we use the other d eigenvectors to form embeddings of the original data. Vector $\boldsymbol{y_v} = [\boldsymbol{f_1}(v), \boldsymbol{f_2}(v), ..., \boldsymbol{f_d}(v)]^T \in \mathbb{R}^d$ is the d-dimensional embedding for $\boldsymbol{x_v}$.

5.2.3 Group-level spatial maps

After embedding the data into a lower dimensional manifold, we obtain a $d \times V$ matrix $Y = [\boldsymbol{h_1}\, \boldsymbol{h_2} \,...\, \boldsymbol{h_V}]$ in which each row corresponds to an eigenvector as

described in the previous section. We now model this matrix Y as a linear mixture of a number of spatial independent features as indicated in equation (5.7). The common independent spatial features A can be extracted by applying ICA.

The overall LEICA workflow is illustrated in Figure 5.1.

The resulting spatial components extracted by the ICA are continuous values at each voxel, and hence some form of thresholding is employed to determine the final spatial maps. Group t-test based and mixture-model based thresholding methods have been proposed in the literature and used in GIFT and MELODIC; these methods are applicable to our model as well. But it has been shown that a simple thresholding approach based on the absolute values of the voxels also works well [87] and is consistent with the FastICA algorithm that we use. Therefore our thresholding procedure consists of converting the spatial components to Z-scores (zero mean and unit variance) and then thresholding based on absolute Z values.

5.2.4 Evaluation strategies

In this section, we introduce a number of approaches to evaluate our model, which will be applied to resting state and working memory data sets of the Human Connectome project [165]. Given the extracted group-level spatial maps, we show that thresholded spatial maps capture important qualitative properties of resting-state networks, as well as functional organization compatible with working memory task execution. Furthermore, the results are comparable to those obtained with state-of-the-art methods. Next, we show that our model is able to extract more

functionally cohesive and stable spatial components than other methods, and exhibits better generalizability from smaller subgroups.

5.2.4.1 Similarity with other methods

The current state-of-the-art and most widely used tools to tackle the problem addressed here are GIFT and MELODIC. The two toolboxes implement the same concatenated ICA model with slight variations on how the group-level PCA estimation and thresholding are conducted. While their results are comparable, MELODIC seems to have better reproducibility due to the instability of GIFT's t-test based thresholding [87]. Hence, we focus our comparison with MELODIC (but also show results for GIFT). Our model employs different estimation and inferencing procedures than MELODIC. In addition, the final values of the spatial maps in our model are Z-scores based on the absolute strengths of the ICA components, while the values in MELODIC are Z-statistics based on mixture modeling [147]. Thus, we compare thresholded maps which are positive with different magnitude ranges in the two models. We use the cosine similarity measure to compare the corresponding thresholded maps, a commonly used similarity measure in high-dimensional positive spaces.

Specifically, after generating the spatial maps from both models, we apply the corresponding thresholding procedures (the threshold values are adjusted such that the spatial maps have approximately the same number of voxels on average). Let a_1, a_2 be the two thresholded spatial maps from the two models; the cosine similarity

is defined as:

$$c = \frac{a_1 \cdot a_2}{\|a_1\|_2 \|a_2\|_2} \tag{5.12}$$

5.2.4.2 Functional connectivity structure

Functional MRI data have been used to capture large-scale functional connectivity structure in the brain. Voxels in the same intrinsic functional network are expected to have similar activity patterns, which translates into correlated voxel time series. In fact, most seed-based analysis [166] and clustering approaches [167] are based on this assumption. We develop an approach to capture the functional connectivity profile of each spatial map as follows.

Let the rows of the matrix A be a set of thresholded spatial maps, and let W be the full correlation matrix of all pairs of voxels, averaged over all subjects. The i-th row (or column) of W is the correlation between voxel i and the rest of the voxels. We define the correlation maps corresponding to the given thresholded spatial maps by:

$$A_s = A \cdot W \tag{5.13}$$

The i-th row of A_s defines in a certain sense the correlation profile of the i-th spatial map in A. That is, each component of this profile is the weighted sum of the correlations of all the voxels for the spatial component, weighted by the magnitude of the corresponding voxel. Note that our approach differs from commonly used measures such as the average correlation of the voxels in a region of interest (ROI) [168,169]. Note also that the magnitude of a voxel in a spatial component indicates

the importance of this voxel for this component. This weighted sum provides a normalized correlation map for each spatial component over all voxels. It should be pointed out, however, that using average and weighted average correlations generates very similar results.

After generating A_s, we can threshold A_s to maintain only the strongest correlations, followed by comparing the thresholded correlation map $\bar{A}_s$ with A. If A represents inherent functional connectivity networks of the brain, we expect the voxels in each spatial component to be highly correlated with the voxels in the same component, and we expect the correlations with the rest of the voxels to be relatively weak. Therefore, the thresholded correlation map $\bar{A}_s$ should be very similar to A. If $\bar{A}_s$ is not similar with A, this would indicate that there are voxels outside the component that are very correlated with the voxels in the component. Finally, the similarity measure we use to compare $\bar{A}_s$ and A is the average cosine similarity over all the corresponding pairs of spatial maps between $\bar{A}_s$ and A.

5.2.4.3 Reproducibility and consistency

We introduce two tests to evaluate the reproducibility and consistency of our method. Similar tests have been proposed in [87].

Let A_1, A_2 be the matrices defining d extracted spatial maps using two different sets of subjects from the same dataset ("population"). Each row corresponds to a non-thresholded spatial map that has been converted to zero mean and unit norm. We define the cross correlation of the spatial maps to be the matrix $C = A_1 A_2^T$.

Each element C_{ij} is the Pearson correlation coefficient of the i-th row of A_1 and the j-th row of A_2; hence, $C_{ij} = 1$ if and only if the two components are identical. As in [87], we use the the parameter

$$p = \frac{1}{d} tr(CC^T) \tag{5.14}$$

to estimate the overlap of the two subspaces spanned by the rows of A_1 and A_2. The value of p quantifies how similar the two subspaces are, although the individual spatial maps may differ significantly even when the two subspaces are similar.

Another measure corresponds to the one-to-one matching of the components. Let A_1, A_2 be two matrices such that each row is a thresholded or non-thresholded spatial component with zero mean and unit norm, and $\tilde{A}_1$, $\tilde{A}_2$ be respectively row permutations of A_1 and A_2. We define the average one-to-one matching score to be

$$q = \max(\frac{1}{d} tr(\tilde{A}_1 \tilde{A}_2^T)) \tag{5.15}$$

corresponding to all possible row permutations of the two matrices. In this case, we are matching the rows of A_1 and the rows of A_2 such that the overall correlation score is maximized.

To evaluate our model, we evaluate the p and q measures for several scenarios. Let S be the complete dataset with N subjects, and let S_1, S_2 be two sets of subjects selected at random from the complete dataset. We run our algorithm and both MELODIC and GIFT on the two groups S_1 and S_2. We then compute the p scores

on the non-thresholded spatial components and the q scores on both thresholded and non-thresholded spatial components. The two groups are selected as follows

1. S_1, S_2 are two subgroups randomly selected from S each with $\{N/5, 3N/10, 2N/5, N/2\}$ subjects.

2. S_1 is a subgroup randomly selected from S with $\{N/5, 3N/10, 2N/5, N/2\}$ subjects, $S_2 = S$.

Both experiments are repeated with random selection of subgroups and the averaged scores are reported. Test results from 1) illustrate how reproducible the model is, with varying number of available subjects. Results from 2) illustrate the consistency between a small subgroup and the whole group.

5.2.5 fMRI datasets

We use two fMRI datasets to evaluate the methods described above: a resting state fMRI dataset and a working memory task fMRI dataset.

5.2.5.1 Resting state fMRI dataset

The resting state fMRI dataset we use is the 100 healthy unrelated subject dataset from the WU-Minn Human Connectome Project [165]. Each subject was involved in four 15-minute runs with TR=0.72 seconds totaling 1200 frames per run. All frames were sampled into 91,282 grayordinates (a combination of cortex vertices and subcortical voxels). The data was then temporally preprocessed and de-noised using the FIX approach [170, 171]. The resulting images were then aligned using

MSM registration [172]. Full details of this dataset can be found in publications from the project [165, 173]. Each voxel's time series was normalized for each run. Note that, out of the 91,282 grayordinates in the resting state fMRI data, we employed cortical data only (59,412 vertices of the left and right hemispheres).

