Robust and Generalizable Machine Learning through Generative Models, Adversarial Training, and Physics Priors

by

Houpu Yao

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved July 2019 by the
Graduate Supervisory Committee:

Yi Ren, Chair
Yongming Liu
Baoxin Li
Yezhou Yang
Hamidreza Marvi

ARIZONA STATE UNIVERSITY

August 2019

ABSTRACT

Machine learning has demonstrated great potential across a wide range of applications such as computer vision, robotics, speech recognition, drug discovery, material science, and physics simulation. Despite its current success, however, there are still two major challenges for machine learning algorithms: limited robustness and generalizability.

The robustness of a neural network is defined as the stability of the network output under small input perturbations. It has been shown that neural networks are very sensitive to input perturbations, and the prediction from convolutional neural networks can be totally different for input images that are visually indistinguishable to human eyes. Based on such property, hackers can reversely engineer the input to trick machine learning systems in targeted ways. These adversarial attacks have shown to be surprisingly effective, which has raised serious concerns over safety-critical applications like autonomous driving. In the meantime, many established defense mechanisms have shown to be vulnerable under more advanced attacks proposed later, and how to improve the robustness of neural networks is still an open question.

The generalizability of neural networks refers to the ability of networks to perform well on unseen data rather than just the data that they were trained on. Neural networks often fail to carry out reliable generalizations when the testing data is of different distribution compared with the training one, which will make autonomous driving systems risky under new environment. The generalizability of neural networks can also be limited whenever there is a scarcity of training data, while it can be expensive to acquire large datasets either experimentally or numerically for engineering applications, such as material and chemical design.

In this dissertation, we are thus motivated to improve the robustness and generalizability of neural networks. Firstly, unlike traditional bottom-up classifiers, we use a pre-trained generative model to perform top-down reasoning and infer the label information. The proposed generative classifier has shown to be promising in handling input distribution shifts. Secondly, we focus on improving the network robustness and propose an extension to adversarial training by considering the transformation invariance. Proposed method improves the robustness over state-of-the-art methods by 2.5% on MNIST and 3.7% on CIFAR-10. Thirdly, we focus on designing networks that generalize well at predicting physics response. Our physics prior knowledge is used to guide the designing of the network architecture, which enables efficient learning and inference. Proposed network is able to generalize well even when it is trained with a single image pair.

DEDICATION

To my parents and grandparents.

ACKNOWLEDGMENTS

First of all, I would like to express my sincere gratitude to my advisor Dr. Yi Ren, for his guidance, encouragement, continuous support, and patience on me over the past years. His passion, dedication, and rigorous altitude towards research have educated me and guided me throughout this Ph.D. study. I'm grateful to all the effort he had on me, which really helped me in so many aspects. And I would always remember how he taught me to do solid research, and to solve the real problems.

I would like to thank my committee members, Dr. Yongming Liu, Dr. Yezhou Yang, Dr. Hamid Marvi, and Dr. Baoxin Li for their precious time and valuable advice. I had an enjoyable experience working with Dr. Liu and his students on several interesting topics. I really appreciated his broad knowledge, kindness and generosity. Dr. Yang gave me many good suggestions on my research and offered me lots of opportunities to learn from his students, as well as the access to his GPU resources.

And I want to thank all other people that I have been working with during these years: Ruijin Cang, Thurston Sexton, Malcolm Regan, Guangyu Nie, Jingjing Wen, Zhe Wang, Yassine Mazboudi, Yi Gao, Yang Yu, Yutian Pang, Rui Dai. I learned a lot from you, and it was a great pleasure to have worked with you.

I would like to take the opportunity to thank Joshua Steel and his family, Kai Zhou, Yunyi Kang, Xiangwei Peng, Beibei Liu, Zuyuan Zhang, Panpan Xu, Hongchen Lu, and all those friends who have cared me and have made my life more colorful during this period.

Lastly, I would like to express my deepest thanks to my girlfriend Meilin Zeng and my parents that have always supported me wholeheartedly along this Ph.D. journey and beyond.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

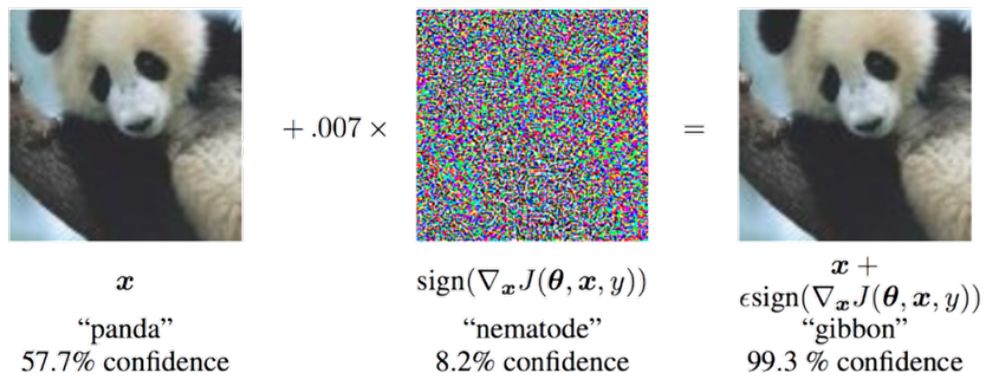
Chapter 1

INTRODUCTION

1.1 Machine Learning and Its Limitations

Neural networks have achieved much success over the past decades. They have broken many records in computer vision (He *et al.*, 2015; Szegedy *et al.*, 2016; Xie *et al.*, 2017b), speech recognition(Graves *et al.*, 2013; Amodei *et al.*, 2016), natural language processing (Kumar *et al.*, 2016; Devlin *et al.*, 2018), and control (Mnih *et al.*, 2015; Silver *et al.*, 2017). Besides, it has also demonstrated great potential in various engineering fields such as astronomy (Graff *et al.*, 2014), molecule and drug discovery (Gilmer *et al.*, 2017; Chen *et al.*, 2018), particle physics (Henrion *et al.*, 2017; de Oliveira *et al.*, 2017), fault diagnosis and prognostics (Jia *et al.*, 2016), material and structure design (Cang and Ren, 2016; Cang *et al.*, 2017; Sosnovik and Oseledets, 2017a), and fluid simulation (Tompson *et al.*, 2016; Chu and Thuerey, 2017).

While deep neural networks have achieved near-human performance in almost all machine perception tasks, it is found that these models can be very sensitive to small but carefully designed input perturbations, as shown in Fig. 1.1. Such property allows the attackers to fool a machine in targeted ways by reverse engineering the inputs (Szegedy *et al.*, 2013; Goodfellow *et al.*, 2014b; Akhtar and Mian, 2018). Recent studies have demonstrated attacks on different neural networks, for example, in image classification (Nguyen *et al.*, 2015; Moosavi-Dezfooli *et al.*, 2016; Kurakin *et al.*, 2016a; Eykholt *et al.*, 2018), detection and segmentation (Hendrik Metzen *et al.*, 2017; Xie *et al.*, 2017a), image retrieval (Sharif *et al.*, 2016), and reinforcement learning (Huang *et al.*, 2017b; Kos and Song, 2017). Furthermore, it has also been demonstrated that these attacks can be successful under real-world settings (Kurakin *et al.*, 2016b; Papernot *et al.*, 2017; Athalye and Sutskever, 2017; Evtimov *et al.*, 2017), posing much threat to applications such as autonomous driving, surveillance, and biomedical, where safety can be critical.



$+ .007 \times$
$=$
$\boldsymbol{x}$
"panda"
57.7% confidence
$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"nematode"
8.2% confidence
$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"gibbon"
99.3 % confidence

Figure 1.1: Limited Robustness of Neural Network under Adversarial Attack. Image to the left is the original input, which will be classified correctly as a "panda" with 57.7% confidence. However, with very little additional noise added along the adversarial gradient direction (plotted in the middle image), the perturbed image (plotted in the right) will be classified into "gibbon" with 99.3% confidence. However, to human beings, the resultant adversarial image is not visually different to the original image. Source: Goodfellow *et al.* (2014b)

Another limitation for neural networks is their generalizability can be limited. It is known that standard neural networks have difficulties at generalization under input distribution shifts (as shown in Fig. 1.2a). For example, neural networks can be difficult to recognize novel object sets correctly (Stringer and Rolls, 2002). Furthermore, images composed of highly structured patterns outside the data distribution have be manufactured to fool the networks (Nguyen *et al.*, 2015). These findings raise the concern that neural network based systems, autonomous driving vehicles for example, does not perform well in new environment. The problem for generalizability can also occur when the training set is small (as shown in Fig. 1.2b). Typically, in order to generalize well, (a large amount of) data samples are required for neural networks to fully cover the data distribution. Huge computer vision datasets have been created to this end: MNIST dataset (established in 1998) contains 70K handwritten digits (Deng, 2012), ImageNet (established in 2010) consists of more than 14 million images in 21k different classes (Deng *et al.*, 2009). ModelNet (established in 2015) consists of more than 127K CAD models from more than 600 different classes (Wu *et al.*, 2015). However, it can be very expensive either experimentally or computationally to acquire dataset at large scale for engineering applications such as in the design of drug, molecule, and material (Cang *et al.*, 2018a).

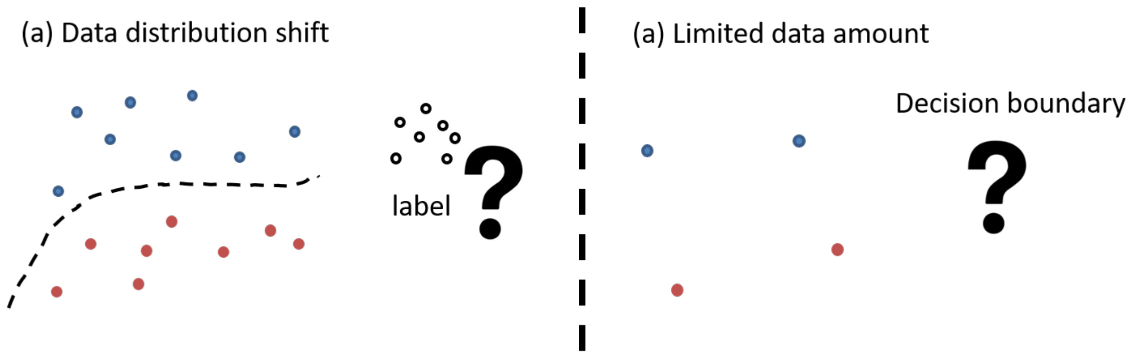


Figure 1.2: Limited generalizability of Neural Networks under (a) data distribution shift, and (b) limited data amount.

1.2 Research Tasks

To this end, we will explore three approaches to address the limitations of neural networks in robustness and generalization. These approaches are listed bellow.

Task1: Build a Classifier based on Generative Model to handle distribution shift. In this task, we design a classifier based on generative model for computer vision related applications. Traditional feedforward neural network (for computer vision) takes in an image, and output its predicted label after several convolutional and fully connected layers. This process is essentially a bottom-up signal processing approach. The concerns over the limited generalizability of neural networks have led

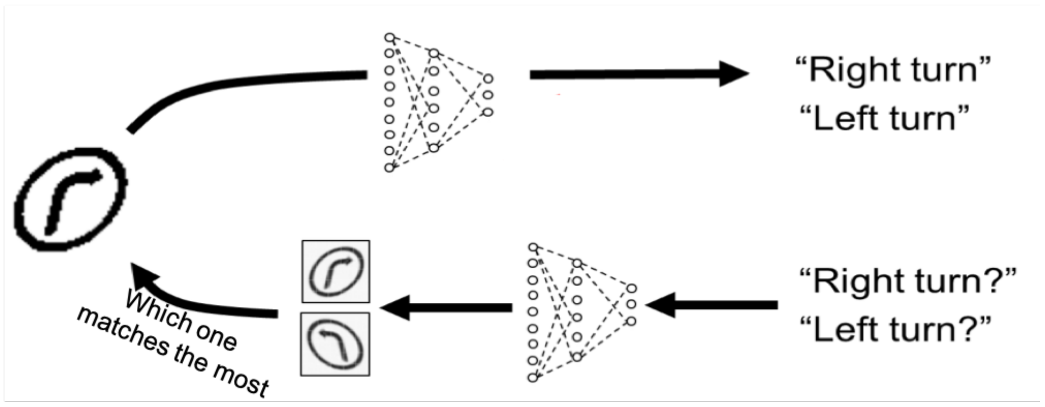


Figure 1.3: Illustration of the Difference between Proposed Generative Classifier and Traditional Feed Forward Classifier. Upper part illustrates the traditional feed-forward classifier, which is bottom-up signal process. Bottom part illustrates proposed generative classifier, which is top-down and involves inference.

to the question of whether semantic attributes or physical components of the input are truly understood by feedforward, albeit deep, networks. Furthermore, recent work has been shown that human vision follows a top-down fashion, and can perform reasoning better (George *et al.*, 2017). To this end, we investigate the possibility of using a generative model to perform classification though a top-down reasoning approach, as illustrated in Fig. 1.3. Specifically, we propose a conditional Variational Auto-Encoder (c-VAE) that learns both the decomposition of inputs and the distributions of the resulting components during training. During testing, the latent variables and input label of the generator are jointly optimized to find the best match between the given input and the output of the generator. The optimized label will be regarded as the network prediction. This is consistent with human reasoning, as when we see something we are not very familiar with, we will think hard in our brain if we have seen it somewhere or not. We will show that this top-down reasoning approach is able to handle data distribution shift better.

Task2: Defend adversarial attacks through transformation-invariant Adversarial training. In this task, we propose adversarial training with transformation invariant attack to defend adversarial attacks and improve model robustness. Existing work has shown that adversarial attacks can also be vulnerable under natural input transformations, and random transformations has been used to preprocessing the input images to improve model robustness (Guo *et al.*, 2017). It was later shown, however, this approach creates a gradient obfuscation effect and can be broken by transformation-invariant (robust) attacks (Athalye *et al.*, 2018). The feasibility of transformation-invariant (robust) attacks in real-world has been demonstrated in (Athalye and Sutskever, 2017). On the other hand, experiments show that adversarial training with an ensemble of different models can be used to effectively avoid gradient obfuscation (Tramèr *et al.*, 2017). Furthermore, independent work has shown that human vision, which is robust to adversaries, is invariant to mild natural input transformations (Ullman *et al.*, 2016).

As illustrate in Fig. 1.4, different background color represents the data distribution of different classes, and the boundary represents the optimum decision boundary. Transformation augmentation constraining the classifier with transformations of inputs, by assuming that transformations preserve labels. Adversarial training constraining the classifier with ϵ-balls around data points, by assuming that labels are preserved within these balls (Madry *et al.*, 2017); Our approach is constraining the classifier with both transformations and the transformed ϵ-balls.

Task3: Design data-driven models based on physics prior knowledge for better generalization with small data. In this task, we propose a method to incorporate the prior knowledge in physics to guide the designing of neural network architectures and predict the physics response when training data is limited. Since

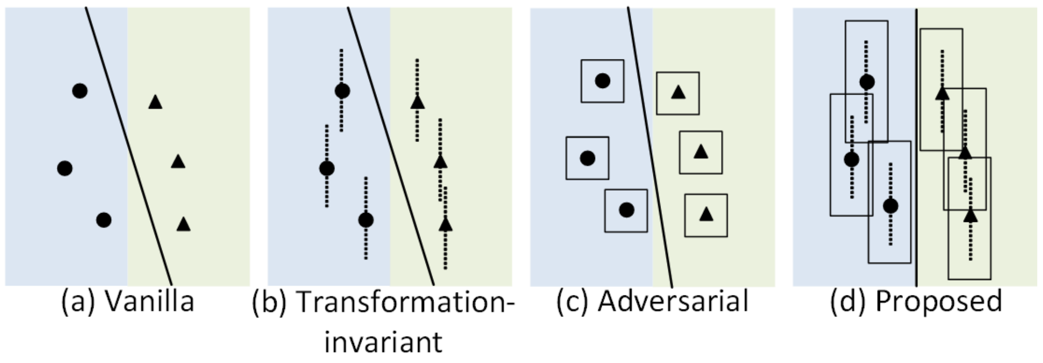


Figure 1.4: Illustration of the Difference between different Training Methods: (a) Vanilla, (b) transformation-invariant: constraining the classifier with transformations of inputs, by assuming that transformations preserve labels, (c) adversarial training: constraining the classifier with ϵ-balls around data points, by assuming that labels are preserved within these balls; (d) proposed: constraining the classifier with both transformations and the transformed ϵ-balls.

only limited information can be provided from the data side, extra information from other sources would be necessary to address such problem. As depicted in Fig. 1.5, our learning approach that takes the advantage of both physics understanding (in term of physics relations) and new developments in data science. Unlike traditional neural networks, proposed FEA-Net is learning framework specially designed to have physics prior knowledge embedded in the network architecture. We first propose FEA convolution to model the governing Partial Differential Equations (PDEs) in mechanical analysis, which links the Finite Element Analysis (FEA) formulation and its solution algorithms to the underlying mathematical representation of the convolutional neural network. Secondly, we will turn the fix-point iterative solver into the form of a convolutional neural network. Compared with purely physics based approach, our method is more memory efficient. And compared with purely data-driven method, our method is more data efficient.

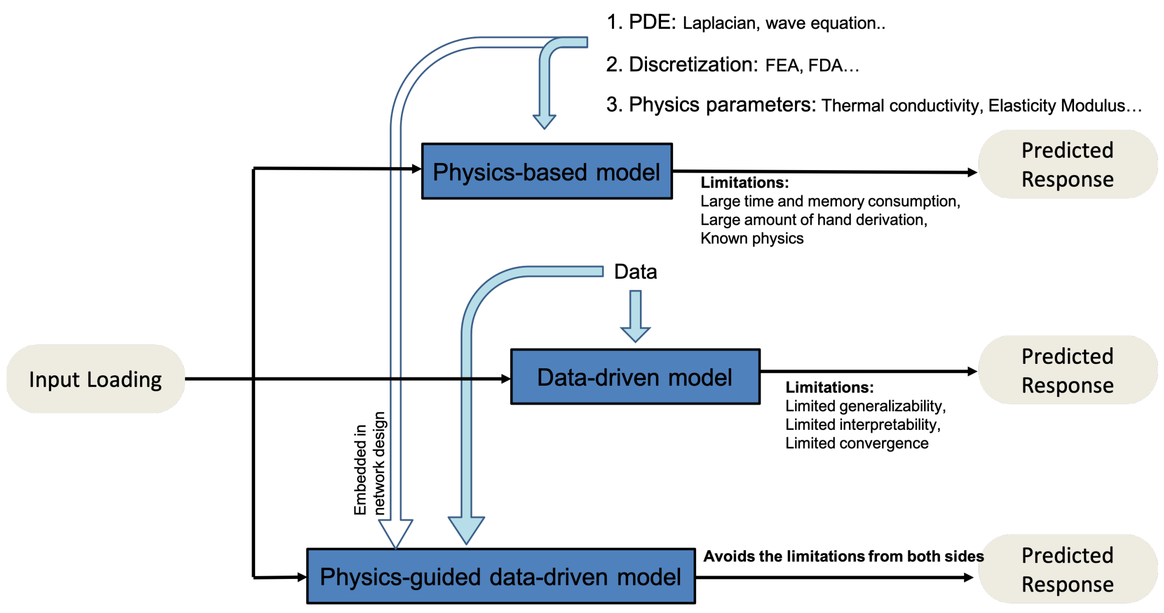

1. PDE: Laplacian, wave equation..
2. Discretization: FEA, FDA...
3. Physics parameters: Thermal conductivity, Elasticity Modulus...
Physics-based model
Limitations:
Large time and memory consumption,
Large amount of hand derivation,
Known physics
Predicted Response
Data
Input Loading
Data-driven model
Limitations:
Limited generalizability,
Limited interpretability,
Limited convergence
Predicted Response
Embedded in network design
Physics-guided data-driven model
Avoids the limitations from both sides
Predicted Response

Figure 1.5: Illustration of Proposed Physics-guided Data-driven Learning.

1.3 Outline of the Dissertation

The rest of this dissertation is arranged as follows: The second chapter introduces the technical background and related works on machine learning, generative models, adversarial attack and defense, and machine learning for physics. In the third chapter, we propose a novel classification algorithm based on generative models that can classify objects outside of training distribution successfully. In the fourth chapter, we propose transformation-invariant adversarial training to address the limitation in robustness. In the fifth chapter, we focused on designing a network architecture based on physics prior knowledge that has good generalizibility in predicting physics response with limited training data. We conclude the dissertation in the sixth chapter.

Chapter 2

TECHNICAL BACKGROUND

2.1 Machine Learning

2.1.1 Neural networks

The very building block of neural network is called perceptron, which is inspired by neuron cells in our brain. Neuron cells receive electric impulses through dentrites, and these impulses will be combined and processed before sending out to other cells. Likewise, as depicted in Fig.2.1, a perceptron takes in some values X, weight them differently using W, apply a non-linear function f, and then output a new value Y.

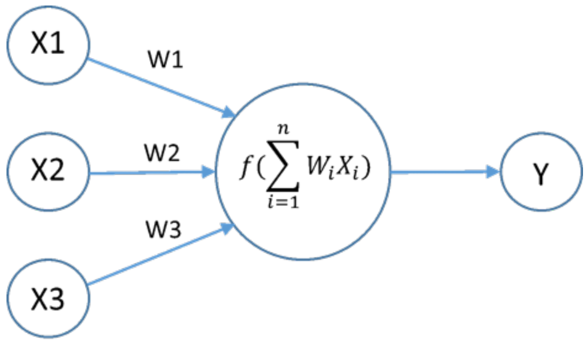


Figure 2.1: Illustration of A Single Perceptron.

The training of a perceptron is often formulated as an optimization problem, which is, finding the best weights W that minimize the difference between the true label and the label predicted by perceptron:

$$W^* = \arg\max_{W} \left(f(W \cdot x) - y\right) \tag{2.1}$$

where $\cdot$ denotes matrix vector production.

Similar to our brain, where neuron cells are interconnected with each other, perceptrons can also be grouped together and stacked upon each other to form a larger and deeper structure (see Fig.2.2 for example). This structure is called Multi-Layer Perceptron (MLP), which is also referred to as Neural Network (NN). Based on the universal approximation theorem, given enough nodes, a two layered neural network is capable of approximating any functions (Hornik, 1991a).

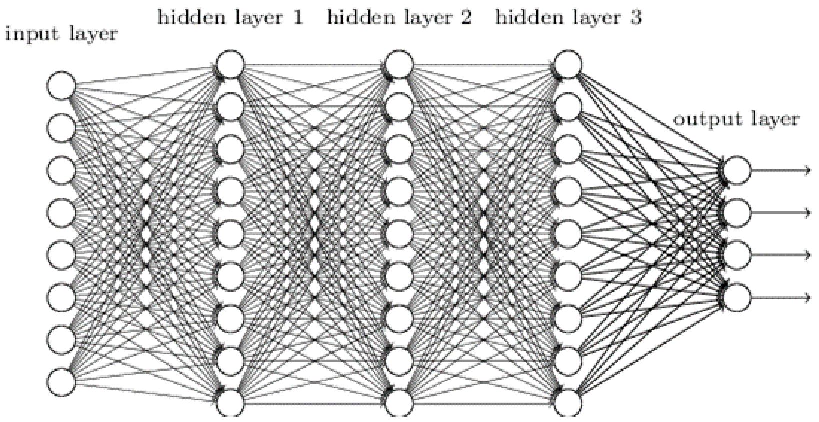


Figure 2.2: Example of A 4-layer MLP. It takes 8 variables as the input, and has an output of 4 variables. (Figure adopted from Internet)

Training of a neural network can also be formulated into an optimization problem similar to perceptron:

$$\begin{aligned} x^{i+1} &= f(W^i \cdot x^i), \text{for} i = 1, 2, ..., k \\ W^* &= \arg\max_W L(x^k, y) \end{aligned} \tag{2.2}$$

where $W = \{W^1, W^2, ..., W^k\}$, k is the total number of layers, and W^i is the weight on each layer. L is a pre-defined loss function. Because the nested nature of neural networks, Back-propagation, as a special case of chain rule for computing derivation, has been proposed to solve Eq.2.2 efficiently (Rumelhart *et al.*, 1985).

For more efficient learning of vision related problems, LeCun *et al.* (1989) proposed Convolutional Neural Networks (CNN) by replacing some fully connected layers with

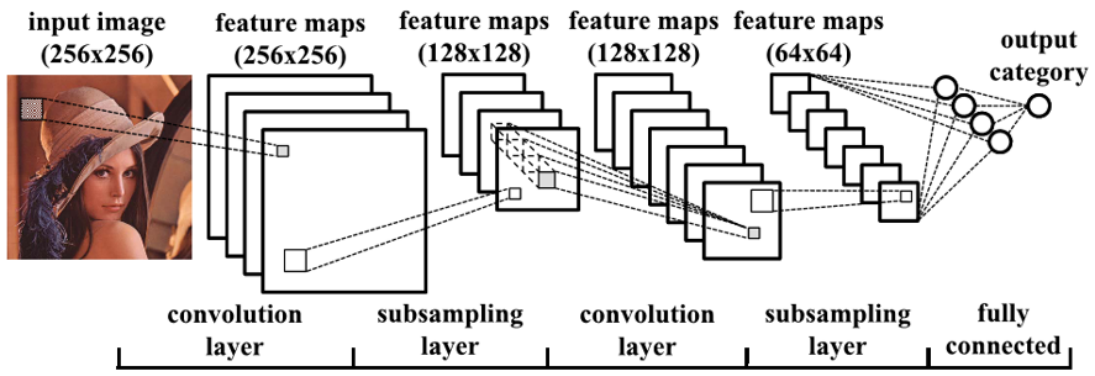


Figure 2.3: Example of 2D convolutional neural network. (Figure adapted from Internet.)

convolution. As shown in Fig.2.3, convolutions are done on the input image, followed by down-sampling layers to reduce the resolution and thus the computation cost. Finally, the feature maps are vectorized and feed to fully connected layers to obtain the final output. Because CNN does not vectorize the original input image, the spatial relationship between pixels in the input matrix can be preserved. Secondly, since the weight matrix is shared across different locations, the number of unknown parameters in the network can be efficiently reduced. CNN has shown to be very suitable for various vision-based tasks like object recognition (LeCun *et al.*, 1989; Krizhevsky *et al.*, 2012), detection (Girshick *et al.*, 2014; Ren *et al.*, 2015), generation (Radford *et al.*, 2015; Isola *et al.*, 2017), and segmentation (Long *et al.*, 2015; Ronneberger *et al.*, 2015).

Various optimization techniques have been specially designed to solve the optimization problem Eq.2.2 and train neural network better (Duchi *et al.*, 2011; Zeiler, 2012; Kingma and Ba, 2014). One of the most advanced optimizers is the Adaptive Moment Estimation (Adam) (Kingma and Ba, 2014), which is an optimization method that adaptively computes the learning rates for each parameter. It estimates the first and second order moment (mean and variance) of the gradients respectively:

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t \\ \mu_t &= \beta_2 \mu_{t-1} + (1-\beta_2) g_t^2 \end{aligned} \tag{2.3}$$

The updating rule for the variables of our interest is:

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\hat{\rho}_t} + \epsilon} \hat{m}_t \tag{2.4}$$

where $\hat{m}_t$ and $\hat{\rho}_t$ are the estimated moment after correction:

$$\begin{aligned} \hat{m}_t &= \frac{m_t}{1-\beta_1^t} \\ \hat{\mu}_t &= \frac{\mu_t}{1-\beta_2^t} \end{aligned} \tag{2.5}$$

The default value for β_1, β_2, and ϵ are set to 0.9 , 0.999, and 10^{-8}, respectively. It has been shown empirically that Adam works very well in many cases and compares favorably to other optimizers.

2.1.2 Deep generative models

A generative model is a model that generates a probabilistic distribution approximately matches with the data distribution. Notable generative models in the field of comptuer vision include Restricted Boltzmann Machine (RBM) (Hinton and Salakhutdinov, 2006; Lee *et al.*, 2009), Variational Autoencoder (VAE) (Kingma and Welling, 2013), PixelCNN (van den Oord *et al.*, 2016a), Generative Adversarial Network (Goodfellow *et al.*, 2014a), among many others.

VAE approximates the data distribution by maximizing the variational lower bound of the data log-likelihood (Kingma and Welling, 2013), and requires both the prior and the output to be modeled as parametric families of distributions. Novel extensions to VAE include DRAW (Gregor *et al.*, 2015), which generates images by incorporating a recurrent attention mechanism, conditional Variational Autoencoder (c-VAE), which performs segmentation and classification by incorporating image label into the model (Sohn *et al.*, 2015), and disentangling CVAE (disVAE) (Yan *et al.*, 2016), which uses multiple networks to separate image foreground from background.

Another type of generative model is designed to model the data distributions directly without any approximation. One notable example is PixelCNN (van den Oord *et al.*, 2016a), which models each pixel as conditioned on all other observed pixels. The major shortage of these models is their high computational cost. Various attempts, e.g., gates (van den Oord *et al.*, 2016b), ResNet-style structures (Salimans *et al.*, 2017), and long-range spatial independency (Reed *et al.*, 2017) have been made to improve the computational efficiency of PixelCNN.

Instead of explicitly defining the density function, Generative Adversarial Network (GAN) (Goodfellow *et al.*, 2014a; Radford *et al.*, 2015) and its many variants (e.g., (Chen *et al.*, 2016a; Dumoulin *et al.*, 2016; Zhu *et al.*, 2017; Reed *et al.*, 2016; Zhang *et al.*, 2016a)) inherently does so by finding the equilibrium of a min-max game between a generator and a discriminator. The difficulty in finding a good Nash equilibrium for GAN has led to many proposals of enhancements (Salimans *et al.*, 2016; Tolstikhin *et al.*, 2017; Berthelot *et al.*, 2017; Durugkar *et al.*, 2016; Zhang *et al.*, 2016b; For and Tools, 2018), in particular for preventing mode collapse (Zhao *et al.*, 2016; Metz *et al.*, 2016; Arjovsky *et al.*, 2017; Ghosh *et al.*, 2017).

In addition, hybrid models have been proposed to achieve better generation qualities or specific goals: PixelVAE (Gulrajani *et al.*, 2016) combines VAE and PixelCNN to capture small details while uses fewer autoregression layers than PixelCNN. VAE-GAN (Larsen *et al.*, 2015) combines VAE with GAN to obtain a feature-wise error measurement provided by the discriminator, resulting in generations with higher visual fidelity. Adversarial Autoencoder (AAE) (Makhzani *et al.*, 2015; Makhzani and Frey, 2017) replaces the KL divergence of autoencoder with GAN loss, so that it can handle discrete latent distribution.

2.2 Adversarial Attacks and Defense

It has been shown that neural networks can be very sensitive to input perturbations, and even a very slight difference in the input can cause the prediction to be different. This intriguing property has raised many interests on both attacking and defending side.

2.2.1 Attack

Given $(x, y) \in \mathcal{D}$ as an image-label pair from a dataset $\mathcal{D}$, a classifier $f(\cdot, \theta) : \mathbb{R}^d \to [0, 1]^k$ with parameters θ maps input images to the softmax outputs, and a loss function $L(\cdot, \cdot) : [0, 1]^k \times [0, 1]^k \to \mathbb{R}$, the untargeted attack can be formulated as the problem of finding:

$$x^{adv} = \underset{x' \in \mathcal{N}_\epsilon(x)}{\arg\max}\, L(y, f(x', \theta)) \tag{2.6}$$

where $\mathcal{N}_\epsilon(x)$ is a L_p ball around x with radius ϵ. Different distance definition and different ways to solve the optimization problem give birth to a wide variety of adversarial attack methods. Attacks with $p = \infty$ (Szegedy *et al.*, 2013; Kurakin *et al.*, 2016b,c; Madry *et al.*, 2017), $p = 2$ (Carlini and Wagner, 2017; Moosavi-Dezfooli *et al.*, 2016), $p = 1$ (Chen *et al.*, 2017), and $p = 0$ (Su *et al.*, 2017) have been proposed.

The first adversarial attack on convolutional neural networks is Fast Gradient Signed Method (FGSM) (Szegedy *et al.*, 2013). Based on L_∞ distance measurement, FGSM simplifies the optimization into a single step gradient descent:

$$x^{adv} = x + \epsilon \cdot \text{sign}\nabla_x L(y, x, \theta) \tag{2.7}$$

Since only a single gradient computation is involved in FGSM attack, it can be very computationally efficient. In the meantime, FGSM is effective enough to destroy vanilla neural networks without any defense mechanisms.

Based on FGSM, many other L_∞ based attacks has been developed. For example, instead of taking a single step towards the gradient direction, Basic Iterative Method (BIM) is proposed to perform adversarial attack by performing gradient ascent. And Kurakin *et al.* (2016b,c) further adds noise to the initial image in performing gradient computation. Furthermore, Madry *et al.* (2017) utilized Projected Gradient Descent with random perturbation to solver the optimization problem, which has shown to be extremely successful in attacking neural networks. Instead of explicit gradient computation, Uesato *et al.* (2018) adopted Simultaneous Perturbation Stochastic Approximation (SPSA) (Spall *et al.*, 1992) to approximate the adversarial gradient and is able to attack models with gradient obfuscation (Athalye *et al.*, 2018).

Lots of adversarial attacks based on L_2 distance have been proposed as well, such as C&W (Carlini and Wagner, 2017) and DeepFool (Moosavi-Dezfooli *et al.*, 2016). And Elastic Net loss has also been adopted in Chen *et al.* (2017) for even better attack performance. Furthermore, the extreme L_0 based attack has been proposed, and solved with evolutionary optimizer (Su *et al.*, 2017). It is shown that, with adversarial perturbation on as few as a single pixel can be sufficient to fool the network.

