UNIVERSITY OF CALIFORNIA,
IRVINE

Quantization of Neural Networks

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Mathematics

by

Ziang Long

2022

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

Firstly, I would like to express my gratitude to my advisor Prof. Jack Xin for the continuous and unconditional support throughout the five years of my Ph.D. study. I could not appreciate more the guidance and the opportunity and could not have imagined a better mentor.

I also would like to thank Prof Yifeng Yu, my co-advisor, for all the interesting mathematics questions that motivated me, and inspired me.

Besides my advisor, I would like to thank my collaborator, Dr. Penghang Yin, for being a role model as a peer and offering great help on my research.

I also thank Prof Long Chen for his Blog and career advice.

Last but the most importantly, I would like to thank my wife, Dr.Hongxuyang Yu for her love and has been always standing by me.

Portion of the text of Chapter 2 is a reprint of the material as it appears in Global convergence and geometric characterization of slow to fast weight evolution in neural network training for classifying linearly non-separable data, Inverse Problem and Imaging, 2021.

Portion of the text of Chapter 3 is a reprint of the material as it appears in Learning quantized neural nets by coarse gradient method for nonlinear classification, Research in Mathematical Science, 2021.

My work was partially supported by NSF grants IIS-1632935, DMS-1854434, DMS-1924548, and DMS-1924935.

VITA

Ziang Long

EDUCATION

Doctor of Philosophy in Mathematics	**2022**
University of California, Irvine	*Irvine, California*
Master of Science in Mathematical Finance	**2017**
Rutgers University	*New Brunswick, New Jersey*
Bachelor of Science in Mathematics	**2015**
Nankai University	*Tianjin*

RESEARCH EXPERIENCE

Graduate Research Assistant	**2017–2022**
University of California, Irvine	*Irvine, California*

TEACHING EXPERIENCE

Teaching Assistant	**2017–2022**
University of California, Irvine	*Irvine, California*

REFEREED JOURNAL PUBLICATIONS

Global convergence and geometric characterization of slow to fast weight evolution in neural network training for classifying linearly non-separable data **2021**
Inverse Problems and Imaging

Learning quantized neural nets by coarse gradient method for nonlinear classification **2021**
Research in the Mathematical Sciences

ABSTRACT OF THE DISSERTATION

Quantization of Neural Networks

By

Ziang Long

Doctor of Philosophy in Mathematics

University of California, Irvine, 2022

Professor Jack Xin, Chair

We study the dynamics of gradient descent in learning neural networks for classification problems. Unlike in existing works, we consider the linearly non-separable case where the training data of different classes lie in orthogonal subspaces.

Deep neural networks (DNNs) are quantized for efficient inference on resource-constrained platforms. However, training deep learning models with low-precision weights and activations involves a demanding optimization task, which calls for minimizing a stage-wise loss function subject to a discrete set-constraint. While numerous training methods have been proposed, existing studies for full quantization of DNNs are mostly empirical. From a theoretical point of view, we study practical techniques for overcoming the combinatorial nature of network quantization. Specifically, we investigate a simple yet powerful projected gradient-like algorithm for quantizing two-layer convolution networks, by repeatedly moving one step at float weights in the negative direction of a heuristic *fake* gradient of the loss function (so-called coarse gradient) evaluated at quantized weights. For the first time, we prove that under mild conditions, the sequence of quantized weights recurrently visit the global optimum of the discrete minimization problem for training fully quantized network. We also show numerical evidence of the recurrence phenomenon of weight evolution in training quantized deep networks.