5.2.5.2 Working memory task fMRI dataset

We employed the working memory task dataset of the 100 healthy unrelated subjects from the WU-Minn Human Connectome Project [165]. Participants were presented with blocks of trials that consisted of pictures of places, tools, faces and body parts. Within each run, the 4 different stimulus types were presented in separate blocks such that half the blocks used a 2-back memory task and half used a 0-back memory task. Individual subject data includes 2 runs of 405 frames each, preprocessed using FEAT [174]. Full details of this dataset can be found in [165,173]. Here, we analyzed only data from 2-back memory blocks as follows. To account for head motion, motion parameters were regressed out of the time series with AFNI's [175] 3dDeconvolve, which was also used to removed slow baseline drifts. Subsequently, time series data were z scored. To account for the cue stimulus indicating block type at the beginning of each block, data corresponding to the first 12 seconds of each block were removed. The remaining data segment of the block, plus 2 seconds due to hemodynamic lag, were considered to reflect working memory related data. In this way, 25 frames for each block were used, totaling 200 frames once the data were concatenated. Each voxel's time series was normalized for each

run.

5.2.6 Model parameters choice

Choosing an appropriate model order d (the number of non-constant eigenvectors; see equation 5.11) depends on the data. Since model order selection is not the main focus of this paper, we use an empirical value. In [173] a model order of 30 was used to conduct group analysis on both cortical and subcortical data; 23 of the extracted spatial maps were suggested to be meaningful based on visual inspection of the maps. As we are only using cortical data, a model order of $d = 20$ was employed here.

The determination of the number k of nearest neighbors used in constructing the similarity graph (see equation 5.9) should also be based on the data. The value of k needs to be large enough to include the vertices that are closely correlated with a reference voxel, but not so large as to include too many weak correlations in which case the model becomes noisier. The nearest neighbors of a voxel v are those voxels that are strongly correlated with it. To select a consistent threshold cut-off, we Z-scored the correlations and utilized a threshold of $Z > 2$. Specifically, for each voxel, its correlation values with all other voxels (corresponding to a single row or column of the group-level correlation matrix) were converted to Z-scores and thresholded. This procedure selects the neighbors that are the most functionally coupled to the voxel in question. In this manner, we determine the number of nonzero values obtained after thresholding for each voxel, and average this count

across all voxels. This average value is then taken as the value of k.

5.3 Results

5.3.1 Resting state dataset

5.3.1.1 Group-level spatial maps

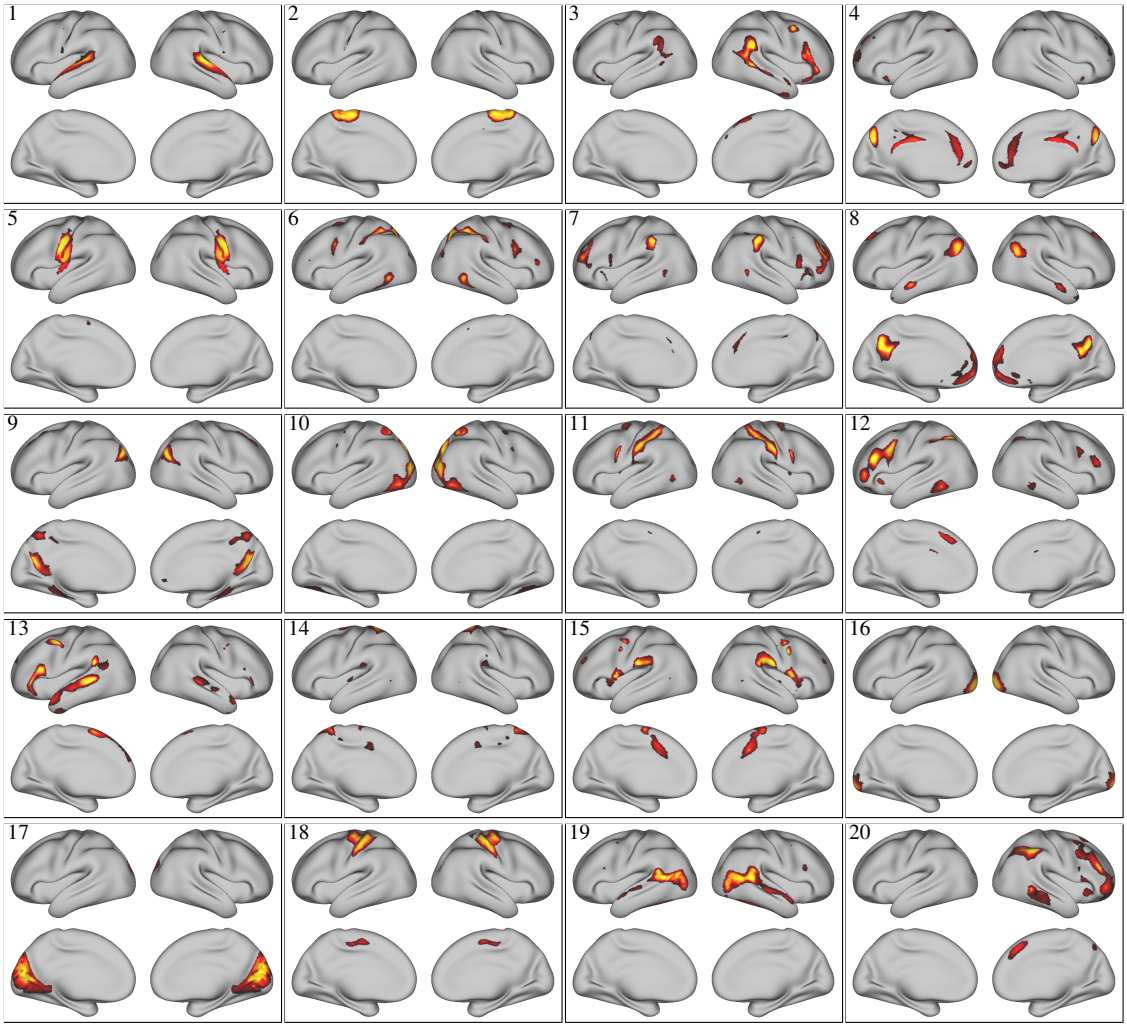


Figure 5.2: Twenty group spatial maps generated by LEICA on the 100-subject HCP preprocessed resting state fMRI dataset. Spatial maps were thresholded at $|Z| > 2$.

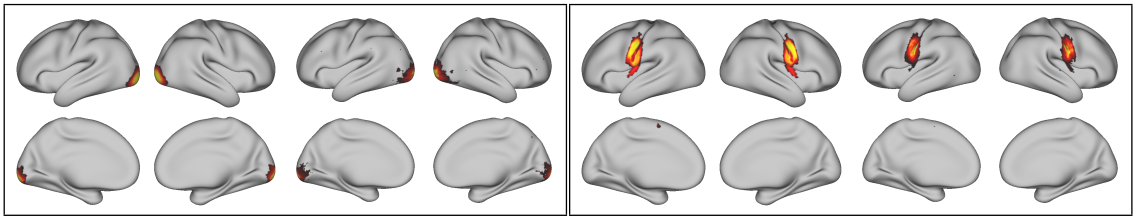
As described in section 5.2.6, an important parameter that is estimated from the data by the model is the number k of nearest neighbors used in constructing the similarity graph. Based on the resting state dataset, $k = 1742$. Figure 5.2 shows the 20 group-level thresholded spatial maps extracted by LEICA on the resting state dataset. The extracted independent spatial components were converted to Z scores and thresholded at $|Z| > 2$.

5.3.1.2 Similarity score

We first visually compared the spatial maps from LEICA and MELODIC. Subsequently, we computed the similarity scores of pairs putatively capturing the same brain networks. Out of the 20 LEICA spatial maps, 18 of them exhibited good matches with the corresponding MELODIC spatial maps. The cosine similarity scores are presented in Table 5.1.

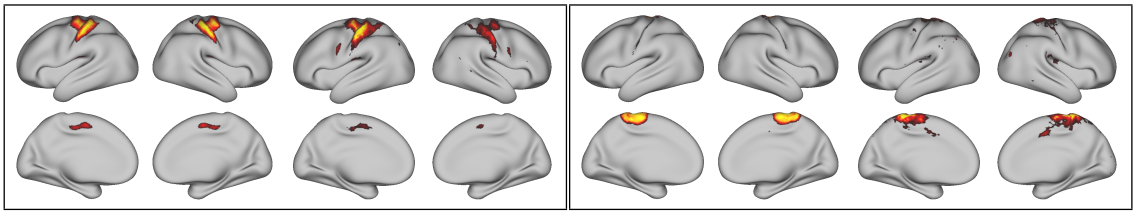
SM #	c	SM #	c
16	0.96	4	0.71
5	0.90	9	0.71
10	0.87	20	0.68
18	0.83	19	0.61
17	0.80	7	0.60
8	0.78	3	0.59
2	0.77	12	0.57
13	0.72	11	0.56
15	0.72	6	0.53

Table 5.1: Cosine similarity scores for MELODIC and LEICA spatial maps (SM) for the resting state dataset. The numbering of the spatial maps is the same as in Figure 5.2.