What's more, it has been shown that adversarial attacks are transferable among models with different architectures and even learning algorithms (Papernot *et al.*, 2016a), suggesting that a surrogate model can be used to perform adversarial attacks on black-box models.

2.2.2 Defense

Various attempts have been made to defend adversarial attacks. Some of the most popular approaches are: (1) To suppress the adversarial perturbation of the input.

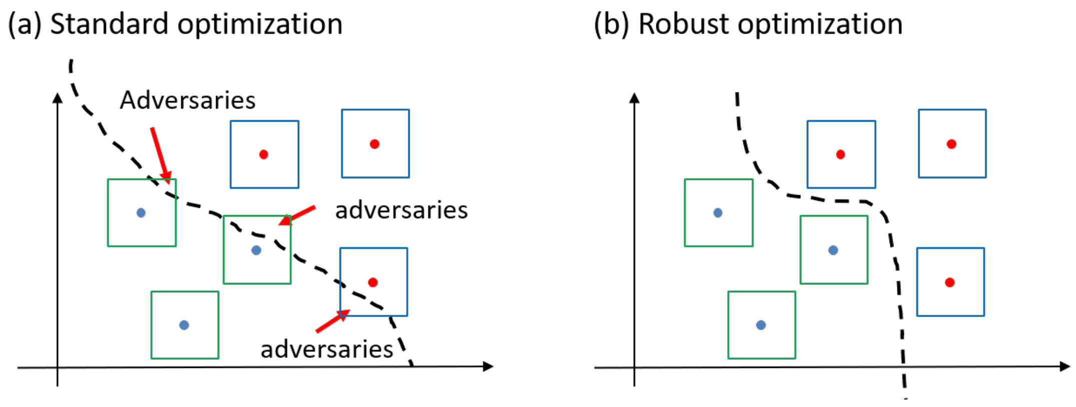


Figure 2.4: From Standard Optimization to Robust Optimization. Box denotes the L_∞ cube with length ϵ around the data point.

Examples include feature squeezing (Xu *et al.*, 2017a), JPEG compression (Das *et al.*, 2018), denoising autoencoder (Gu and Rigazio, 2014) and its variants (Liao *et al.*, 2018). (2) To hide useful gradient information from the attacker. Examples include thermometer encoding (Roy *et al.*, 2018), defense distillation (Papernot *et al.*, 2016c), and random input transformation (Guo *et al.*, 2017). (3) Adversarial training or robust optimization (Kurakin *et al.*, 2016b; Tramèr *et al.*, 2017; Madry *et al.*, 2017), which formulates a minimax problem which targets the worst case performance of the model under adversarial attack as illustrated in Fig. 2.4.

Purposefully or not, many existing defense methods results in gradient obfuscation and their robustness is not reliable (Athalye *et al.*, 2018). As a result, these models can be either vulnerable to black-box attacks (Papernot *et al.*, 2016a; Tramèr *et al.*, 2017) or easily defeated by tailored white-box attacks (Athalye *et al.*, 2018). Robust optimization, or adversarial training, remains to be one of the few defense mechanisms that do not suffer from gradient obfuscation. Adversarial training can be formulated as the following minimax problem:

$$\min_{\theta} \mathop{\mathbb{E}}_{x,y\sim\mathcal{D}} \left[\max_{x' \in \mathcal{N}_\epsilon(x)} L(y, f(x', \theta)) \right] \tag{2.8}$$

Recent work has shown that this minimax problem can be decoupled and solved sequentially by first performing adversarial attack on the model then train the model with adversaries (Kurakin *et al.*, 2016b; Madry *et al.*, 2017). Current best defense method is PGD adversarial training (Madry *et al.*, 2017) and TRADES (Zhang *et al.*, 2019b), which all comes from this formulation.

2.3 Physics Based Learning

People also started to explore the possibility to use deep neural networks to predict the physics to solve related engineering problems. Some of the seminal work has investigated the application of deep learning in thermal (Sheikholeslami *et al.*, 2019) and fluid (Tompson *et al.*, 2016; Chu and Thuerey, 2017) simulation, structure analysis (Wang *et al.*, 2019; Finol *et al.*, 2018) and optimization (Sosnovik and Oseledets, 2017b; Cang *et al.*, 2018b), material property prediction (Bouman *et al.*, 2013; Li *et al.*, 2019) and design (Bessa *et al.*, 2017; Cang *et al.*, 2018a), system monitoring (Zhao *et al.*, 2019) and calibration (Yao *et al.*, 2019). Despite all these progresses in deep learning; however, almost all neural networks suffer from one or many of the following problems: (1) The model is not quite interpretable, which makes it difficult to interpret what we have learned exactly and further verify the correctness of the prediction. (2) The generalizability is limited, which means it is hard to predict the system response when the training data is limited. (3) There is no guaranteed convergence for traditional deep neural networks, which makes the network design difficult and lots of trials would be needed.

Recent attempts have also been made to design more efficient data-driven models for different physics problems, and several seminal works have been done to build hybrid learning mechanisms with physics knowledge. The performance of FEA has been enhanced by learning better integration rule (Oishi and Yagawa, 2017), or element information (Capuano and Rimoli, 2019). These works use neural network as a module under the FEA framework. Parallel works have been done to improve the convergence and accuracy of finite difference analysis (FDA) solvers to initial value problems (IVP) through learning the optimum filters (Hsieh *et al.*, 2018), or by building a hybrid model with ODE information hard-coded (Yu *et al.*, 2018). Furthermore, it has been observed that there exist some similarities between different FDA solvers and some neural network structures (Lu *et al.*, 2017b). Based on this finding, (Long *et al.*, 2018) proposed PDE-Net based on finite difference scheme and reported promising result in system identification.

Chapter 3

ROBUST CLASSIFICATION THROUGH GENERATION

3.1 Introduction

We demonstrate that a generative model can be designed to perform classification tasks under challenging settings, including input distribution shifts, overlapping objects, and adversarial attacks. Specifically, we propose a conditional variational auto-encoder that learns both the decomposition of inputs and the distributions of the resulting components. During test, we jointly optimize the latent variables of the generator and the relaxed component labels to find the best match between the given input and the output of the generator. The model demonstrates promising performance at recognizing novel component combinations from a traffic sign dataset, and overlapping components from the multiMNIST dataset. Experiments also show that the proposed model achieves high robustness on MNIST and NORB datasets, in particular for high-strength gradient attacks and non-gradient attacks.

Neural network architectures have been developed to achieve human-level performance on standard vision tasks (He *et al.*, 2015; Szegedy *et al.*, 2016). However, it is acknowledged that feedforward networks have a difficulty at generalization under input distribution shifts, e.g., novel object sets (Stringer and Rolls, 2002) and objects with overlaps (Sabour *et al.*, 2017). Neuro-Evolution of Augmenting Topologies (NEAT) (Stanley and Miikkulainen, 2002), which evolves an image defined by a compositional pattern producing network (CPPN) (Stanley, 2007) to maximize a classification target. Furthermore, studies have shown that networks, even with high standard test accuracy, can suffer from imperceptible adversarial attacks (Goodfellow *et al.*, 2014b). While neither distribution shifting or adversarial attacks are common cases in standard test environments for image classifiers (Papernot *et al.*, 2016b; Lu *et al.*, 2017a), the demonstrated risk of existing models have raised concerns over their real-world applications (e.g., autonomous driving and security surveillance) where failed classification for a short amount of time can be catastrophic. Indeed, evidence showed that state-of-the-art classification models have a drastically different accuracy changing pattern than human beings in classifying image sequences with diminishing details (Ullman *et al.*, 2016), suggesting that the two have different feature learning behaviors.

This concern over model generalizability and robustness is intrinsic to classifiers that perform bottom-up signal processing. Alternative models that integrate bottom-up processing with top-down reasoning through recursive inference have been studied (Chen *et al.*, 2008; George *et al.*, 2017). Of particular interest are recursive compositional models (e.g., AND-OR templates (George *et al.*, 2017)) that learn to match deformable objects and infer graph states for detection and recognition. These models take advantage of highly structured generators, e.g., by explicitly modeling object edges and surfaces. However, the intrinsic trade-off between model and computational complexity (for both model learning and recognition/classification) may hamper their application to general inputs for which the recognition or classification tasks depend on a richer set of features. With the rapid advance in generative models (e.g. GAN (Brock *et al.*, 2018), PixelCNN (Salimans *et al.*, 2017), and VAE (Kingma and Welling, 2013; Yan *et al.*, 2016)), it is tempting to investigate top-down classification mechanisms that incorporate more flexible generators than compositional models.

To this end, we present in this paper a classification algorithm where input images are classified by minimizing the difference between the target image and the output of a customized variational auto-encoder. Following the argument that attribute recognition is key to object classification (Farhadi *et al.*, 2009; Lampert *et al.*, 2009), we focus on the classification of attributes, or more specifically, the existence of object components from the input image, while assuming that the follow-up mapping from these components to the object class is established. As shown in Fig. 3.1, the model is built upon a variational auto-encoder whose decoder is composed of separate sub-networks, where each attribute is associated with a sub-network. The model is trained on a set of labeled images, e.g., traffic signs along with a binary encoding of their symbolic components, to decompose and reconstruct image components corresponding to the attributes through the sub-networks. When images have individual labels, such as MNIST, our model is reduced to separate generative models trained for each label. After the conditional generative model has been pre-trained, we jointly search the latent variables and the label to find a generation that is similar to the input the most. The label value after optimization can be used as the classification result. Among all generative models, variational auto-encoder is chosen as an example to perform classification.

The rest of this chapter is structured as follows: We first elaborate on the proposed generative model, its loss design, training settings, and the classification algorithm in Sec. 4.2. We verify the model in Sec. 3.3 through benchmark studies on MNIST, smallNORB, and a customized traffic sign dataset. Sec. 3.3.3 discusses the connection of this chapter to existing methods and future directions.

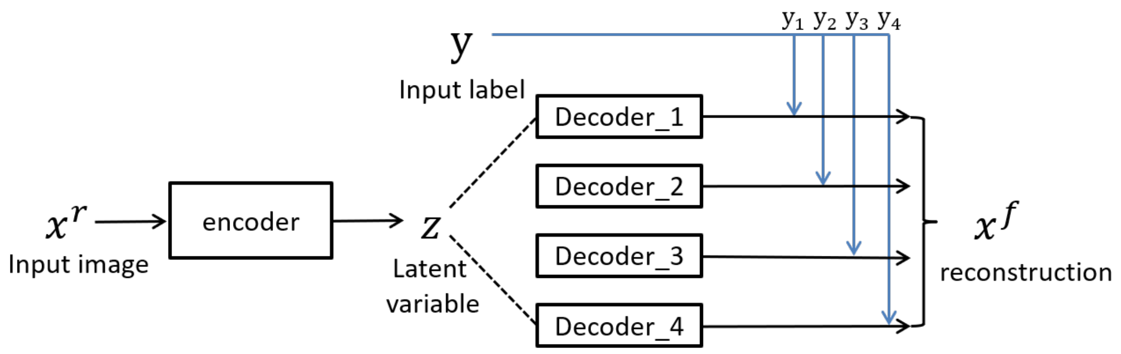


Figure 3.1: Schematic of the Proposed Generative Model with Sub-networks Each Generating Images Corresponding to an Image Component. Here, as an example, we assume that there are a total of four components.

3.2 Classification through Generation

3.2.1 Generation

In this chapter, we customize a variational auto-encoder (VAE) to have multiple sub-networks as its decoder (as shown in Fig. 3.1). VAE is chosen among all these different generative models because of its simplicity and good convergence property.

Let n be the number of sub-networks, y be the image attributes (i.e., the binary encoding of image components), x^r and x^f be the input and output images of the network, respectively, and z be the latent variables. We have $z = Enc(x^r, \theta) \in \mathbb{R}^p$ as the encoder, and $x^f(z, y, \phi) = \sum_i^n y_i Dec_i(z_i, \phi_i)$ as the decoder, where $z_i = Enc_i(x^r, \theta) \in \mathbb{R}^{p/n}$ is the ith segment of z. θ and $\phi = \{\phi_i\}_{i=1}^n$ are weights for the encoder and decoders, respectively. We note that a higher dimensional latent space improves reconstruction performance, yet tends to cause mode collapse across sub-networks, and vice versa.

Since the classification performance is sensitive to both the reconstruction quality (as the classification decision is based on pixel matching) and the ability of image decomposition achieved through the sub-networks (so that novel component combinations can be correctly classified), a balance between the two performance metrics is required. These considerations lead to the following training loss, which will be minimized through stochastic gradient descent:

$$\begin{aligned} l(\theta, \phi | D) &= \sum_{\{x^r, y\} \in D} \left(||x^f(z, y, \phi) - x^r||_2^2 + C \cdot y^T KL(z) \right) \\ z &= Enc(x^r, \theta, \epsilon) \end{aligned} \tag{3.1}$$

where $KL(z)$ is the sub-network-wise KL divergence of the distribution of z.

3.2.2 Classification

We consider classification as an inverse problem of jointly finding z^* and y^* that minimize the difference between the generated image x^f and a target x^r in the image space:

$$\begin{aligned} z^*, y^* &= \text{argmin}_{z \in \mathbb{R}^p, y \in \{0,1\}^n} f(z, y) \\ f(z, y) &\equiv \|x^f(z, y, \phi) - x^r\|_2, \end{aligned} \tag{3.2}$$

Note that Eq. (3.2) is a combinatorial problem, as an enumeration of the binary attributes is needed. To reduce the computational complexity, we relax each attribute to be continuous within $[0, 1]$, so that the optimization problem differentiable and can be solved efficiently through gradient descent and backpropagation. This process is illustrated in Fig. 3.2. Furthermore, due to the non-convexity of the problem, we use the encoded z from x^r as the initial guess for faster search. y is initialized as a vector of 0.5.

In addition, we observed that an object component absent in the input image can be falsely introduced to the optimized x^f to account for the adversarial noise added to the image. To address this issue, we introduced a L_1 penalty to regularize non-zero attributes with weight c. The intuition is that a component should be discarded (i.e., sub-network outputs suppressed) if its addition to the generated image does not improve the reconstruction quality significantly. We also found that minimizing the

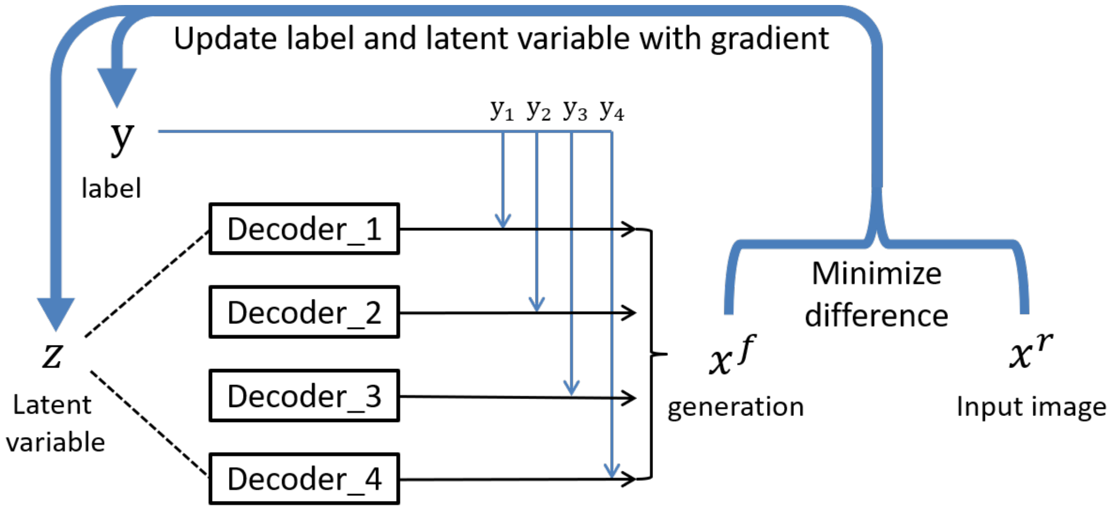


Figure 3.2: Schematic of the Classification through the Pre-trained Generative Model. The direction of the arrow corresponds to the direction of gradient flow during testing.

image-wise difference after a sigmoid transformation d further improves the defense performance. With these modifications, the classifier solves the following problem:

$$z^*, y^* = \text{argmin}_{z \in \mathbb{R}^p, y \in [0,1]^n} f(z, y) \equiv \|d(x^f(z, y, \phi)) - d(x^r)\|_2 + c\|y\|_1, \tag{3.3}$$

where $d(x) = \text{sigmoid}\big(\beta \cdot x + b\big)$.

Upon convergence, z^* and y^* obtained via Eq. (3.3) will be passed through Algorithm 1 to derive the classification result: First, if the lowest reconstruction loss found $f(z^*, y^*)$ is larger than a pre-set threshold l, the input image will be classified as noise. This step filters out adversarial attacks using non-gradient methods. If the image passes this filter, we apply two thresholds $y_l < y_u$ to y^*: an element y_i^* is set to 0 if $y_i^* < y_l$, and 1 if $y_i^* > y_u$. For the remaining elements of y^* between y_l and y_u, we will enumerate over all binary combinations of this subset to generate a candidate set $\mathcal{Y}$. We then compute $z^\dagger(y) = \text{argmin}_{z \in \mathbb{R}^p} f(z, y)$ for $y \in \mathcal{Y}$ and set the classification result as $y^\dagger = \text{argmin}_{y \in \mathcal{Y}} f(z^\dagger(y), y)$. The hyperparameters ($l$, y_l, y_u, β, b, and c) can be tuned using a small validation set of adversarial images through gradient and non-gradient attacks.

```
Algorithm 1: The proposed classification algorithm
   input      : Images $x$ and the trained generative model
   output     : Predicted label $y$
   parameter: $(l, y_l, y_u, \beta, b, c)$
 1 Load a pre-trained generative model;
 2 Pre-processed input image $\hat{x} = d(x_r)$;
 3 Encode the pre-processed image $\hat{z} = Enc(\hat{x})$;
 4 Obtain $z^*, y^*$ and the minimum objective function value $f(z^*, y^*)$ from
     Eq. (3.3) with the initial $\hat{z}$ and $y_i = 0.5$
 5 if $f(z^*, y^*) < l$ then
 6     Set $y_i = 0$ for all $y_i^* < y_l$ and $y_i = 1$ for all $y_i^* > y_u$;
 7     Set $\mathcal{I} = \{i | y_i^* \in [y_l, y_u]\}$;
 8     Enumerate over $y_i \in \{0, 1\}$ for $i \in \mathcal{I}$ to generate a candidate set $\mathcal{Y}$;
 9     Compute $z^\dagger(y) = \text{argmin}_{z \in \mathbb{R}^p} f(z, y)$ for $y \in \mathcal{Y}$;
10     Find $y^\dagger = \text{argmin}_{y \in \mathcal{Y}} f(z^\dagger(y), y)$;
11 else
12     Return "Image does not belong to any known class."
13 end
```

3.3 Experiments and Results

In this section, we demonstrate the efficacy of our method through several experiments. First, we test the perfromance of our algorithm on three datasets (MNIST, smallNORB, and traffic signs) along with the binarization defense (Xu *et al.*, 2017b) under the simplest FGSM attacks with increasing perturbation levels. Secondly, we ask the network to perform classification with distribution shift in testing data, in which cases the benchmarks completely fails. Examples on the traffic sign data and CPPN MNIST samples were given to preliminarily validate of the capability of proposed algorithm to handle data distribution shift.

3.3.1 Implementation details

Datasets: The proposed model will be tested on MNIST, NORB (LeCun *et al.*, 2004), multiMNIST (Sabour *et al.*, 2017), and a Traffic Sign (TS) dataset. For NORB, similar to Sabour *et al.* (2017), we downscale the image resolution to 48-by-48. The multiMNIST dataset is synthesized by stacking two MNIST images into an image of resolution 36-by-36, which will result in 80% overlap between two digits on average. Every image label in multiMNIST has 2 ones and 8 zeros. A visualization of the multiMNIST dataset can be found in Fig. 3.9. The TS dataset is prepared with affine transformation to mimic traffic signs at different view angles. There are four different types of images in the dataset: “left turn", “right turn", “no left turn", and “no right turn". These images are in a resolution of 64-by-64, with examples visualized in Fig. 3.8. There are four components for TS images to represent the components: circle, slash, left arrow, and right arrow. For example, a “No right turn" image has label $[1, 1, 0, 1]$ since it has circle, slash, and right arrow.

Adversaries: We prepared adversarial images on MNIST and smallNORB with FGSM attacks. Different noise levels (ϵ) were used to perform FGSM attacks on images that can be classified correctly by the discriminative classifier. Pixel values are bounded within $[0, 1]$ after adversarial perturbations. If an adversarial image is misclassified with a 90% confidence or greater, the attack is considered successful and this adversarial image is included in the adversarial dataset. For MNIST, we generated 500 adversarial images for perturbations with magnitudes $\epsilon/C = 0.1, 0.2, 0.3$, and 0.4, where ϵ is the L_∞ norm of the attack vector and C is the averaged maximum pixel value of the dataset ($C = 1$ for MNIST, MultiMNIST, and TS, and $C = 0.82$ for smallNORB). For smallNORB, we generate 1700 adversarial images under all these attack levels. For TS, we generated 600 FGSM images in total for $\epsilon = 0.06$ and 360 FGSM images for $\epsilon = 0.59$.

CPPN Samples: We evolve CPPNs using NEAT (Stanley and Miikkulainen, 2002) to create artificial CPPN images outside of the training data distribution. The fitness of the evolution is defined as $1 - ||\hat{y} - y_{target}||$, where $\hat{y}$ is the classification of the CPPN image by the discriminative model and y_{target} is the target class. Evolved images with fitness values over 0.9 are stored. Duplicated examples are removed. A population of 50 CPPNs is used during the evolution. The CPPN comprises tangent, hat, sine, inverse and rectified linear activation units. We generated 331 and 334 NEAT images for MNIST and TS respectively.

Generator: Different conditional VAEs are customized and trained for each dataset. The number of sub-networks is equal to the number of total components in the dataset, which is 10 for MNIST and multiMNIST, 5 for NORB, and 4 for TS. The latent dimensions are 4, 64, and 4 for MNIST, smallNORB, and TS. The number of sub-networks are 10, 5, and 3, correspondingly. Once the network has been trained, it is expected to decompose the input object into different components. To demonstrate the efficacy of the learning of decomposition, we visualize the latent space of each sub-networks trained on different datasets in Fig. 3.3. It can be seen that the decomposition has been done successfully as expected. One unexpected observation from the results is that the model is able to remove data redundancy automatically, which can be seen from the TS dataset: To recall, the first component of a TS image represents the existence of a circle. Since the circle exists in all TS images, the learned generative model combines the circle into the generation of the arrows and leaves the

Table 3.1: Hyperparameters Settings Used in the Experiments

	l	y_l	y_u	β	b	c
MNIST	0.20	n/a	n/a	5.0	-2.0	0.0001
smallNORB	n/a	n/a	n/a	5.0	-2.0	0.0001
TS	0.25	0.6	0.7	7	-5	0.001

first sub-network blank.

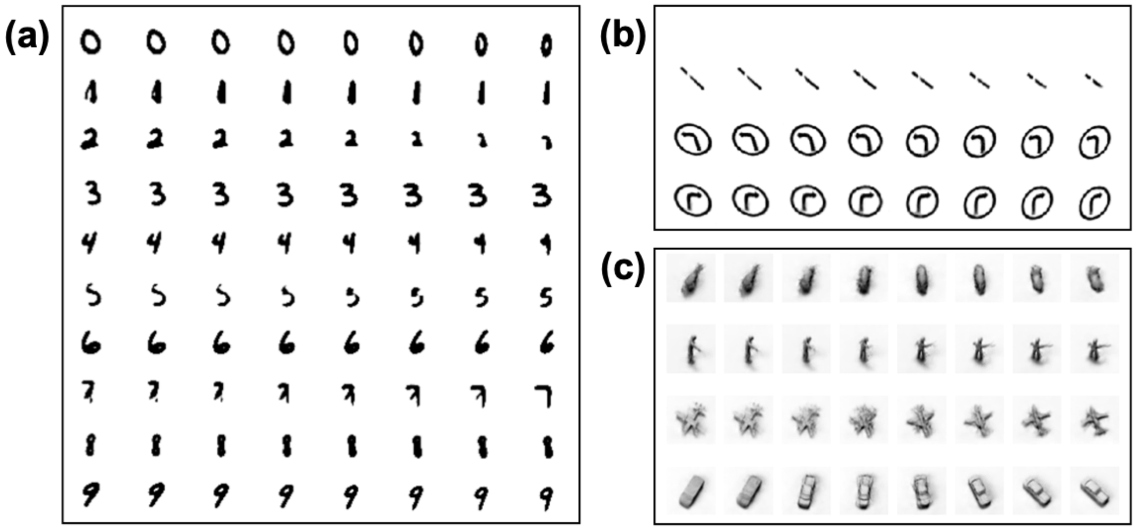


Figure 3.3: Sampling the latent space of each generator trained on (a) multiMNIST, (b) TS, and (c) NORB. Each sub-network is dedicated to one component. Samples of multiMNIST training images can be found in Fig. 3.9.

Hyper-parameters: The hyper-parameters for classification are tuned through grid search based on the classification accuracy of validation samples randomly drawn from the training datasets. The size of the validation set is 320 for all datasets. The resulting hyper-parameters are summarizes in Tab. 3.1. Note that for MNIST and smallNORB, the candidate set $\mathcal{Y}$ is directly formed as all one-hot labels. Since we have not tested non-gradient attack on smallNORB, parameter l is not used in this case.

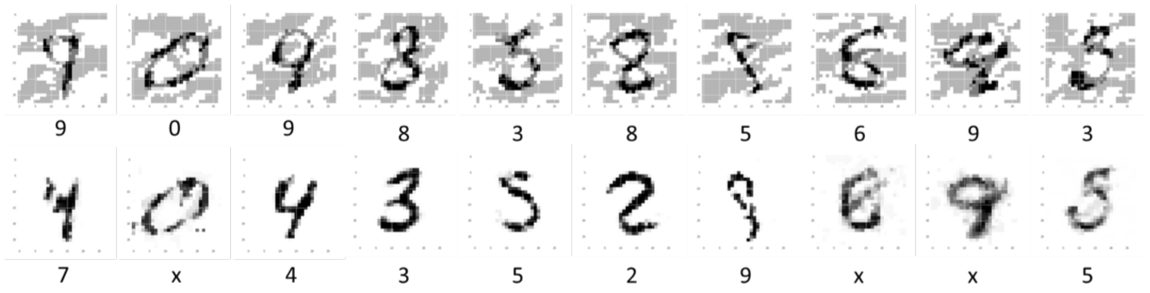


Figure 3.4: Examples of Succeeded Classification on FGSM Adversaries.

3.3.2 Results

Classification accuracy under FGSM attacks

We first compare our method with the binarization defense (Xu *et al.*, 2017b) on FGSM attacked MNIST samples in Tab. 3.2. It can be seem that the performance of binarization drops quickly with increasing attack magnitudes, while the proposed model is able to maintain high accuracy even under relatively high attack magnitude. To further examine the cases where our model fails, we visualize the successfully classified and misclassified FGSM samples under $\epsilon = 0.4$ in Fig. 3.4 and Fig 3.5, respectively. We note that many of the misclassified images are hard to be recognized even for human beings.

Table 3.2: Comparisons between Baseline (Binarization) and the Proposed Method on Robustness under FGSM Attacks with Increasing Perturbation Levels.

	ϵ	0.0	0.1	0.2	0.3	0.4
MNIST	baseline	**0.97**	**0.95**	0.90	0.76	0.51
MNIST	proposed	0.95	0.87	**0.91**	**0.87**	**0.82**
NORB	baseline	59.8	18.3	2.8	0.8	0.2
NORB	proposed	**87.1**	**61.9**	**40.1**	**24.6**	**18.8**

An additional comparison is performed through church window plots for the standard feedforward classifier, the binarization defense, and our algorithm on sample

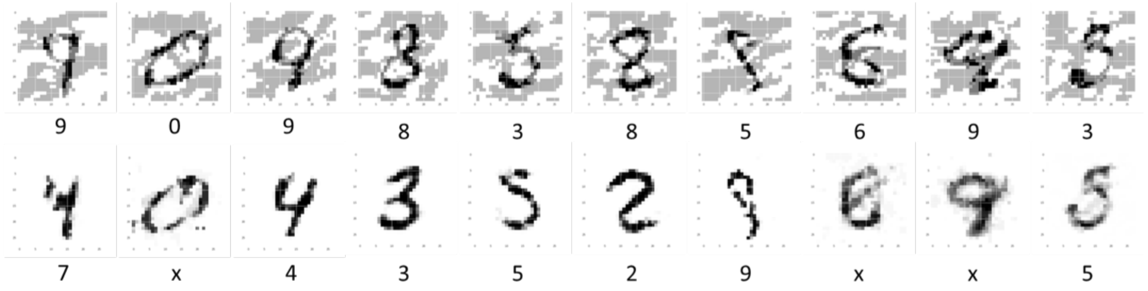


Figure 3.5: Examples of Failed Classification on FGSM Adversaries. "x" denotes the reconstruction error has exceeded the threshold, and the image is regraded as noise.

images from MNIST and NORB in Fig. 3.6. The two dimensions of the plots are the gradient direction of the correct label and a random orthogonal direction, both derived by attacking the feedforward models. Each color represents a different label, with the color at the origin the correct label. Perturbed images on the boundaries as well as the misclassified images with the smallest perturbations are visualized. It can been seen that after binary filtering, the model can still get confused even with small noises, while the proposed method classifies robustly on these samples.

Classification of CPPN samples

While binarize defense almost fails in these cases, our proposed method is still able to maintain satisfactory performance. If the classification result is different from the targeted label with above 50% confidence, we will treat it as misclassified. Fig. 3.7 lists samples of the CPPN images, their binarization, and the corresponding generations for all three datasets we tested. We see that the feedforward classifier is fooled completely by the CPPN images with consistently high confidence for the targeted labels. And even after binarization, many CPPN images can still be misclassified with high confidence. However, when it comes to proposed generative classifiers, it is able to identify 90.6% and 100% CPPN samples on MNIST and TS dataset based on the reconstruction error. This is because the generator is not trained on such highly

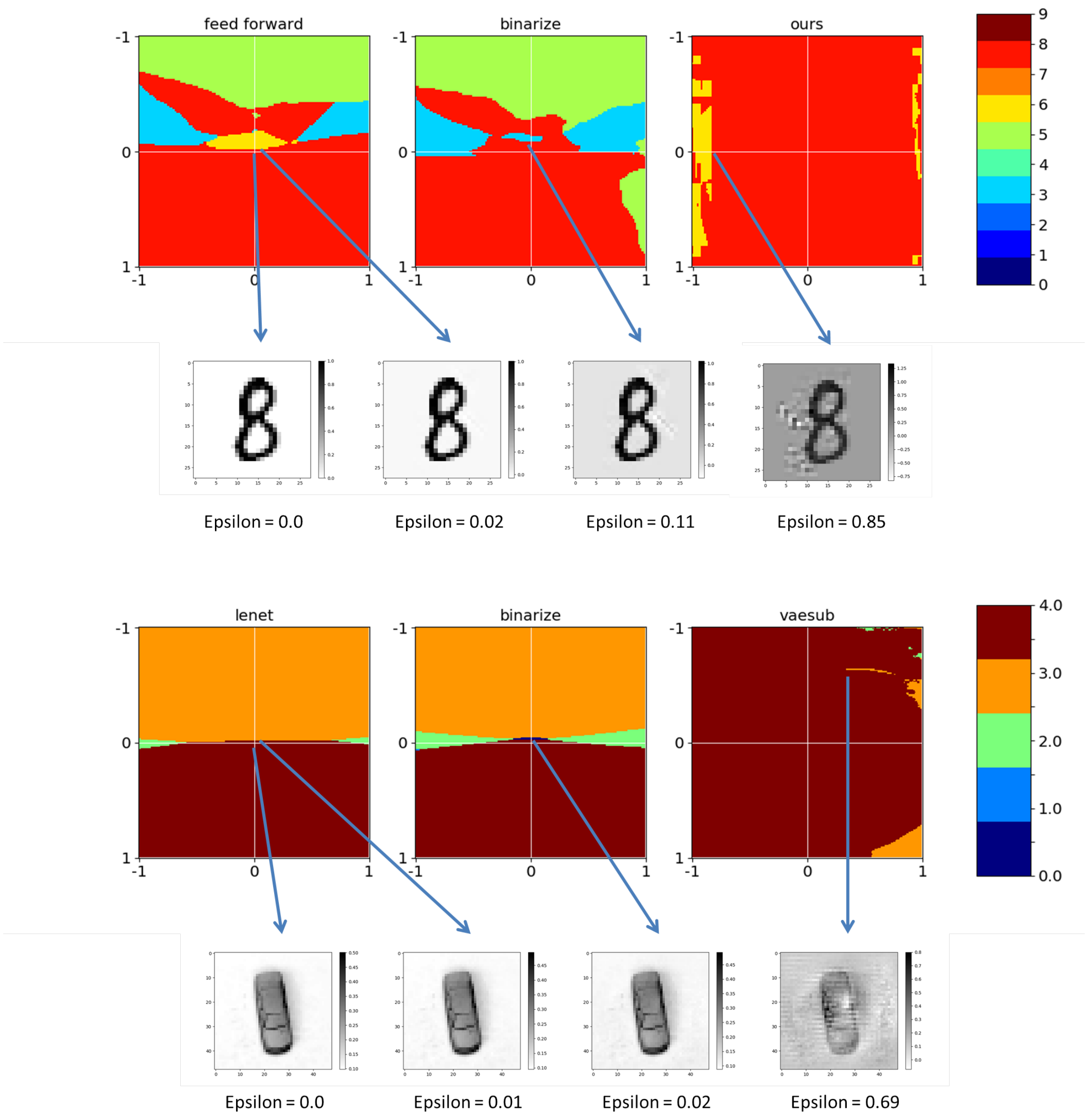


Figure 3.6: Color images from left to right are the church window plots for feedforward classifier, the binarization defense, and proposed generative classifier. A sample is drawn from MNIST (upper) and NORB (bottom) for demonstration. Adversarial images on with the smallest perturbations that gets misclassified are plotted underneath.