Chapter 1

Introduction

1.1 Convergence and Slow-to-Fast Weight Evolution in DNN training for Classifying Linearly Non-Separable Data

Deep neural networks (DNN) have achieved remarkable performances in image and speech classification tasks among other AI applications in recent years; for examples, see [19, 25, 35, 40]. Although there have been numerous theoretical contributions to understand their success, the learning process in the actual network training remains largely empirical. One interesting phenomenon is that over-parametrized DNN's trained by stochastic gradient descent generalize [32, 33] instead of overfitting the training data contrary to conventional statistical learning. Though several convergence results are proved in the over-parameterized regime for deep networks [13, 28, 2], the network weights move only in a small neighborhood of the random initialization and so their dynamics are very localized. Partly, this may be attributed to the exceedingly large number of neurons in convergence theory, far surpassing what is used in practice where the weights evolve significantly from random start through hundreds of epochs in training to reach best prediction accuracy.

Chapter 2 addresses how the weights evolve towards a global minimum of loss function as the number of neurons increases from the feature dimension (the least necessary) to the over-parametrized regime. To facilitate analysis, our model network structure is motivated by [6] on classifying linearly separable data. We instead study a linearly non-separable multi-category classification problem with an emphasis on the dynamics of weights in terms of the two time scales of evolution and a geometric characterization of the transition time. Our training data of the two classes will lie in orthogonal sub-spaces, which extends the data configuration in [5] where the subspace of each class is one dimensional for an XOR detection problem. Orthogonality of input data from the two classes implies that the training process in each class can be analyzed independently of the other. In the one-dimensional case [5], each weight update does not increase the loss on any sample point. In the multi-dimensional case here, we find that during gradient descent weight update, it is not possible that the loss is non-increasing in the point-wise sense (on each input data). Instead, the population loss is decreasing (i.e. in the sense of expectation). The population loss here is based on the hinge loss function and the network activation function is ReLU. Under a mild non-degenerate data condition, we prove that all critical points of our non-convex and non-smooth population loss function are global minima. Similar landscapes (a local minimum is a global minimum) are known for deep networks with activation functions that are either strictly convex [29], or real analytic and strictly increasing [33].

In DNN training, one observes that the network learning consists of alternating phases: plateaus where the validation error remains fairly constant and periods of rapid improvement where a lot of progress is made over a few epochs. Prior to our work, [11] studied slow and fast weight dynamics in a solvable model while minimizing a binary cross entropy or hinge loss function on linearly separable data.

In the regression context, [4] came across such two time-scale phenomenon in training a two-linear-layer convolutional network with prescribed ground truth and unit Gaussian input data. This particular data assumption makes it possible to readily derive the closed-form expressions of the population loss and gradient, and then analyze the energy landscape and convergence of the gradient descent algorithm.

In chapter 2, we study network weight dynamics in training a one-hidden-layer ReLU network via hinge loss minimization on multi-category classification of linearly non-separable data lying in n orthogonal sub-spaces. Our main contributions are:

- We discovered a geometric condition (GC) to characterize the transition time T from the first (slow) phase of weight evolution to the second fast weight convergence. The condition says that the convex hull of the weights on the unit sphere contains the origin, see Fig. 2.1 for an illustration. Equivalent geometric conditions are also derived (Lemma 2.4). In the first (slow) phase, the weight directions spread out over the unit sphere to satisfy GC.

- We obtain upper bound on T in terms of data distribution function provided that the network weights are uniformly bounded during training which we observed numerically.

- We give probabilistic bounds on the validity of geometric condition for random initialization, which suggests that the larger the number of neurons, the more likely GC holds and the earlier the fast phase of evolution begins.

- We prove the global convergence of gradient descent training algorithm under the uniformly bounded weight assumption. In case of positive network bias, we prove a global Lipschitz gradient property of the loss function and sub-sequential convergence of weights to a global minimum. In case of zero network bias, we prove that the loss function has Lipschitz gradient away from the origin and is piece-wise C^1.

- We prove that all critical points of the population loss function are global minima under a non-degenerate data condition.
- We provide numerical examples to substantiate our theory, extend the data assumption, and illustrate the weight dynamics as the network size increases towards the over-parametrized regime. We visualize the feature and weight vectors in DNNs on MNIST data in connection with our model findings.