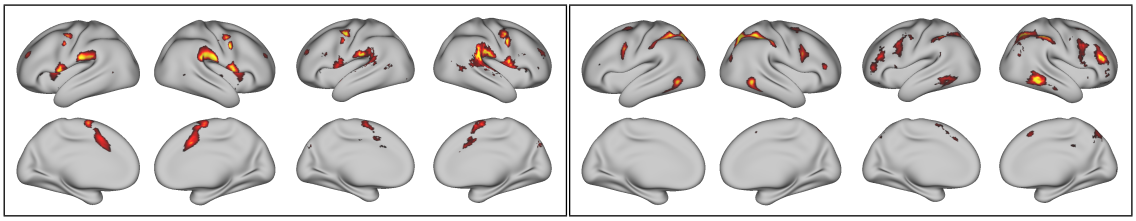
(a) $c = 0.96$

(b) $c = 0.90$



(c) $c = 0.83$

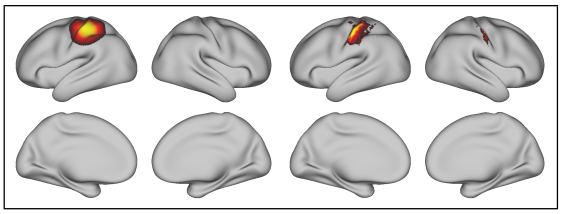
(d) $c = 0.77$



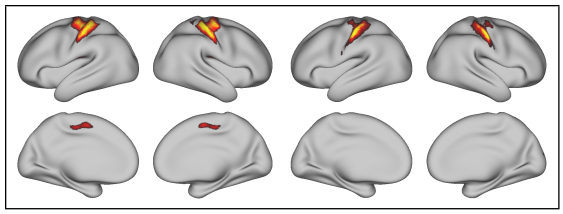
(e) $c = 0.72$

(f) $c = 0.53$

Figure 5.3: Six sample pairs of spatial maps with different cosine similarity scores c. In each frame, the spatial maps on the left are generated by LEICA and those on the right are generated by MELODIC. Both models appear to capture the same components (even in the case with least similarity, $c = 0.53$). The components from our model appear to be smoother than those by MELODIC.



(a) An artificial spatial map (left) generated by drawing a circle around a vertex near the left motor network (with Gaussian weights) and its corresponding correlation map (right). Cosine similarity score: $c = 0.42$.



(b) The motor network generated by LEICA (left) and its corresponding correlation map (right). Cosine similarity score: $c = 0.84$.

Figure 5.4: Two spatial maps and their corresponding correlation maps. The correlation maps were created by computing the weighted sum of the correlation maps for each individual vertex as described in section 5.2.4.2.

In Figure 5.3 we show sample pairs with varying similarity scores. In all cases, the pairs are visually very similar, and even the least similar pair ($c = 0.53$, Figure 5.3(f)), shows good correspondence. Furthermore, LEICA spatial maps are qualitatively smoother than those produced by MELODIC.

Two LEICA components (1 and 14) did not match components generated by MELODIC. Nevertheless, the two maps appear to be related to meaningful functional networks. We tentatively identify spatial map 1 to be related to the auditory network, and spatial map 14 to be related to the fronto-parietal network important for attention and executive function.

Next, we describe a quantitative comparison between LEICA and both MELODIC and GIFT.

5.3.1.3 Recovery of functional connectivity structure

We start with an example to illustrate the validation strategy described in section 5.2.4.2. In Figure 5.4 we show two spatial maps (left in each frame) and the corresponding correlation maps (right). In (a), to generate an artificial map, a sphere with a geodesic radius of 40 mm was generated around a vertex near the motor network and weighted by a Gaussian kernel. In (b), the spatial map corresponded to a motor-related network generated by LEICA. In both cases, correlation maps were created by computing the weighted sum of the correlation maps for each individual vertex (in the spatial maps), as described in section 5.2.4.2. When a spatial map does not closely correspond to a meaningful intrinsic network, such as the case in

SM #	LEICA	MELODIC	GIFT
16	0.96	0.95	0.94
5	0.93	0.93	0.93
18	0.84	0.86	0.86
11	0.82	0.83	0.82
17	0.81	0.79	0.81
6	0.79	0.79	0.81
9	0.79	0.78	0.79
10	0.79	0.78	0.78
13	0.78	0.78	0.78
2	0.78	0.78	0.77
8	0.77	0.77	0.76
7	0.76	0.77	0.76
12	0.74	0.76	0.75
3	0.74	0.74	0.73
14	0.73	0.70	0.67
20	0.73	0.61	0.62
1	0.73	0.54	0.56
19	0.67	0.51	0.50
4	0.66	0.44	0.48
15	0.66	0.18	0.43
average	**0.77**	0.71	0.73

Table 5.2: Cosine similarity between 20 spatial maps generated by LEICA, MELODIC, and GIFT and their correlation maps for the resting state dataset. The numbering of spatial maps (SM #) applies to LEICA; the values for MELODIC and GIFT are sorted in decreasing order.

(a), the similarity score was low ($c = 0.42$), and its correlation map is relatively poor (in (a), compare the two leftmost and the rightmost surfaces). When a spatial map better captures an intrinsic network, such as the case in (b), both images matched well and the cosine similarity score was high ($c = 0.84$).

To evaluate the model, we followed the approach above and compared the individual spatial maps generated by LEICA, MELODIC, and GIFT with the corresponding correlation maps (Table 5.2), just as done in Figure 5.4 (b). LEICA exhibited cosine similarity scores (between component maps and correlation maps)

Resting State	GIFT	MELODIC	LEICA
20 subjects			
p	0.82	0.80	**0.88**
q - non-thresholded	0.79	0.83	**0.83**
q - thresholded	0.48	**0.82**	0.77
30 subjects			
p	0.84	0.85	**0.91**
q - non-thresholded	0.81	0.83	**0.85**
q - thresholded	0.60	**0.84**	0.80
40 subjects			
p	0.89	0.87	**0.92**
q - non-thresholded	0.82	0.84	**0.86**
q - thresholded	0.76	**0.84**	0.81
50 subjects			
p	0.91	0.89	**0.93**
q - non-thresholded	0.87	0.85	**0.89**
q - thresholded	0.82	0.85	**0.85**

(a) Reproducibility results based on pairs of subgroups

Resting State	GIFT	MELODIC	LEICA
20 subjects			
p	0.73	0.71	**0.92**
q - non-thresholded	0.73	0.68	**0.86**
q - thresholded	0.46	0.65	**0.82**
30 subjects			
p	0.75	0.74	**0.94**
q - non-thresholded	0.74	0.69	**0.90**
q - thresholded	0.52	0.68	**0.86**
40 subjects			
p	0.79	0.76	**0.95**
q - non-thresholded	0.75	0.72	**0.90**
q - thresholded	0.61	0.70	**0.87**
50 subjects			
p	0.81	0.77	**0.96**
q - non-thresholded	0.79	0.74	**0.91**
q - thresholded	0.68	0.72	**0.87**

(b) Reproducibility results of one subgroup relative to the whole group

Table 5.3: Reproducibility results based on pairs of subgroups each with 20, 30, 40, 50 subjects for the resting state dataset. p quantifies the similarity of the two subspaces and q measures the one-to-one matching similarity.

that were similar or higher than the two other algorithms, indicating that the method uncovered meaningful component maps that represent intrinsic functional networks.

5.3.1.4 Reproducibility results

We first describe how the results for GIFT and MELODIC were generated. For GIFT, the subject-level dimensionality was first reduced to $m = 150$ using PCA, followed by the temporal concatenation across all reduced data. As stated in [88,89] the subject-specific PCA should not over-reduce the data ($m > d$); usually the larger the m the better, and we chose m to be large for best performance. PCA was then applied on the concatenated data to reduce the dimensionality to $d = 20$, followed by ICA to extract the independent components. For MELODIC, the temporal concatenation PCA was implemented using the MIGP algorithm to compute group-level principal components with dimension of 4500, as used in the original paper [88] for the same dataset. The MELODIC toolbox was then used to extract 20 independent components.

The reproducibility results generated by cross validating a pair of subgroups and by cross validating one subgroup relative to the complete group are shown in Table 5.3. Overall LEICA achieved the most reproducible and stable results among the models. From Table 5.3(a), we see that given any pair of distinct subgroups, LEICA was able to extract a set of group spatial features that was most stable most often. These results are important because they indicate that the inter-subject variability was modeled better using Laplacian eigenmaps than PCA. In Table 5.3(b),

the values from LEICA are also significantly higher, which indicates that LEICA was able to extract more consistent group spatial features when fewer subjects are used.