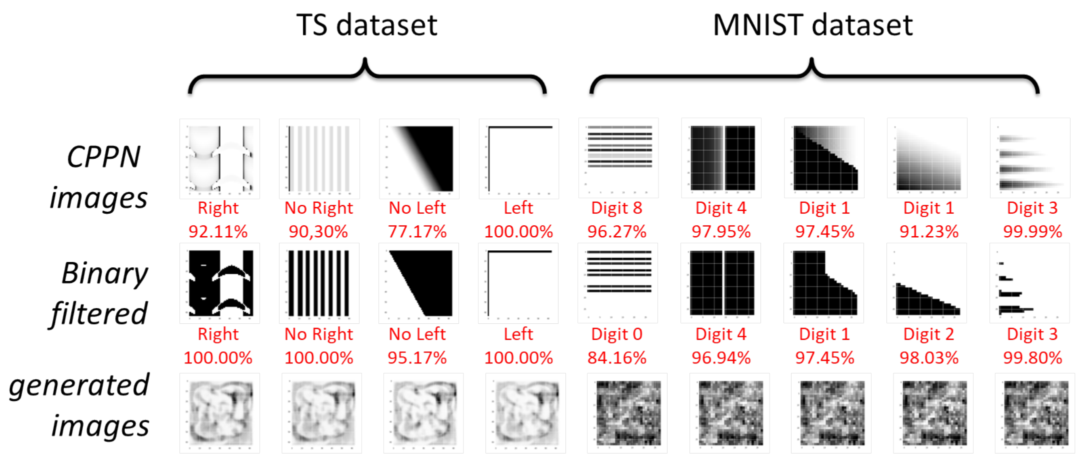


Figure 3.7: Examples of classification results from non-gradient (NEAT) attacks on MNIST and TS. From top to bottom: CPPN images from NEAT, binarized images, and generated images from the proposed method. For each image, we label them by their classification from the feedforward classifiers and the corresponding confidence.

structured pattern, thus it won't be able to reconstruct them out during testing phase either. This leads to a high reconstruction error exceeds the pre-defined threshold. And they will be classified into the noise category based on Algorithm 1.

Classification of novel objects

In this experiment, we first train our model on the TS dataset by only using images from "left turn", "right turn" and "no right turn" categories. After the model is trained, its performance will be evaluated on images from "No left turn" category. This experiment is set up in a way that the network is asked to recognize novel objects not included in the training set.

A traditional feedforward CNN classifier will completely fail under this setting, where 99% of the "no left turn" images are classified as "left turn". This is due to the fact that the classifier will associate the "left arrow" feature with the "left turn" category during the training phase, and this correlation strongly affects the prediction during the testing phase. In contrast, our proposed model learns to decompose the

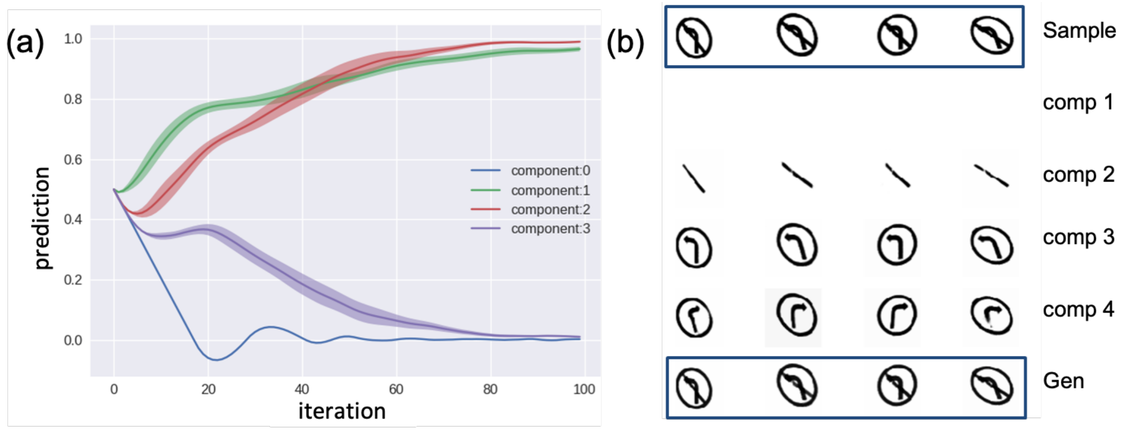


Figure 3.8: Classification Result on Novel Objects. Left: convergence of component labels during the classification process, with ±3 standard deviation over all "no left turn" test images. Right: the inputs (top), their sub-network generations (middle), and the optimal final generations (bottom).

TS dataset into components after training (as shown in Fig. 3.8b), and by combining these components together, it correctly recognizes 95% of all novel testing images.

Classification of overlapping objects

We now demonstrate that the proposed model is able to handle objects with large overlap using the multiMNIST dataset. As shown in Fig. 3.3, the proposed model learns the decomposition of individual digits successfully after training. Applying the learned model to classifying the test dataset leads to a classification accuracy of 65.6%. Successful and failed test samples and their classification results are shown in Fig. 3.9. While the accuracy on multiMNIST is lower than that of a CapsNet (95%) (Sabour *et al.*, 2017), investigation of the model performance shows that many of the misclassified samples are truly difficult to be separated even for human beings. It should also be noted that the accuracy of CapsNet is resulted from 60 million training data, while our model successfully decomposes the learns the component-wise latent spaces with only 128k data points. We expect improved classification accuracy by increasing the capacity of the generative model.

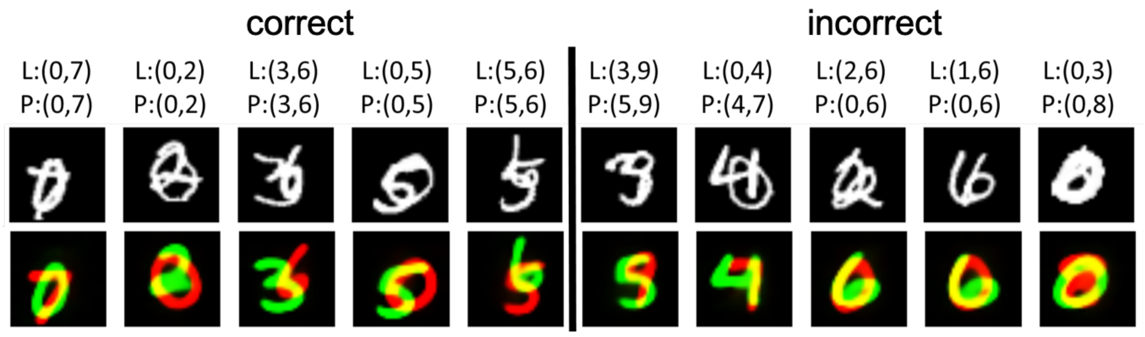


Figure 3.9: Examples of Classification Results on MultiMNIST. First row: input test images. Second row: predicted images. L and P denotes ground truth and predicted labels, respectively.

3.3.3 Discussion

The fundamental challenge towards robust and generalizable classification is the lack of unsupervised prediction capability of existing data-driven methods, i.e., from the given data points, we are not able to predict the subset of the image space where data from each label should reside, leaving class boundaries of a feedforward classifier incorrect or vulnerable to attacks. Existing adversarial training mechanisms use the fact that vicinity of a given data point in the image space should possess the same label, and push the class boundaries accordingly to defend attacks with small perturbations. These, however, do not consider the fact that images with large perturbations from the data points, can still be perceptually similar to these points. The proposed method in this chapter can be considered as an attempt to infer class labels far away from the data. The method is similar to K-nearest neighbour in that the classification is performed based on minimum distances. However, instead of measuring the distances to data points, we do so with the manifolds defined by the generative models, thus leveraging the structure of the data.

While the presented experiments show promise of the proposed classification method, following future works can be done to better characterize the applicability and limitations of the proposed method. (1) Deeper investigations are needed to determine if measuring distances in certain feature space will be more effective than in the image space, as suggested by Sabour *et al.* (2017). The fact that feedforward classifiers can achieve high test accuracy suggests that they can map data points to a feature space where a relatively simple classifier can be applied. Thus it will be interesting to see if learning structures of data points in this feature space, instead of the image space, can lead to more generalizable classifiers. (2) The proposed method still requires training labels. Given existing learning mechanisms that perform clustering and generation (e.g., infoGAN (Chen *et al.*, 2016b)), it would be interesting to investigate how our algorithm can be integrated with such mechanisms. (3) Scalability to rich datasets is a valid concern of the current model, as it will require a large number of sub-networks when applied to datasets with a large range of objects and components. One potential solution, as discussed in (George *et al.*, 2017), is to build shared lower-level layers across the sub-networks. (4) In addition, the proposed model used a simple summation to assemble outputs from the sub-networks, which may fail to work for cases where spatial overlap of object components should be considered. One solution could be to introduce a depth variable to guide the assembly. The variable will be modeled as an output of the decoder in addition to the sub-networks, and will be optimized during classification.

3.4 Conclusions

In this paper, we investigate the utility of a tailored conditional variational autoencoder as a classifier, and test its generalizability and robustness under challenging tasks. Results show that the proposed training and classification formulations lead to promising performance: First, the model can recognize overlapping objects and novel component combinations that do not exist in the training phase. Second, our model is able to defend against adversarial attacks well, in particular under higher attack magnitudes and under none-gradient attacks.

Chapter 4

ROBUST CLASSIFICATION THROUGH TRANSFORMATION AUGMENTATION

4.1 Introduction

The vulnerability of neural networks under adversarial attacks has led to rapid development of theories and algorithms on both attack side (Szegedy *et al.*, 2013; Kurakin *et al.*, 2016b,c; Madry *et al.*, 2017; Carlini and Wagner, 2017; Moosavi-Dezfooli *et al.*, 2016; Chen *et al.*, 2017; Su *et al.*, 2017) and defense side (Xu *et al.*, 2017a; Das *et al.*, 2018; Gu and Rigazio, 2014; Liao *et al.*, 2018; Roy *et al.*, 2018; Papernot *et al.*, 2016c; Guo *et al.*, 2017; Kurakin *et al.*, 2016b; Tramèr *et al.*, 2017; Madry *et al.*, 2017). However, most existing defense mechanisms are later shown to be vulnerable under stronger attacks (Evtimov *et al.*, 2017; Athalye *et al.*, 2018). Adversarial training through robust optimization remains to be one of the few defense mechanisms that do not suffer gradient obfuscation, and so far adversarial training with PGD attack achieves the best empirical robustness under l_∞ attacks (Athalye *et al.*, 2018; Madry *et al.*, 2017).

Existing work has revealed that adversarial attacks can become ineffective under small natural image transformations, and therefore random transformations can be used to pre-process the inputs and improve model robustness (Guo *et al.*, 2017). It was later shown, however, by aggregating attack losses from different transformations together, the resultant adversarial image will not be affected under these transformations (Athalye *et al.*, 2018; Dong *et al.*, 2019). The feasibility of such transformation-invariant attacks in real-world applications has also been demonstrated (Athalye and

Sutskever, 2017). To our best knowledge, however, no existing defense approaches explicitly tackle the transformation-invariant adversarial attacks.

It is further revealed in Tsipras *et al.* (2018) that there exists an intrinsic trade-off between model robustness and standard accuracy, which may explain the limited robustness currently achieved, e.g., on CIFAR-10 (Tsipras *et al.*, 2018) and ImageNet (Kurakin *et al.*, 2016d). They also suggested suggested that model robustness relies on the use of robust features, and that training driven by standard accuracy tends to bias the model towards non-robust features (Tsipras *et al.*, 2018). On the other hand, experiments have shown that human vision, which is robust to adversaries, is invariant to mild natural input transformations (Ullman *et al.*, 2016). Drawing on these findings, it is reasonable to question whether constraints on transformation invariance may force image classifiers to learn and rely on robust features only and, in turn, acquire better robustness.

This chapter is thus motivated to investigate the following hypothesis: *Model robustness can be enhanced through training against transformation-invariant attacks.* As a preliminary study, we consider transformations including input cropping, rotation, and zooming. We assume that under appropriate parameter settings, these transformations are content preserving, i.e., the transformed images keep salient features, and can still be correctly classified by human beings. We apply adversarial training to the proposed model, and simulate transformation-invariant adversaries during the attack phases of the training. A comparison between the proposed method and the standard adversarial training is illustrated in Fig. 1.4.

Experiments on MNIST (LeCun *et al.*, 1989) and CIFAR-10 (Krizhevsky *et al.*, 2014) lead to the following key findings:

1. Imposing transformation invariance on clean or adversarial training does not improve model robustness;

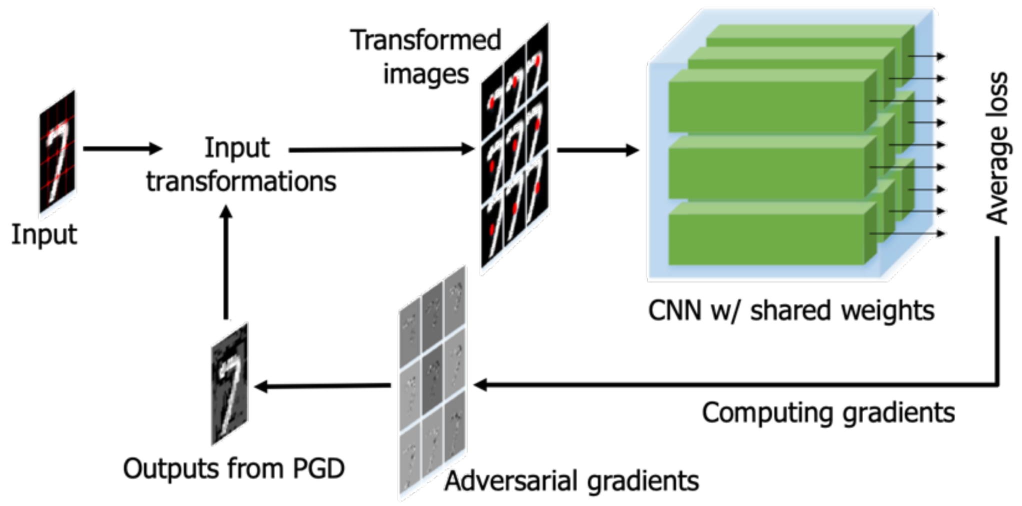


Figure 4.1: Schematic of the Proposed Learning Architecture

2. Adversarial training using transformation-invariant attacks significantly improves the empirical model robustness from the state of the art (baseline model (Madry *et al.*, 2017)): 93.2% to 97.2% for MNIST, and 47.3% to 54.4% for CIFAR-10, both on l_∞ attacks with attack bounds $\epsilon = 0.3$ and $\epsilon = 8/255$, respectively.

3. The proposed learning architecture and training method significantly improve learning efficiency from the state of the art.

4. The proposed method is more effective at learning robust features.

4.2 Proposed Method

As shown in Fig. 4.1, the proposed learning architecture passes an ensemble of transformations of an input through copies of a shared convolutional neural network, before aggregating the network outputs. Below we introduce the architecture and implementation details.

Decision making Let the shared network be $f(\cdot, \theta)$, and the set of input transformations be $\mathcal{T}$. The probabilistic prediction of an input x follows:

$$y_{pred} = \frac{1}{|\mathcal{T}|} \sum_{T \in \mathcal{T}} f(T(x), \theta). \tag{4.1}$$

Aggregated loss The aggregated loss for data (x, y) is defined as:

$$J(x, y; \theta) = \sum_{T \in \mathcal{T}} L(y, f(T(x), \theta)) \tag{4.2}$$

For classification tasks, $L(\cdot, \cdot)$ is the cross-entropy.

Adversarial training under robust attacks The training objective follows robust optimization, which is a min-max problem:

$$\min_{\theta} \mathop{\mathbb{E}}_{x,y \sim D} \left[\max_{x^{adv} \in \mathcal{N}_{\epsilon}(x)} J(x^{adv}, y; \theta) \right] \tag{4.3}$$

As discussed in Sec. 2.2.2, this minimax problem can be decoupled and solved sequentially. The inner maximization problem is an attack robust against *all transformations*, and is similar to the Expectation of Transformation (EoT) method (Athalye *et al.*, 2018; Athalye and Sutskever, 2017). However, to our best knowledge, EoT has not yet been incorporated into adversarial training.

We use untargeted PGD attacks to approximately solve the inner maximization problem in Eq. (4.3). The attack parameters for different dataset follows the baseline model proposed in Madry *et al.* (2017), which are also summarized in Tab. 4.1. Term a and t denote the step size and number of gradient descent steps in one attack, respectively.

Network architecture Our network for MNIST and CIFAR-10 follow the wider networks in Madry *et al.* (2017).

Table 4.1: Hyper-parameters used for adversarial training.

Dataset	ϵ	a	t
MNIST	0.3	0.01	40
CIFAR-10	$8/255$	$2/255$	7

Transformations The input transformations we tested in this chapter include cropping, rotation, and zooming. For cropping, the configuration is shown in Fig. 4.2. The number of crops in the case of CIFAR-10 is limited by the fact that we only have 8 GPUs in parallel, each of which handles one copy of the shared network. For rotation, we use the built-in rotation function from TensorFlow. For zooming, we first apply cropping and then rescale the image back to the original size through bilinear interpolation.

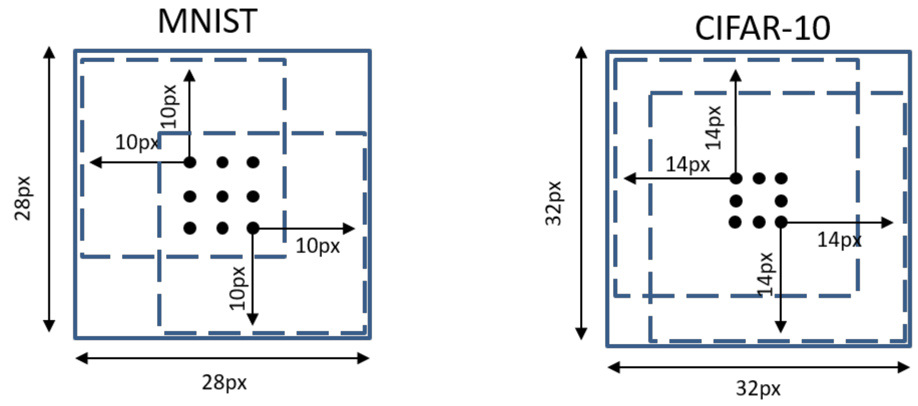


Figure 4.2: Location and Size of the Crops for MNIST and CIFAR-10. Black dots denote the center of the crops.

4.3 Results and Discussions

We first evaluate the empirical robustness of the proposed method against various existing attacks with different transformation settings. We will focus on image cropping as the transformation applied to the input, and show that with this simple setting our model can already consistently outperform the state of the art on evaluation tasks. Mixed results from applying rotation and zooming will then be discussed. We highlight that training the model with separated attacks for individual input transformations is not effective. We perform a thorough investigation on whether our method relies on gradient obfuscation in the second part. Lastly, attempts are made to explain the performance gain from our model by showing that incorporating transformation invariance in an adversarial training setting increases the probability at learning robust features and dropping the non-robust ones.

4.3.1 Evaluation of empirical network robustness

White-box robustness The white-box robustness of our model is compared with baseline approach on MNIST and CIFAR-10 under FGSM (Szegedy *et al.*, 2013), BIM (Kurakin *et al.*, 2017), PGD (Madry *et al.*, 2017), and C&W (Carlini and Wagner, 2017) attacks. For fair comparison, we follow the parameters for adversarial training in Madry *et al.* (2017) to train proposed model. The parameters for adversarial attack in training is listed in Fig.4.1. The testing attack parameters for FGSM, PGD, and BIM attacks also follow Tab. 4.1. For C&W attacks, we use learning rate 0.2 and 40 steps with a Lagrange multiplier of 1.0.

We compared our model with baseline model trained with standard optimization and robust optimization. The white-box robustness for these models are summarized in Tab. 4.2, together with accuracy on clean test data. With transformation-invariant adversarial training, our model is able to achieve comparable or higher accuracy than the baseline model both with and without adversarial training. More importantly, our model is able to achieve significantly higher robustness under all tested attacks. In particular, the proposed method improves white-box PGD robustness from 93.2% to 95.7% on MNIST, and from 47.3% to 54.4% on CIFAR-10.

In addition, we test how robustness changes along with the number of iterations

Table 4.2: Robustness on MNIST and CIFAR-10 Against White-box Attacks. "None": clean test accuracy, B_{nat}, B_{adv}: Baseline model without and with adversarial training; P_{nat}, P_{adv}: Proposed model without and with transformation-invariant adversarial training.

MNIST

attack	None	FGSM	BIM	PGD	C&W
B_{nat}	98.9	7.0	0.0	0.0	3.2
P_{nat}(Ours)	**99.3**	5.7	0.8	0.5	20.8
B_{adv}	98.4	95.2	92.5	93.2	91.7
P_{adv}(Ours)	99.2	**96.9**	**95.0**	**95.7**	**96.0**

CIFAR-10

attack	None	FGSM	BIM	PGD	C&W
B_{nat}	95.2	12.8	0.0	4.1	0.0
P_{nat}(Ours)	**95.6**	15.2	0.0	13.0	0.3
B_{adv}	87.3	56.4	48.36	47.3	19.51
P_{adv}(Ours)	87.9	**59.4**	**52.9**	**54.4**	**22.89**

(t) in PGD attacks. To do so, we increase t from 0 to 20 while fixing the attack bound ϵ and the step size a, and compare the performance between P_{adv} and B_{adv} (Fig. 4.3). The proposed model consistently out-performs the baseline. It should also be noted that our model is comparable to TRADES Zhang *et al.* (2019b), although a more rigorous comparison is needed (e.g., Zhang *et al.* (2019b) uses ResNet-18 for CIFAR-10, while we followed the model in Madry *et al.* (2017)). Specifically, on CIFAR-10 with 20-step PGD and $\epsilon = 8/255$, TRADES achieves robustness (clean accuracy) of 56.6% (84.9%) for $1/\lambda = 6$ and 49.1% (88.6%) for $1/\lambda = 1$. In comparison, our model achieves a robustness (clean accuracy) of 50.3% (87.7%).

Furthermore, we compare the white-box robustness under PGD attacks beyond the attack bounds ϵ used during adversarial training. For MNIST, we test $\epsilon \in [0, 1]$

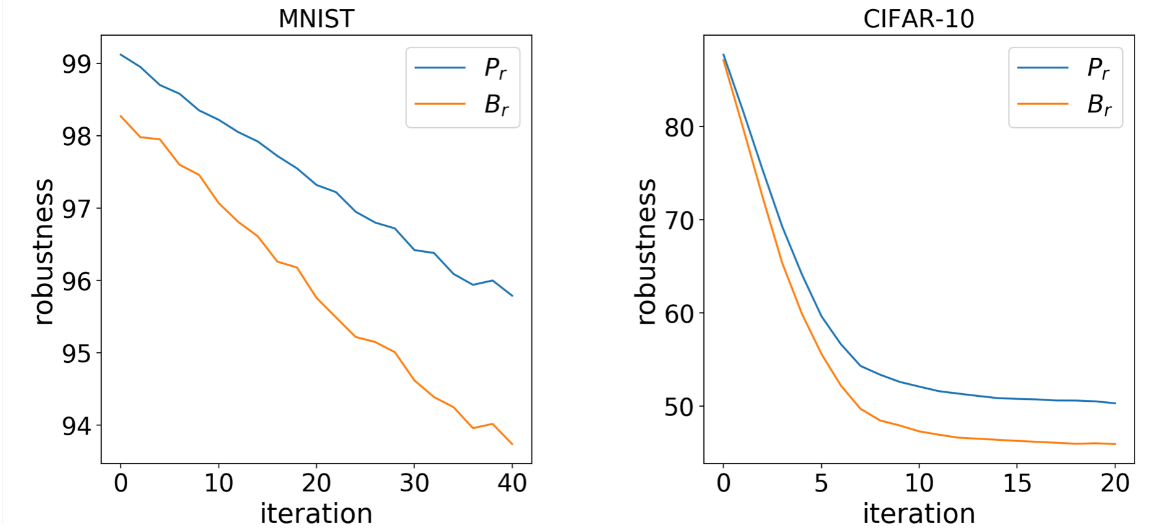


Figure 4.3: Model robustness for a range of number of PGD steps.

where $\epsilon = 1$ represents the maximal l_∞ attack strength. For CIFAR-10, we test $\epsilon \in [0, 35/255]$. The comparisons are shown in Fig. 4.4. Still, the proposed method exhibits consistently higher robustness than the baseline under all attack bounds.

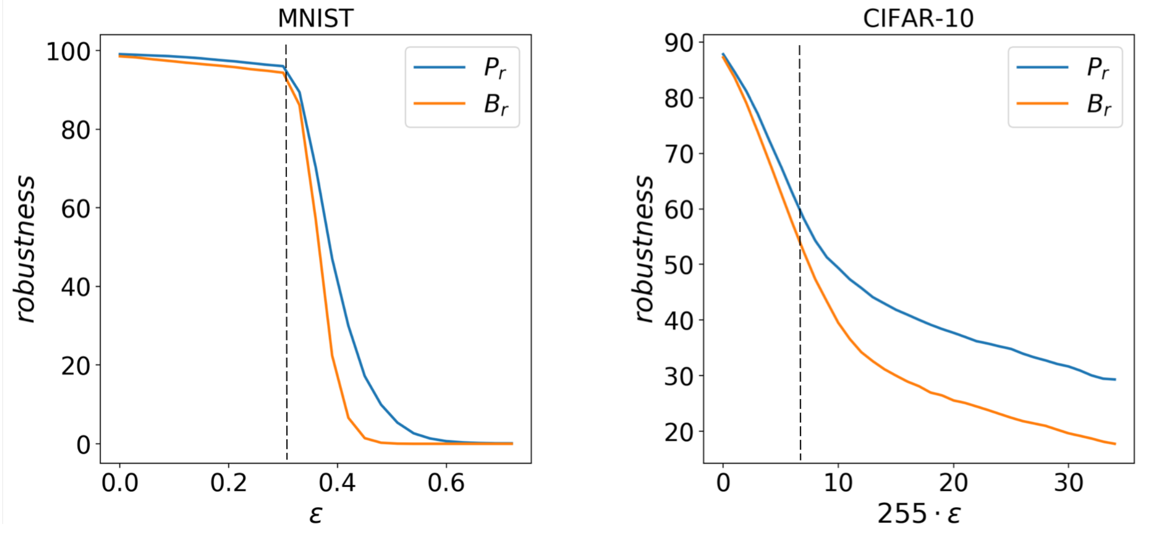


Figure 4.4: Model robustness for a range of attack bounds. The attack bounds used for training are marked as black vertical lines.

Sensitivity of hyper-parameters on white-box robustness We study the influence of the number of input crops and the crop size on model robustness using MNIST: For the former, we fix the cropping size to 20 and vary the number of crops from 1 to 64. The locations of crops are shown in Fig. 4.1b. For the latter, we fix the number of crops to 9 and vary the size of the crops from 12 to 24. Both training and test use $\epsilon = 0.3$. The white-box PGD accuracy of these models are summarized in Tab. 4.3.

Table 4.3: Parametric Study on the Number of Crops and the Crop Size Used in the Proposed Model

cropping size	12	16	20	24	28
clean testing	11.3	98.6	**99.2**	99.1	98.4
PGD white-box	11.1	94.2	**95.7**	94.9	93.2
number of crops	1	4	9	36	64
clean testing	98.3	99.0	**99.2**	**99.2**	**99.2**
PGD white-box	92.1	95.1	95.7	**96.1**	**96.1**

It can be seen that increasing the number of crops helps to improve both clean and adversarial accuracy, with diminishing effect. On the other hand, a sweet spot exists for the crop size: Larger cropping sizes tend to improve clean test accuracy, yet inevitably lead to reduced number of crops and robustness, while making the size too small will reduce the clean test accuracy significantly, which upper bounds the robustness. For the remaining experiments, we use 9 crops each with size 20 for MNIST and 8 crops with size 28 for CIFAR-10.

Effect of transformation ensemble on model robustness To better understand the effect of transformation ensemble on model robustness, we further conduct an ablation study where we remove the ensemble effect during training and test phases separately and monitor how the robustness changes in each of the cases.

No ensemble in the test phase: Here we train the model as proposed, and test the white box robustness of each individual network copies in the ensemble. The copies

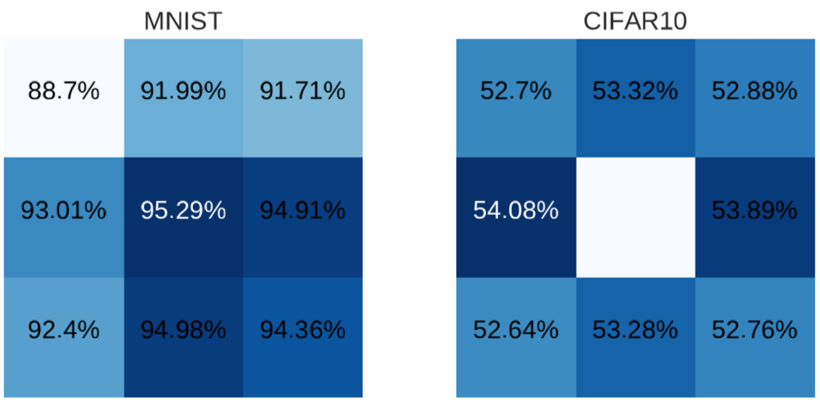


Figure 4.5: White-box Robustness of Individual Network Copies Within the Ensemble Under PGD Attacks. Locations in the grid correspond to cropping locations.

are only different in their input transformation layers. The robustness of individual copies range from 88.7% to 95.3% on MNIST, and 52.6% to 54.1% on CIFAR-10 (Fig. 4.5). These values are lower than the robustness with ensemble (95.7% and 54.4%), suggesting that ensemble is effective during test.

No ensemble in the training phase: We now investigate the effect of the ensemble in adversarial training on model robustness. Specifically, we remove the influence of ensemble on the generation of adversaries, in which case the training becomes standard adversarial training with training data replaced by their cropped copies. When using nine cropping windows, this leads to a 9-fold data augmentation. During adversarial training, this experiment setting results in adversaries that are not necessarily transformation invariant. In the test phase, we perform the same ensemble operation as in Eq. (4.1). On MNIST, these settings lead to a model robustness of 93.4% and clean test accuracy of 98.7%, which is comparable to baseline and worse than the proposed.

This experiment reveals the critical role of transformation-invariant attacks in improving robustness from standard adversarial training. We conjecture that when the model is attacked by individual transformations, the resulting adversaries may lead to contradictory gradient directions for model refinement, i.e., refining the model with respect to attacks for one particular transformation may not help (or even worsen) the robustness under other transformations.

Effect of rotation and zooming Investigation on the effect of input rotation and zooming leads to mixed results, which is listed in Tab. 4.4. On MNIST, rotation yields more robust models while zooming does not; on CIFAR-10, model robustness improves with mild rotation angles (maximum 4 degrees); yet larger angles (maximum 30 degrees) and zooming show limited effects. It is worth noting that combining different transformations may lead to extra improvement in robustness. Specifically, we tested the combination of two models on MNIST, trained separately with cropping (cropping size of 20, 9 crops) and rotation (4 orientations). The model reaches 97.2% white-box robustness, while the individual models have 95.7% with cropping and 95.5% with rotation.

Table 4.4: Robustness and standard accuracy of models trained on transformation-invariant attacks. Baseline robustness (standard accuracy) is 93.2 (98.4) on MNIST and 47.3(87.3) on CIFAR-10.

MNIST			
$\lvert\mathcal{T}\rvert$	1	4	9
cropping	92.1 (98.3)	95.1 (99.0)	95.7 (99.2)
rotation	93.3 (98.7)	95.5 (99.1)	96.1(99.3)
zooming	90.7 (99.2)	91.5 (98.5)	93.6 (99.2)
CIFAR-10			
$\lvert\mathcal{T}\rvert$	1	4	8
cropping	45.1 (80.5)	53.3 (83.1)	54.4 (87.9)
rotation	46.5 (78.6)	54.4 (86.8)	53.8 (87.8)

Challenge with rotation and zooming on CIFAR-10: It is interesting to note that rotation augmentation on CIFAR-10 only works well under small rotation angles. The robustness and standard accuracy will be reduced to 48.1% and 78.9% if we increase the maximum rotation angle from 4 degrees to 30 degrees. We conjecture that this is because larger rotation leads to an increased number of robust features for CIFAR-10 (e.g., the same image patterns of different angles). With the fixed network capacity, this may lead to unsuccessful learning. Such phenomenon, however, does not have significant influence on the robust features on MNIST (which are strokes of different orientations, see Fig. 4.10), and thus allows a network with fixed capacity to perform relatively equally across cropping, rotation, and zooming. We thus believe that increasing the network capacity or building in rotation invariance may help improving the robustness and accuracy at a larger rotation angles, but this investigation will be left as future work due to the resultant high training costs.

Convergence issue with saturation: A recent study discovered the sensitivity of model robustness to input distribution (Ding *et al.*, 2018). Inspired by this discovery, we tested the effect of saturation on model robustness *in the original input space.* However, our current implementation of robust adversarial training with image saturation suffers from explosion of the adversarial gradient, which is intrinsic to the transformation. A smooth approximation of the saturation operation needs to be introduced before a proper evaluation of its effect on model robustness can be performed.