1.2 Learning Quantized Neural Nets by Coarse Gradient Method

Deep neural networks (DNNs) have achieved remarkable success in a number of domains including computer vision [25, 35], reinforcement learning [31, 40] and natural language processing [9]. However, due to the huge number of model parameters, the deployment of DNNs can be computationally and memory intensive. As such, it remains a great challenge to deploy DNNs on mobile electronics with low computational budget and limited memory storage.

To address this challenge, research efforts have been made to the quantizing weights and activations of DNNs while maintaining their performance. Quantization methods train DNNs with the weights and activation values being constrained to low-precision arithmetic rather than the conventional floating-point representation in full-precision. [22, 51, 7, 50, 30, 52], which offer the feasibility of running DNNs on CPUs rather than GPUs in real-time. For example, the XNOR-Net [34] with binary weights and activations sees 58$\times$ faster convolutional operations and 32$\times$ memory savings.

The approximation power of weight quantized DNNs was investigated in [15, 12], while the recent paper [39] studies the approximation power of DNNs with discretized activations. On the computational side, training quantized DNNs typically calls for solving a large-scale optimization problem, yet with extra computational and mathematical challenges. Although people often quantize both the weights and activations of DNNs, they can be viewed as two relatively independent subproblems. Weight quantization basically introduces an additional set-constraint that characterizes the quantized model parameters, which can be efficiently carried out by projected gradient type methods [10, 26, 27, 48, 20, 46]. Activation quantization (i.e., quantizing ReLU), on the other hand, involves a stair-case activation function with zero derivative almost everywhere (a.e.) in place of the sub-differentiable ReLU. Therefore, the resulting composite loss function is piece-wise constant and cannot be minimized via the (stochastic) gradient method due to the vanished gradient.

To overcome this issue, a simple and hardware friendly approach is to use a straight-through estimator (STE) [17, 3, 44]. More precisely, one can replace the a.e. zero derivative of quantized ReLU with an ad-hoc surrogate in the backward pass, while keeping the original quantized function during the forward pass. Mathematically, STE gives rise to a *biased* first-order oracle computed by an unusual chain rule. This first-order oracle is not the gradient of the original loss function because there exists a mismatch between the forward and backward passes. Throughout Chapter 3 and Chapter 4, this STE-induced type of "gradient" is called coarse gradient. While coarse gradient is not the true gradient, in practice it works as it miraculously points towards a descent direction (see [44] for a thorough study in the regression setting). Moreover, coarse gradient has the same computational complexity as standard gradient. Just like the standard gradient descent, the minimization procedure of training activation quantized networks simply proceeds by repeatedly moving one step at current point in the opposite of coarse gradient with some step size. The performance of the resulting coarse gradient method, e.g. convergence property, naturally relies on the choice of STE. How to choose a proper STE so that the resulting training algorithm is provably convergent is still poorly understood, especially in the nonlinear classification setting.

The idea of STE dated back to the classical perceptron algorithm [37, 38] for binary classification. Specifically, the perceptron algorithm attempts to solve the empirical risk minimization problem:

$$\min_{\boldsymbol{w}} \sum_{i=1}^{N} (\mathrm{sign}(\boldsymbol{x}_i^\top \boldsymbol{w}) - y_i)^2, \tag{1.1}$$

where $(\boldsymbol{x}_i, y_i)$ is the i^{th} training sample with $y_i \in \{\pm 1\}$ being a binary label; for a given input $\boldsymbol{x}_i$, the single-layer perceptron model with weights $\boldsymbol{w}$ outputs the class prediction $\mathrm{sign}(\boldsymbol{x}_i^\top \boldsymbol{w})$. To train perceptrons, Rosenblatt [37] proposed the following iteration for solving (1.1) with the step size $\eta > 0$:

$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^t - \eta \sum_{i=1}^{N} (\mathrm{sign}(\boldsymbol{x}_i^\top \boldsymbol{w}^t) - y_i) \cdot \boldsymbol{x}_i, \tag{1.2}$$