5.3.2 Working memory dataset

5.3.2.1 Group-level spatial maps

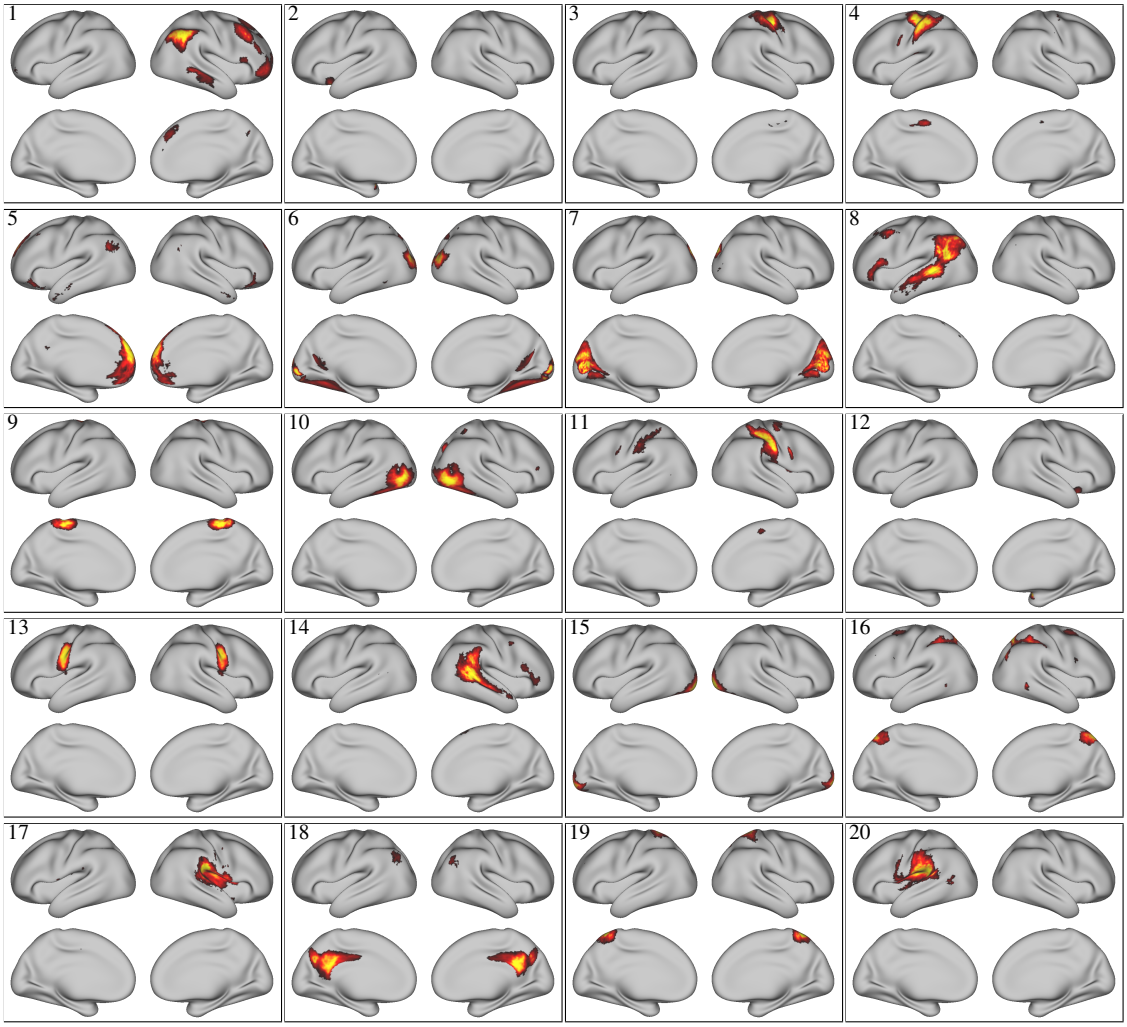


Figure 5.5: Twenty group spatial maps generated by LEICA on the 100-subject HCP preprocessed working memory task dataset. Spatial maps were thresholded at $|Z| > 2$.

The method estimated the number of nearest neighbors used in constructing the similarity graph to be $k = 938$ for the working memory dataset. Figure 5.5 shows the 20 group-level independent thresholded spatial maps extracted by LEICA on the 2-back working memory fMRI dataset. The extracted spatial components were also converted to Z-scores and thresholded at $|Z| > 2$.

5.3.2.2 Similarity score

Out of the 20 LEICA spatial maps, 18 exhibited relatively good matches with those from MELODIC (cosine similarity scores are presented in Table 5.4). Thus, the two models captured similar sets of components.

SM #	c	SM #	c
13	0.97	8	0.67
7	0.95	11	0.63
10	0.93	20	0.58
15	0.86	16	0.56
4	0.81	19	0.44
1	0.77	5	0.41
9	0.76	14	0.40
6	0.75	17	0.38
18	0.71	3	0.35

Table 5.4: Cosine similarity scores for spatial maps (SM) from MELODIC and LEICA on the working memory dataset. The numbering of the spatial maps is the same as in Figure 5.5.

5.3.2.3 Recovery of functional connectivity structure

As in the case of the resting state dataset, we compared spatial maps obtained by LEICA, MELODIC, and GIFT with the corresponding correlation maps (Table 5.5). For the task dataset, LEICA exhibited cosine similarity scores (between component maps and correlation maps) that were consistently higher than the two other algorithms, indicating that the method better uncovered functional networks during working memory performance.

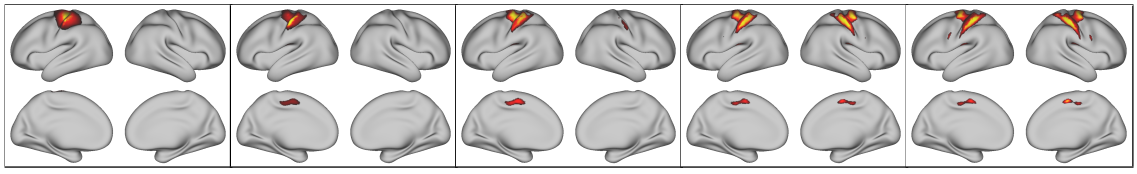
SM #	LEICA	MELODIC	GIFT
2	0.99	0.97	0.99
12	0.99	0.84	0.91
13	0.97	0.80	0.84
9	0.96	0.78	0.81
3	0.93	0.77	0.80
4	0.93	0.74	0.78
11	0.85	0.74	0.77
19	0.85	0.73	0.74
17	0.79	0.72	0.73
15	0.76	0.70	0.71
6	0.74	0.70	0.69
18	0.72	0.64	0.68
5	0.70	0.62	0.67
16	0.69	0.59	0.63
10	0.69	0.55	0.61
20	0.68	0.51	0.60
14	0.68	0.45	0.53
1	0.67	0.44	0.47
8	0.65	0.40	0.34
7	0.61	0.26	0.29
average	**0.79**	0.65	0.68

Table 5.5: Cosine similarity between 20 spatial maps and their correlation maps for LEICA, MELODIC, GIFT on the working memory dataset . The numbering of spatial maps (SM #) only applies to LEICA, and the values for MELODIC and GIFT are sorted in decreasing order.

5.3.2.4 Reproducibility results

Along the same lines of section 5.3.1.4, we chose the subject-level PCA dimension $m = 90$ for GIFT, and a dimension of 200 for MIGP in MELODIC. The reproducibility results for the three models are shown in Table 5.6. Compared with the results for resting state (Table 5.3), all reproducibility scores dropped considerably, reflecting the rather limited amount of task data (200 frames per subject vs. 4800 frames per subject). However, the performance of LEICA did not decline as much as observed for the other two models, highlighting the advantage of the method when more limited data are available, as is typical in most fMRI research.

5.4 Discussion



(a) $k = 50$ (b) $k = 200$ (c) $k = 1000$ (d) $k = 2500$ (e) $k = 5000$

Figure 5.6: The motor networks extracted by LEICA on the resting state dataset with different values of k.