Learning efficiency We train the proposed and baseline models with increasing training data sizes and compare model robustness and accuracy along the data size. We still use the PGD attacks with parameters from Tab. 4.1 for training. A comparison of the robustness of obtained models is shown in Tab. 4.5. It can be seen that our models consistently achieves higher robustness with different amount of training data. Or in other words, proposed model require less training data for the same robustness or accuracy levels.

Table 4.5: Comparison on learning efficiency. A: clean test accuracy, R: white-box robustness under PGD attacks

MNIST						
$\|\mathcal{D}\|$	0.3k	1k	3k	10k	30k	60k
A(B_{adv})	91.7	**96.6**	96.5	97.8	98.5	98.4
A(P_{adv})	**93.4**	96.3	**97.6**	**98.5**	**98.8**	**99.2**
R(B_{adv})	35.0	73.6	80.5	86.7	88.0	93.2
R(P_{adv})	**68.2**	**82.6**	**89.0**	**93.7**	**95.6**	**95.7**

CIFAR-10						
$\|\mathcal{D}\|$	0.3k	1k	3k	10k	30k	60k
A(B_{adv})	43.1	52.2	62.5	**73.2**	80.0	87.3
A(P_{adv})	**43.3**	**55.0**	**67.4**	72.1	**80.2**	**87.9**
R(B_{adv})	10	12.8	21.4	33.1	45.3	47.4
R(P_{adv})	**12.7**	**17.6**	**27.2**	**37.2**	**47.8**	**54.5**

4.3.2 Gradient obfuscation

We have shown that our model achieves high white-box robustness. However, as discussed in Athalye *et al.* (2018), intentionally or not, many models are based on gradient obfuscation (shattering, masking, and explosion/vanishing) and their robustness is not reliable. We first note that the proposed model does not create shattering since it has no randomness, and does not create gradient explosion or vanishing since it does not contain long recurrence. In the following, we investigate whether gradient masking exists in our model with black-box attacks, non-gradient based attacks, and visualization of the loss landscape.

Black-box attack Models with gradient masking tend to have very low black box robustness. To investigate the black-box robustness of the proposed model, we perform PGD attacks on four source models to generate adversarial samples: B_{nat}, B_{adv}, P_{nat}, and P_{adv}. We again use $\epsilon = 0.3$ and $\epsilon = 8/255$ for MNIST and CIFAR-10, respectively. These samples are tested on different models, and the results are summarized in Tab. 4.6. The rows and columns correspond to different test and source models, respectively. Diagonal elements are the white-box accuracy. It can be seen that proposed model achieves some improvements under all tested black-box attacks as well.

Table 4.6: Robustness on MNIST and CIFAR-10 against black-box PGD attacks

MNIST

	B_{nat}	P_{nat}	B_{adv}	P_{adv}	worst case
B_{nat}	0.0	8.8	85.6	94.8	0.0
P_{nat}(Ours)	39.3	0.6	63.6	89.5	0.6
B_{adv}	96.7	96.5	93.2	95.5	93.2
P_{adv}(Ours)	**97.9**	**97.4**	**98.2**	**95.7**	**95.7**

CIFAR-10

	B_{nat}	P_{nat}	B_{adv}	P_{adv}	worst case
B_{nat}	0.0	0.2	79.8	68.0	0.0
P_{nat}(Ours)	0.0	0.2	**81.1**	**69.7**	0.0
B_{adv}	86.1	86.1	47.4	66.9	47.4
P_{adv}(Ours)	**86.2**	**86.3**	69.5	54.4	**54.4**

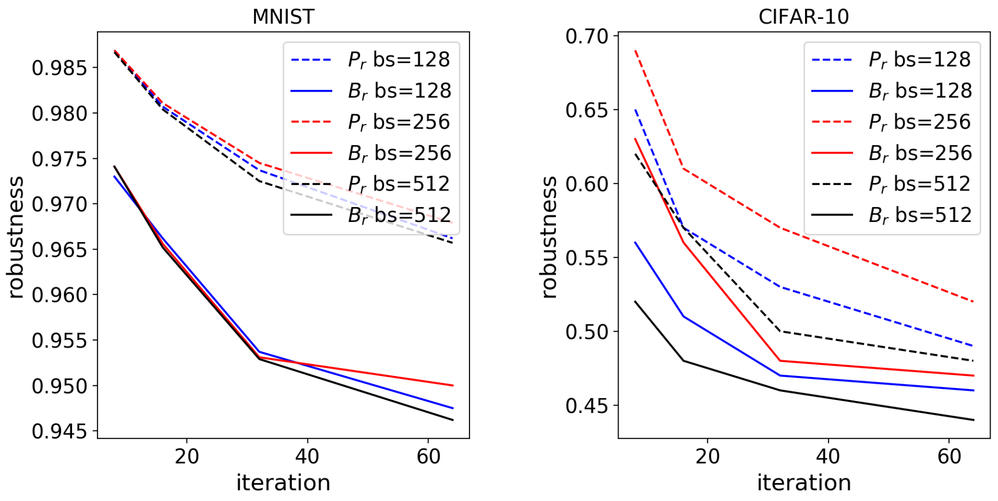


Figure 4.6: Network Robustness Under SPSA Attack.

None-gradient based attacks We tested the performance of B_{adv} and P_{adv} under white-box SPSA attacks Uesato *et al.* (2018). Since SPSA uses estimated gradient instead of direct differentiation, it is less likely to be affected by gradient masking. The performance comparison of proposed model with baseline model is shown in Fig. 4.6. It can be seen that the proposed method consistently out-performs the baseline under different SPSA iteration and batch size, which again indicates that the proposed model does not rely on gradient masking.

Loss landscape As discussed in (Athalye *et al.*, 2018), models with gradient obfuscation, i.e., gradient masking, shattering, and explosion or vanishing, can be attacked by tailored attacks. Our model does not create shattering since it has no randomness; it also does not create gradient explosion or vanishing since it does not contain long recurrence. In this experiment, we visualize the loss landscapes around random test points to check if gradient masking effects exists or not. As shown in Fig. 4.7, the gradient landscapes are informative and smooth. Therefore, the reported robustness does not rely on gradient obfuscation and the improvements are reliable.

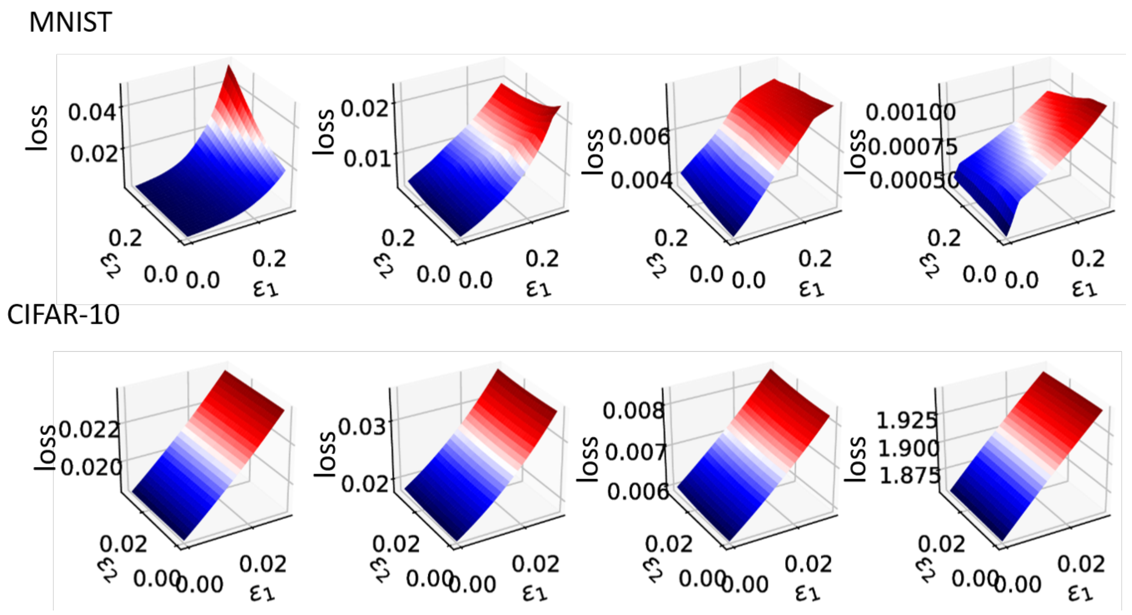


Figure 4.7: Loss landscapes around random test samples on MNIST (top row) and CIFAR-10 (bottom row). ϵ_1 is the adversarial gradient direction and ϵ_2 is a random direction orthogonal to the adversarial gradient.

4.3.3 The influence of input cropping-invariant attacks on the learning of robust features

From the experiments we observe that cropping is effective as an input transformation at improving model robustness. In the following, we provide preliminary explanations to this finding through two toy cases inspired by (Tsipras *et al.*, 2018): 1) We use a Gaussian data model to show that using cropped inputs for adversarial training leads to a higher probability of both dropping non-robust features and learning robust ones; 2) we then conduct a binary classification task on digits "5" and "7" from MNIST, and empirically show that incorporating cropping leads to more successful learning of robust features.

Preliminary analysis of the proposed method *A binary classification task*: Consider a data model consisting of input-label pairs (x, y) sampled from a distribution $\mathcal{G}$:

$$y \overset{u.a.r}{\sim} \{-1, 1\}, \ x_i \overset{i.i.d.}{\sim} \mathcal{N}(\eta_i y, 1), \tag{4.4}$$

where $\mathcal{N}(\mu, \sigma^2)$ is a normal distribution with mean μ and variance σ^2, and $\eta_i \in [0, 1]$ represents the correlation between x_i (the ith element of x) and y. We consider a linear classifier with parameters w, $f(x) := \text{sign}(w^T x)$. With regularization, adversarial training solves:

$$\min_{w:||w||_2 \leq 1} \max_{||\delta||_\infty \leq \epsilon} \mathbb{E}_{\mathcal{D}} \left[\max(0, 1 - y w^T (x + \delta)) \right]. \tag{4.5}$$

From Tsipras *et al.* (2018), the solution to Eq. (4.5) follows a simple rule: With a finite dataset $\mathcal{D}$ drawn from $\mathcal{G}$, if $|\mathbb{E}_{\mathcal{D}}[y x_i]| \geq \epsilon$, w_i is assigned a non-zero weight, or otherwise w_i is zero. See Fig. 4.8a for an example. When $\mathcal{D}$ is infinite, we derived the true robust features $\mathcal{R} := \{x_i | \eta_i \geq \epsilon\}$. Without loss of generality, we will consider $\eta_1 = 1$ and $\eta_2 = \eta_3 = ... = \eta_{d+1} = \eta < \epsilon$ in the following analysis.

Robust features under the proposed model: We consider a simple aggregation over transformations $\mathcal{T} := \{T_k\}_{k=1}^K$, which leads to the classifier $f(x) := \text{sign}(\theta^T z)$, where $z := (1/K) \sum_{k=1}^K T_k(x)$, and $T_k(x)$ is the kth transformed feature. Let z_i be the ith element of z. With input cropping, we have $z_i = (1/K) \sum_{j \in N_i} x_j$, where N_i is a set of image pixels for the ith aggregated feature. For simplicity, we assume that $N_i = \{i, i+1, ..., i+K\}$ for $i = 1, ..., d+1-K$. This makes N_1 the only set that contains x_1 (the robust feature). We have the following distribution in the aggregated feature space:

$$z_1 \sim \mathcal{N}(\rho, \frac{1}{K}), \ z_i \overset{i.i.d.}{\sim} \mathcal{N}(\eta y, \frac{1}{K}), \ \forall i \geq 2, \tag{4.6}$$

where $\rho = \frac{y(1+(T-1)\eta)}{K}$. When $\rho \geq \epsilon$, and under infinite data size, adversarial training leads to $\theta_1 = 1$ and all other weights as 0s. Thus z_1 is the robust (aggregated) feature.

Distribution of sample means: Denote the sample means of the correlations as $\hat{\eta}_i$ for x_i, and $\hat{\gamma}_i$ for z_i. Under a finite data size N, $\hat{\eta}_i$ and $\hat{\gamma}_i$ follow the following distributions:

$$\begin{aligned} \hat{\eta}_1 \sim \mathcal{N}(y, \frac{1}{N}),\ \hat{\eta}_i \overset{i.i.d.}{\sim} \mathcal{N}(\eta y, \frac{1}{N}),\ \forall i \geq 2, \\ \hat{\gamma}_1 \sim \mathcal{N}(\rho, \frac{1}{NK}),\ \hat{\gamma}_i \overset{i.i.d.}{\sim} \mathcal{N}(\eta y, \frac{1}{NK}),\ \forall i \geq 2, \end{aligned} \tag{4.7}$$

Effectiveness of dropping non-robust features: The probabilities of correctly dropping a non-robust feature are $\Phi((\epsilon-\eta)N)$ and $\Phi((\epsilon-\eta)NK)$ before and after applying the transformation, respectively, where Φ is the cumulative distribution function for $\mathcal{N}(0,1)$. Since $\epsilon - \eta > 0$ for non-robust features and the number of transformations $K > 1$, the transformation improves the effectiveness of dropping non-robust features.

Effectiveness of learning robust features: The probabilities of correctly learning the robust feature are $p(\hat{\eta}_1 > \epsilon) = 1 - \Phi((\epsilon - 1)N)$ and $p(\hat{\gamma}_1 > \epsilon) = 1 - \Phi((\epsilon - \rho)NK)$ before and after applying the transformation, respectively. We can derive from here that $p(\hat{\eta}_1 > \epsilon) < p(\hat{\gamma}_1 > \epsilon)$ iff

$$2 \log K + Nh(\rho) > 0, \tag{4.8}$$

where $h(\rho) = \epsilon^2(K-1) + (\rho^2 K - 1) - 2\epsilon(\rho K - 1)$ is a quadratic function of ρ. Given K, ϵ, and ρ, Eq. (4.8) sets a condition on N for which a model that incorporates transformation will have a higher probability of correctly extracting the robust feature. Specifically, we can derive the following conclusions: (1) When ρ is close to 1, $p(\hat{\eta}_1 > \epsilon) < p(\hat{\gamma}_1 > \epsilon)$ for any $N > 0$. We can show in particular that when $\rho = 1$,

$$\epsilon^2(K-1) + (\rho^2 K - 1) - 2\epsilon(\rho K - 1) = (K-1)(1-\epsilon)^2 > 0, \tag{4.9}$$

thus any $N > 0$ satisfies Eq. (4.8). (2) For $\rho \in [1/K, \rho^*]$, where ρ^* is the larger root of $h(\rho) = 0$, $p(\hat{\eta}_1 > \epsilon) < p(\hat{\gamma}_1 > \epsilon)$ if

$$N < \frac{-2 \log K}{h(\rho)}. \tag{4.10}$$

This preliminary analysis shows that the probability for learning an aggregated feature with high correlation (e.g., a set of neighbouring pixels that are unique to one class) is higher than that for learning an individual feature with equally high correlation (with the label). On the other hand, for aggregated features that are moderately correlated with the label (e.g., a pixel unique to a class surrounded by noises), there is an upper bound on the data size for which incorporating transformation in adversarial training will gain an advantage.

Empirical examination Here we formulate a binary classification problem using digits "5" and "7" from MNIST. We use a cropping size of 26 and a model that aggregates 9 cropped inputs. Correlation values for all features with and without cropping are computed (Fig. 4.8a). With $\epsilon = 0.2$ and all training data (10k in total), the robust features are identified for a linear model defined on the original images and a separate linear model defined on the features aggregated from cropped images. The resultant categorization of robust and non-robust features is considered as the ground truth.

We then perform the same computation on random batches of the data (400 for each), mimicking adversarial training under small dataset. For each batch, the learned robust features are recorded. By combining results from 200 random batches, we compute the percentage of batches that each feature is being correctly classified as robust or non-robust. A comparison is shown in Fig. 4.8b. The model with input cropping achieves higher success rate at learning robust features and dropping non-robust ones: Without cropping, there are 16 robust features not always regarded as robust, and 32 non-robust features regarded as robust in some cases. With cropping, these numbers reduce to 9 and 10, respectively.

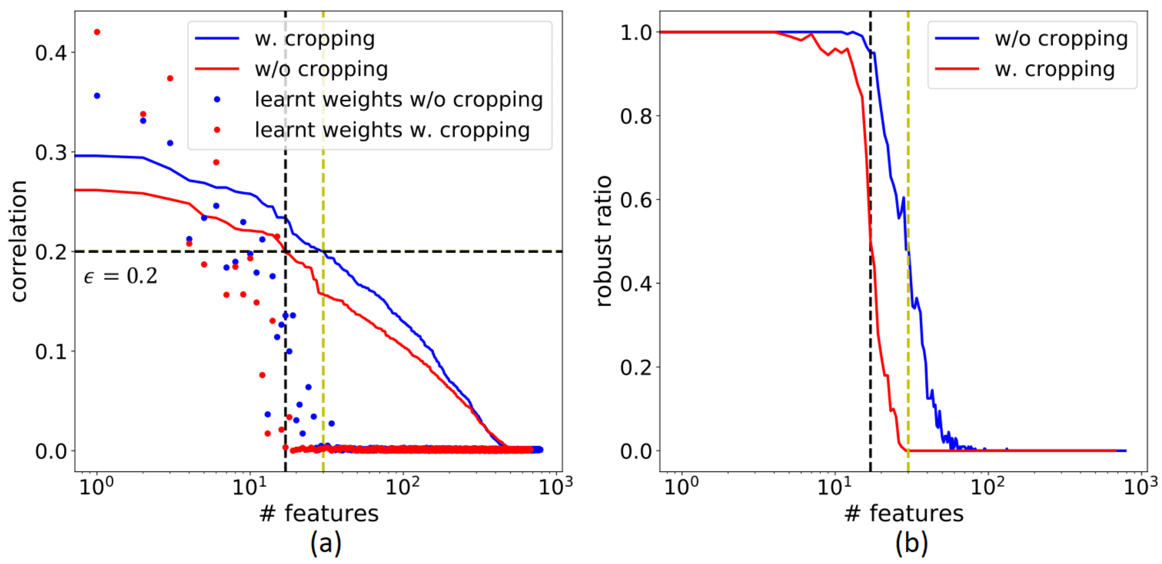


Figure 4.8: (a) Correlation of each feature to labels w/ and w/o cropping aggregation. Dots are classifier weights derived from adversarial training based on Eq. (4.5). (b) The ratio that a feature is regarded as robust across 200 random batches. Dashed lines in black and yellow separate robust and non-robust features for models w/ and w/o cropping aggregation, respectively. The ideal ratio curve would be constantly 1 to the left of the dashed line and 0 to the right.

Network Visualization Lastly, we qualitatively evaluate the efficacy of our model at learning robust features, by considering such features as human-interpretable image patterns. We first visualize the adversarial gradients from P_{nat} and P_{adv} in Fig. 4.9. The result shows that incorporating cropping invariance alone does not yield meaningful adversarial gradients. However, combined with adversarial training, the adversarial gradients become interpretable.

We further investigate network filters from MNIST. Consistent with (Madry *et al.*, 2017), the first-layer filters for baseline model are sparse. In comparison, our model increases the number of non-zero filters from 3 to 22. Considering that these filters serve as denoisers with learned thresholds, more filters would allow richer information to be passed to the next layer of the network. We also compare the second-layer filters from baseline and our model in Fig. 4.10. Our model is able to produce more human-interpretable strokes, which may explain its improvement in robustness.

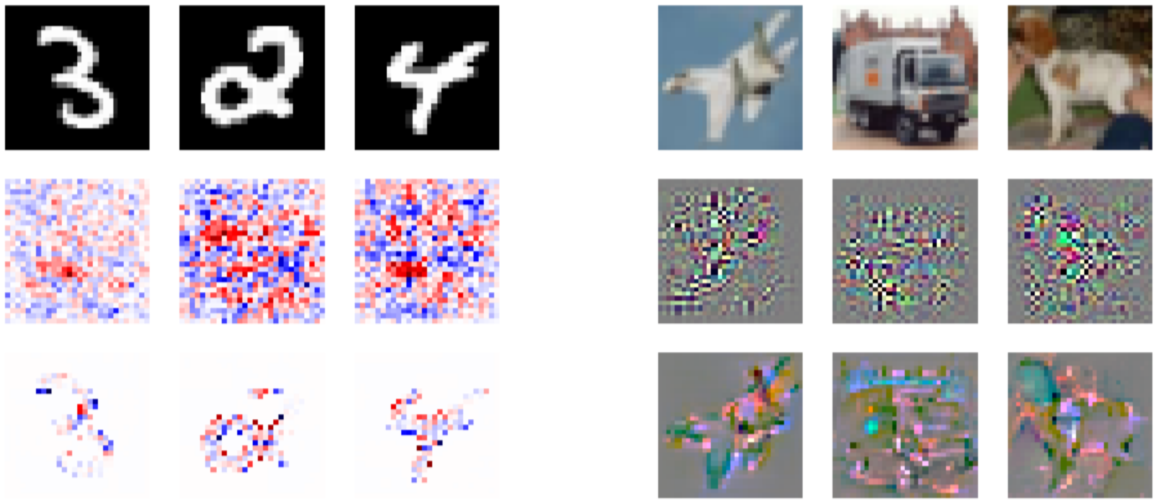


Figure 4.9: Visualization of adversarial gradients. Images from top to bottom are random clean samples, adversarial gradients from P_{nat}, and those from P_{adv}.

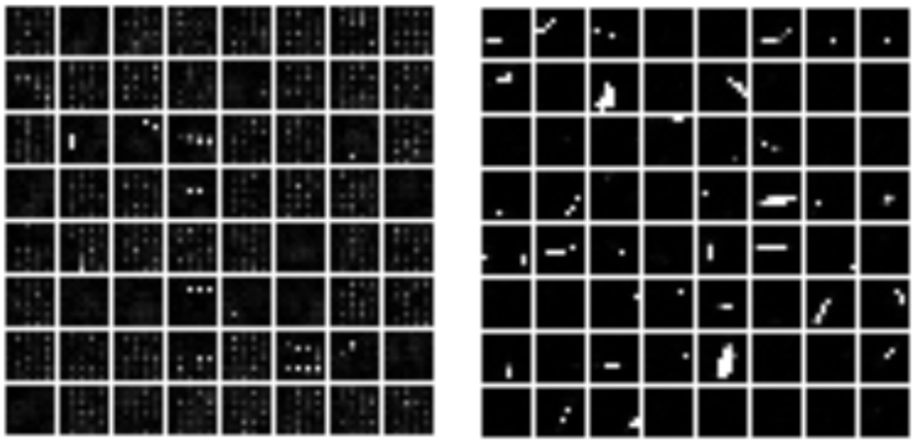


Figure 4.10: Visualization of the second-layer filters from networks trained on MNIST (left: baseline, right: proposed). The proposed method is able to learn more meaningful strokes.

4.4 Related Work

Random input transformation Guo et al. (Guo *et al.*, 2017) proposed using random transformations to pre-processing the input images to improve model robustness. It was later shown, however, that this approach creates a gradient masking effect and can be broken by robust attacks (Athalye *et al.*, 2018). Unlike (Guo *et al.*, 2017), we consider the transformation as part of our model during the adversarial training process.

Bag of features Studies on bag of features (BoF) (Csurka *et al.*, 2004; Jurie and Triggs, 2005; Zhang *et al.*, 2007; Brendel and Bethge, 2018) proposed the aggregation of clusters of local image features for classification. While our model also takes in multiple inputs as in BoF models, we assume that these images are *all* content preserving, and do not rely on an aggregation for classification.

Ensemble adversarial training Attacks from an ensemble of black-box models have been used to effectively avoid gradient masking in one-step adversarial training (Tramèr *et al.*, 2017). While our model also uses an ensemble of attacks, these attacks are white-box and multi-step. Importantly, these attacks do not cause gradient masking.

4.5 Limitations and Future work

Limitations *Computational cost*: The proposed model requires higher computational cost due to the computation of transformation-invariant attacks. As the effectiveness of the proposed method is positively correlated with the number of transformations (shown experimentally in Tab. 4.3 and theoretically in Sec. 4.3.3 for image cropping), the success of the method depends on the availability of parallel GPUs for computing attack gradients. Due to this limitation, our experiments on the restricted ImageNet dataset (Tsipras *et al.*, 2018) have so far achieved limited success. Specifically, with cropping size 168 and four crops, the proposed model achieves 92.83% and 96.93% for robustness and standard accuracy, which are comparable to 92.75% and 96.83% for baseline model. We hypothesize that increasing the cropping size and the number of crops will further improve the model robustness.

Verification : Existing formal verification (Katz *et al.*, 2017) and certification (Xu *et al.*, 2009; Wong and Kolter, 2018; Wong *et al.*, 2018) methods rely on the linearity of the decision function. Our model aggregates softmax outputs (Eq. (4.1)), which makes existing tools not directly applicable. It will be necessary to investigate whether the aggregation can be performed before softmax.

4.6 Conclusions

In this chapter we investigated a learning architecture that incorporates input transformations into adversarial training, and showed that the model (1) achieves better empirical robustness than the state of the art on MNIST and CIFAR-10, (2) is more data efficient, and (3) is more effective at extracting robust features, by using image cropping as an ensemble of transformations. Importantly, we showed that while constraining the model to be transformation invariant (through data augmentation) does not help improve model robustness, incorporating transformation-invariant attacks in training plays a critical role in achieving this goal. The proposed model is computationally more costly than standard adversarial training. The generalization of the success from cropping to other transformations is currently limited, potentially due to the limited capacities of the tested models.

Chapter 5

PHYSICS-GUIDED DATA-DRIVEN MODEL FOR EFFICIENT MECHANICAL RESPONSE PREDICTION

5.1 Introduction

Machine learning has achieved great success in computer vision (Krizhevsky *et al.*, 2012), speech recognition(Amodei *et al.*, 2016), natural language processing (Devlin *et al.*, 2018), and control (Silver *et al.*, 2017) with huge amount of training samples. Inspired by these success, deep neural networks have also been widely used to solve problems in physics recently (Sheikholeslami *et al.*, 2019; Tompson *et al.*, 2016; Chu and Thuerey, 2017; Wang *et al.*, 2019; Finol *et al.*, 2018; Sosnovik and Oseledets, 2017b; Cang *et al.*, 2018b; Bouman *et al.*, 2013; Li *et al.*, 2019; Bessa *et al.*, 2017; Cang *et al.*, 2018a; Zhao *et al.*, 2019; Yao *et al.*, 2019). However, deep neural networks suffer from poor generalizability when the training data is limited. In the meantime, large dataset can be difficult or expensive to obtain for many engineering applications (Cang *et al.*, 2018a). Our main objective of this chapter is to accurately predict the response of a physics system based on limited observed data. In this chapter, we designed convolutional neural networks to model the physics from response to loading and vise versa with the prior knowledge in physics.

Unlike previous approaches, an integration between purely data-based and physics-based model is considered in this chapter. As illustrated in Fig. 1.5, our prior knowledge in physics principles is used to guide the designing of data-driven model structures, and the problems on both ends can be mitigated in this way. Specifically, we use the prior knowledge in FEA and its numerical solvers to design the structure of the proposed FEA-Net: (1) Inspired by FEA and its local support property, we first design FEA convolution operation that is physically meaningfully. (2) Based on FEA convolution, we design a framework for learning and inference for higher efficiency and accuracy.

Different network structures are designed to maximize the performance of both training and inference: (1) The network is designed to reversely map the system response to its corresponding loading at the training stage. This mapping is extremely simple, which makes the network training very computationally efficient. It can be very accurate at the same time, which enables the learning to perform well on even a single training data point, i.e. to perform one-shot learning. Depending on the extent of the physics prior, either network filters or physics parameters can be learned. (2) We construct another network to map the system loading to response by re-using the convolution operations from the training stage. The structure of the inference network is backed up by fix-point iterative solvers and thus have theoretical convergence w.r.t network depth.

As a summary, FEA-Net has the following advantages over traditional purely data-driven models:

- The filters of FEA-Net are physically meaningful. Thus, FEA-Net is interpretable and we can infer the physics knowledge from the trained network.
- FEA-Net is more generalizable, and can perform single-shot learning with only a single image pair.
- FEA-Net has guaranteed prediction accuracy convergence w.r.t. network depth.
- Furthermore, the proposed method is a general framework and more advanced network structures like multi-grid can be added easily.

And when compared with purely model-based approach, it has the following advantages:

- FEA-Net doesn't require the physics model.
- FEA-Net consumes far less memory, and can be easier to be parallelized on GPU devices.
- FEA-Net is much easier to extend to other problems. For example, the same architecture can be easily adapted to solve problems with different physics field and spatial dimensions.

An outline of the paper is as follows: Sec.5.2 formulates the problem and reviews some of the related work on both deep learning and computational mechanics side. In Sec.5.3, we introduce the FEA convolution and extend it to handle multi-physics and multi-phase problems. Based on FEA convolution, we design FEA-Net structure for multi-physics (homogeneous thermoelasticity) and multi-phase (bi-phase elasticity) in Sec.5.4. Numerical examples are given in Sec.5.5. Sec.5.6 concludes the paper.

5.2 Preliminaries and Background

5.2.1 Problem formulation

To start with, we make the following assumptions: (1) All observations are in image form. This assumption makes the use of convolutional neural networks as data-driven model possible. (2) Consider linear physics only. We start with simpler linear physics first, as it is easier to prove the convergence of the proposed algorithm. Similar idea is also adopted in Kawaguchi (2016); Hsieh *et al.* (2018). Future work will extend this framework to non-linear physics.

We state our general goal as: *Build a model to predict the system response when*

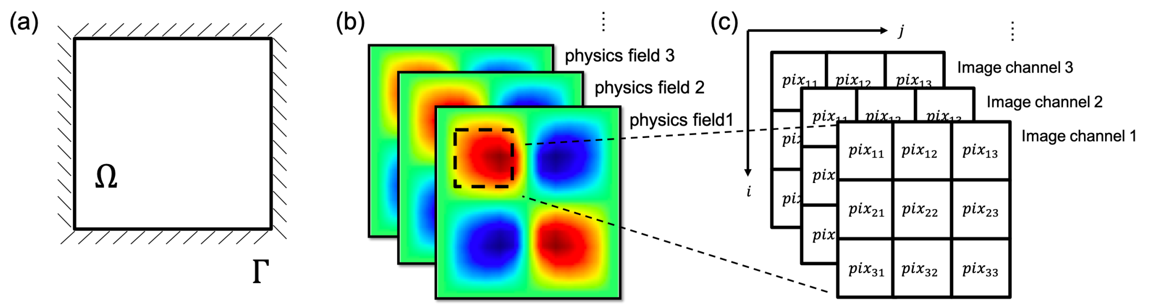


Figure 5.1: (a) Solution domain Ω with boundary condition. (b) Load/ response heatmap can be viewed as images. (c) Image pixels and their coordinate system.

a new loading image is given, given a dataset $\mathcal{D}$ that contains observed loading image $V \in R^{(N,N,P)}$ and response image $U \in R^{(N,N,Q)}$ pair. N is the spatial resolution of the images, P and Q are the number of input and output channels for loading and response images respectively. More specific problem definitions will be given in Sec. 5.4.1 and Sec. 5.4.2.

In the rest of this paper, we assume that the solution domain Ω is 2D and square-shaped as depicted in Fig. 5.2.1a. There can be multiple different physics field in Ω, which can be expressed as several different heatmaps (Fig. 5.2.1b). And these heatmaps can be viewed as a multiple channel image, as shown in Fig. 5.2.1c. For example, we have three channels for thermoelasticity problems: x- and y- directional displacement (or force) and temperature (or heat flux).

5.2.2 Neural networks

A L-layer (fully connected) neural network is a function

$$y(x|W) = f^L(W^L, ...f^2(W^2, f^1(W^1, x))) \tag{5.1}$$

with parameters $W = \{W^1, W^2...W^L\}$ and input x. The function f is called activation function, which acts on all components of the input vector. During the training phase, the network weights are determined by minimizing the difference between network output and observations. It is found that, given enough nodes, neural networks with non-linear activation function have the potential to approximate any complicated functions (Hornik, 1991b). However, how to design the network to be more efficient for different problems is always an open question.

A plethora of research has been done to design more efficient and effective neural networks among deep learning and computer vision communities. Some of the biggest breakthroughs can be summarized as: (1) Replacing some fully connected layers with convolutions (LeCun *et al.*, 1989) (as shown in Fig. 5.2a). In this way, Convolutional Neural Network (CNN) mimics the human visual system and captures the spatial correlations better. It has shown to be very suitable for various vision-based tasks like object recognition (LeCun *et al.*, 1989; Krizhevsky *et al.*, 2012), detection (Girshick *et al.*, 2014; Ren *et al.*, 2015), generation (**?**Isola *et al.*, 2017), and segmentation (Long *et al.*, 2015; Ronneberger *et al.*, 2015). (2) The invention of residual networks (ResNet) (He *et al.*, 2016). Through the short-cut residual connections, Res-Net style network can avoid the notorious "gradient vanishing" problem and make the training of network with thousands of layers possible. Compared with previous neural networks, ResNet and its various extensions (Huang *et al.*, 2017a; Xie *et al.*, 2017b) can almost always achieve better convergence and higher accuracy. (3) The development of one-shot learning algorithms (Fei-Fei *et al.*, 2006; Santoro *et al.*, 2016). Based on either Bayesian theory (Fei-Fei *et al.*, 2006) or external network memory and attention mechanism (Santoro *et al.*, 2016), these models can be very data efficient and partially mitigates the need for big data for network training.