We note that the above perceptron algorithm is not the same as gradient descent algorithm. Assuming the differentiability, the standard chain rule computes the gradient of the i^{th} sample loss function by

$$(\mathrm{sign}(\boldsymbol{x}_i^\top \boldsymbol{w}^t) - y_i) \cdot (\mathrm{sign})'(\boldsymbol{x}_i^\top \boldsymbol{w}^t) \cdot \boldsymbol{x}_i. \tag{1.3}$$

Comparing (1.3) with (1.2), we observe that the perceptron algorithm essentially uses a coarse (and fake) gradient as if $(\mathrm{sign})'$ composited in the chain rule was the derivative of identity function being the constant 1.

The idea of STE was extended to train deep networks with binary activations [17]. Successful experimental results have demonstrated the effectiveness of the empirical STE approach. For example, [3] proposed a STE variant which uses the derivative of sigmoid function instead of identity function. [21] used the derivative of hard tanh function, i.e., $1_{\{|x|\leq 1\}}$, as an STE in training binarized neural networks. To achieve less accuracy degradation, STE was later employed to train DNNs with quantized activations at higher bit-widths [22, 51, 7, 8, 47], where some other STEs were proposed including the derivatives of standard ReLU ($\max\{x, 0\}$) and clipped ReLU ($\min\{\max\{x, 0\}, 1\}$).

Regarding the theoretical justification, it has been established that the perceptron algorithm in (1.2) with identity STE converges and perfectly classifies linearly separable data; see for examples [43, 14] and references therein. Apart from that, to our knowledge, there had been almost no theoretical justification of STE until recently: [44] considered a two-linear-layer network with binary activation for regression problems. The training data is assumed to be instead linearly non-separable, being generated by some underlying model with true parameters. In this setting, [44] proved that the working STE is actually non-unique and that the coarse gradient algorithm is descent and converges to a valid critical point if choosing the STE to be the proxy derivative of either ReLU (i.e., $\max\{x, 0\}$) or clipped ReLU function (i.e., $\min\{\max\{x, 0\}, 1\}$). Moreover, they proved that the identity STE fails to give a convergent algorithm for learning two-layer networks, although it works for single-layer perception.

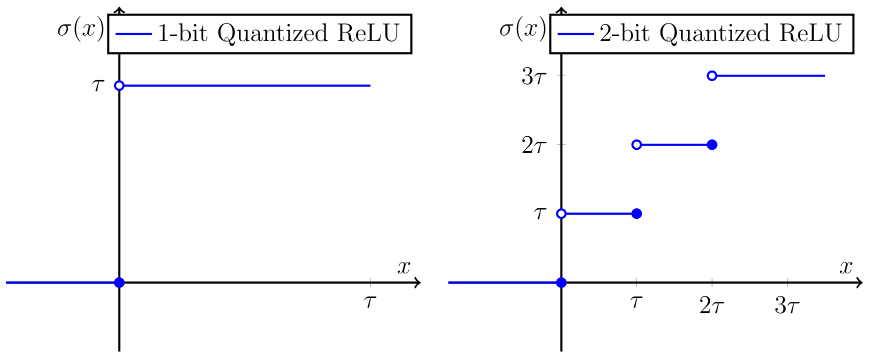


Figure 1.1: Quantized activation functions. τ is a value determined in the network training; see section 3.7

Fig. 1.1 shows examples of 1-bit (binary) and 2-bit (ternary) activations. We see that a quantized activation function zeros out any negative input, while being increasing on the positive half. Intuitively, a working surrogate of the quantized function used in backward pass should also enjoy this monotonicity, as conjectured by [44] which proved the effectiveness of coarse gradient for two specific STEs: derivatives of ReLU and clipped ReLU, and for binarized activation. In Chapter 3, we take a further step towards understanding the convergence of coarse gradient methods for training networks with *general quantized activations* and for *classification of linearly non-separable data. A major analytical challenge we face here is that the network loss function is not in closed analytical form, in sharp contrast to [44].* We present more general results to provide meaningful guidance on how to choose STE in activation quantization. Specifically, we study multi-category classification of linearly non-separable data by a two-linear-layer network with multi-bit activations and hinge loss function. We establish the convergence of coarse gradient methods for a broad class of surrogate functions. More precisely, if a function $g : \mathbb{R} \to \mathbb{R}$ satisfies the following properties:

- $g(x) = 0$ for all $x \leq 0$,
- $g'(x) \geq \delta > 0$ for all $x > 0$ with some constant δ,

then with proper learning rate, the corresponding coarse gradient method converges and perfectly classifies the non-linear data when g' serves as the STE during the backward pass. This gives the affirmation of a conjecture in [44] regarding good choices of STE

Training fully quantized DNN requires solving a challenging optimization problem with piecewise constant (and non-convex) training loss functions and a discrete set-constraint. That is, one considers the following constrained optimization problem for training quantized neural nets:

$$\min_{\boldsymbol{w}} \; f(\boldsymbol{w}) := \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[\ell(\boldsymbol{w};\boldsymbol{x})] \quad \text{subject to} \quad \boldsymbol{w} \in \mathcal{Q} \tag{1.4}$$

where $\ell(\boldsymbol{w};\boldsymbol{x})$ is the loss function for sample $\boldsymbol{x}$, which is *discrete-valued* as non-linear activations are also quantized; $\mathcal{Q}$ is the set of quantized weights. For general constrained minimization, the classical projected gradient descent (PGD):

$$\boldsymbol{w}^{t+1} = \mathrm{proj}_{\mathcal{Q}}\left(\boldsymbol{w}^t - \eta_t\,\mathbb{E}[\nabla_{\boldsymbol{w}}\ell(\boldsymbol{w}^t;\boldsymbol{x})]\right)$$

is considered. Here $\mathrm{proj}_{\mathcal{Q}}$ is the projection onto set $\mathcal{Q}$ for quantizing float weights to ones at low bit-width, giving a weight quantization scheme. However, with quantized activations, the gradient of loss function $\nabla_{\boldsymbol{w}}\ell(\boldsymbol{w};\boldsymbol{x})$ is almost everywhere (a.e.) zero, leaving the standard back-propagation and hence PGD inapplicable.

In Chapter 4, we study the following iterative algorithm for training fully quantized networks

$$\begin{cases}\boldsymbol{y}^{t+1} = \boldsymbol{y}^t - \eta_t\,\mathbb{E}[\tilde{\nabla}_{\boldsymbol{w}}\ell(\boldsymbol{w}^t;\boldsymbol{x})]\\ \boldsymbol{w}^{t+1} = \mathrm{proj}_{\mathcal{Q}}(\boldsymbol{y}^{t+1}),\end{cases}\tag{QUANT}$$

where $\tilde{\nabla}_{\boldsymbol{w}}\ell$ denotes some heuristic modification of the vanished $\nabla_{\boldsymbol{w}}\ell$ based on the so-called straight-through estimator (STE) [3, 17], rendering a valid search direction. Compared with PGD which can be recast as the two-step iteration:

$$\begin{cases}\boldsymbol{y}^{t+1} = \boldsymbol{w}^t - \eta_t\,\mathbb{E}[\nabla_{\boldsymbol{w}}\ell(\boldsymbol{w}^t;\boldsymbol{x})]\\ \boldsymbol{w}^{t+1} = \mathrm{proj}_{\mathcal{Q}}(\boldsymbol{y}^{t+1})\end{cases}\tag{PGD}$$

another key difference is that, in the gradient step, float weights $\boldsymbol{y}^{t+1}$ is updated by perturbing $\boldsymbol{y}^t$ instead of the current projection $\boldsymbol{w}^t$.