In the present paper, we propose a nonlinear ICA-based model for extracting group-level spatial maps from multi-subject fMRI datasets. There are two key elements to our method. The first element is the construction of a k-nearest neighbor graph based on the group-average correlation matrix, which makes the final results less noisy and captures intrinsic functional connectivity. The second element is the use of a nonlinear dimensionality reduction stage based on Laplacian eigenmaps,

Working Memory	GIFT	MELODIC	LEICA
20 subjects			
p	0.35	0.36	**0.68**
q - non-thresholded	0.41	0.41	**0.70**
q - thresholded	0.27	0.41	**0.68**
30 subjects			
p	0.47	0.45	**0.76**
q - non-thresholded	0.51	0.49	**0.77**
q - thresholded	0.39	0.49	**0.74**
40 subjects			
p	0.51	0.52	**0.81**
q - non-thresholded	0.55	0.55	**0.81**
q - thresholded	0.49	0.54	**0.78**
50 subjects			
p	0.56	0.55	**0.84**
q - non-thresholded	0.59	0.57	**0.84**
q - thresholded	0.55	0.56	**0.80**

(a) Reproducibility results based on pairs of subgroups

Working Memory	GIFT	MELODIC	LEICA
20 subjects			
p	0.49	0.52	**0.73**
q - non-thresholded	0.53	0.52	**0.72**
q - thresholded	0.33	0.50	**0.69**
30 subjects			
p	0.60	0.58	**0.79**
q - non-thresholded	0.63	0.57	**0.79**
q - thresholded	0.42	0.55	**0.76**
40 subjects			
p	0.66	0.65	**0.83**
q - non-thresholded	0.60	0.63	**0.82**
q - thresholded	0.52	0.60	**0.79**
50 subjects			
p	0.69	0.67	**0.86**
q - non-thresholded	0.64	0.65	**0.84**
q - thresholded	0.61	0.62	**0.81**

(b) Reproducibility results of one subgroup relative to the whole group

Table 5.6: Reproducibility results based on subgroups each with 20, 30, 40, 50 subjects on the working memory dataset.

which seeks to identify a manifold subspace common to the group. The mapping is such that it preserves the correlation structure among the voxels as much as possible, which we propose is important to capture functional brain networks during resting and task states.

5.4.1 Model flexibility

LEICA has more parameters than traditional PCA-ICA models. In particular, the construction of the similarity graph requires the specification of the edge weights and the number k of nearest neighbors. A number of techniques exist for specifying edge weights, including Pearson's correlation coefficient, the heat kernel, and binarized weights. As noted before, we believe that the thresholded Pearson's correlation coefficient is appropriate for our application. The number k of nearest neighbors needs to be determined, as we illustrated earlier. Also, the final similarity graph can be fine tuned. In this paper, we have provided practical and simple ways to select the parameters that are particular to our model. These methods seem to work well in practice and may offer more flexibility than found in other models.

5.4.2 Number of nearest neighbors k

LEICA provides a principled method for determining the value of k based on z scoring, which produces good results on both the resting state and working memory datasets. However, we note that the model is not very sensitive to the exact value of k. Figure 5.6 shows motor-related networks extracted by the method applied to

resting state data with different values of k. When k is rather small, say $k = 50$ or even $k = 200$, the resulting motor networks do not represent the true network well. For $k = 1000$, the motor networks are accurately extracted but the left and right hemispheres are separated. When $k = 2500$, the results are nearly identical to what was extracted automatically (Figure 5.2). For $k = 5000$, which we consider as "large" (nearly three times the value of $k = 1742$ obtained by the method), the motor network becomes somewhat noisy but the result is still quite reasonable. We note that for the range $k \in [1200, 2500]$ there seems to be no appreciable differences in the results obtained by the method.

5.4.3 Model order d

The general problem of estimating the model order (here, the number of components) amounts to estimating the intrinsic dimensionality of a nonlinear manifold from a given data sample, a notoriously difficult problem [176]. Moreover, in the context of our application, we note that it is not clear that there is a single "correct" value that is needed. A technique used to estimate the number of components for fMRI data is based on Akaike's information criterion (AIC) and the minimum description length (MDL), both of which make significant technical assumptions about the model [177]. For fMRI datasets, MELODIC and GIFT use this technique for their model parameter estimation but the accuracy is not guaranteed [87]. Another technique is based on a statistical shape model [87]. In the present investigation, we decided against using such techniques due to their considerable assumptions. Previ-

ous empirical results for fMRI data indicate that the appropriate model parameter lies between 20 and 100, depending on the dataset. Here, we found that the model order of 20 was adequate for the two datasets investigated, as shown by the high quality of the spatial components generated, but we note that the exploration of methods to estimate model order was beyond the scope of the paper.

5.4.4 Computational complexity and execution time

When the number of subjects is very large, it becomes computationally difficult to concatenate the data across all subjects along the temporal dimension to perform the typical estimation procedures (such as PCA) due to memory limitations. This issue is handled in GIFT by using successive PCAs to reduce the dimensionality of each individual's data before concatenation across subjects, which is then followed by running PCA again on the concatenated data. In MELODIC, a recently proposed algorithm called MIGP is used to compute an approximation of PCA on the temporally concatenated data. In contrast, LEICA computes the correlation matrix of each individual subject, followed by averaging these correlation matrices over the subjects. We then build a k-nearest neighbor graph based on the resulting correlation matrix. This is followed by computing the Laplacian eigenmaps on the sparse correlation matrix, which includes a sparse eigenvector decomposition that has a fast implementation, and can be easily boosted by GPU platforms as shown in [178].

The execution times of MELODIC, GIFT, and our model on the 100 sub-

ject HCP resting state and working memory datasets are shown in Table 5.7. All programs were run on the same machine with Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz CPU. Overall, LEICA ran much faster than both algorithms on the resting state dataset, and slower than MELODIC and faster than GIFT on the working memory dataset. This can be explained by the fact that the execution time of MELODIC is dominated by the MIGP step, which is quadratic in terms of the length T of the time series, while the execution time of LEICA is linear in T ($T = 4800$ for the resting state dataset and $T = 200$ for the working memory dataset). The MIGP step of MELODIC was implemented in MATLAB (we used the scripts provided by HCP) and the ICA step was from the MELODIC toolbox. Both GIFT and our model were implemented in Python (as the original GIFT toolbox is not compatible with CIFTI data format used by HCP, we implemented the model in Python).

	MELODIC	GIFT	LEICA
resting state	2d 5h 17m	8h 42m	5h 33m
working memory	8m 47s	20m 15s	15m 29s

Table 5.7: Execution times for MELODIC, GIFT, and LEICA on the 100-subject resting state and working memory datasets from the Human Connectome Project.

5.5 Conclusion

In this chapter, we present a novel model – called LEICA – involving a non-linear dimensionality reduction procedure followed by ICA to identify group-level spatial features from multi-subject fMRI datasets, and applied it to resting state and

working memory data. The spatial maps extracted by our model were shown to be related to meaningful functional networks, and are comparable or better than those generated by one of the current state-of-the-art models (MELODIC). Evaluation of the functional connectivity structure of spatial component maps revealed that LEICA detects functionally cohesive maps. Moreover, LEICA spatial maps are at least as functionally cohesive or better than those detected by MELODIC and GIFT for resting state data, and more functionally cohesive than by MELODIC and GIFT for working memory data. Tests of reproducibility showed that LEICA is at least as stable as MELODIC and GIFT when ample data are available (resting state dataset), and more stable than these methods when more limited data are available (working memory dataset). Finally, our method is computationally efficient. Our software is available at `https://github.com/liuchihuang/LEICA`.

Chapter 6: Conclusion and Future Work

6.1 Conclusion

This thesis has investigated reliability problems in machine learning, including robustness, calibration, and reproducibility.

For adversarial robustness, we propose an approach with feature prioritization and regularization. The attention maps learned in the model favor robust features and focuses sharply on the region of interest. The learned gradient maps align perfectly with salient data characteristics, which further proves that our model heavily relies on the robust features. We showed that our model not only significantly improves adversarial robustness, but also improves standard accuracy.

Then we address the confidence calibration problem by proposing a new form of label smoothing. Under some reasonable assumptions, we propose a form of smooth label where the values are based on class similarities, and show that the resulting loss function directly optimizes the objective of confidence calibration. We adopt feature based similarities or semantic similarity, and demonstrate through extensive experiments that our method significantly outperforms state-of-the-art techniques.

Next, we try to address the two problems at the same time. We propose to a method that is able to significantly improve the calibration performance while

having impressive adversarial robustness. We leverage the information in the representation space learned from a network, and train a confidence prediction network for calibrated confidence output using the distances in the representation space.

Finally, we address the reproducibility problem in applying machine learning algorithms to extract functional brain networks from group fMRI data. We use Laplacian eigenmaps as the data reduction step in order to preserve the correlation information in the original data as best as possible. This nonlinear map is robust relative to noise in the data and to inter-subject variability. We show that LEICA detects functionally cohesive maps that are much more reproducible than the state-of-the-art methods.

6.2 Future Work

Our work in Chapter 2 employs regularizer to change the decision boundary of the neural network and improve robustness. There are an extensive amount of works using various regularizers, including TRADES [48], entropic regularization [179], adversarial logit pairing [45], ensemble diversity [180], local linearization [181], and hypersphere embedding regularization [182]. They are inspired from different perspectives, and new types of regularizers can be explored.

Most works that improve confidence calibration are post-hoc based, i.e., they transform the network's predictions based on a held-out validation set to improve calibration after the model is trained. Our work in Chapter 3 is one of the few that are non-post-hoc. Post-hoc approaches generally work well but they rely on

a well-sampled validation set and require that the testing data is homogeneous to the validataion set, which significantly restricted their application. In our work we considered several similarity metrics, and it could be extended by exploring more measures. Also, instead of having class-level smooth labels, using individual sample labels could be more beneficial.