Shown in Fig. 5.2b, Fully Convolutional Network (FCN) is a special type of CNN that only contains convolutional operations (Long *et al.*, 2015). It takes in images as input and outputs another image of the same resolution with per-pixel label. Since only convolution operation is involved, FCN is very computationally efficient and can handle inputs of arbitrary size. FCN is designed to perform pixel-wise labeling,

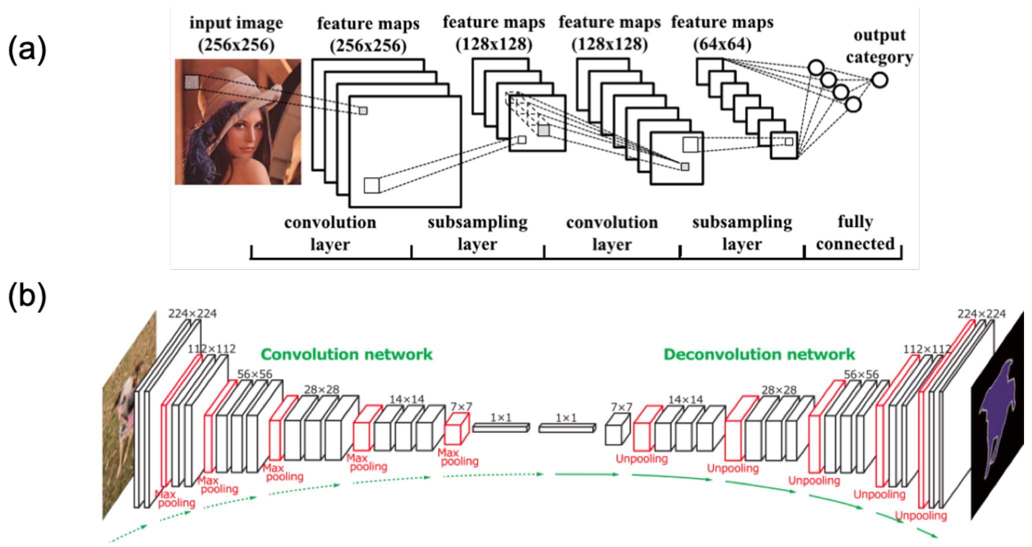


Figure 5.2: (a): CNN, (b): FCN (figure adopted from Ren *et al.* (2015))

and has shown to be very powerful in semantic segmentation (Long *et al.*, 2015; Ronneberger *et al.*, 2015).

5.2.3 Finite Element Analysis

As a quick review, the core idea behind Finite Element Analysis (FEA) is to approximate the potential field with piece-wise lower-order functions (Hughes, 2012). In practice, it involves discretizing the solution domain with smaller meshes, which transforms the original PDE into a system of linear equations:

$$K \cdot u = v \tag{5.2}$$

where v and u are the vectors of system loading and response on the FEA nodes, and K is the global stiffness matrix which is obtained by assembling all individual element stiffness matrices K^e:

$$K^e = \int_{\Delta} B^T C B d\Omega \tag{5.3}$$

where C is the constitutional matrix depends on the material property, B is the geometry matrix decided by the element shape and order, and Δ is the finite element.

While numerous numerical solvers exist for solving Eq. (5.2), most of them involves iteratively computing the residual (Yang and Mittal, 2014):

$$r = v - K \cdot u \tag{5.4}$$

As an example, the simplest Jacobi solver has the following form:

$$u_{t+1} = \omega D^{-1} \cdot r_t + u_t \tag{5.5}$$

where ω is a hyper-parameter, and D is the diagonal part of matrix K.

5.3 FEA Convolution

This section is organized into the following parts: We start by introducing the FEA convolution to model PDE for homogeneous material, and generalize it to handle multi-physics problems in Sec. 5.3.1. Proposed FEA convolution is then further extended to multi-phase problems in Sec. 5.3.2. How the gradient of FEA convolution can be obtained is discussed in Sec. 5.3.3.

5.3.1 FEA convolution for multi-physics problem

For physics process, its system loading and response need to satisfy some underlying PDE. Based on finite element analysis, there exists a "local support property": The loading at any node is related to only the response at its surrounding nodes. Thus, in image space, any pixel value in image V is only related to the pixel values in U at its neighboring region. This relationship is formalized into the following theorem:

Theorem 1. *The mapping from system response image $U \in R^{(N,N,Q)}$ to system loading image $V \in R^{(N,N,P)}$ can be modeled with a convolution operation for homogeneous material:*

$$V = W \circledast U \tag{5.6}$$

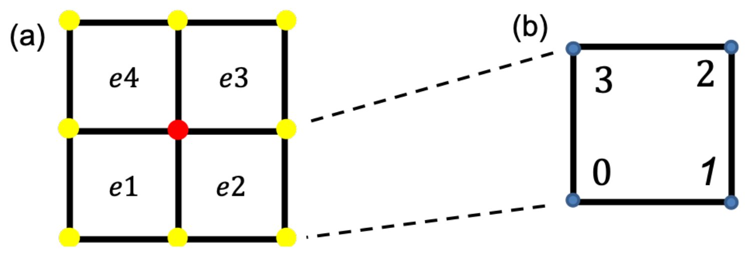


Figure 5.3: (a) The response of $node(i, j)$ (plotted in red) is affected by the loading on the nodes of the four surrounding elements only. (b) Node numbering convention for a element.

where $\circledast$ and $W \in R^{(R,R,P,Q)}$ denotes the convolution operator and filter, and R can be any odd number larger than 3.

Proof. We give the proof with single input and output component ($P = Q = 1$), which can be extended to other cases naturally. Under FEA perspective, the relationship between U and V can be defined by Eq. (5.2), with element stiffness matrix defined in Eq. (5.3). It is worth noting that, if the physics problem is unchanged and the material is homogeneous everywhere, then its constitutional matrix C will be the same for all elements. Furthermore, if the mesh is uniform and of the same order, then the shape matrix N will be the same as well. Under these hypotheses, the element stiffness matrices K^e will be the same everywhere.

As an example, we use the simplest 4 node linear element to discretize the underlying PDE in this paper. Following the numbering convention in Fig. 5.3, such discretization will lead us to a system of linear equations:

$$\begin{aligned} V_{ij} = {} & K^{e1}_{13} \cdot U_{i+1,j-1} + K^{e1}_{23} \cdot U_{i+1,j} + K^{e1}_{33} \cdot U_{i,j} + K^{e1}_{43} \cdot U_{i,j-1} \\ & + K^{e2}_{14} \cdot U_{i+1,j} + K^{e2}_{24} \cdot U_{i+1,j+1} + K^{e2}_{34} \cdot U_{i,j+1} + K^{e2}_{44} \cdot U_{i,j} \\ & + K^{e3}_{11} \cdot U_{i,j} + K^{e3}_{21} \cdot U_{i,j+1} + K^{e3}_{31} \cdot U_{i-1,j+1} + K^{e3}_{41} \cdot U_{i-1,j} \\ & + K^{e4}_{12} \cdot U_{i,j-1} + K^{e4}_{11} \cdot U_{i,j} + K^{e4}_{32} \cdot U_{i-1,j} + K^{e4}_{42} \cdot U_{i-1,j-1} \end{aligned} \tag{5.7}$$

where e1 to e4 denotes the four neighbouring elements of $node(i, j)$ as Fig. 5.3a shows. The subscript of the element stiffness matrix goes from 1 to 4, which corresponds to the node index inside a particular element as Fig. 5.3b shows.

By comparing Eq. (5.6) and Eq. (5.7), we can obtain a 3-by-3 FEA convolution kernel W explicitly:

$$W = \begin{bmatrix} K^e_{42} & K^e_{32} + K^e_{41} & K^e_{31} \\ K^e_{43} + K^e_{12} & K^e_{11} + K^e_{22} + K^e_{33} + K^e_{44} & K^e_{34} + K^e_{21} \\ K^e_{13} & K^e_{23} + K^e_{14} & K^e_{24} \end{bmatrix} \tag{5.8}$$

Moreover, based on the formulation of higher-order elements, we can easily obtain the analytical expression of larger FEA convolution kernels. For example, a 5-by-5 kernel can be obtained with second-order elements, and a 7-by-7 kernel can be obtained with third-order elements. □

In the rest of this paper, we assume that the filters are obtained from linear element and with spatial size 3 by 3. The terms in W all have their physics meaning. For example, W_{22} (and W_{11}) represents the loading at a particular pixel, when there is only a unit response at that pixel itself exist (or the upper left of that pixel). The convolution kernel can be obtained in a closed-form if the physics is perfectly known, otherwise we will need to learn it from data. In Sec. 5.5, we will verify the proposed kernel and learning approach by comparing the learned filter with its analytical value. Below we give examples of the analytical FEA convolution kernel for some known physics problems.

Example 5.3.1. The expression of W for thermal conduction problems is:

$$W^{tt} = \frac{\kappa}{3} \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{5.9}$$

where κ is the thermal conductivity coefficient.

The derivation of this kernel is in Appendix A.1. It is interesting to note that W^{tt} is actually a Laplacian filter. The reason for this is that the governing equation for thermal c onduction is Laplacian (Poisson) equation.

Example 5.3.2. The convolution for elasticity has two input and output channels, representing the x- and y- directional components of loading and response respectively. The interactions from the same input and output channels are:

$$\left(W^{xx}\right)^T = W^{yy} = \frac{E}{4(1-\nu^2)} \begin{bmatrix} -(1-\nu/3) & -2(1+\nu/3) & -(1-\nu/3) \\ 4\nu/3 & 8(1-\nu/3) & 4\nu/3 \\ -(1-\nu/3) & -2(1+\nu/3) & -(1-\nu/3) \end{bmatrix} \tag{5.10a}$$

where E and ν are Young's modulus an Poisson's ratio respectively. And the coupling terms between the two channels are:

$$W^{xy} = W^{yx} = \frac{E}{2(1-\nu)} \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix} \tag{5.10b}$$

The derivation of these kernels is included in Appendix A.2. Again, FEA convolutional kernel for elasticity also has many interesting properties: The non-coupling terms are symmetric along both axis, and they are just rotated versions of each other. The coupling terms are diagonally symmetric with many zeros, because axial loading does not cause any shear effects for homogeneous material.

Example 5.3.3. The analytical FEA convolution kernel for the coupling effect between thermal and elasticity is:

$$\begin{aligned} -(W^{xt})^T = W^{yt} &= \frac{\alpha E}{6(1-\nu)} \begin{bmatrix} 1 & 4 & 1 \\ 0 & 0 & 0 \\ -1 & -4 & -1 \end{bmatrix} \\ W^{tx} = W^{ty} &= 0 \end{aligned} \tag{5.11}$$

where α is the thermal expansion coefficient.

The derivation of this kernel is included in Appendix A.3. Since the coupling between thermal and elasticity is a one-way coupling, W^{tx} and W^{ty} are all zeros.

Example 5.3.4. The dynamics of thermoelasticity can be expressed into a convolution form of:

$$\begin{bmatrix} V^x \\ V^y \\ V^t \end{bmatrix} = \begin{bmatrix} W^{xx} & W^{xy} & W^{xt} \\ W^{yx} & W^{yy} & W^{yt} \\ W^{tx} & W^{ty} & W^{tt} \end{bmatrix} \circledast \begin{bmatrix} U^x \\ U^y \\ U^t \end{bmatrix} \tag{5.12}$$

This can be obtained by simply combining Example 5.3.1 to Example 5.3.3 together. Both input and output image has three channels, which corresponds to x- and y-directional mechanical component and thermal component. And the overall filter for the coupling field is a tensor with dimension $R^{(3,3,3,3)}$.

5.3.2 FEA convolution for multi-phase problem

Now we demonstrate how this idea can be extended to multi-phase problems. Without loss of generality, we use bi-phase elasticity as an example. A binary-valued image H is first introduced to represent the material phase. The pixel value of H represents which material phase exists at the specific spatial location (element). The resolution of H is set to $N-1$ by $N-1$, as the number of elements is one less than the number of nodes in FEA with linear element.

For bi-phase elasticity, the loading image $V \in R^{(N,N,2)}$ would be related to both the response image U and the phase image H. We define FEA convolution for bi-phase material as:

$$V = \Theta(\rho) \otimes (U, H) \tag{5.13}$$

where $\otimes$ and $\Theta \in R^{(2,P,Q,S,S)}$ are the FEA convolution operator and kernel for bi-phase elasticity. For 2D elasticity, we have $P = Q = 2$, which represents the x- and y- component. And for linear element we have $S = 4$, which represents there are four nodes in each element. And Θ is further assumed to be a depended on some physics hyper-parameters, e.g. $\rho = \{E, \nu\}$ for elasticity.

Following the numbering convention in Fig. 5.3, with four-node linear finite element of the same size, the relationship between phase, response, and loading images can be obtained with FEA as:

$$\begin{aligned} V_{ij}^{q} = \sum_{h=0}^{1} \Big(& C_{hijp}^{e1} \cdot (h + (-1)^h H_{i,j-1}) + C_{hijp}^{e2} \cdot (h + (-1)^h H_{i,j}) \\ & + C_{hijp}^{e3} \cdot (h + (-1)^h H_{i-1,j}) + C_{hijp}^{e4} \cdot (h + (-1)^h H_{i-1,j-1}) \Big) \end{aligned} \tag{5.14}$$

where $e1$ to $e4$ still denotes the four neighbouring elements of $node(i,j)$ as Fig. 5.3a shows. h is either 0 or 1, representing which material phase is under consideration. And C_{hijp} is obtained from U and W as:

$$C_{hijp}^{e1} = \sum_{q=1}^{Q} \Big(W_{31}^{hpq} \cdot U_{i+1,j-1}^{q} + W_{32}^{hpq} \cdot U_{i+1,j}^{q} + W_{33}^{hpq} \cdot U_{i,j}^{q} + W_{34}^{hpq} \cdot U_{i,j-1}^{q} \Big) \tag{5.15a}$$

$$C_{hijp}^{e2} = \sum_{q=1}^{Q} \Big(W_{41}^{hpq} \cdot U_{i+1,j}^{q} + W_{42}^{hpq} \cdot U_{i+1,j+1}^{q} + W_{43}^{hpq} \cdot U_{i,j+1}^{q} + W_{44}^{hpq} \cdot U_{i,j}^{q} \Big) \tag{5.15b}$$

$$C_{hijp}^{e3} = \sum_{q=1}^{Q} \Big(W_{11}^{hpq} \cdot U_{i,j}^{q} + W_{12}^{hpq} \cdot U_{i,j+1}^{q} + W_{13}^{hpq} \cdot U_{i-1,j+1}^{q} + W_{14}^{hpq} \cdot U_{i-1,j}^{q} \Big) \tag{5.15c}$$

$$C_{hijp}^{e4} = \sum_{q=1}^{Q} \Big(W_{21}^{hpq} \cdot U_{i,j-1}^{q} + W_{22}^{hpq} \cdot U_{i,j}^{q} + W_{23}^{hpq} \cdot U_{i-1,j}^{q} + W_{24}^{hpq} \cdot U_{i-1,j-1}^{q} \Big) \tag{5.15d}$$

The FEA convolution for bi-phase material is illustrated in Fig. 5.4. The filter Θ is applied to both a 2-by-2 region in H and a 3-by-3 region in U at the same location. Similar to conventional convolution, the filter will be shifted by one pixel every time and applied to different regions of the images. For homogeneous material, Eq. (5.14) will be reduced to Eq. (5.6) by setting $H \equiv 1$.

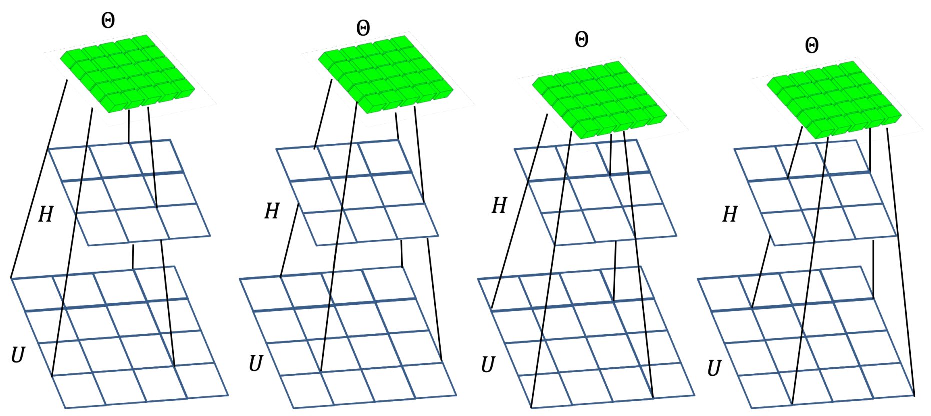


Figure 5.4: Illustration of the Bi-phase FEA Convolution. Both H and U is involved in FEA convolution. While Θ is fixed, H and U involved in the computation will be shifted by 1 pixel each time.

Example 5.3.5. The bi-phase FEA convolutional kernel Θ for elasticity can be obtained by splitting the element stiffness matrix (which can be found in Eq. (A.12) in the Appendix):

$$\Theta^{h00} = \frac{E_h}{12(1-\nu_h^2)} \begin{bmatrix} -2\nu_h+6 & -\nu_h-1 & \nu_h-1 & 2\nu_h \\ -\nu_h-1 & -2\nu_h+6 & 2\nu_h & \nu_h-1 \\ \nu_h-1 & 2\nu_h & -2\nu_h+6 & -\nu_h-1 \\ 2\nu_h & \nu_h-1 & -\nu_h-1 & -2\nu_h+6 \end{bmatrix} \tag{5.16a}$$

$$\Theta^{h11} = \frac{E_h}{16(1-\nu_h^2)} \begin{bmatrix} -2\nu_h+6 & 2\nu_h & \nu_h-1 & -\nu_h-1 \\ 2\nu_h & -2\nu_h+6 & -\nu_h-1 & \nu_h-1 \\ \nu_h-1 & -\nu_h-1 & -2\nu_h+6 & 2\nu_h \\ -\nu_h-1 & \nu_h-1 & 2\nu_h & -2\nu_h+6 \end{bmatrix} \tag{5.16b}$$

where E_h and ν_h represents the Young's Modulus and Poisson ration for different material phases. And the coupling filters between different input and output channels are:

$$\Theta^{h01} = \frac{E_h}{8(1-\nu_h^2)} \begin{bmatrix} \nu_h+1 & 1-3\nu_h & -\nu_h-1 & 3\nu_h-1 \\ 3\nu_h-1 & -\nu_h-1 & 1-3\nu_h & \nu_h+1 \\ -\nu_h-1 & 3\nu_h-1 & \nu_h+1 & 1-3\nu_h \\ 1-3\nu_h & \nu_h+1 & 3\nu_h-1 & -\nu_h-1 \end{bmatrix} \tag{5.17a}$$

$$\Theta^{h01} = \frac{E_h}{8(1-\nu_h^2)} \begin{bmatrix} \nu_h+1 & 3\nu_h-1 & -\nu_h-1 & 1-3\nu_h \\ 1-3\nu_h & -\nu_h-1 & 3\nu_h-1 & \nu_h+1 \\ -\nu_h-1 & 1-3\nu_h & \nu_h+1 & 3\nu_h-1 \\ 3\nu_h-1 & \nu_h+1 & 1-3\nu_h & -\nu_h-1 \end{bmatrix} \tag{5.17b}$$

5.3.3 Gradient of FEA convolution

Since FEA convolution for thermoelasticity is actually a standard multi-channel convolution, standard deep learning packages like Tensorflow [1] can be directly used to obtain its gradient and to perform back-propagation (Rumelhart *et al.*, 1988). However, the gradient for bi-phase FEA convolution in Eq. (5.13) needs to be explicitly defined for efficient computation.

Since V is a function of U, E and Θ (or its physical parameter ρ) in bi-phase convolution, there will be three different partial derivatives w.r.t. V needs to be computed. Based on Eq. (5.14), the gradient of the output V w.r.t. input U can be derived as:

$$\begin{aligned} \frac{\partial V}{\partial U_{ij}^q} = \sum_{h=0}^{1} \Big(& \sum_{p=1}^{P} \frac{\partial C_{hijp}^{e1}}{\partial U_{ij}^q} \cdot (h + (-1)^h H_{i,j-1}) + \sum_{p=1}^{P} \frac{\partial C_{hijp}^{e2}}{\partial U_{ij}^q} \cdot (h + (-1)^h H_{i,j}) \\ & + \sum_{p=1}^{P} \frac{\partial C_{hijp}^{e3}}{\partial U_{ij}^q} \cdot (h + (-1)^h H_{i-1,j}) + \sum_{p=1}^{P} \frac{\partial C_{hijp}^{e4}}{\partial U_{ij}^q} \cdot (h + (-1)^h H_{i-1,j-1}) \Big) \end{aligned} \tag{5.18}$$

[1]https://www.tensorflow.org/

where:

$$\frac{\partial C^{e1}_{hijp}}{\partial U^q_{ij}} = \Theta^{hpq}_{31} \cdot \hat{V}^p_{i+1,j-1} + \Theta^{hpq}_{32} \cdot \hat{V}^p_{i+1,j} + \Theta^{hpq}_{33} \cdot \hat{V}^p_{i,j} + \Theta^{hpq}_{34} \cdot \hat{V}^p_{i,j-1} \tag{5.19a}$$

$$\frac{\partial C^{e2}_{hijp}}{\partial U^q_{ij}} = \Theta^{hpq}_{41} \cdot \hat{V}^p_{i+1,j} + \Theta^{hpq}_{42} \cdot \hat{V}^p_{i+1,j+1} + \Theta^{hpq}_{43} \cdot \hat{V}^p_{i,j+1} + \Theta^{hpq}_{44} \cdot \hat{V}^p_{i,j} \tag{5.19b}$$

$$\frac{\partial C^{e3}_{hijp}}{\partial U^q_{ij}} = \Theta^{hpq}_{11} \cdot \hat{V}^p_{i,j} + \Theta^{hpq}_{12} \cdot \hat{V}^p_{i,j+1} + \Theta^{hpq}_{13} \cdot \hat{V}^p_{i-1,j+1} + \Theta^{hpq}_{14} \cdot \hat{V}^p_{i-1,j} \tag{5.19c}$$

$$\frac{\partial C^{e4}_{hijp}}{\partial U^q_{ij}} = \Theta^{hpq}_{21} \cdot \hat{V}^p_{i,j-1} + \Theta^{hpq}_{22} \cdot \hat{V}^p_{i,j} + \Theta^{hpq}_{23} \cdot \hat{V}^p_{i-1,j} + \Theta^{hpq}_{24} \cdot \hat{V}^p_{i-1,j-1} \tag{5.19d}$$

And $\hat{V}$ is the gradient propagated to this FEA convolution during backpropagation, if there are several FEA convolutions stacked upon each other. The value of $\hat{V}$ is set to 1 if there is no external gradient passing in.

The gradient of the output V w.r.t. to the material phase H in Eq. (5.14) is relatively simpler:

$$\begin{aligned}\frac{\partial V}{\partial H_{ij}} = \sum_{h=0}^{1}\sum_{p=1}^{P}\Big(& C^{e1}_{hijp}(h-(-1)^h) \cdot \hat{V}^p_{i,j-1} + C^{e2}_{hijp}(h-(-1)^h) \cdot \hat{V}^p_{i,j} \\ & + C^{e3}_{hijp}(h-(-1)^h) \cdot \hat{V}^p_{i-1,j} + C^{e4}_{hijp}(h-(-1)^h) \cdot \hat{V}^p_{i-1,j-1}\Big)\end{aligned} \tag{5.20}$$

Recall that we have assumed that the physics interaction W is related to a set of hidden physics parameters $\rho \in R^Q$, then the gradient of the output F in Eq. (5.14) w.r.t. to ρ can be obtained as:

$$\begin{aligned}\frac{\partial V}{\partial \rho} = \sum_{h=0}^{1}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{p=1}^{P}\Big(& \frac{\partial C^{e1}_{hijp}}{\partial \rho} \cdot (h+(-1)^h H_{i,j-1}) + \frac{\partial C^{e2}_{hijp}}{\partial \rho} \cdot (h+(-1)^h H_{i,j}) \\ & + \frac{\partial C^{e3}_{hijp}}{\partial \rho} \cdot (h+(-1)^h H_{i-1,j}) + \frac{\partial C^{e4}_{hijp}}{\partial \rho} \cdot (h+(-1)^h H_{i-1,j-1})\Big)\end{aligned} \tag{5.21}$$

where:

$$\frac{\partial C^{e1}_{hijp}}{\partial \rho} = \sum_{p=1}^{P}\Big(\frac{\partial \Theta^{hpq}_{31}}{\partial \rho} \cdot \hat{V}^p_{i+1,j-1} + \frac{\partial \Theta^{hpq}_{32}}{\partial \rho} \cdot \hat{V}^p_{i+1,j} + \frac{\partial \Theta^{hpq}_{33}}{\partial \rho} \cdot \hat{V}^p_{i,j} + \frac{\partial \Theta^{hpq}_{34}}{\partial \rho} \cdot \hat{V}^p_{i,j-1}\Big) \tag{5.22a}$$

$$\frac{\partial C^{e2}_{hijp}}{\partial \rho} = \sum_{p=1}^{P} \Big(\frac{\partial \Theta^{hpq}_{41}}{\partial \rho} \cdot \hat{V}^p_{i+1,j} + \frac{\partial \Theta^{hpq}_{42}}{\partial \rho} \cdot \hat{V}^p_{i+1,j+1} + \frac{\partial \Theta^{hpq}_{43}}{\partial \rho} \cdot \hat{V}^p_{i,j+1} + \frac{\partial \Theta^{hpq}_{44}}{\partial \rho} \cdot \hat{V}^p_{i,j} \Big) \tag{5.22b}$$

$$\frac{\partial C^{e3}_{hijp}}{\partial \rho} = \sum_{p=1}^{P} \Big(\frac{\partial \Theta^{hpq}_{11}}{\partial \rho} \cdot \hat{V}^p_{i,j} + \frac{\partial \Theta^{hpq}_{12}}{\partial \rho} \cdot \hat{V}^p_{i,j+1} + \frac{\partial \Theta^{hpq}_{13}}{\partial \rho} \cdot \hat{V}^p_{i-1,j+1} + \frac{\partial \Theta^{hpq}_{14}}{\partial \rho} \cdot \hat{V}^p_{i-1,j} \Big) \tag{5.22c}$$

$$\frac{\partial C^{e4}_{hijp}}{\partial \rho} = \sum_{p=1}^{P} \Big(\frac{\partial \Theta^{hpq}_{21}}{\partial \rho} \cdot \hat{V}^p_{i,j-1} + \frac{\partial \Theta^{hpq}_{22}}{\partial \rho} \cdot \hat{V}^p_{i,j} + \frac{\partial \Theta^{hpq}_{23}}{\partial \rho} \cdot \hat{V}^p_{i-1,j} + \frac{\partial \Theta^{hpq}_{24}}{\partial \rho} \cdot \hat{V}^p_{i-1,j-1} \Big) \tag{5.22d}$$

If nothing is known about the underlying physics, we can just set ρ to Θ itself. If we have some prior knowledge on the underlying physics, for example we know it is an elasticity problem, then the computation of $\partial\Theta/\partial\rho$ can be obtained from Eq. (5.16) and Eq. (5.17). Furthermore, if we know the material is homogeneous, Eq. (5.18) to (5.21) can be largely simplified into the gradient of conventional 2D convolutions.

5.4 FEA-Net

This section is divided into to parts, where we build FEA-Net for multi-physics and multi-phase problems respectively. To maximize the efficiency, different network architectures are designed for learning and inference: We model the inverse mapping from system response to its corresponding loading during the training stage, and another network architecture is built to map the system loading to response during the inference stage.

5.4.1 FEA-Net for Multi-physics Problems

We use homogeneous thermoelasticity as an example to demonstrate how to design the learning and inference architecture for multi-physics based on the FEA convolution. For thermoelasticity, $V \in R^{(N,N,3)}$ (and $U \in R^{(N,N,3)}$) has three channels: x-

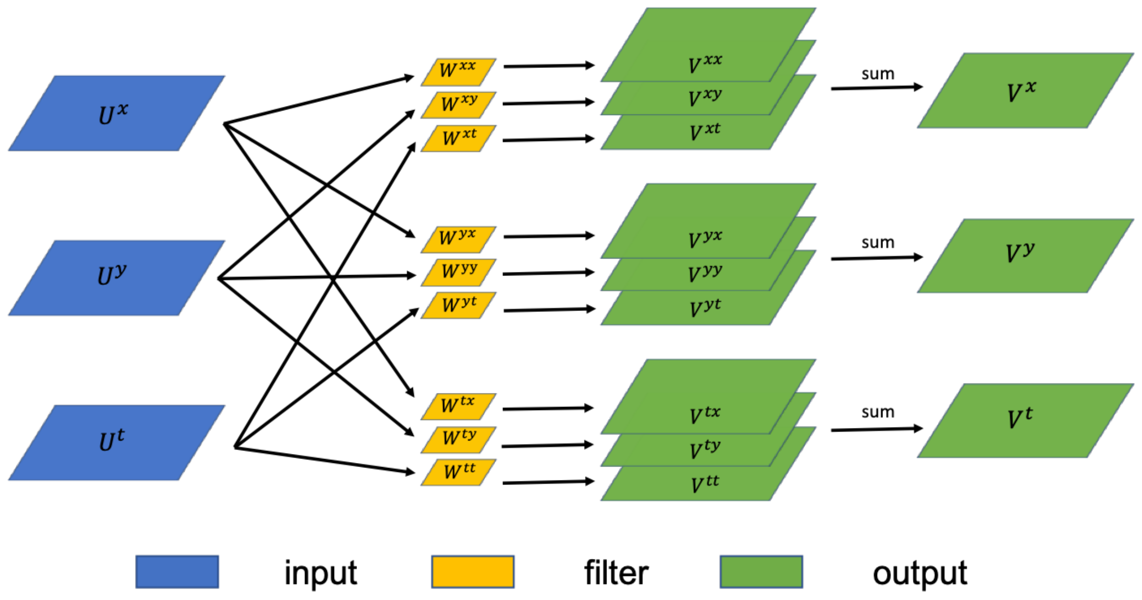


Figure 5.5: Learning Architecture for Homogeneous Thermoelasticity.

and y- directional mechanical loading (response) and heat flux (temperature). As defined in Eq. (5.12), the mapping from U to V is a convolutional operation:

$$V = W \circledast U := f_1(U, W) \tag{5.23}$$

with $W \in R^{(3,3,3,3)}$. This relationship can be further expressed with a single layer network with linear activation as illustrated in Fig. 5.5. The input and output to the network are the response image U and the predicted loading image V respectively.

Given a training dataset $\mathcal{D}$, the optimum filter W can be obtained by minimizing the difference between the observed system loading V and the predicted loading:

$$W^* = \arg\min_W \mathbb{E}_{(V,U)\sim\mathcal{D}} L\Big(V, \tilde{V}\Big) \tag{5.24}$$

where $\tilde{V} = f_1(U, W)$ is the network prediction, $L(\cdot,\cdot) : \mathbb{R}^{(N,N,3)} \times \mathbb{R}^{(N,N,3)} \to \mathbb{R}$ is a pre-defined loss function, which is chosen as the L_2 norm in this paper. FEA convolution will extract the information of the governing PDE during training.

Once FEA convolution has been trained, we can use it to construct the mapping from V to U and predict the system response when a new loading is applied. The core idea is to transform the iterative solvers (as in Eq. (5.4)) into a convolutional network based on the FEA convolution. We will demonstrate with the basic Jacobi solver (as in Eq. (5.5)) for its simplicity; however, it is worth noting that the proposed method can be applied with other more advanced iterative solvers as well.

Physically, the diagonal matrix D in Eq. (5.5) corresponds to the interaction between U^q_{ij} (response component q at $node(i,j)$) and V^q_{ij} (loading component q at $node(i,j)$). Under the FEA convolution perspective, this interaction can actually be expressed in the term W^{xx}_{22}, W^{yy}_{22}, or W^{tt}_{22} for thermoelasticity. Thus, the matrix-vector production $D^{-1} \cdot x$ can be reformulated into an element-wise production $P * X$, with $*$ denotes the element-wise operator.

We further define boundary condition operator $\mathcal{B}$, which specifies the Dirichlet boundary condition on Γ. What it does is to reset the value u on Γ to ground-truth. By substituting Eq. (5.6) into Eq. (5.5) and apply the boundary condition operator, we have:

$$U_{t+1} = \mathcal{B}\Big(\omega * P * (V - W \circledast U_t) + U^t\Big) \tag{5.25}$$

The details can be found in Appendix Appendix B. Because most of the computation of Eq. (5.25) lies in computing the FEA convolution, it can be viewed as stacking FEA convolutions upon each other. By setting the initial guess $U_0 = V$, Eq. (5.25) can be further visualized as a convolutional neural network (as in Fig. **??**).

The resultant network architecture is similar to both Fully Convolutional Network (FCN) (Long *et al.*, 2015) and the cutting-edge densely connected ResNet (Huang *et al.*, 2017a), as it is composed of only convolutions and has "short-cuts" across different layers. Similar to FCN, since no fully connected layer is involved, FEA-Net can handle inputs of different size without any problem. Most importantly, aside from the similarity on the surface, physics knowledge is inherently embedded in FEA-Net. Since FEA-Net is designed based on the fix-point iterative solver, so it has certifiable convergence w.r.t. network depth during inference.

Proposition 1. *The output of the inference network will converge to the ground-truth with increasing network depth, if the network filters have been learned accurately.*

Proof. See Appendix B for proof details. Numerical examples can be found in Sec. 5.5.3. □

Proposition 2. *The network filters can be learned accurately with a single image pair, if there is no linear correlation between different channels in the training loading images.*

Proof. See Appendix C for proof details. Numerical examples can be found in Sec. 5.5.1. □

Putting Proposition 1 and Proposition 2 together, it can be seen that our model is able to perform inference with certifiable convergence with a single training image pair.