Chapter 2

Convergence and Slow-to-Fast Weight Evolution in DNN training for classifying Linearly Non-Separable Data

2.1 Problem Setup

In this chapter, we consider the multi-category classification problem in the d-dimensional space $\mathcal{X} = \mathbb{R}^d$. Let $\mathcal{Y} = [n] := \{1, 2, \cdots, n\}$ be the set of labels, and $\{\mathcal{D}_i\}_{i=1}^n$ be n probabilistic distributions over $\mathcal{X} \times \mathcal{Y}$. Throughout the theoretical analysis of this chapter, we make the following assumptions on the data:

1. **(Separability)** There are n orthogonal subspaces $V_i \subseteq \mathcal{X}$ for $i \in [n]$ with $\bigoplus_{i=1}^n V_i = \mathcal{X}$,

such that

$$\mathbb{P}_{(\boldsymbol{x},y)\sim\mathcal{D}_i}\left[\boldsymbol{x}\in V_i \quad\text{and}\quad y=i\right]=1.$$

2. **(Boundedness of data)** For $i\in[n]$, There exist positive constants m_i and M_i, such that

$$\mathbb{P}_{(\boldsymbol{x},y)\sim\mathcal{D}_i}\left[m_i\le|\boldsymbol{x}|\le M_i\right]=1.$$

3. **(Boundedness of p.d.f.)** For $i\in[n]$, let p_i be the probability density function of distribution $\mathcal{D}_i$ restricted on V_i. For any $\boldsymbol{x}\in V_i$ with $m_i<|\boldsymbol{x}|<M_i$, it holds that

$$0<p_{\min}\le p_i(\boldsymbol{x})\le p_{\max}<\infty.$$

Later on, we denote $\mathcal{D}$ to be the evenly mixed distribution of $\mathcal{D}_i$'s. For notation simplicity, we let $m=\min\limits_{i\in[n]} m_i$, $M=\max\limits_{i\in[n]} M_i$ and $d_i=\dim V_i$.

We consider a two-layer neural network with k hidden neurons. Denote by $\boldsymbol{W}=[\boldsymbol{w}_1,\cdots,\boldsymbol{w}_k]\in\mathbb{R}^{d\times k}$ the weight matrix in the hidden layer. For any input data $\boldsymbol{x}\in\mathcal{X}=\mathbb{R}^d$, we have

$$h_j=\langle\boldsymbol{w}_j,\boldsymbol{x}\rangle-b_j \quad\text{and}\quad f_i=\sum_{j=1}^{n}v_{i,j}\sigma(h_j)$$

and the neural net outputs

$$f\left(\boldsymbol{W};\boldsymbol{x}\right)=\boldsymbol{V}\sigma\left(\boldsymbol{W}^\top\boldsymbol{x}\right)=\left[f_1,\cdots,f_n\right]^\top, \tag{2.1}$$

where $\sigma:=\max(\cdot,0)$ is the ReLU function acting element-wise, and the bias $b_j\ge 0$ and $\boldsymbol{V}=(v_{i,j})$ are constants. Throughout this chapter, we assume the following:

Assumption 2.1. $\boldsymbol{V}=(v_{i,j})\in\mathbb{R}^{k\times n}$ *satisfies*

1. *For any $i\in[n]$, there exists some $j\in[n]$ such that $v_{i,j}>0$.*

2. *If $v_{i,j} > 0$ then $v_{r,j} < 0$ for all $r \neq i$ and $r \in [n]$.*

3. *There exists some constant $v > 0$ such that $|v_{i,j}| = v$.*

One can show that as long as $k \geq n$, such $\boldsymbol{V}$ can be constructed easily.