Our work in Chapter 4 provides promising results on improving robustness and calibration of a neural network at the same time. While we used prototypical learning, other types of model that provides useful representation space could be considered, such as metric learning. We can also try this problem from another aspect, such as changing the adversarial training algorithm or adding regularizers that optimize both robustness and calibration.

For applying machine learning in neuroscience, there is a huge room for progress. We employed traditional machine learning algorithms such as the Laplacian Eigenmaps and ICA. With the impressive achievements in machine learning and deep learning, investigating on how to incorporate more advanced techniques will be very interesting.

Bibliography

[1] Kevin P Murphy et al. Naive bayes classifiers. *University of British Columbia*, 18(60):1–8, 2006.

[2] George AF Seber and Alan J Lee. *Linear regression analysis*, volume 329. John Wiley & Sons, 2012.

[3] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

[4] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[5] Johan AK Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

[6] Tapas Kanungo, David M Mount, Nathan S Netanyahu, Christine D Piatko, Ruth Silverman, and Angela Y Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE transactions on pattern analysis and machine intelligence*, 24(7):881–892, 2002.

[7] Zoran Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 2, pages 28–31. IEEE, 2004.

[8] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[9] Martin J McKeown, Scott Makeig, Greg G Brown, Tzyy-Ping Jung, Sandra S Kindermann, Anthony J Bell, and Terrence J Sejnowski. Analysis of fmri data by blind separation into independent spatial components. Technical report, DTIC Document, 1997.

[10] 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.

[11] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[13] Muhammad Imran Razzak, Saeeda Naz, and Ahmad Zaib. Deep learning for medical image processing: Overview, challenges and the future. In *Classification in BioApps*, pages 323–350. Springer, 2018.

[14] Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, and Joel S Emer. Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE*, 105(12):2295–2329, 2017.

[15] Shihao Gu, Bryan Kelly, and Dacheng Xiu. Empirical asset pricing via machine learning. Technical report, National Bureau of Economic Research, 2018.

[16] Maithra Raghu and Eric Schmidt. A survey of deep learning for scientific discovery. *arXiv preprint arXiv:2003.11755*, 2020.

[17] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[18] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2015.

[19] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[20] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[21] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*, pages 372–387. IEEE, 2016.

[22] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

[23] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

[24] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017.

[25] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[26] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018.

[27] Jonathan Uesato, Brendan O'donoghue, Pushmeet Kohli, and Aaron Oord. Adversarial risk and the dangers of evaluating against weak attacks. In *International Conference on Machine Learning*, pages 5025–5034. PMLR, 2018.

[28] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

[29] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519. ACM, 2017.

[30] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

[31] Cihang Xie, Zhishuai Zhang, Jianyu Wang, Yuyin Zhou, Zhou Ren, and Alan Yuille. Improving transferability of adversarial examples with input diversity. *arXiv preprint arXiv:1803.06978*, 2018.

[32] Yucheng Shi, Siyu Wang, and Yahong Han. Curls & whey: Boosting black-box adversarial attacks. *arXiv preprint arXiv:1904.01160*, 2019.

[33] Nina Narodytska and Shiva Kasiviswanathan. Simple black-box adversarial attacks on deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1310–1318. IEEE, 2017.

[34] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 15–26. ACM, 2017.

[35] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. *arXiv preprint arXiv:1804.08598*, 2018.

[36] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

[37] Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Prior convictions: Black-box adversarial attacks with bandits and priors. *arXiv preprint arXiv:1807.07978*, 2018.

[38] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *arXiv preprint arXiv:1907.07174*, 2019.

[39] Anish Athalye and Nicholas Carlini. On the robustness of the cvpr 2018 white-box adversarial example defenses. *arXiv preprint arXiv:1804.03286*, 2018.

[40] Nicholas Carlini and David Wagner. Defensive distillation is not robust to adversarial examples. *arXiv preprint arXiv:1607.04311*, 2016.

[41] Taesik Na, Jong Hwan Ko, and Saibal Mukhopadhyay. Cascade adversarial machine learning regularized with a unified embedding. *arXiv preprint arXiv:1708.02582*, 2017.

[42] Dong Yin, Kannan Ramchandran, and Peter Bartlett. Rademacher complexity for adversarially robust generalization. *arXiv preprint arXiv:1810.11914*, 2018.

[43] Aman Sinha, Hongseok Namkoong, and John Duchi. Certifying some distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571*, 2017.

[44] Andrew Slavin Ross and Finale Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[45] Harini Kannan, Alexey Kurakin, and Ian Goodfellow. Adversarial logit pairing. *arXiv preprint arXiv:1803.06373*, 2018.

[46] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 854–863. JMLR. org, 2017.

[47] Farzan Farnia, Jesse M Zhang, and David Tse. Generalizable adversarial training via spectral normalization. *arXiv preprint arXiv:1811.07457*, 2018.

[48] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P Xing, Laurent El Ghaoui, and Michael I Jordan. Theoretically principled trade-off between robustness and accuracy. *arXiv preprint arXiv:1901.08573*, 2019.

[49] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *arXiv preprint arXiv:1801.09344*, 2018.

[50] Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5283–5292, 2018.

[51] Eric Wong, Frank Schmidt, Jan Hendrik Metzen, and J Zico Kolter. Scaling provable adversarial defenses. In *Advances in Neural Information Processing Systems*, pages 8400–8409, 2018.

[52] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*, pages 97–117. Springer, 2017.

[53] Ruediger Ehlers. Formal verification of piece-wise linear feed-forward neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, pages 269–286. Springer, 2017.

[54] Matthias Hein and Maksym Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *Advances in Neural Information Processing Systems*, pages 2266–2276, 2017.

[55] Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. Evaluating the robustness of neural networks: An extreme value theory approach. *arXiv preprint arXiv:1801.10578*, 2018.

[56] Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Duane Boning, Inderjit S Dhillon, and Luca Daniel. Towards fast computation of certified robustness for relu networks. *arXiv preprint arXiv:1804.09699*, 2018.

[57] Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. In *Advances in Neural Information Processing Systems*, pages 4939–4948, 2018.

[58] Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against adversarial attacks using high-level representation guided denoiser. *arXiv preprint arXiv:1712.02976*, 2017.

[59] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. *arXiv preprint arXiv:1812.03411*, 2018.

[60] Lukas Schott, Jonas Rauber, Matthias Bethge, and Wieland Brendel. Towards the first adversarially robust neural network model on mnist. *arXiv preprint arXiv:1805.09190*, 2018.

[61] Xuanqing Liu and Cho-Jui Hsieh. From adversarial training to generative adversarial networks. *arXiv preprint arXiv:1807.10454*, 2018.

[62] Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[63] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1321–1330. JMLR. org, 2017.

[64] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.

[65] Ananya Kumar, Percy S Liang, and Tengyu Ma. Verified uncertainty calibration. In *Advances in Neural Information Processing Systems*, pages 3792–3803, 2019.

[66] Jize Zhang, Bhavya Kailkhura, and T Han. Mix-n-match: Ensemble and compositional methods for uncertainty calibration in deep learning. *arXiv preprint arXiv:2003.07329*, 2020.

[67] John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[68] Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *Icml*, volume 1, pages 609–616. Citeseer, 2001.

[69] Meelis Kull, Telmo Silva Filho, and Peter Flach. Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers. In *Artificial Intelligence and Statistics*, pages 623–631. PMLR, 2017.

[70] Meelis Kull, Miquel Perello Nieto, Markus Kängsepp, Telmo Silva Filho, Hao Song, and Peter Flach. Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with dirichlet calibration. In *Advances in Neural Information Processing Systems*, pages 12316–12326, 2019.

[71] Kanil Patel, William Beluch, Bin Yang, Michael Pfeiffer, and Dan Zhang. Multi-class uncertainty calibration via mutual information maximization-based binning. *arXiv preprint arXiv:2006.13092*, 2020.

[72] Kartik Gupta, Amir Rahimi, Thalaiyasingam Ajanthan, Thomas Mensink, Cristian Sminchisescu, and Richard Hartley. Calibration of neural networks using splines. *arXiv preprint arXiv:2006.12800*, 2020.

[73] Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip HS Torr, and Puneet K Dokania. Calibrating deep neural networks using focal loss. *arXiv preprint arXiv:2002.09437*, 2020.

[74] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[75] Rafael Müller, Simon Kornblith, and Geoffrey Hinton. When does label smoothing help? *arXiv preprint arXiv:1906.02629*, 2019.

[76] Sunil Thulasidasan, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *Advances in Neural Information Processing Systems*, pages 13888–13899, 2019.

[77] 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.