5.4.2 FEA-Net for multi-phase problems

From Sec. 5.3.2, we know that the system loading image V, response image U, and phase image H should satisfy the following relationship:

$$V = \Theta(\rho) \otimes (U, H) := f_2(U, H, \rho) \tag{5.26}$$

where Θ is the FEA convolutional filter for bi-phase material, which is parametrized by the physics parameters ρ. Depending on the availability of the training dataset, three different learning problems can be formulated:

Problem 5.4.1. Assume that the material property ρ is known, we wish to learn the material phase image H based on the observed system loading and response pair (V, U).

This particular situation can happen when the material properties of each phase can be obtained from historical database or from experimental testing, such as indentation testing. The micro-structure information is unknown. This training process can be formulated into an optimization problem:

$$H^* = \arg\min_H \mathbb{E}_{(V_i,U_i,\rho_i)\sim\mathcal{D}_1} L\Big(V_i, f_2(\rho_i, H, U_i)\Big) \tag{5.27}$$

where $\mathcal{D}_1$ is the training set that contains the loading and response pair obtained with the same material phase.

Proposition 3. *The material phase H can also be correctly learned in any sub-region $\Phi \subset \Omega$, as long as the material property ρ is known and (V, U) image pair has been observed in Φ.*

Proof. See Appendix D.1 for proof details. Numerical examples can be found in Sec. 5.5.2. □

Problem 5.4.2. Assume the material phase information H is known, we wish to learn the material properties information ρ based on the observed system loading and response pair (V, U).

This particular situation can happen when the material micro-structure information is observed from measurements, such as optical imaging and scanning electron microscope imaging. However, the material properties of each phase are unknown. Such training process can be formulated as another optimization problem:

$$\rho^* = \arg\min_\rho \mathbb{E}_{(V_i,U_i,H_i)\sim\mathcal{D}_2} L\Big(V_i, f_2(\rho_i, H, U_i)\Big) \tag{5.28}$$

where $\mathcal{D}_2$ is a different training set, which contains the loading, response, and material phase pair obtained under the same material property.

Proposition 4. *Given material phase image H, only a single image pair (V, U) is needed to estimate the material property correctly on both phases, as long as: (1) H contains both phases, and (2) V has none-zero value*

Proof. See Appendix D.2 for proof details. Numerical examples can be found in Sec. 5.5.2. □

Problem 5.4.3. Assume that we know the loading and response (V, U), and we wish to estimate both material property and phase together.

The joint estimation of both material phase and property can be formulated as:

$$\rho*, H^* = \arg\min_{\rho, H} \mathbb{E}_{(V_i,U_i)\sim\mathcal{D}_3} L\Big(V_i, f_2(\rho, H, U_i)\Big) \tag{5.29}$$

This is a more difficult problem, and we will empirically show that it is also solvable under our framework.

Furthermore, we can put some constraints on the training process if we know which physics parameters are involved. As the simplest example, if we roughly know the range of the physics parameter ρ, we can perform projected gradient descent by:

$$\rho = clip(\rho, \rho_l, \rho_u) \tag{5.30}$$

where ρ_l and ρ_u are the lower and upper bound of ρ. If we have a better prior knowledge of the distribution of material property, we can have a tighter constraint to make the training even more efficient.

The inference network structure for multi-phase problems can also be obtained from Eq.(5.5). By subtracting the diagonal terms from Θ, the expression of P for bi-phase material can be obtained as:

$$\begin{aligned} P_{ijn} = 1/\sum_{h=0}^{1} \Big(&\Theta_{33}^{hpq} \cdot (h + (-1)^h H_{i,j-1}) + \Theta_{44}^{hpq} \cdot (h + (-1)^h H_{i,j}) \\ &+\Theta_{11}^{hpq} \cdot (h + (-1)^h H_{i-1,j}) + \Theta_{22}^{hpq} \cdot (h + (-1)^h H_{i-1,j-1})\Big) \end{aligned} \tag{5.31}$$

Similar to multi-physics problems, by substituting Eq. (5.13) and Eq. (5.31) into Eq. (5.5) we will have the convolutional form of the Jacobi solver:

$$U_{t+1} = \mathcal{B}\Big(\omega * P * \big(V - \Theta \otimes (U_t, H) + U^t\big)\Big) \tag{5.32}$$

As with Eq. (5.25), certifiable convergence w.r.t. network depth can also be obtained with Eq. (5.32).

5.5 Experiments and Results

This section is arranged as follows: In the first three parts, we verify our learning algorithm for different problems in multi-physics and multi-phase, as well as the convergence of our inference architecture. In the fourth and fifth part, our method is compared to purely data-based and physics-based model respectively.

5.5.1 Verification of learning on multi-physics

We verify if the proposed method can learn physically meaningful filters correctly for multi-physics problem in this subsection. The training data is generated with different material properties: We have Young's modulus E ranging from 0.1 TPa to 0.4 TPa, Poisson ratio ranging from 0.2 to 0.35, thermal conductivity from $10W/(m \cdot K)$ to $14W/(m \cdot K)$, thermal expansion ratio from $11/°C$ to $15/°C$. The material property used to generate training data is also used to obtain the reference filter value based on Eq. (5.9) to Eq. (5.12). The network is randomly initialized, and trained with a single image pair.

It is observed that the training of all networks is converging well, and the loss value is approaching zero with relative L_2 error less than $1e-5$. We show three examples of the network filters obtained with different training data in Tab. 5.1. The learned W_{11}^{xx}, W_{11}^{xy}, W_{11}^{tx}, W_{11}^{xt} is listed with their reference value. Still, the value of

Table 5.1: Examples of the learned filter elements with different training data.

Physical parameters				Filter	Reference	Predicted
$E(TPa)$	ν	$\kappa(W/(m\cdot K))$	$\alpha(10^{-5}/^{\circ}C)$			
0.23	0.289	11.82	12.92	W_{11}^{xx}	-56.7054195	-56.7053258
				W_{11}^{xy}	40.4512309	40.4511605
				W_{22}^{tx}	-0.159391382	-0.159391115
				W_{11}^{tt}	-4.56652389	-4.56652390
0.196	0.299	12.97	12.96	W_{11}^{xx}	-48.3777489	-48.3777004
				W_{11}^{xy}	34.8800133	34.8799752
				W_{11}^{tx}	-0.150795392	-0.150795228
				W_{11}^{tt}	-5.26968980	-5.26968981
0.228	0.273	12.92	11.82	W_{11}^{xx}	-55.8988893	-55.8988283
				W_{11}^{xy}	39.1483937	39.1483472
				W_{11}^{tx}	-0.168541618	-0.168541427
				W_{11}^{tt}	-5.91251479	-5.91251480

these filter elements is getting very close to the reference value.

Another comparison is made by fixing the material property and vary the initialization. The result is shown in Tab. 5.2. It can be seen that the reference filter has very nice symmetry property. More importantly, for all the random loading/ response data pair, the learned filter is matching with the reference value very well.

The previous two experiments verify the correctness of Proposition 2. In the third experiment, we test how the training algorithm performs if the premise is violated. The loading we applied to generate training data is shown in Fig. 5.6, where two loading channels can be correlated. This violates the premise of Proposition 2 (see

Table 5.2: Comparison on the reference value of the network filter W^{xx} with predictions on 100 random loading images.

Filter	Reference	Prediction	
		mean	std
W^{xx}_{11}	-52.2454463	-52.24507827	0.00147689
W^{xx}_{12}	22.19275595	22.19259812	0.00062905
W^{xx}_{13}	-52.2454463	-52.24509037	0.00144434
W^{xx}_{21}	-126.68364854	-126.68278956	0.00347672
W^{xx}_{22}	417.96357038	417.96069933	0.01163251
W^{xx}_{23}	-126.68364854	-126.68276882	0.00357171
W^{xx}_{31}	-52.2454463	-52.24508064	0.00149684
W^{xx}_{32}	22.19275595	22.19259831	0.0006441
W^{xx}_{33}	-52.2454463	-52.24508812	0.0014402

Appendix C for more details). Although it is difficult to visually tell that the response data has a linear correlation, it is in fact rank deficient. Our network is still able to minimize the loss on such data; however, the learned filter can be different every time depending on the initialization. An easy way to mitigate this problem to always apply random loading to obtain the system response. Another alternative solution is to add more prior knowledge to the network, for example, the relationship between network filters and physics parameters (as with solution to Problem1).

5.5.2 *Verification of learning on multi-phase*

In this part, we first verify that learning of material phase and property as well as inference works well under a wide range of different settings like learning rate, initialization, and problem complexity. The constrain in Eq. (5.30) is set to $E \in (0, 0.5)TPa$ and $\nu \in (0, 0.5)$. Note this is a very loose constrain that could be satisfied

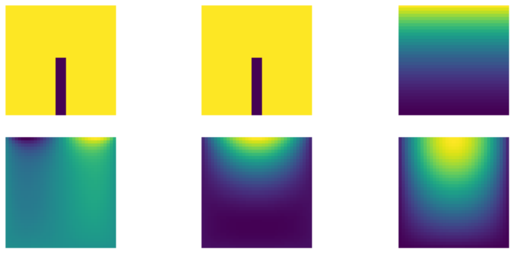


Figure 5.6: An example of the loading and response image pair that will fail the proposed training. First and second rows are loading response images. First to last columns are different channels for x-directional y-directional and thermal components.

by almost all known material. We define the relative estimation error of a variable x as:

$$\epsilon = \frac{|x^{pred} - x^{ref}|_2}{|x^{ref}|_2} \tag{5.33}$$

Problem1: Material phase estimation

We first show how our method performs in solving Problem 5.4.1 with different complexity of material phase. Circular inclusions of different size, shape, and location are used in this experiment, as shown in Fig. 5.7a. All phase images H in this experiment are at a resolution of 50 by 50. The inclusion has a Young's modulus of $0.241TPa$ and a Poisson ratio of 0.36. The second material has a Young's modulus of $0.2TPa$ and a Poisson ratio of 0.25. As discussed in Sec. 5.4.2, FEA-Net should be able to perform one-shot learning, so only a single loading/ response image pair is used for training in this experiment. The training data is generated with FEA.

We initialize the network filter Θ with ground truth material property ρ, while it's material phase H is initialized randomly. We use Adam optimizer (Kingma and Ba, 2014) with a learning rate of 10^{-2} to run Eq. (5.27). The convergence of the training process is visualized in Fig. 5.7b. For all phase configurations considered, starting from random initials, the estimated material phase is converging as the training iteration increases. The caption shows the relative phase estimation error ϵ, which is

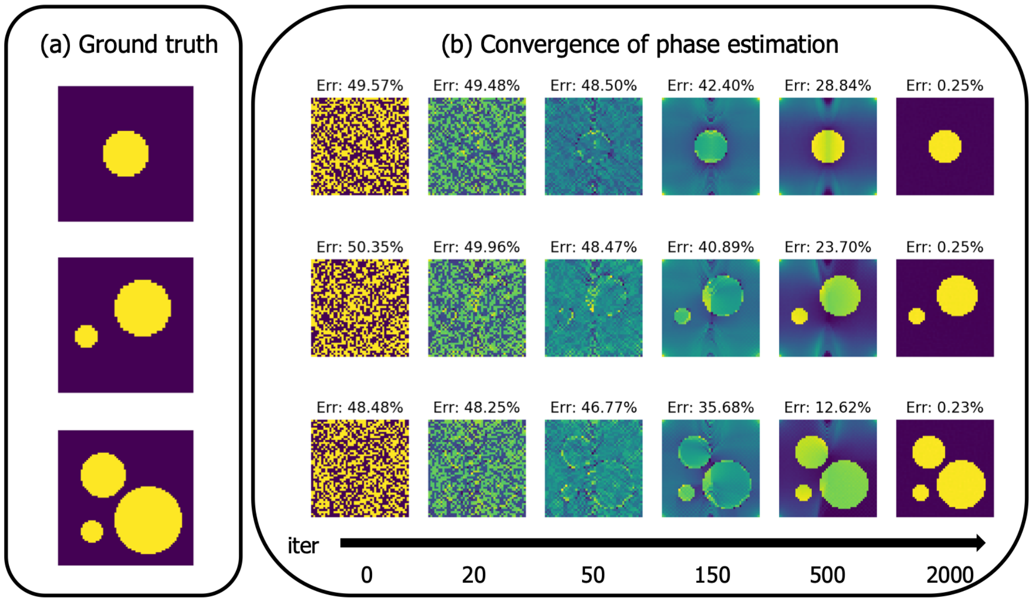
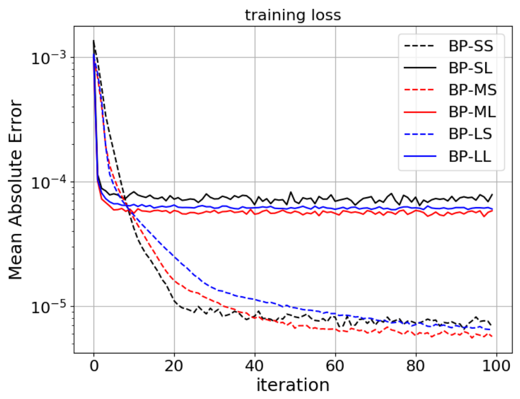


Figure 5.7: Learning convergence of material phase with different inclusions. (a): material phase ground truth (b): predicted phase at different iteration starting from random initialization.

dropping below 0.25% for all phase configurations.

A second experiment is designed to further evaluate the influence that image resolution and learning rate has on material phase estimation. The single inclusion material shown in Fig. 5.7 is used here, with image resolution goes from 25 to 50 and 75. The learning rate used is either 10^{-1} or 10^{-2}. We use "SS" to abbreviate small image resolution and small learning rate, "ML" to abbreviate medium image resolution and large learning rate, and so on.

We initialized the material phase image H randomly. The convergence of the network training loss and the prediction error in the material phase is shown in Fig. 5.8. It can be seen that the training is successful under all these settings. And a larger learning rate tends to increase the speed of convergence, with a side effect of a relatively larger error rate. This is because larger learning rate will make it harder to converge to the global optimum. Another observation is that, the material phase information is getting much harder to be estimated correctly when the resolution



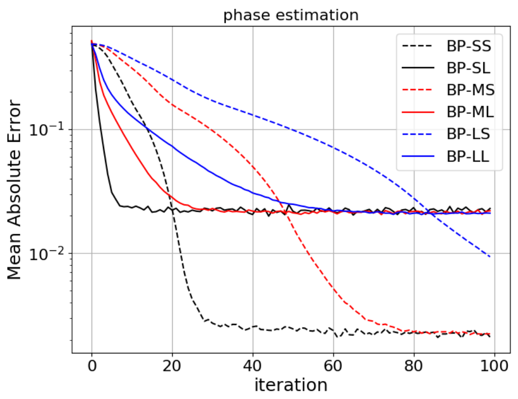


Figure 5.8: Material Phase Estimation with Different Resolution and Learning rate. (a): Convergence of the loss. (b): Convergence of the prediction error.

is increased. This is because the search space is getting significantly larger when we increase the image resolution, thus making the optimization problem much more difficult.

Problem2: Material property estimation

We first test how the learning of material property performs with random initialization. Again, only a single image pair is used for training. The data is generated with the single circular inclusion phase configuration (as shown by Fig. 5.7a) under a resolution of 50-by-50. The material properties are $0.241TPa, 0.36$ and $0.2TPa, 0.25$ for inclusion and exclusion material.

We initialize a total of 100 different network copies, with E ranges from 0 to 0.5 TPa and ν ranges from 0 to 0.5 for both phases. Adam optimizer with a learning rate of 10^{-3} is used to run Eq. (5.28). It is worth noting that the training is converging across all 100 samples, which demonstrate that our algorithm is robust towards different initialization. The convergence of one randomly picked network is visualized in Fig. 5.9, where the training process is very stable and all parameters are converging within 150 iterations.

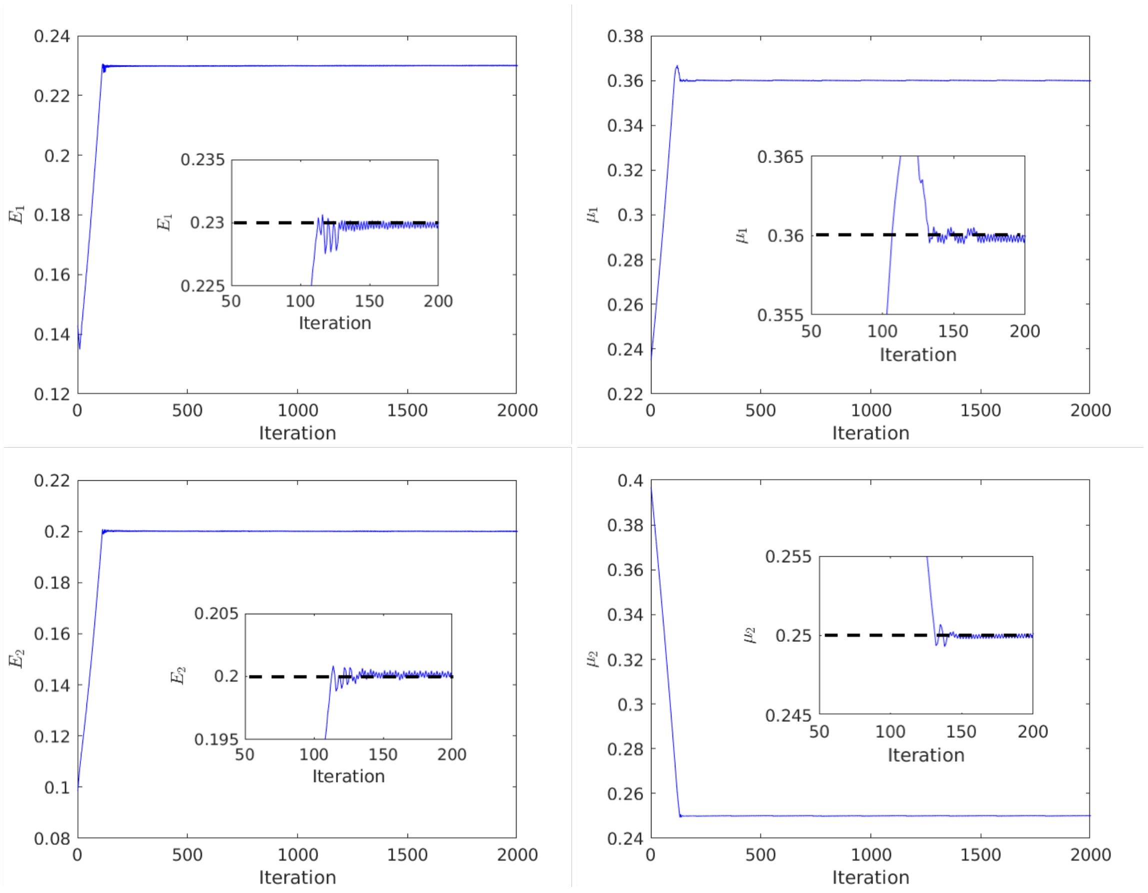


Figure 5.9: Convergence of Different Material Properties. Reference solution is marked with dashed lines.

Furthermore, we test the robustness of our training algorithm with data obtained from different material properties. We fix inner material property to be $E = 0.2TPa$ and $\nu = 0.25$, and the surrounding material is set to have E vary from 0.05 TPa to 0.441 TPa and ν vary from 0.1 to 0.4. A total of 25 different data points are generated in this way. We initialize the network with ground truth material phase, material property of $0.01TPa$ and 0.1 for both phases. Again, Adam optimizer with a learning rate of 10^{-3} is used to run Eq. (5.28). The training is converging for all training data to an error rate below 1%, suggesting that our method is very robust in predicting material properties.

Problem3: Joint material phase and property estimation

In this part, we assume both the material phase and property is unknown and will estimate them from the loading/ response pair. Material phase images from Fig. 5.13a are used to generate the training data. Inclusion (and exterior material) has Young's modulus of 0.241 TPa (and 0.2 TPa) and a Poisson ratio of 0.36 (and 0.25).

The network is initialized with random phase matrix H with value continuously ranging from 0 and 1. Young's modulus and Poisson ratio for both materials are also randomly initialized from 0 to 0.5. For optimizer, we choose truncated Newton's method, and constrain Young's modulus to between 0 to 0.5 TPa and Poisson ration between 0 to 0.5. The termination criteria of the optimizer is set to whenever the gradient is less than 1e-9. Line search is used to determine the optimum step size.

The learned material phase and property are shown in Fig. 5.10 and Tab. 5.3, respectively. Yellow and blue color in Fig. 5.10 represents different material phases. It is interesting to see that the color for inclusions and exclusion in Fig. 5.10 can shuffle (case 6 for single inclusion, and almost all cases for three inclusions). In the meantime, their estimated material properties for two phases are also shuffled in Tab. 5.3. This is because shuffling the phase and material property together actually corresponds to exactly the same physics problem. In other words, they are equally good solutions to the optimization problem, and the original problem does not have a unique solution.

The estimation for Young's modulus is very accurate. The average error rate is 0.3% and 2.5% for different materials. The error on the Poisson ratio is larger, especially on two inclusions. This can be improved by an extra round of post-processing: Initialize the network again with binarized predicted material phase image, and solve Problem 5.4.2 again to focus on learning the material property.

Table 5.3: Learned material properties with random initializations. The reference material property is (0.241 TPa, 0.36) and (0.2 TPa, 0.2) for both phases. inc1 to inc3 denotes different number of inclusions in the material, which can be seen from Fig. 5.10.

		Young's Modulus (TPa)		Poisson Ratio	
Init. \ Est.		Material 0	Material 1	Material 0	Material 1
inc1	case1	0.241	0.197	0.362	0.241
	case2	0.241	0.191	0.362	0.212
	case3	0.241	0.189	0.361	0.203
	case4	0.241	0.191	0.361	0.214
	case5	0.241	0.192	0.362	0.215
	case6	0.190	0.241	0.205	0.360
inc2	case1	0.241	0.204	0.275	0.294
	case2	0.241	0.203	0.274	0.295
	case3	0.241	0.204	0.275	0.294
	case4	0.241	0.203	0.274	0.294
	case5	0.241	0.204	0.275	0.294
	case6	0.241	0.203	0.275	0.294
inc3	case1	0.198	0.241	0.241	0.363
	case2	0.242	0.196	0.364	0.234
	case3	0.200	0.242	0.250	0.364
	case4	0.196	0.244	0.234	0.371
	case5	0.193	0.234	0.221	0.373
	case6	0.197	0.244	0.239	0.372

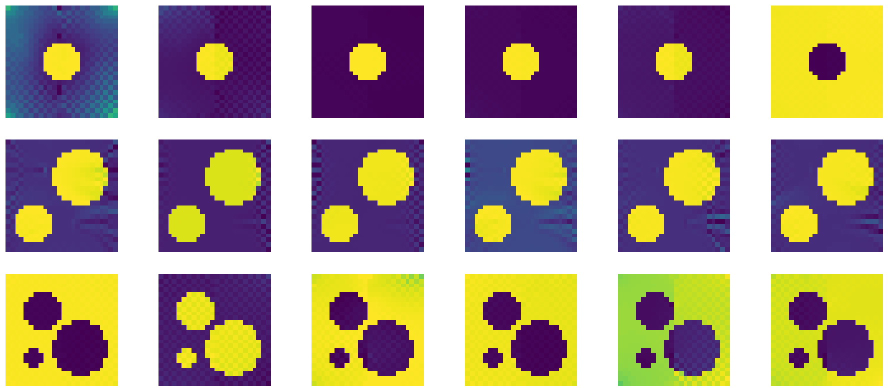


Figure 5.10: Visualization of the Estimated Phase Obtained from Joint Optimization of Material Phase and Property. Different columns are obtained from different random initialization. Yellow and blue color denote material phase 0 and 1 respectively. The predicted material property corresponding to these material phases are listed in Tab. 5.3.

5.5.3 Verification of response prediction

In this subsection, we demonstrate that the response prediction for multi-physics and multi-phase is converging w.r.t. network depth. From Sec. 5.5.1 and Sec. **??** we have seen that FEA-Net is capable to learn either physically meaningful filters or physics parameters robustly and accurately. Based on this, we investigate the convergence of the inference architecture of FEA-Net by assuming that the convolutional kernel has been learned correctly.

For multi-physics problem, we use material property at: $E = 0.212TPa$, $\nu = 0.288$, $\kappa = 16W/(m \cdot K)$, and $\alpha = 1.2e^{-5}/°C$ to generate the reference response. For multi-phase problem, the second material property is set at $E = 0.23TPa$ and $\nu = 0.275$. On the other hand, based on these physics parameters, the reference network filters can be obtained based on Eq. (5.9) to Eq. (5.12) for thermoelasticity , and Eq. (5.16), Eq. (5.17) for bi-phase elasticity.

Since only convolution operation is involved in FEA-Net, it is able to handle inputs

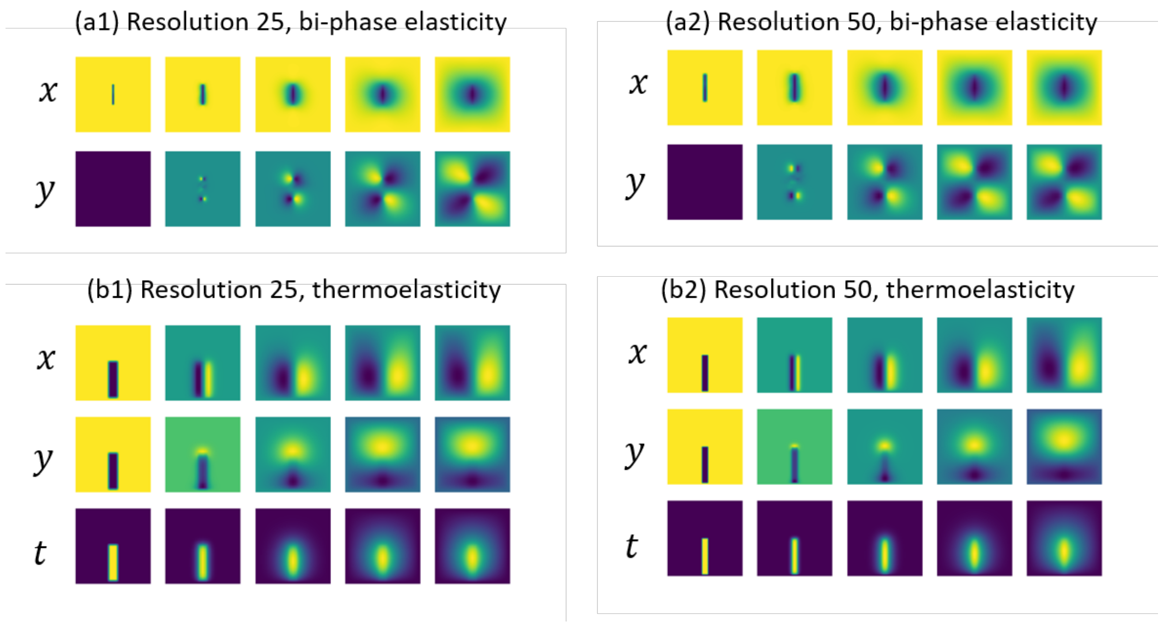


Figure 5.11: Visualization of FEA-Net inference output. Different rows corresponds to x- and y- directional response. From left to right columns are the network input and output at 10th, 100th, 500th, and 4000th layers.

of different resolutions. Thus, we also test the inference performance with loading image of resolution 25 and 50. The predicted response at different network layers is visualized in Fig. 5.11a and Fig. 5.11b for thermoelasticity and bi-phase elasticity respectively. It can be seen that they are all converging well, and there is no visual difference between FEA-Net prediction at 4000 layers and ground truth for these cases.

We further visualize the convergence of the network prediction error w.r.t. the network depth in Fig. 5.12. Interestingly, the loading image at a lower resolution is converging much faster than its higher resolution counterpart. It is conceivable that predicting the response at a higher resolution is a harder task. We expect improvements can be made in at least three aspects: (1) Use larger filter size, which covers a larger region and has a larger receptive field. This is analogous to using higher-order finite elements as discussed in Theorem 1. (2) Form FEA convolution

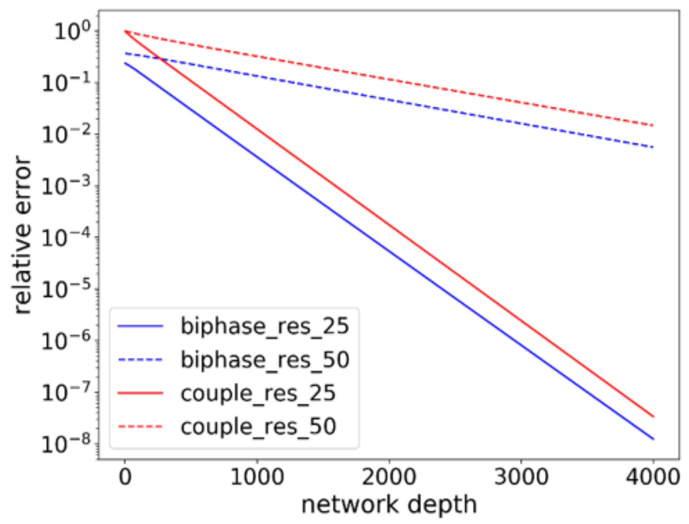


Figure 5.12: Error convergence for bi-phase elasticity and homogeneous thermoelasticity coupling under resolution of 25 and 50.

block with several layers of FEA convolution layers, and use it to replace the current single FEA convolution. Stacking several convolutions together also leads to a larger receptive field. (2) Building better network architectures based on more advanced solvers like multi-grid. The multi-grid network tends to converge much faster, as it computes the response at different resolutions.

5.5.4 Comparison with deep neural networks

In this section, we compare FEA-Net with data-driven approaches in predicting homogeneous elasticity problems. Because FCN can handle image input of different resolution, it is used as the benchmark for comparison. As the simplest example, we train both networks on single-phase elasticity and ask it to make a prediction when a new loading is applied.

Dataset and network setup

To compare the data efficiency, our training set only includes 4 loading and response image pairs obtained from numerical simulation under different loading conditions. As

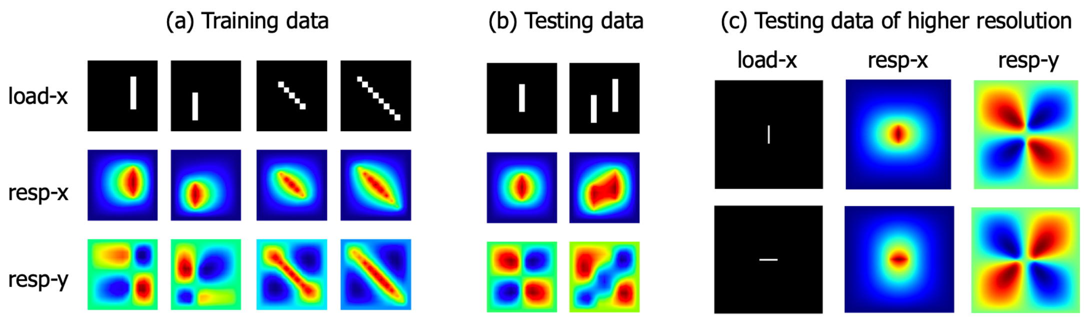


Figure 5.13: Visualization of training and testing data. (a): Training set at a resolution of 12 by 12. (b): Testing data at resolution 12 by 12. (c): Testing data at resolution 60 by 60. Loading is applied uniformly along the x direction only, with locations visualized as the white regions in the plot.

shown in Fig. 5.13a, the white lines in the first row are the locations where a uniform x-directional force is applied. The second and third rows are the displacement along with x and y directions respectively. Different columns correspond to different training data pair. The material used to generate these data pairs has an elasticity modulus of 0.2 TPa and Poisson's ratio of 0.25. To thoroughly investigate the generalization capability of different models, we created four different testing cases. These testing set are are shown in Fig. 5.13b and Fig. 5.13c, which is composed of different loading conditions and image resolutions. Learning on such few amount of data can be a very challenging task for purely data-driven approaches like FCN, but can be handled by proposed FEA-Net.

The benchmark FCN model takes in the system loading image and outputs the predicted response image. As shown in Fig. 5.14, it has 7 layers: The first layer has 2 input channels and 64 output channels, the middle 5 layers have 64 input and 64 output channels, and the output layer has 64 input channels and 2 output channels. The filter size is kept as 3x3, which is the same as FEA-Net. ReLU activation is applied after every layer except the last one. Under such setting, the network contains over 4k filters and 186k trainable variables. The training objective of FCN is to minimize the predicted response with the reference system response. We

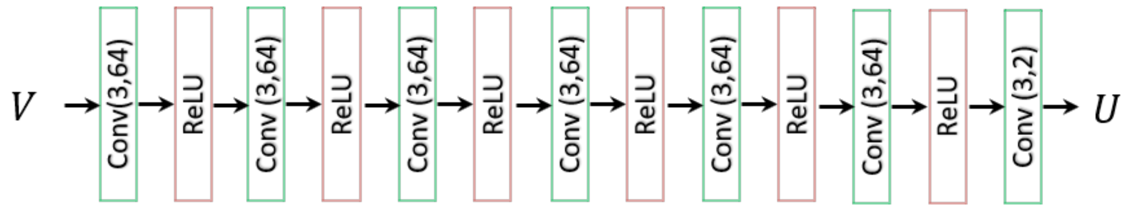


Figure 5.14: Architecture of the baseline FCN model. It has 7 convolutional layers and a forward pas has a total of 20k convolutional operations.

build this network in Tensorflow and train it with Adam optimizer (Kingma and Ba, 2014).

The training architecture of FEA-Net is still one layer, with response image as input and loading image as output. The inference architecture of FEA-Net we used has 5k layers, each layer has 4 filters, which also leads to a total of 20k convolution operations.