[78] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

[79] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[80] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems*, pages 13153–13164, 2019.

[81] Alex Graves. Practical variational inference for neural networks. In *Advances in neural information processing systems*, pages 2348–2356, 2011.

[82] Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, volume 6. International Conference on Representation Learning, 2018.

[83] Monya Baker. 1,500 scientists lift the lid on reproducibility. *Nature News*, 533(7604):452, 2016.

[84] C Glenn Begley and John PA Ioannidis. Reproducibility in science: improving the standard for basic and preclinical research. *Circulation research*, 116(1):116–126, 2015.

[85] Marcus R Munafò, Brian A Nosek, Dorothy VM Bishop, Katherine S Button, Christopher D Chambers, Nathalie Percie Du Sert, Uri Simonsohn, Eric-Jan Wagenmakers, Jennifer J Ware, and John PA Ioannidis. A manifesto for reproducible science. *Nature human behaviour*, 1(1):1–9, 2017.

[86] Russell A Poldrack, Chris I Baker, Joke Durnez, Krzysztof J Gorgolewski, Paul M Matthews, Marcus R Munafò, Thomas E Nichols, Jean-Baptiste Poline, Edward Vul, and Tal Yarkoni. Scanning the horizon: towards transparent and reproducible neuroimaging research. *Nature reviews neuroscience*, 18(2):115–126, 2017.

[87] Gaël Varoquaux, Sepideh Sadaghiani, Philippe Pinel, Andreas Kleinschmidt, Jean-Baptiste Poline, and Bertrand Thirion. A group model for stable multi-subject ica on fmri datasets. *Neuroimage*, 51(1):288–299, 2010.

[88] Stephen M Smith, Aapo Hyvärinen, Gaël Varoquaux, Karla L Miller, and Christian F Beckmann. Group-pca for very large fmri datasets. *Neuroimage*, 101:738–749, 2014.

[89] Vince D Calhoun, Tulay Adali, Godfrey D Pearlson, and JJ Pekar. A method for making group inferences from functional mri data using independent component analysis. *Human brain mapping*, 14(3):140–151, 2001.

[90] Christian F Beckmann and Stephen M Smith. Tensorial extensions of independent component analysis for multisubject fmri analysis. *Neuroimage*, 25(1):294–311, 2005.

[91] Ying Guo and Giuseppe Pagnoni. A unified framework for group independent component analysis for multi-subject fmri data. *NeuroImage*, 42(3):1078–1093, 2008.

[92] Chihuang Liu and Joseph JaJa. Feature prioritization and regularization improve standard accuracy and adversarial robustness. *arXiv preprint arXiv:1810.02424*, 2018.

[93] Chihuang Liu and Joseph JaJa. Class-similarity based label smoothing for generalized confidence calibration. *arXiv preprint arXiv:2006.14028*, 2020.

[94] Chihuang Liu, Joseph JaJa, and Luiz Pessoa. Leica: Laplacian eigenmaps for group ica decomposition of fmri data. *NeuroImage*, 169:363–373, 2018.

[95] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

[96] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[97] Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Analysis of classifiers' robustness to adversarial perturbations. *Machine Learning*, 107(3):481–508, 2018.

[98] Anish Athalye and Ilya Sutskever. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397*, 2017.

[99] J Zico Kolter and Eric Wong. Provable defenses against adversarial examples via the convex outer adversarial polytope. *arXiv preprint arXiv:1711.00851*, 2017.

[100] Aaditya Prakash, Nick Moran, Solomon Garber, Antonella DiLillo, and James Storer. Deflecting adversarial attacks with pixel deflection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8571–8580, 2018.

[101] Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*, 2017.

[102] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

[103] Krishnamurthy Dvijotham, Sven Gowal, Robert Stanforth, Relja Arandjelovic, Brendan O'Donoghue, Jonathan Uesato, and Pushmeet Kohli. Training verified learners with learned verifiers. *arXiv preprint arXiv:1805.10265*, 2018.

[104] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. There is no free lunch in adversarial robustness (but there are unexpected benefits). *arXiv preprint arXiv:1805.12152*, 2018.

[105] Thomas Tanay, Jerone TA Andrews, and Lewis D Griffin. Built-in vulnerabilities to imperceptible adversarial perturbations. *arXiv preprint arXiv:1806.07409*, 2018.

[106] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *arXiv preprint arXiv:1801.00553*, 2018.

[107] Paul Hongsuck Seo, Zhe Lin, Scott Cohen, Xiaohui Shen, and Bohyung Han. Progressive attention networks for visual attribute prediction. *arXiv preprint arXiv:1606.02393*, 2016.

[108] Saumya Jetley, Nicholas A Lord, Namhoon Lee, and Philip HS Torr. Learn to pay attention. *arXiv preprint arXiv:1804.02391*, 2018.

[109] Chuanbiao Song, Kun He, Liwei Wang, and John E Hopcroft. Improving the generalization of adversarial training with domain adaptation. *arXiv preprint arXiv:1810.00740*, 2018.

[110] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[111] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[112] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2722–2730, 2015.

[113] Miles Brundage, Shahar Avin, Jack Clark, Helen Toner, Peter Eckersley, Ben Garfinkel, Allan Dafoe, Paul Scharre, Thomas Zeitzoff, Bobby Filar, et al. The malicious use of artificial intelligence: Forecasting, prevention, and mitigation. *arXiv preprint arXiv:1802.07228*, 2018.

[114] Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017.

[115] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3967–3976, 2019.

[116] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[117] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

[118] Alexey Dosovitskiy and Thomas Brox. Generating images with perceptual similarity metrics based on deep networks. In *Advances in neural information processing systems*, pages 658–666, 2016.

[119] David Berthelot, Colin Raffel, Aurko Roy, and Ian Goodfellow. Understanding and improving interpolation in autoencoders via an adversarial regularizer. *arXiv preprint arXiv:1807.07543*, 2018.

[120] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[121] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *Iclr*, 2(5):6, 2017.

[122] Clemens-Alexander Brust and Joachim Denzler. Not just a matter of semantics: the relationship between visual similarity and semantic similarity. *arXiv preprint arXiv:1811.07120*, 2018.

[123] Thomas Deselaers and Vittorio Ferrari. Visual and semantic similarity in imagenet. In *CVPR 2011*, pages 1777–1784. IEEE, 2011.

[124] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[125] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196, 2014.

[126] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[127] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

[128] 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.

[129] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[130] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models.

[131] Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. Wikipedia2vec: an optimized tool for learning embeddings of words and entities from wikipedia. *arXiv preprint arXiv:1812.06280*, 2018.

[132] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

[133] Lassi Meronen, Christabella Irwanto, and Arno Solin. Stationary activations for uncertainty calibration in deep learning. *arXiv preprint arXiv:2010.09494*, 2020.

[134] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *arXiv preprint arXiv:1904.12843*, 2019.

[135] Ranganath Krishnan and Omesh Tickoo. Improving model calibration with accuracy versus uncertainty optimization. *arXiv preprint arXiv:2012.07923*, 2020.

[136] David Stutz, Matthias Hein, and Bernt Schiele. Confidence-calibrated adversarial training: Generalizing to unseen attacks. In *International Conference on Machine Learning*, pages 9155–9166. PMLR, 2020.

[137] Yao Qin, Xuezhi Wang, Alex Beutel, and Ed H Chi. Improving uncertainty estimates through the relationship with adversarial robustness. *arXiv preprint arXiv:2006.16375*, 2020.

[138] Bharat Biswal, F Zerrin Yetkin, Victor M Haughton, and James S Hyde. Functional connectivity in the motor cortex of resting human brain using echo-planar mri. *Magnetic resonance in medicine*, 34(4):537–541, 1995.

[139] Michael W Cole, Danielle S Bassett, Jonathan D Power, Todd S Braver, and Steven E Petersen. Intrinsic and task-evoked network architectures of the human brain. *Neuron*, 83(1):238–251, 2014.

[140] Michael D Greicius, Ben Krasnow, Allan L Reiss, and Vinod Menon. Functional connectivity in the resting brain: a network analysis of the default mode hypothesis. *Proceedings of the National Academy of Sciences*, 100(1):253–258, 2003.

[141] Madiha J Jafri, Godfrey D Pearlson, Michael Stevens, and Vince D Calhoun. A method for functional network connectivity among spatially independent resting-state components in schizophrenia. *Neuroimage*, 39(4):1666–1681, 2008.

[142] Michael D Greicius, Gaurav Srivastava, Allan L Reiss, and Vinod Menon. Default-mode network activity distinguishes alzheimer's disease from healthy aging: evidence from functional mri. *Proceedings of the National Academy of Sciences of the United States of America*, 101(13):4637–4642, 2004.

[143] Abigail G Garrity, Godfrey D Pearlson, Kristen McKiernan, Dan Lloyd, Kent A Kiehl, and Vince D Calhoun. Aberrant "default mode" functional connectivity in schizophrenia. *American journal of psychiatry*, 164(3):450–457, 2007.

[144] JS Damoiseaux, CF Beckmann, EJ Sanz Arigita, F Barkhof, Ph Scheltens, CJ Stam, SM Smith, and SARB Rombouts. Reduced resting-state brain activity in the "default network" in normal aging. *Cerebral cortex*, 18(8):1856–1864, 2008.

[145] Irit Weissman-Fogel, Massieh Moayedi, Keri S Taylor, Geoff Pope, and Karen D Davis. Cognitive and default-mode resting state networks: Do male and female brains "rest" differently? *Human brain mapping*, 31(11):1713–1726, 2010.

[146] Erik Barry Erhardt, Srinivas Rachakonda, Edward J Bedrick, Elena A Allen, Tülay Adali, and Vince D Calhoun. Comparison of multi-subject ica methods for analysis of fmri data. *Human brain mapping*, 32(12):2075–2095, 2011.

[147] Christian F Beckmann and Stephen M Smith. Probabilistic independent component analysis for functional magnetic resonance imaging. *IEEE transactions on medical imaging*, 23(2):137–152, 2004.

[148] Yongmei Michelle Wang, Robert T Schultz, R Todd Constable, and Lawrence H Staib. Nonlinear estimation and modeling of fmri data using spatio-temporal support vector regression. In *Biennial International Conference on Information Processing in Medical Imaging*, pages 647–659. Springer, 2003.

[149] Rasmus M Birn, Ziad S Saad, and Peter A Bandettini. Spatial heterogeneity of the nonlinear dynamics in the fmri bold response. *Neuroimage*, 14(4):817–826, 2001.

[150] Karla L Miller, Wen-Ming Luh, Thomas T Liu, Antigona Martinez, Takayuki Obata, Eric C Wong, Lawrence R Frank, and Richard B Buxton. Nonlinear temporal dynamics of the cerebral blood flow response. *Human brain mapping*, 13(1):1–12, 2001.

[151] Xiaoping Xie, Zhitong Cao, and Xuchu Weng. Spatiotemporal nonlinearity in resting-state fmri of the human brain. *Neuroimage*, 40(4):1672–1685, 2008.

[152] Bertrand Thirion and Olivier Faugeras. Nonlinear dimension reduction of fmri data: the laplacian embedding approach. In *Biomedical Imaging: Nano to Macro, 2004. IEEE International Symposium on*, pages 372–375. IEEE, 2004.

[153] Peter Mannfolk, Ronnie Wirestam, Markus Nilsson, Freddy Ståhlberg, and Johan Olsrud. Dimensionality reduction of fmri time series data using locally linear embedding. *Magnetic Resonance Materials in Physics, Biology and Medicine*, 23(5-6):327–338, 2010.

[154] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. Dimensionality reduction: a comparative review. *J Mach Learn Res*, 10:66–71, 2009.

[155] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[156] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[157] Kilian Q Weinberger and Lawrence K Saul. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In *AAAI*, volume 6, pages 1683–1686, 2006.

[158] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

[159] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[160] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

[161] Andre F Marquand, Koen V Haak, and Christian F Beckmann. Functional corticostriatal connection topographies predict goal directed behaviour in humans. *bioRxiv*, page 169151, 2017.

[162] Koen V Haak, Andre F Marquand, and Christian F Beckmann. Connectopic mapping with resting-state fmri. *NeuroImage*, 2017.

[163] Stephen M Smith, Mark Jenkinson, Mark W Woolrich, Christian F Beckmann, Timothy EJ Behrens, Heidi Johansen-Berg, Peter R Bannister, Marilena De Luca, Ivana Drobnjak, David E Flitney, et al. Advances in functional and structural mr image analysis and implementation as fsl. *Neuroimage*, 23:S208–S219, 2004.

[164] Michael D Fox, Abraham Z Snyder, Justin L Vincent, Maurizio Corbetta, David C Van Essen, and Marcus E Raichle. The human brain is intrinsically organized into dynamic, anticorrelated functional networks. *Proceedings of the National Academy of Sciences of the United States of America*, 102(27):9673–9678, 2005.

[165] David C Van Essen, Stephen M Smith, Deanna M Barch, Timothy EJ Behrens, Essa Yacoub, Kamil Ugurbil, WU-Minn HCP Consortium, et al. The wu-minn human connectome project: an overview. *Neuroimage*, 80:62–79, 2013.

[166] BT Thomas Yeo, Fenna M Krienen, Jorge Sepulcre, Mert R Sabuncu, Danial Lashkari, Marisa Hollinshead, Joshua L Roffman, Jordan W Smoller, Lilla Zöllei, Jonathan R Polimeni, et al. The organization of the human cerebral cortex estimated by intrinsic functional connectivity. *Journal of neurophysiology*, 106(3):1125–1165, 2011.

[167] R Cameron Craddock, G Andrew James, Paul E Holtzheimer, Xiaoping P Hu, and Helen S Mayberg. A whole brain fmri atlas generated via spatially constrained spectral clustering. *Human brain mapping*, 33(8):1914–1928, 2012.

[168] Adriana Di Martino, Anouk Scheres, Daniel S Margulies, AMC Kelly, Lucina Q Uddin, Zarrar Shehzad, B Biswal, Judith R Walters, F Xavier Castellanos, and Michael P Milham. Functional connectivity of human striatum: a resting state fmri study. *Cerebral cortex*, 18(12):2735–2747, 2008.

[169] Tianzi Jiang, Yong He, Yufeng Zang, and Xuchu Weng. Modulation of functional connectivity during the resting state and the motor task. *Human brain mapping*, 22(1):63–71, 2004.

[170] Ludovica Griffanti, Gholamreza Salimi-Khorshidi, Christian F Beckmann, Edward J Auerbach, Gwenaëlle Douaud, Claire E Sexton, Enikő Zsoldos, Klaus P Ebmeier, Nicola Filippini, Clare E Mackay, et al. Ica-based artefact removal and accelerated fmri acquisition for improved resting state network imaging. *Neuroimage*, 95:232–247, 2014.

[171] Gholamreza Salimi-Khorshidi, Gwenaëlle Douaud, Christian F Beckmann, Matthew F Glasser, Ludovica Griffanti, and Stephen M Smith. Automatic denoising of functional mri data: combining independent component analysis and hierarchical fusion of classifiers. *Neuroimage*, 90:449–468, 2014.

[172] Emma C Robinson, Saad Jbabdi, Matthew F Glasser, Jesper Andersson, Gregory C Burgess, Michael P Harms, Stephen M Smith, David C Van Essen, and Mark Jenkinson. Msm: a new flexible framework for multimodal surface matching. *Neuroimage*, 100:414–426, 2014.

[173] Stephen M Smith, Christian F Beckmann, Jesper Andersson, Edward J Auerbach, Janine Bijsterbosch, Gwenaëlle Douaud, Eugene Duff, David A Feinberg, Ludovica Griffanti, Michael P Harms, et al. Resting-state fmri in the human connectome project. *Neuroimage*, 80:144–168, 2013.

[174] Mark W Woolrich, Brian D Ripley, Michael Brady, and Stephen M Smith. Temporal autocorrelation in univariate linear modeling of fmri data. *Neuroimage*, 14(6):1370–1386, 2001.

[175] Robert W Cox. Afni: software for analysis and visualization of functional magnetic resonance neuroimages. *Computers and Biomedical research*, 29(3):162–173, 1996.

[176] Francesco Camastra and Antonino Staiano. Intrinsic dimension estimation: Advances and open problems. *Information Sciences*, 328:26–41, 2016.

[177] Mati Wax and Thomas Kailath. Detection of signals by information theoretic criteria. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 33(2):387–392, 1985.

[178] Yu Jin and Joseph F Jaja. A high performance implementation of spectral clustering on cpu-gpu platforms. In *Parallel and Distributed Processing Symposium Workshops, 2016 IEEE International*, pages 825–834. IEEE, 2016.

[179] Gauri Jagatap, Ameya Joshi, Animesh Basak Chowdhury, Siddharth Garg, and Chinmay Hegde. Adversarially robust learning via entropic regularization. *arXiv preprint arXiv:2008.12338*, 2020.

[180] Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. Improving adversarial robustness via promoting ensemble diversity. In *International Conference on Machine Learning*, pages 4970–4979. PMLR, 2019.

[181] Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and Pushmeet Kohli. Adversarial robustness through local linearization. *arXiv preprint arXiv:1907.02610*, 2019.

[182] Tianyu Pang, Xiao Yang, Yinpeng Dong, Kun Xu, Jun Zhu, and Hang Su. Boosting adversarial training with hypersphere embedding. *arXiv preprint arXiv:2002.08619*, 2020.