Experiment results

We train both FEA-Net and FCN with Adam optimizer till converge. The convergence of their loss for the first 400 iterations is shown in Fig. 5.15a. Note that the magnitude of the loss is not directly comparable, since FEA-Net is defined on the difference between reference loading and predicted loading while FCN is defined on the difference between reference response and network predicted response. However, while FCN is still not converging at 400 iterations with a learning rate of 10^{-3}, FEA-Net is converging around 150 steps under the same learning rate. And if we further increase the learning rate for FEA-Net to 10^{-2}, it is able to converge within 20 steps. And similar to training traditional neural networks, there is also a trade-off in learning rate. With larger learning rate the algorithm will learn faster, while able to learn more stable at a smaller learning rate.

Another major difference is that, while the filters from traditional convolutional neural networks like FCN is totally not interpretable, proposed FEA-Net is designed

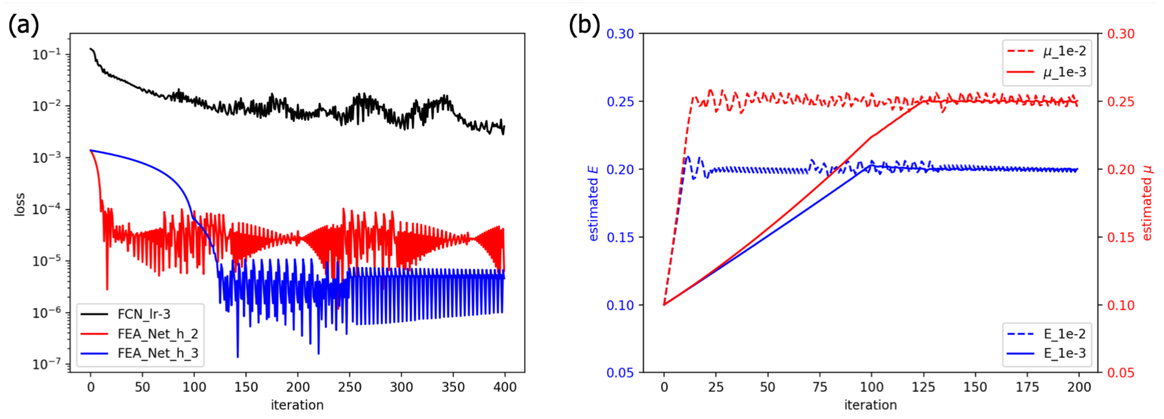


Figure 5.15: (a) Convergence of the training loss for FCN with learning rate 10^{-3} and FEA-Net with learning rate 10^{-2} and 10^{-3}. (b) Convergence of the network training in material property estimation with random initialization. Reference value is $E = 0.2$ and $\nu = 0.25$. (The value of E is scaled by $1e - 12$)

to have physics knowledge embedded and we can infer the physics parameters from its filters. As shown in Fig. 5.15b, FEA-Net successfully learns the correct physics parameters under different learning rate. Similar to the convergence of the loss value, there is also a trade-off between estimation accuracy and learning speed.

Once the network has been trained, we can use it to predict the response image given a new testing loading image. A visual comparison between FCN and FEA-Net is shown in the second and third columns in Fig. 5.16. It can be seen that FCN is able to predict the first testing case well, which shows that our FCN model is reasonable and its training is successful. However, FCN is not able to make correct predictions for all other testing cases which are getting more and more different than the loading images. Although it seems that FCN is getting the correct trend, its prediction is still far away from the ground truth. Such result suggests that there is a big problem with the generalizability of FCN. We also visualize the prediction of FEA-Net with 5000 layers in the fourth and fifth columns in Fig. 5.16. It can be seen that there is almost no visual difference between the network predictions and the ground truth. That is to say, proposed model can generalize well with limited training data.

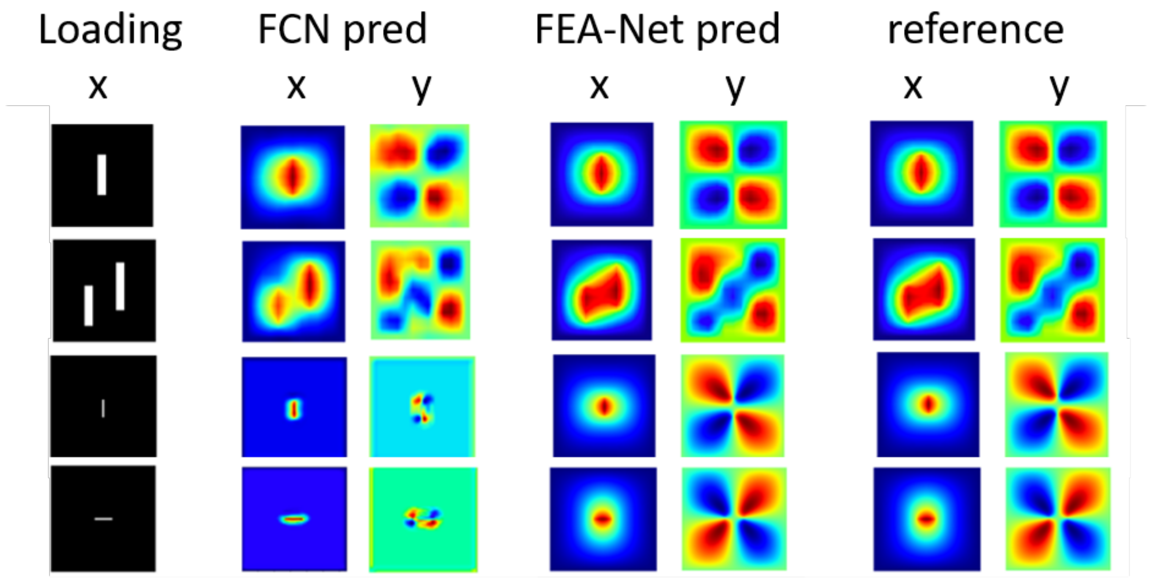


Figure 5.16: Comparison of the network prediction from FCN and FEA-Net. Top to bottom rows corresponds to different test loading cases and the predictions. Loading is applied uniformly along the x direction only. Two columns in each section correspond to x and y directional components.

Table 5.4: Comparison on Memory Usage. n is the resolution.

Problem	thermal		elasticity		bi-phase elasticity		thermoelasticity	
	2D	3D	2D	3D	2D	3D	2D	3D
FEA	$152n^2$	$440n^3$	$304n^2$	$1320n^3$	$304n^2$	$1304n^3$	$456n^2$	$1760n^3$
Proposed	$8n^2 +$ 72	$8n^3 +$ 216	$16n^2 +$ 288	$24n^3 +$ 1944	$24n^2 +$ 288	$32n^3 +$ 1944	$24n^2 +$ 648	$32n^3 +$ 3456
Ratio	19.0	55.0	19.0	55.0	12.7	40.8	19.0	55.0

5.5.5 Comparison with FEA

We derive the memory consumption for FEA-Net and traditional FEA in solving bi-phase elasticity problem. In this subsection, we use n to denote the resolution of the loading image. The memory consumption for traditional FEA is estimated by considering only the storage for the sparse representation of the stiffness matrix

K and the loading vector. The loading will be stored with double-precision floating numbers (8 bytes), which costs $8 * 2 * n^2$ byte memory as each node has two loading components. And since the bandwidth of the stiffness matrix for 2D elasticity is roughly 18, it requires $18*8*n^2$ byte to store the values of stiffness matrix with sparse representation. Besides, an additional $18 * 4 * 2 * n^2$ byte memory is needed to store the row and column index of the sparse matrix with unsigned long int. Summing them together, FEA requires $304n$ byte memory in total. On the other hand, the convolutional filter is shared across all layers in FEA-Net and only very few amount of memory is needed. We estimate the cost for FEA-Net as the storage needed for the filter W (which is $4 * 9 * 8$ byte), loading image with 2 channels ($8 * 2 * n^2$ byte), and the phase image ($8 * n^2$ byte). Summing them together, FEA-Net requires $24n + 288$ byte memory in theory, which is 12.7x less than FEA. The benefit in storage saving can be more significant for 3D problems or problems involving in multi-physics where the bandwidth of the stiffness matrix is larger. For example, the bandwidth of the stiffness matrix is increased to 81 for 3D thermal problems, the memory savings can be $1/55$ with the proposed method. A list of the comparison is given in Tab. 5.4 on the memory consumption of different situations.

We further use a real-world problem to compare the memory consumption of the proposed method with FEA. We have a pipeline system installed between the year 1949 to 1961, and wish to analyze its micro-structure to monitor its health conditions. As in (Dahire *et al.*, 2018), it is known to us that the pipeline is composed of two phases, ferrite and pearlite. Since we know the material property and wish to learn its phase from loading/ response data, this fails into Problem 5.4.1 discussed in Sec. 5.4.2. The loading and response image pairs we used has a resolution of 150, which is shown in the first and second columns in Fig. 5.17. The learned phase images for different samples are shown in the third column of Fig. 5.17. And the reference phase images

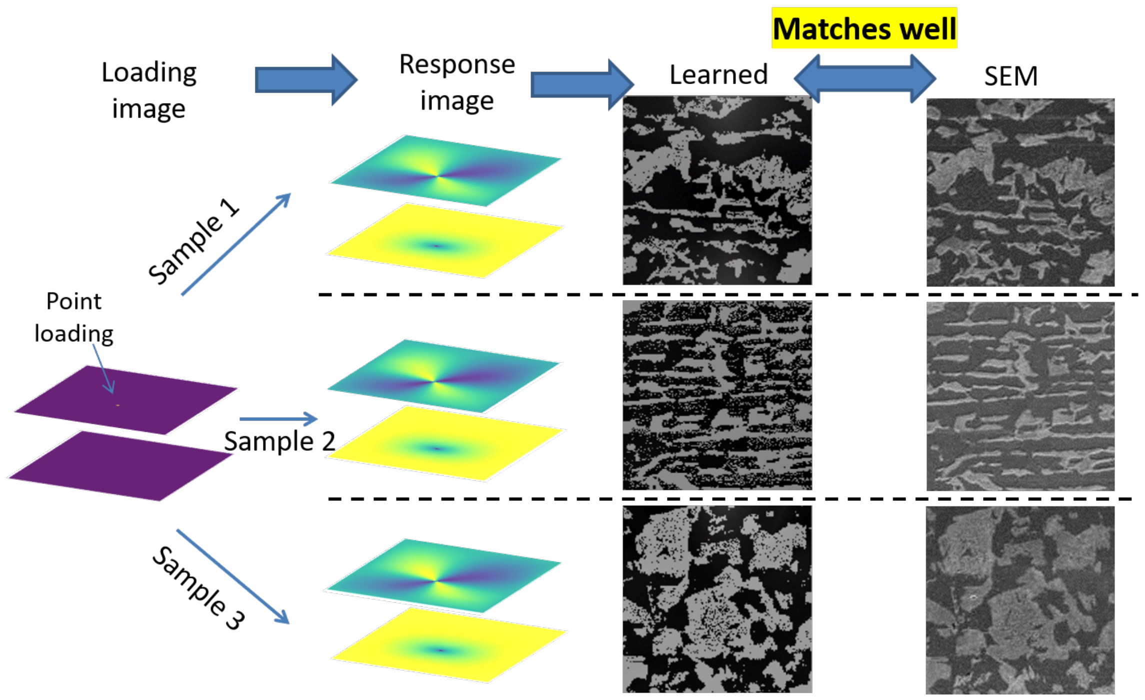


Figure 5.17: Estimated phase of the pipeline samples. The first column shows the loading image, with the response image on different samples shown in the second column. The third and fourth column shows the learned material phase and phase obtained form SEM scan.

obtained from Scanning Electron Microscope (SEM) are shown in the last column of Fig. 5.17. The learned material phase images and reference ones match very well with each other.

The peak memory consumption of the proposed method under our Tensorflow implementation is shown in Fig. 5.18. It can be seen that the proposed method has larger memory consumption a lower DOF, which is caused by the overhead of Tensorflow implementation. However, as DOF increases, the memory consumption of our implementation is approaching the theoretical bound. And our method will start to consume less memory than the baseline lower bound starting from 3 million DOF.

As for the cost in computational time, if sparse matrix-vector production is used, the complexity for traditional FEA should be the same to FEA-Net. However, since convolution can be implemented very efficiently on GPU, there could be an improve-

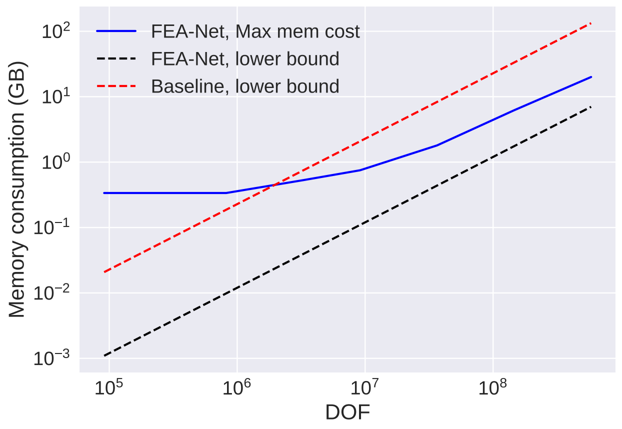


Figure 5.18: Comparison on Memory Cost with Standard FEA. Red and black dashed lines are the estimated lower bound of the memory consumption. Blue curve is the peak memory consumption of our model obtained from experiment.

ment on the time consumption considering the benefit from the hardware side.

5.6 Discussion and Conclusion

Motivated by the success and limitations of both data-driven models and physics-based models, we present a hybrid learning approach to predict the physics response with limited training data samples. The proposed method is a very flexible model, which can have different physics prior knowledge added easily. It has good interpretability, as the network filters are designed to reflect the PDE behind the physics behavior. Theoretical analysis and empirical experiments have shown that the proposed method is very data efficient in learning and has good convergence at predicting both multi-physics and multi-phase problems.

Furthermore, there are many interesting directions worth pursuing based on this study: (1) By setting the second material to have zero mechanical property, the proposed method can handle topology optimization. (2) More efficient inference architectures can be built with more advanced solvers like multi-grid. (3) Multi-grid solvers can be used to model material homogenization at a different scale such as in (Liu *et al.*, 2019). (4) Extending current networks to non-linear to model the non-linearity in the material property. (5) Use larger convolution filters or more convolutional layers for higher efficiency. (6) Incorporate graph convolutions to handle irregular mesh.

Chapter 6

CONCLUSION

6.1 Summary

In this dissertation, we explored two major issues of neural networks: robustness and generalizability. Three ideas were tested to solve these problems: generative model, transformation-invariant adversarial training, and utilizing prior knowledge in physics.

We first designed a classifier based on the generative model, which partially addresses both problems in robustness and generalizability. A customized conditional Variational Auto-Encoder is tailored as a classifier, and tested its generalizability and robustness under challenging tasks. Results showed that the proposed algorithm leads to promising performance: First, the model can recognize overlapping objects and novel component combinations that do not exist in the training phase. Second, our model is able to defend against adversarial attacks well under higher attack magnitudes.

Then we proposed transformation-invariant adversarial training on feed forward classifiers, and improved the robustness over state-of-the-art algorithms. We investigated the effectiveness of a specific set of transformations. The results showed that the resultant model improves empirical robustness over standard adversarial training without gradient obfuscation effects, and can be more data efficient than baseline model.

Lastly, we introduced physics-guided learning and built a network that generalizes very well in predicting physics response with limited samples. A general framework was proposed to integrate physics prior knowledge into a data-driven approach based on FEA theory. As shown by examples in elasticity, proposed hybrid method is generalizable even with very limited data samples, and has certifiable convergence compared with purely data-driven approaches.

6.2 Future Work

For the generative model, it is worth considering: (1) Sharing the features across different sub-networks, which has the potential to speed up the network training and inference speed, (2) Using more powerful generators, which will be able to cover more complicated data distributions (3) Testing its performance on larger datasets such as CelebA or ImageNet.

For transformation invariant adversarial training, improvements can be made in the following aspects: (1) Using the idea in YOPO (Zhang *et al.*, 2019a) to further improve training speed, by re-using the adversarial gradients of the upper layers of the network. (2) Learning the optimum natural image transformations with Spatial Transformer Network instead using a pre-defined set of transformations.

There are many interesting directions worth exploring with physics-guided learning: (1) By setting the second material to have zero mechanical property, the proposed method can handle topology optimization. (2) More efficient inference architectures can be built with more advanced solvers like multi-grid. (3) Multi-grid solvers can be used to model material homogenization at a different scale such as in Liu *et al.* (2019). (4) Extending current networks to non-linear to model the non-linearity in the material property. (5) Use larger convolution filters or more convolutional layers for higher efficiency.

REFERENCES

Akhtar, N. and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey", arXiv preprint arXiv:1801.00553 (2018).

Amodei, D., S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin", in "International Conference on Machine Learning", pp. 173–182 (2016).

Arjovsky, M., S. Chintala and L. Bottou, "Wasserstein GAN", arXiv URL `http://arxiv.org/abs/1701.07875` (2017).

Athalye, A., N. Carlini and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples", arXiv preprint arXiv:1802.00420 (2018).

Athalye, A. and I. Sutskever, "Synthesizing robust adversarial examples", arXiv preprint arXiv:1707.07397 (2017).

Berthelot, D., T. Schumm and L. Metz, "BEGAN: Boundary Equilibrium Generative Adversarial Networks", arXiv pp. 1–9, URL `http://arxiv.org/abs/1703.10717` (2017).

Bessa, M., R. Bostanabad, Z. Liu, A. Hu, D. W. Apley, C. Brinson, W. Chen and W. K. Liu, "A framework for data-driven analysis of materials under uncertainty: Countering the curse of dimensionality", Computer Methods in Applied Mechanics and Engineering **320**, 633–667 (2017).

Bouman, K. L., B. Xiao, P. Battaglia and W. T. Freeman, "Estimating the material properties of fabric from video", in "Proceedings of the IEEE international conference on computer vision", pp. 1984–1991 (2013).

Brendel, W. and M. Bethge, "Approximating cnns with bag-of-local-features models works surprisingly well on imagenet", (2018).

Brock, A., J. Donahue and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis", arXiv preprint arXiv:1809.11096 (2018).

Cang, R., H. Li, H. Yao, Y. Jiao and Y. Ren, "Improving direct physical properties prediction of heterogeneous materials from imaging data via convolutional neural network and a morphology-aware generative model", Computational Materials Science **150**, 212–221 (2018a).

Cang, R. and M. Y. Ren, "Deep network-based feature extraction and reconstruction of complex material microstructures", in "ASME 2016 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference", pp. V02BT03A008–V02BT03A008 (American Society of Mechanical Engineers, 2016).

Cang, R., Y. Xu, S. Chen, Y. Liu, Y. Jiao and M. Y. Ren, “Microstructure representation and reconstruction of heterogeneous materials via deep belief network for computational material design”, Journal of Mechanical Design **139**, 7, 071404 (2017).

Cang, R., H. Yao and Y. Ren, “One-shot optimal topology generation through theory-driven machine learning”, arXiv preprint arXiv:1807.10787 (2018b).

Capuano, G. and J. J. Rimoli, “Smart finite elements: A novel machine learning application”, Computer Methods in Applied Mechanics and Engineering **345**, 363–381 (2019).

Carlini, N. and D. Wagner, “Towards evaluating the robustness of neural networks”, in “2017 IEEE Symposium on Security and Privacy (SP)”, pp. 39–57 (IEEE, 2017).

Chen, H., O. Engkvist, Y. Wang, M. Olivecrona and T. Blaschke, “The rise of deep learning in drug discovery”, Drug discovery today **23**, 6, 1241–1250 (2018).

Chen, P.-Y., Y. Sharma, H. Zhang, J. Yi and C.-J. Hsieh, “Ead: elastic-net attacks to deep neural networks via adversarial examples”, arXiv preprint arXiv:1709.04114 (2017).

Chen, X., Y. Duan, R. Houthooft, J. Schulman, I. Sutskever and P. Abbeel, “InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets”, arXiv URL `http://arxiv.org/abs/1606.03657` (2016a).

Chen, X., Y. Duan, R. Houthooft, J. Schulman, I. Sutskever and P. Abbeel, “Infogan: Interpretable representation learning by information maximizing generative adversarial nets”, in “Advances in Neural Information Processing Systems”, pp. 2172–2180 (2016b).

Chen, Y., L. Zhu, C. Lin, H. Zhang and A. L. Yuille, “Rapid inference on a novel and/or graph for object detection, segmentation and parsing”, in “Advances in neural information processing systems”, pp. 289–296 (2008).

Chu, M. and N. Thuerey, “Data-driven synthesis of smoke flows with cnn-based feature descriptors”, ACM Transactions on Graphics (TOG) **36**, 4, 69 (2017).

Csurka, G., C. Dance, L. Fan, J. Willamowski and C. Bray, “Visual categorization with bags of keypoints”, in “Workshop on statistical learning in computer vision, ECCV”, vol. 1, pp. 1–2 (Prague, 2004).

Dahire, S., F. Tahir, Y. Jiao and Y. Liu, “Bayesian network inference for probabilistic strength estimation of aging pipeline systems”, International Journal of Pressure Vessels and Piping **162**, 30–39 (2018).

Das, N., M. Shanbhogue, S.-T. Chen, F. Hohman, S. Li, L. Chen, M. E. Kounavis and D. H. Chau, “Shield: Fast, practical defense and vaccination for deep learning using jpeg compression”, arXiv preprint arXiv:1802.06816 (2018).

de Oliveira, L., M. Paganini and B. Nachman, "Learning particle physics by example: location-aware generative adversarial networks for physics synthesis", Computing and Software for Big Science **1**, 1, 4 (2017).

Deng, J. D. J., W. D. W. Dong, R. Socher, L.-J. L. L.-J. Li, K. L. K. Li and L. F.-F. L. Fei-Fei, "ImageNet: A large-scale hierarchical image database", 2009 IEEE Conference on Computer Vision and Pattern Recognition pp. 2–9 (2009).

Deng, L., "The mnist database of handwritten digit images for machine learning research [best of the web]", IEEE Signal Processing Magazine **29**, 6, 141–142 (2012).

Devlin, J., M.-W. Chang, K. Lee and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding", arXiv preprint arXiv:1810.04805 (2018).

Ding, G. W., K. Y.-C. Lui, X. Jin, L. Wang and R. Huang, "On the sensitivity of adversarial robustness to input data distributions", (2018).

Dong, Y., T. Pang, H. Su and J. Zhu, "Evading defenses to transferable adversarial examples by translation-invariant attacks", arXiv preprint arXiv:1904.02884 (2019).

Duchi, J., E. Hazan and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization", Journal of Machine Learning Research **12**, Jul, 2121–2159 (2011).

Dumoulin, V., I. Belghazi, B. Poole, A. Lamb, M. Arjovsky, O. Mastropietro and A. Courville, "Adversarially Learned Inference", arXiv preprint arXiv:1606.00704 pp. 1–15, URL `http://arxiv.org/abs/1606.00704` (2016).

Durugkar, I., I. Gemp and S. Mahadevan, "Generative Multi-Adversarial Networks", arXiv pp. 1–14, URL `http://arxiv.org/abs/1611.01673` (2016).

Evtimov, I., K. Eykholt, E. Fernandes, T. Kohno, B. Li, A. Prakash, A. Rahmati and D. Song, "Robust physical-world attacks on machine learning models", arXiv preprint arXiv:1707.08945 (2017).

Eykholt, K., I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno and D. Song, "Robust physical-world attacks on deep learning visual classification", in "Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition", pp. 1625–1634 (2018).

Farhadi, A., I. Endres, D. Hoiem and D. Forsyth, "Describing objects by their attributes", in "Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on", pp. 1778–1785 (IEEE, 2009).

Fei-Fei, L., R. Fergus and P. Perona, "One-shot learning of object categories", IEEE transactions on pattern analysis and machine intelligence **28**, 4, 594–611 (2006).

Finol, D., Y. Lu, V. Mahadevan and A. Srivastava, "Deep convolutional neural networks for eigenvalue problems in mechanics", International Journal for Numerical Methods in Engineering (2018).

For, N. and A. M. Tools, “Progressive Growing of GANs for Improved Quality, Stability, and Variation”, pp. 1–25 (2018).

George, D., W. Lehrach, K. Kansky, M. Lázaro-Gredilla, C. Laan, B. Marthi, X. Lou, Z. Meng, Y. Liu, H. Wang *et al.*, “A generative vision model that trains with high data efficiency and breaks text-based captchas”, Science p. eaag2612 (2017).

Ghosh, A., V. Kulharia, V. Namboodiri, P. H. S. Torr and P. K. Dokania, “Multi-Agent Diverse Generative Adversarial Networks”, arXiv URL `http://arxiv.org/abs/1704.02906` (2017).

Gilmer, J., S. S. Schoenholz, P. F. Riley, O. Vinyals and G. E. Dahl, “Neural message passing for quantum chemistry”, in “Proceedings of the 34th International Conference on Machine Learning-Volume 70”, pp. 1263–1272 (JMLR. org, 2017).

Girshick, R., J. Donahue, T. Darrell and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation”, in “Proceedings of the IEEE conference on computer vision and pattern recognition”, pp. 580–587 (2014).

Goodfellow, I., J. Pouget-Abadie and M. Mirza, “Generative Adversarial Networks”, arXiv preprint arXiv: ... pp. 1–9, URL `http://arxiv.org/abs/1406.2661` (2014a).

Goodfellow, I. J., J. Shlens and C. Szegedy, “Explaining and Harnessing Adversarial Examples”, arXiv pp. 1–11, URL `http://arxiv.org/abs/1412.6572` (2014b).

Graff, P., F. Feroz, M. P. Hobson and A. Lasenby, “Skynet: an efficient and robust neural network training tool for machine learning in astronomy”, Monthly Notices of the Royal Astronomical Society **441**, 2, 1741–1759 (2014).

Graves, A., A.-r. Mohamed and G. Hinton, “Speech recognition with deep recurrent neural networks”, in “2013 IEEE international conference on acoustics, speech and signal processing”, pp. 6645–6649 (IEEE, 2013).

Gregor, K., I. Danihelka, A. Graves, D. Jimenez Rezende and D. Wierstra, “DRAW: A Recurrent Neural Network For Image Generation”, arXiv pp. 1462–1471 (2015).

Gu, S. and L. Rigazio, “Towards deep neural network architectures robust to adversarial examples”, arXiv preprint arXiv:1412.5068 (2014).

Gulrajani, I., K. Kumar, F. Ahmed, A. A. Taiga, F. Visin, D. Vazquez and A. Courville, “PixelVAE: A Latent Variable Model for Natural Images”, arXiv pp. 1–9, URL `http://arxiv.org/abs/1611.05013` (2016).

Guo, C., M. Rana, M. Cisse and L. van der Maaten, “Countering adversarial images using input transformations”, arXiv preprint arXiv:1711.00117 (2017).

He, K., X. Zhang, S. Ren and J. Sun, “Deep Residual Learning for Image Recognition”, arXiv **7**, 3, 171–180 (2015).

He, K., X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition", in "Proceedings of the IEEE conference on computer vision and pattern recognition", pp. 770–778 (2016).

Hendrik Metzen, J., M. Chaithanya Kumar, T. Brox and V. Fischer, "Universal adversarial perturbations against semantic image segmentation", in "Proceedings of the IEEE International Conference on Computer Vision", pp. 2755–2764 (2017).

Henrion, I., J. Brehmer, J. Bruna, K. Cho, K. Cranmer, G. Louppe and G. Rochette, "Neural message passing for jet physics", (2017).

Hinton, G. E. and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks", science **313**, 5786, 504–507 (2006).

Hornik, K., "Approximation capabilities of multilayer feedforward networks", Neural networks **4**, 2, 251–257 (1991a).

Hornik, K., "Approximation capabilities of multilayer feedforward networks", Neural networks **4**, 2, 251–257 (1991b).

Hsieh, J.-T., S. Zhao, S. Eismann, L. Mirabella and S. Ermon, "Learning neural pde solvers with convergence guarantees", (2018).

Huang, G., Z. Liu, K. Q. Weinberger and L. van der Maaten, "Densely connected convolutional networks", in "Proceedings of the IEEE conference on computer vision and pattern recognition", vol. 1, p. 3 (2017a).

Huang, S., N. Papernot, I. Goodfellow, Y. Duan and P. Abbeel, "Adversarial Attacks on Neural Network Policies", arXiv URL `http://arxiv.org/abs/1702.02284` (2017b).

Hughes, T. J., *The finite element method: linear static and dynamic finite element analysis* (Courier Corporation, 2012).

Isola, P., J.-Y. Zhu, T. Zhou and A. A. Efros, "Image-to-image translation with conditional adversarial networks", arXiv preprint (2017).

Jia, F., Y. Lei, J. Lin, X. Zhou and N. Lu, "Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data", Mechanical Systems and Signal Processing **72**, 303–315 (2016).

Jurie, F. and B. Triggs, "Creating efficient codebooks for visual recognition", in "Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on", vol. 1, pp. 604–610 (IEEE, 2005).

Katz, G., C. Barrett, D. L. Dill, K. Julian and M. J. Kochenderfer, "Reluplex: An efficient smt solver for verifying deep neural networks", in "International Conference on Computer Aided Verification", pp. 97–117 (Springer, 2017).

Kawaguchi, K., "Deep learning without poor local minima", in "Advances in neural information processing systems", pp. 586–594 (2016).

Kingma, D. P. and J. Ba, "Adam: A method for stochastic optimization", arXiv preprint arXiv:1412.6980 (2014).

Kingma, D. P. and M. Welling, "Auto-Encoding Variational Bayes", arXiv URL `http://arxiv.org/abs/1312.6114` (2013).

Kos, J. and D. Song, "Delving into adversarial attacks on deep policies", arXiv preprint arXiv:1705.06452 (2017).

Krizhevsky, A., V. Nair and G. Hinton, "The cifar-10 dataset", online: http://www. cs. toronto. edu/kriz/cifar. html (2014).

Krizhevsky, A., I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", in "Advances in neural information processing systems", pp. 1097–1105 (2012).

Kumar, A., O. Irsoy, P. Ondruska, M. Iyyer, J. Bradbury, I. Gulrajani, V. Zhong, R. Paulus and R. Socher, "Ask me anything: Dynamic memory networks for natural language processing", in "International Conference on Machine Learning", pp. 1378–1387 (2016).

Kurakin, A., I. Goodfellow and S. Bengio, "Adversarial examples in the physical world", Arxiv , c, 1–15, URL `http://arxiv.org/abs/1607.02533` (2016a).

Kurakin, A., I. Goodfellow and S. Bengio, "Adversarial examples in the physical world", arXiv preprint arXiv:1607.02533 (2016b).

Kurakin, A., I. Goodfellow and S. Bengio, "Adversarial machine learning at scale", arXiv preprint arXiv:1611.01236 (2016c).

Kurakin, A., I. Goodfellow and S. Bengio, "Adversarial machine learning at scale", arXiv preprint arXiv:1611.01236 (2016d).

Kurakin, A., I. J. Goodfellow and S. Bengio, "Adversarial machine learning at scale", (2017), URL `https://arxiv.org/abs/1611.01236`.

Lampert, C. H., H. Nickisch and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer", in "Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on", pp. 951–958 (IEEE, 2009).

Larsen, A. B. L., S. K. Sønderby and O. Winther, "Autoencoding beyond pixels using a learned similarity metric", arXiv URL `http://arxiv.org/abs/1512.09300` (2015).

LeCun, Y., B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition", Neural computation **1**, 4, 541–551 (1989).

LeCun, Y., F. J. Huang and L. Bottou, "Learning methods for generic object recognition with invariance to pose and lighting", in "Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on", vol. 2, pp. II–104 (IEEE, 2004).

Lee, H., R. Grosse, R. Ranganath and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations", Proceedings of the 26th Annual International Conference on Machine Learning ICML 09 **2008**, 1–8, URL `http://portal.acm.org/citation.cfm?doid=1553374.1553453` (2009).

Li, X., Z. Liu, S. Cui, C. Luo, C. Li and Z. Zhuang, "Predicting the effective mechanical property of heterogeneous materials by image based modeling and deep learning", Computer Methods in Applied Mechanics and Engineering **347**, 735–753 (2019).

Liao, F., M. Liang, Y. Dong, T. Pang, J. Zhu and X. Hu, "Defense against adversarial attacks using high-level representation guided denoiser", in "Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition", pp. 1778–1787 (2018).

Liu, Z., C. Wu and M. Koishi, "A deep material network for multiscale topology learning and accelerated nonlinear modeling of heterogeneous materials", Computer Methods in Applied Mechanics and Engineering **345**, 1138–1168 (2019).

Long, J., E. Shelhamer and T. Darrell, "Fully convolutional networks for semantic segmentation", in "Proceedings of the IEEE conference on computer vision and pattern recognition", pp. 3431–3440 (2015).

Long, Z., Y. Lu, X. Ma and B. Dong, "Pde-net: Learning pdes from data", in "Proceedings of the 35th International Conference on Machine Learning (ICML 2018)", (2018).

Lu, J., H. Sibai, E. Fabry and D. Forsyth, "Standard detectors aren't (currently) fooled by physical adversarial stop signs", arXiv preprint arXiv:1710.03337 (2017a).

Lu, Y., A. Zhong, Q. Li and B. Dong, "Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations", arXiv preprint arXiv:1710.10121 (2017b).

Madry, A., A. Makelov, L. Schmidt, D. Tsipras and A. Vladu, "Towards deep learning models resistant to adversarial attacks", arXiv preprint arXiv:1706.06083 (2017).

Makhzani, A. and B. J. Frey, "Pixelgan autoencoders", in "Advances in Neural Information Processing Systems", pp. 1972–1982 (2017).

Makhzani, A., J. Shlens, N. Jaitly, I. Goodfellow and B. Frey, "Adversarial autoencoders", arXiv preprint arXiv:1511.05644 (2015).

Metz, L., B. Poole, D. Pfau and J. Sohl-Dickstein, "Unrolled Generative Adversarial Networks", arXiv pp. 1–25, URL `http://arxiv.org/abs/1611.02163` (2016).

Mnih, V., K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning", Nature **518**, 7540, 529 (2015).

Moosavi-Dezfooli, S.-M., A. Fawzi and P. Frossard, “Deepfool: a simple and accurate method to fool deep neural networks”, in “Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition”, pp. 2574–2582 (2016).

Nguyen, A., J. Yosinski and J. 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”, pp. 427–436 (2015).

Oishi, A. and G. Yagawa, “Computational mechanics enhanced by deep learning”, Computer Methods in Applied Mechanics and Engineering **327**, 327–351 (2017).

Papernot, N., P. McDaniel and I. Goodfellow, “Transferability in machine learning: from phenomena to black-box attacks using adversarial samples”, arXiv preprint arXiv:1605.07277 (2016a).

Papernot, N., P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik and A. Swami, “Practical black-box attacks against deep learning systems using adversarial examples”, arXiv preprint arXiv:1602.02697 (2016b).

Papernot, N., P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik and A. Swami, “Practical black-box attacks against machine learning”, in “Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security”, pp. 506–519 (ACM, 2017).

Papernot, N., P. McDaniel, X. Wu, S. Jha and A. Swami, “Distillation as a defense to adversarial perturbations against deep neural networks”, in “2016 IEEE Symposium on Security and Privacy (SP)”, pp. 582–597 (IEEE, 2016c).

Radford, A., L. Metz and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks”, arXiv preprint arXiv:1511.06434 (2015).

Reed, S., Z. Akata, X. Yan, L. Logeswaran, B. Schiele and H. Lee, “Generative Adversarial Text to Image Synthesis”, ICML pp. 1060–1069 (2016).

Reed, S., A. van den Oord, N. Kalchbrenner, S. G. Colmenarejo, Z. Wang, D. Belov and N. de Freitas, “Parallel Multiscale Autoregressive Density Estimation”, arXiv URL `http://arxiv.org/abs/1703.03664` (2017).

Ren, S., K. He, R. Girshick and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks”, in “Advances in neural information processing systems”, pp. 91–99 (2015).

Ronneberger, O., P. Fischer and T. Brox, “U-net: Convolutional networks for biomedical image segmentation”, in “International Conference on Medical image computing and computer-assisted intervention”, pp. 234–241 (Springer, 2015).

Roy, A., C. Raffel, I. Goodfellow and J. Buckman, “Thermometer encoding: One hot way to resist adversarial examples”, (2018).

Rumelhart, D. E., G. E. Hinton and R. J. Williams, "Learning internal representations by error propagation", Tech. rep., California Univ San Diego La Jolla Inst for Cognitive Science (1985).

Rumelhart, D. E., G. E. Hinton, R. J. Williams *et al.*, "Learning representations by back-propagating errors", Cognitive modeling **5**, 3, 1 (1988).

Sabour, S., N. Frosst and G. E. Hinton, "Dynamic routing between capsules", in "Advances in Neural Information Processing Systems", pp. 3859–3869 (2017).

Salimans, T., I. Goodfellow, W. Zaremba, V. Cheung, A. Radford and X. Chen, "Improved Techniques for Training GANs", Nips pp. 1–10 (2016).

Salimans, T., A. Karpathy, X. Chen and D. P. Kingma, "PixelCNN++: Improving the PixelCNN with Discretized Logistic Mixture Likelihood and Other Modifications", arXiv pp. 1–10, URL `http://arxiv.org/abs/1701.05517` (2017).

Santoro, A., S. Bartunov, M. Botvinick, D. Wierstra and T. Lillicrap, "Meta-learning with memory-augmented neural networks", in "International conference on machine learning", pp. 1842–1850 (2016).

Sharif, M., S. Bhagavatula, L. Bauer and M. K. Reiter, "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition", in "Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security", pp. 1528–1540 (ACM, 2016).

Sheikholeslami, M., M. B. Gerdroodbary, R. Moradi, A. Shafee and Z. Li, "Application of neural network for estimation of heat transfer treatment of al2o3-h2o nanofluid through a channel", Computer Methods in Applied Mechanics and Engineering **344**, 1–12 (2019).

Silver, D., J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, "Mastering the game of go without human knowledge", Nature **550**, 7676, 354 (2017).

Sohn, K., H. Lee and X. Yan, "Learning structured output representation using deep conditional generative models", in "Advances in Neural Information Processing Systems", pp. 3483–3491 (2015).

Sosnovik, I. and I. Oseledets, "Neural networks for topology optimization", arXiv preprint arXiv:1709.09578 (2017a).

Sosnovik, I. and I. Oseledets, "Neural networks for topology optimization", arXiv preprint arXiv:1709.09578 (2017b).

Spall, J. C. *et al.*, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation", IEEE transactions on automatic control **37**, 3, 332–341 (1992).

Stanley, K. O., "Compositional pattern producing networks: A novel abstraction of development", Genetic programming and evolvable machines **8**, 2, 131–162 (2007).

Stanley, K. O. and R. Miikkulainen, "Evolving neural networks through augmenting topologies", Evol. Comput. **10**, 2, 99–127, URL `http://dx.doi.org/10.1162/106365602320169811` (2002).

Stringer, S. M. and E. T. Rolls, "Invariant object recognition in the visual system with novel views of 3d objects", Neural Computation **14**, 11, 2585–2596 (2002).

Su, J., D. Vasconcellos Vargas and S. Kouichi, "One pixel attack for fooling deep neural networks", arXiv (2017).

Szegedy, C., V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the inception architecture for computer vision", in "Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition", pp. 2818–2826 (2016).

Szegedy, C., W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow and R. Fergus, "Intriguing properties of neural networks", arXiv preprint arXiv:1312.6199 (2013).

Tolstikhin, I., S. Gelly, O. Bousquet, C.-J. Simon-Gabriel and B. Schölkopf, "AdaGAN: Boosting Generative Models", arXiv pp. 1–31, URL `http://arxiv.org/abs/1701.02386` (2017).

Tompson, J., K. Schlachter, P. Sprechmann and K. Perlin, "Accelerating eulerian fluid simulation with convolutional networks", arXiv preprint arXiv:1607.03597 (2016).

Tramèr, F., A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh and P. McDaniel, "Ensemble adversarial training: Attacks and defenses", arXiv preprint arXiv:1705.07204 (2017).

Tsipras, D., S. Santurkar, L. Engstrom, A. Turner and A. Madry, "Robustness may be at odds with accuracy", (2018).

Uesato, J., B. O'Donoghue, A. v. d. Oord and P. Kohli, "Adversarial risk and the dangers of evaluating against weak attacks", arXiv preprint arXiv:1802.05666 (2018).

Ullman, S., L. Assif, E. Fetaya and D. Harari, "Atoms of recognition in human and computer vision", Proceedings of the National Academy of Sciences **113**, 10, 2744–2749 (2016).

van den Oord, A., N. Kalchbrenner and K. Kavukcuoglu, "Pixel Recurrent Neural Networks", arXiv **48**, URL `http://arxiv.org/abs/1601.06759` (2016a).

van den Oord, A., N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves and K. Kavukcuoglu, "Conditional Image Generation with PixelCNN Decoders", arXiv URL `http://arxiv.org/abs/1606.05328` (2016b).

Wang, Q., G. Zhang, C. Sun and N. Wu, "High efficient load paths analysis with u* index generated by deep learning", Computer Methods in Applied Mechanics and Engineering **344**, 499–511 (2019).

Wong, E. and Z. Kolter, “Provable defenses against adversarial examples via the convex outer adversarial polytope”, in “International Conference on Machine Learning”, pp. 5283–5292 (2018).

Wong, E., F. Schmidt, J. H. Metzen and J. Z. Kolter, “Scaling provable adversarial defenses”, arXiv preprint arXiv:1805.12514 (2018).

Wu, Z., S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang and J. Xiao, “3d shapenets: A deep representation for volumetric shapes”, in “Proceedings of the IEEE conference on computer vision and pattern recognition”, pp. 1912–1920 (2015).

Xie, C., J. Wang, Z. Zhang, Y. Zhou, L. Xie and A. Yuille, “Adversarial examples for semantic segmentation and object detection”, in “International Conference on Computer Vision”, (IEEE, 2017a).

Xie, S., R. Girshick, P. Dollár, Z. Tu and K. He, “Aggregated residual transformations for deep neural networks”, in “Proceedings of the IEEE conference on computer vision and pattern recognition”, pp. 1492–1500 (2017b).

Xu, H., C. Caramanis and S. Mannor, “Robustness and regularization of support vector machines”, Journal of Machine Learning Research **10**, Jul, 1485–1510 (2009).

Xu, W., D. Evans and Y. Qi, “Feature squeezing: Detecting adversarial examples in deep neural networks”, arXiv preprint arXiv:1704.01155 (2017a).

Xu, W., D. Evans and Y. Qi, “Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks”, arXiv URL `http://arxiv.org/abs/1704.01155` (2017b).

Yan, X., J. Yang, K. Sohn and H. Lee, “Attribute2image: Conditional image generation from visual attributes”, in “European Conference on Computer Vision”, pp. 776–791 (Springer, 2016).

Yang, X. I. and R. Mittal, “Acceleration of the jacobi iterative method by factors exceeding 100 using scheduled relaxation”, Journal of Computational Physics **274**, 695–708 (2014).

Yao, H., J. Wen, Y. Ren, B. Wu and Z. Ji, “Low-cost measurement of industrial shock signals via deep learning calibration”, arXiv preprint arXiv:1902.02829 (2019).

Yu, Y., H. Yao and Y. Liu, “Physics-based learning for aircraft dynamics simulation”, in “PHM Society Conference”, vol. 10 (2018).

Zeiler, M. D., “Adadelta: an adaptive learning rate method”, arXiv preprint arXiv:1212.5701 (2012).

Zhang, D., T. Zhang, Y. Lu, Z. Zhu and B. Dong, “You only propagate once: Accelerating adversarial training via maximal principle”, arXiv preprint arXiv:1905.00877 (2019a).

Zhang, H., T. Xu, H. Li, S. Zhang, X. Huang, X. Wang and D. Metaxas, “StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks”, arXiv URL `http://arxiv.org/abs/1612.03242` (2016a).

Zhang, H., T. Xu, H. Li, S. Zhang, X. Huang, X. Wang and D. Metaxas, "StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks", arXiv URL `http://arxiv.org/abs/1612.03242` (2016b).

Zhang, H., Y. Yu, J. Jiao, E. P. Xing, L. E. Ghaoui and M. I. Jordan, "Theoretically principled trade-off between robustness and accuracy", arXiv preprint arXiv:1901.08573 (2019b).

Zhang, J., M. Marszałek, S. Lazebnik and C. Schmid, "Local features and kernels for classification of texture and object categories: A comprehensive study", International journal of computer vision **73**, 2, 213–238 (2007).

Zhao, J., M. Mathieu and Y. LeCun, "Energy-based Generative Adversarial Network", Nips , 2006, 1–15, URL `http://arxiv.org/abs/1609.03126` (2016).

Zhao, R., R. Yan, Z. Chen, K. Mao, P. Wang and R. X. Gao, "Deep learning and its applications to machine health monitoring", Mechanical Systems and Signal Processing **115**, 213–237 (2019).

Zhu, J.-Y., T. Park, P. Isola and A. A. Efros, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks", arXiv URL `http://arxiv.org/abs/1703.10593` (2017).

APPENDIX A

ANALYTICAL FEA CONVOLUTIONAL KERNELS

We derive the analytical form of the FEA convolutional kernels for different physics problems in this appendix. The geometry matrix B in Eq. 5.3 has an expression of:

$$B = LN \tag{A.1}$$

where L is differential operator.

For simplicity, we choose Δ to be the simplest linear element in Eq. 5.3, which makes N has the form of:

$$N = \frac{1}{4}\left[(1-\xi)(1-\eta) \quad (1+\xi)(1-\eta) \quad (1+\xi)(1+\eta) \quad (1-\xi)(1+\eta)\right] \tag{A.2}$$

For thermal and elasticity problems, we have:

$$B^1 = \begin{bmatrix} [1.5]\frac{\partial N_1^e}{\partial \xi} & \frac{\partial N_2^e}{\partial \xi} & \frac{\partial N_3^e}{\partial \xi} & \frac{\partial N_4^e}{\partial \xi} \\ \frac{\partial N_1^e}{\partial \eta} & \frac{\partial N_2^e}{\partial \eta} & \frac{\partial N_3^e}{\partial \eta} & \frac{\partial N_4^e}{\partial \eta} \end{bmatrix} \tag{A.3}$$

$$B^2 = \begin{bmatrix} [1.5]\frac{\partial N_1^e}{\partial \xi} & \frac{\partial N_2^e}{\partial \xi} & \frac{\partial N_3^e}{\partial \xi} & \frac{\partial N_4^e}{\partial \xi} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\partial N_1^e}{\partial \eta} & \frac{\partial N_2^e}{\partial \eta} & \frac{\partial N_3^e}{\partial \eta} & \frac{\partial N_4^e}{\partial \eta} \\ \frac{\partial N_1^e}{\partial \eta} & \frac{\partial N_2^e}{\partial \eta} & \frac{\partial N_3^e}{\partial \eta} & \frac{\partial N_4^e}{\partial \eta} & \frac{\partial N_1^e}{\partial \xi} & \frac{\partial N_2^e}{\partial \xi} & \frac{\partial N_3^e}{\partial \xi} & \frac{\partial N_4^e}{\partial \xi} \end{bmatrix} \tag{A.4}$$

And the constitutional matrix C differs for different problems.

Once we have B and C defined, we can compute each term of the element stiffness matrix by integrating Eq. 5.3. The integral has relatively simple forms in many cases, and analytical solutions can be directly obtained. We use $\hat{K}$ to represent the element stiffness matrix in this appendix to avoid duplication in notation.

A.1 Thermal problem

Thermal problems are governed by Poisson equation:

$$\kappa(u_{,xx} + u_{,yy}) = v \tag{A.5}$$

where u and v denotes temperature and heat flux, and κ is the heat conductivity ratio.

The matrix C has expression:

$$C = \begin{bmatrix} \kappa & 0 \\ 0 & k \end{bmatrix} \tag{A.6}$$

By substituting Eq. A.1 and Eq. A.6 into Eq. 5.3, we have:

$$\hat{K} = \frac{1}{16}\int_{-1}^{1}\int_{-1}^{1} \begin{bmatrix} -(\xi-1)^2-(\eta-1)^2 & \xi^2+\eta^2-2y & \xi^2-\eta^2-2 & \xi^2+\eta^2-2\xi \\ \xi^2+\eta^2-2\eta & -(\xi+1)^2-(\eta-1)^2 & \xi^2+\eta^2+2\xi & \xi^2+\eta^2-2 \\ \xi^2-\eta^2-2 & \xi^2+\eta^2+2\xi & -(\xi+1)^2-(\eta+1)^2 & \xi^2+\eta^2+2\eta \\ \xi^2+\eta^2-2\xi & \xi^2+\eta^2-2 & \xi^2+\eta^2+2\eta & -(\xi-1)^2-(\eta+1)^2 \end{bmatrix} d\xi d\eta \tag{A.7}$$

This integration can be computed analytically:

$$\hat{K} = \frac{\kappa}{6}\begin{bmatrix} -4 & 1 & 2 & 1 \\ 1 & -2 & 1 & 2 \\ 2 & 1 & -4 & 1 \\ 1 & 2 & 1 & -4 \end{bmatrix} \tag{A.8}$$

By further substituting to Eq. 5.8, we have the FEA convolutional kernel for the thermal problem:

$$W^{tt} = \frac{\kappa}{3}\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{A.9}$$

A.2 Elasticity problem

Because both the loading and response for 2D elasticity have both x and y component, the FEA convolution filter $W \in R^{(3,3,2,2)}$ which has 2 input channels and two output channels.

2D plane elasticity problems are governed by the following equilibrium equation:

$$C\nabla^2 u + b = 0 \tag{A.10}$$

where u is the temperature and b is the body force. The matrix C for plane elasticity has expression:

$$C = \frac{E}{1-\nu^2}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix} \tag{A.11}$$

By substituting the constitutional matrix C in Eq. A.11 and the geometry matrix B in Eq. A.3 into Eq. 5.3, we can obtain the corresponding element stiffness matrix:

$$\hat{K} = \frac{E}{16(1-\nu^2)}\begin{bmatrix} 8-\frac{8}{3}\nu & 2\nu+2 & -\frac{4}{3}\nu-4 & 6\nu-2 & \frac{4}{3}\nu-4 & -2\nu-2 & \frac{8}{3}\nu & 2-6\nu \\ -\frac{4}{3}\nu-4 & 2-6\nu & 8-\frac{8}{3}\nu & -2\nu-2 & \frac{8}{3}\nu & 6\nu-2 & \frac{4}{3}\nu-4 & 2\nu+2 \\ \frac{4}{3}\nu-4 & -2\nu-2 & \frac{8}{3}\nu & 2-6\nu & 8-\frac{8}{3}\nu & 2\nu+2 & -\frac{4}{3}\nu-4 & 6\nu-2 \\ \frac{8}{3}\nu & 6\nu-2 & \frac{4}{3}\nu-4 & 2\nu+2 & -\frac{4}{3}\nu-4 & 2-6\nu & 8-\frac{8}{3}\nu & -2\nu-2 \\ 2\nu+2 & 8-\frac{8}{3}\nu & 2-6\nu & \frac{8}{3}\nu & -2\nu-2 & -\frac{4}{3}\nu-4 & 6\nu-2 & \frac{4}{3}\nu-4 \\ 6\nu-2 & \frac{8}{3}\nu & -2\nu-2 & 8-\frac{8}{3}\nu & 2-6\nu & -\frac{4}{3}\nu-4 & 2\nu+2 & \frac{4}{3}\nu-4 \\ -2\nu-2 & -\frac{4}{3}\nu-4 & 6\nu-2 & \frac{4}{3}\nu-4 & 2\nu+2 & 8-\frac{8}{3}\nu & 2-6\nu & \frac{8}{3}\nu \\ 2-6\nu & -\frac{4}{3}\nu-4 & 2\nu+2 & \frac{4}{3}\nu-4 & 6\nu-2 & \frac{8}{3}\nu & -2\nu-2 & 8-\frac{8}{3}\nu \end{bmatrix} \tag{A.12}$$

where the first and second half of the rows (and columns) corresponds to x directional response (and loading). We will start by considering only the relationship between x directional loading and x directional response by extracting the entries from the upper-left section of the matrix:

$$\hat{K}^{xx} = \frac{E}{16(1-\nu^2)}\begin{bmatrix} 8-\frac{8}{3}\nu & -\frac{4}{3}\nu-4 & \frac{4}{3}\nu-4 & \frac{8}{3}\nu \\ -\frac{4}{3}\nu-4 & 8-\frac{8}{3}\nu & \frac{8}{3}\nu & \frac{4}{3}\nu-4 \\ \frac{4}{3}\nu-4 & \frac{8}{3}\nu & 8-\frac{8}{3}\nu & -\frac{4}{3}\nu-4 \\ \frac{8}{3}\nu & \frac{4}{3}\nu-4 & -\frac{4}{3}\nu-4 & 8-\frac{8}{3}\nu \end{bmatrix} \tag{A.13}$$

Based on Eq. 5.8, we can find the FEA convolutional kernel for x directional loading and response:

$$W^{xx} = \frac{E}{4(1-\nu^2)}\begin{bmatrix} -(1-\nu/3) & 4\nu/3 & -(1-\nu/3) \\ -2(1+\nu/3) & 8(1-\nu/3) & -2(1+\nu/3) \\ -(1-\nu/3) & 4\nu/3 & -(1-\nu/3) \end{bmatrix} \tag{A.14}$$

Similarly, the relationship between x directional loading and y directional response W_{xy} can be obtained from the upper right section of Eq. A.12, the relationship between y directional loading and y directional response W^{yy} can be obtained lower right section, and the relationship between y directional loading and x directional response W^{yx} can be obtained from the lower-left section. Since similar approach is used, we skip the repeated derivation and give their expressions directly:

$$\begin{aligned}
W^{xy} = W^{yx} &= \frac{E}{8(1-\nu)} \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix} \\
W^{yy} &= \frac{E}{4(1-\nu^2)} \begin{bmatrix} -(1-\nu/3) & -2(1+\nu/3) & -(1-\nu/3) \\ 4\nu/3 & 8(1-\nu/3) & 4\nu/3 \\ -(1-\nu/3) & -2(1+\nu/3) & -(1-\nu/3) \end{bmatrix} \\
W^{yy} &= \frac{E}{4(1-\nu^2)} \left(\begin{bmatrix} -1 & -2 & -1 \\ 0 & 8 & 0 \\ -1 & -2 & -1 \end{bmatrix} + \frac{\nu}{3} \begin{bmatrix} 1 & -2 & 1 \\ 4 & -8 & 4 \\ 1 & -2 & 1 \end{bmatrix} \right)
\end{aligned} \tag{A.15}$$

A.3 Thermoelasticity problem

The equilibrium equation of the coupled thermoelastic problems can be expressed as the following tensor form:

$$\frac{1}{2} E_{ijkl}(u_{k,lj} + u_{l,kj}) - E_{ijkl}\alpha\delta_{kl}\Delta T_{,j} + b_i = 0 \tag{A.16}$$

where u is the displacement and b is the external body force. α is the thermal expansion coefficient of the isotropic materials. E_{ijkl} is the elastic tensor. By discretization, the matrix form of finite element analysis can be obtained,

$$\begin{bmatrix} K^u & K^{ut} \\ 0 & K^t \end{bmatrix} \begin{bmatrix} u \\ T \end{bmatrix} = \begin{bmatrix} F \\ Q \end{bmatrix} \tag{A.17}$$

The non-coupled stiffness matrix K^u and K^t are the same as previous ones. Only the coupling term K^{ut} is shown here,

$$\hat{K}^{ut} = \frac{\alpha E}{16(\nu-1)} \int_{\Delta} \begin{bmatrix}
(\xi-1)(\eta-1)^2 & -(\xi+1)(\eta-1)^2 & (\eta^2-1)(\xi+1) & -(\eta^2)(\xi-1) \\
(\xi-1)^2(\eta-1) & -(\xi^2-1)(\eta-1) & (\xi^2-1)(\eta+1) & -(\xi-1)^2(\eta+1) \\
-(\xi-1)(\eta-1)^2 & (\xi+1)(\eta-1)^2 & -(\eta^2-1)(\xi+1) & (\eta^2-1)(\xi-1) \\
-(\xi^2-1)(\eta-1) & (\xi+1)^2(\eta-1) & -(\xi+1)^2(\eta+1) & (\xi^2-1)(\eta+1) \\
(\eta^2-1)(\xi-1) & -(\eta^2-1)(\xi+1) & (\xi+1)(\eta+1)^2 & -(\xi-1)(\eta+1)^2 \\
(\xi^2-1)(\eta-1) & -(\xi+1)^2(\eta-1) & (\xi+1)^2(\eta+1) & -(\xi^2-1)(\eta+1) \\
-(\eta^2-1)(\xi-1) & (\eta^2-1)(\xi+1) & -(\xi+1)(\eta+1)^2 & (\xi-1)(\eta+1)^2 \\
-(\xi-1)^2(\eta-1) & (\xi^2-1)(\eta-1) & -(\xi^2-1)(\eta+1) & (\xi-1)^2(\eta+1)
\end{bmatrix} d\Omega \tag{A.18}$$

After integration on [-1,1], we have:

$$\hat{K} = \frac{\alpha E}{6(1-\nu)} \begin{bmatrix} -2 & -2 & -1 & -1 \\ -2 & -1 & -1 & -2 \\ 2 & 2 & 1 & 1 \\ -1 & -2 & -2 & -1 \\ 1 & 1 & 2 & 2 \\ 1 & 2 & 2 & 1 \\ -1 & -1 & -2 & -2 \\ 2 & 1 & 1 & 2 \end{bmatrix} \tag{A.19}$$

where odd and rows corresponds to x and y directional elasticity response. By extracting the entries from the odd rows, the relationship between x directional loading and heat flux can be obtained:

$$\hat{K}^{xt} = \frac{\alpha E}{6(1-\nu)} \begin{bmatrix} -2 & -2 & -1 & -1 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 2 \\ -1 & -1 & -2 & -2 \end{bmatrix} \tag{A.20}$$

Based on Eq. 5.8, we can find it FEA convolutional kernel:

$$W^{xt} = \frac{\alpha E}{6(1-\nu)} \begin{bmatrix} -1 & 0 & 1 \\ -4 & 0 & 4 \\ -1 & 0 & 1 \end{bmatrix} \tag{A.21}$$

Similarly, the FEA convolutional kernel for the relationship can be obtained as:

$$W^{xt} = \frac{\alpha E}{6(1-\nu)} \begin{bmatrix} 1 & 4 & 1 \\ 0 & 0 & 0 \\ -1 & -4 & -1 \end{bmatrix} \tag{A.22}$$

APPENDIX B

FORWARD INFERENCE RELATED PROOF

We use u^* (and $\hat{u}$) to denote the part where its value is known (and unknown). In this way, Eq. 5.2 can be split as:

$$\left[\begin{array}{c:c} K_{00} & K_{01} \\ \hdashline K_{10} & K_{11} \end{array}\right] \left[\begin{array}{c} \hat{u} \\ \hdashline u^* \end{array}\right] = \left[\begin{array}{c} v^* \\ \hdashline \hat{v} \end{array}\right] \tag{B.1}$$

The boundary condition operator $\mathcal{B}$ is defined as:

$$\mathcal{B} \left[\begin{array}{c} u_1 \\ \hdashline u_2 \end{array}\right] = \left[\begin{array}{c} u_1 \\ \hdashline u^* \end{array}\right] \tag{B.2}$$

which makes:

$$\mathcal{B}\left(\left[\begin{array}{c:c} K_{00} & K_{01} \\ \hdashline K_{10} & K_{11} \end{array}\right] \left[\begin{array}{c} u_1 \\ \hdashline u_2 \end{array}\right] \right) = \left[\begin{array}{c:c} K_{00} & K_{01} \\ \hdashline 0 & I \end{array}\right] \left[\begin{array}{c} u_1 \\ \hdashline u^* \end{array}\right] \tag{B.3}$$

This is equivalent to solving:

$$\left[\begin{array}{c:c} K_{00} & K_{01} \\ \hdashline 0 & I \end{array}\right] \left[\begin{array}{c} \hat{u} \\ \hdashline u^* \end{array}\right] = \left[\begin{array}{c} v^* \\ \hdashline u^* \end{array}\right] \tag{B.4}$$

It is obvious that Eq. B.1 and Eq. B.4 actually have the same solution. Thus, FEA-Net will have exactly the same convergence as the numerical solver to its corresponding FEA problem.

APPENDIX C

LEARNING WITH MULTI-PHYSICS RELATED PROOF

The objective of training FEA-Net on linear physics boils down to learning the filter W given observed (V, U) pair:

$$\arg\min_{W} \|W \circledast U - V\|_2^2 \tag{C.1}$$

Since convolution operation is a linear operation, Eq. C.1 is essentially a linear regression problem. For thermoelasticity, it can be decomposed into three different learning problems:

$$\arg\min_{W^{xx},W^{xy},W^{xt}} \left\|W^{xx} \circledast U^x + W^{xy} \circledast U^y + W^{xt} \circledast U^t - V^x\right\|_2^2 \tag{C.2a}$$

$$\arg\min_{W^{yx},W^{yy},W^{yt}} \left\|W^{yx} \circledast U^x + W^{yy} \circledast U^y + W^{yt} \circledast U^t - V^y\right\|_2^2 \tag{C.2b}$$

$$\arg\min_{W^{tx},W^{ty},W^{tt}} \left\|W^{tx} \circledast U^x + W^{ty} \circledast U^y + W^{tt} \circledast U^t - V^t\right\|_2^2 \tag{C.2c}$$

Consider optimizing Eq. C.2a for example, this optimization problem is equivalent to finding the least mean square solution of:

$$W^{xx} \circledast U^x + W^{xy} \circledast U^y + W^{xt} \circledast U^t = V^x \tag{C.3}$$

which can actually be re-organized into a matrix form:

$$\mathbf{U} \cdot \overrightarrow{w} = \overrightarrow{v} \tag{C.4}$$

where $\mathbf{U} \in R^{(n^2,27)}$ and $\overrightarrow{w} \in R^{(27,1)}$ are in the form of:

$$\mathbf{U} = [\mathbf{U}^x, \mathbf{U}^y, \mathbf{U}^t] \tag{C.5a}$$

$$\overrightarrow{w} = [\overrightarrow{w^x}, \overrightarrow{w^y}, \overrightarrow{w^t}]^T \tag{C.5b}$$

and their components have an expression of:

$$\overrightarrow{w^x} = [W^{xx}_{11}, W^{xx}_{12}, W^{xx}_{13}, W^{xx}_{21}, W^{xx}_{22}, W^{xx}_{23}, W^{xx}_{31}, W^{xx}_{32}, W^{xx}_{33}] \tag{C.6a}$$

$$\mathbf{U}^x = [\mathbf{U}^x_{i-1,j-1}, \mathbf{U}^x_{i-1,j}, \mathbf{U}^x_{i+1,j}, \mathbf{U}^x_{i,j-1}, \mathbf{U}^x_{i,j}, \mathbf{U}^x_{i,j+1}, \mathbf{U}^x_{i+1,j-1}, \mathbf{U}^x_{i+1,j}, \mathbf{U}^x_{i+1,j+1}] \tag{C.6b}$$

There is a total of 27 variables to be learned from Eq. C.4. There are two conditions to ensure the problem is well defined: (1) The number of rows is larger or equal to 27. This means that we need to have the image resolution at least 6-by-6. (2) The coefficient matrix $\mathbf{U}$ is column-wise full rank.

Lemma 2. *If different loading channels are linearly dependent, **U** matrix will not be row-wise full rank.*

Proof. We start by assuming there exists such linear dependence:

$$V^x = c_1 V^y + c_2 V^t \tag{C.7}$$

Substituting it into the relationship between loading images and response images in Theorem 1:

$$\begin{aligned} W^{xx} \circledast U^x + W^{xy} \circledast U^y + W^{xt} \circledast U^t = c_1\big(&W^{yx} \circledast U^y + W^{yy} \circledast U^y + W^{yt} \circledast U^t\big) \\ +c_2\big(&W^{tx} \circledast U^x + W^{ty} \circledast U^y + W^{tt} \circledast U^t\big) \end{aligned} \tag{C.8}$$

after simplification we have:

$$\begin{aligned} (W^{xx} - c_1 W^{xy} - c_2 W^{xt}) \circledast U^x + (W^{yx} - c_1 W^{yy} - c_2 W^{yt}) \circledast U^y \\ +(W^{tx} - c_1 W^{ty} - c_2 W^{tt}) \circledast U^t = 0 \end{aligned} \tag{C.9}$$

which can be further re-organize into matrix form:

$$\mathbf{U} \cdot \overrightarrow{c} = 0 \tag{C.10}$$

where:

$$c = [\overrightarrow{w^x} - c_1\overrightarrow{w^y} - c_2\overrightarrow{w^t}, \overrightarrow{w^x} - c_1\overrightarrow{w^y} - c_2\overrightarrow{w^t}, \overrightarrow{w^x} - c_1\overrightarrow{w^y} - c_2\overrightarrow{w^t}]^t \tag{C.11}$$

Thus, matrix $\mathbf{U}$ has column-wise correlation and is not of column-wise full rank. □

For thermoelasticity specifically, we have $W^{xt} = W^{yt} = 0$ and the condition for rank deficiency in Lemma 2 can be further simplified. Since $\overrightarrow{w^t} = 0$, as long as $V^x = c_1 V^y$, the system $\mathbf{U}$ will have multicollinearility. Physically, that means we can not have the loading pointing towards one direction in obtaining the training data.

APPENDIX D

LEARNING WITH MULTI-PHASE RELATED PROOF

D.1 Estimating material phase

From Eq. 5.14 we can see that Eq. 5.26 is a linear function of H if the other variables (ρ, V, U are known). Furthermore, Eq. 5.27 will become a quadratic programming problem if L_2 error measurement is used. Thus, the solution to Problem 5.4.1 is unique. Since the governing PDE is the same everywhere in Ω, the objective Eq. 5.27 also holds for any $\Phi \subset \Omega$. Thus, the material phase in Φ can be obtained from:

$$H^*(q) = \arg\min_H f(\rho, H, V(q), U(q)) \tag{D.1}$$

This means that the learning of material phase can be successful for arbitrary image size.

D.2 Estimating material property

It is obvious that both material phases need to get present in the phase image H, otherwise the other material property will not get involved in the optimization. Now we prove the second condition on V. We can see that f is a function that is linearly related to the FEA convolution kernel W. Furthermore, as can be seen from Eq. 5.10a, Eq. 5.10b for homogeneous material and Eq. 5.16, Eq. 5.17 for bi-phase material, the Young's modulus E term can be extracted from the convolutional kernel. In other words, for elasticity problems, Eq. 5.26 is decomposible w.r.t. Young's modulus E:

$$V = f_2(U, H, E, \nu) = E \cdot \hat{f}_2(U, H, \nu) \tag{D.2}$$

If we have $V \equiv 0$, there will be two possible solutions: $E \equiv 0$ or $\hat{f}(U, H, \nu) \equiv 0$. Thus, we need to have $V(q)$ has non-zero value(s) in order to learn the correct solution.

Learning material property with Eq. 5.28 can be very data efficient: (1) Suppose the material is homogeneous, we have $\rho \in \mathbb{R}^2$ for elasticity problems. In this case, the optimization problem will be well defined if the number of constraints is larger or equal to 2. (2) Suppose that there exist two phases. In this case, we have $\rho \in \mathbb{R}^4$ for elasticity problems. Thus, the resolution of V needs to be larger than 2x2, and at least one element needs to be non-zero. In either case, only a single loading-response pair can be sufficient to define the optimization problem.