The prediction is given by the maximum coordinate index of the network output

$$\hat{y}\left(\boldsymbol{W}; \boldsymbol{x}\right) = \arg\max i \in [n] f_i,$$

ideally $\hat{y}(x) = i$ if $x \in V_i$. The classification accuracy in percentage is the frequency that this occurs (when network output label $\hat{y}$ matches the true label) on a validation data set. Given the data sample $\{\boldsymbol{x}, y\}$, the associated hinge loss function reads

$$l(\boldsymbol{W}; \{\boldsymbol{x}, y\}) := \sum_{i \neq y} \max\left\{0, 1 - f_y + f_i\right\}. \tag{2.2}$$

For network training, we consider the gradient descent algorithm with step size $\eta > 0$

$$\boldsymbol{W}^t = \boldsymbol{W}^{t-1} - \eta \nabla l(\boldsymbol{W}^{t-1}) \tag{2.3}$$

to solve following population loss minimization problem

$$\min_{\boldsymbol{W} \in \mathbb{R}^{d \times k}} l\left(\boldsymbol{W}\right) = \mathop{\mathbb{E}}_{\{\boldsymbol{x}, y\} \sim \mathcal{D}} \left[l\left(\boldsymbol{W}; \{\boldsymbol{x}, y\}\right)\right], \tag{2.4}$$

where the sample loss function $l\left(\boldsymbol{W}; \{\boldsymbol{x}, y\}\right)$ is given by (2.2). Let l_i be the population loss function of data type i. More precisely,

$$l_i(\boldsymbol{W}) = \mathop{\mathbb{E}}_{(\boldsymbol{x}, y) \sim \mathcal{D}_i} \left[l\left(\boldsymbol{W}; \{\boldsymbol{x}, y\}\right)\right] = \mathbb{E}(\boldsymbol{x}, y) \sim \mathcal{D}_i \sum_{r \neq i} \sigma\left(1 - f_i + f_r\right).$$

Thus, we can rewrite the loss function as

$$l(\boldsymbol{W}) = \frac{1}{n}\sum_{i=1}^{n} l_i(\boldsymbol{W}).$$

Note that the population loss function

$$l_i(\boldsymbol{W}) = \sum_{r \neq i} \int_{\{f_i < f_r + 1\}} (1 - f_i + f_r)\, p_i(\boldsymbol{x}) \; d\,\boldsymbol{x}$$

has no closed-form solution even if p is a constant function on its support. We cannot use closed-form formula to analyze the learning process, which makes our work different from many other works.

2.2 Preliminaries

2.2.1 Decomposition

Lemma 2.1. *1. For any $i \in [n]$, if $\boldsymbol{W}_i^* \in \mathbb{R}^{d\times k}$ solves the optimization problem*

$$\min_{\boldsymbol{W} \in V_i^k} \; l_i(\boldsymbol{W}),$$

then $\boldsymbol{W}^* = \sum_{i=1}^{n} \boldsymbol{W}_i^*$ solves the original problem

$$\min_{\boldsymbol{W} \in \mathbb{R}^{d\times k}} \; l(\boldsymbol{W}).$$

2. If $\boldsymbol{W}' = \boldsymbol{W} - \eta \nabla_{\boldsymbol{W}} l_i(\boldsymbol{W})$, then for any $r \neq i$, we have

$$l(\boldsymbol{W}'; \{\boldsymbol{x}, r\}) = l(\boldsymbol{W}; \{\boldsymbol{x}, r\})$$

for almost all $(\boldsymbol{x}, r) \sim \mathcal{D}_r$.

Proof of Lemma 2.1. Note that $\mathcal{X} = \mathbb{R}^d = \bigoplus_{i=1}^n V_i$, we decompose $\boldsymbol{w}_j = \sum_{i=1}^n \boldsymbol{w}_{j,i}$ where $\boldsymbol{w}_{j,i} \in V_i$.

Since V_i's are orthogonal spaces, we have

$$\langle \boldsymbol{w}_j, \boldsymbol{x} \rangle = \langle \boldsymbol{w}_{j,i}, \boldsymbol{x} \rangle$$

if $\boldsymbol{x} \in V_i$. Now, assume $\boldsymbol{x} \in V_i$, let

$$\boldsymbol{W}_i = [\boldsymbol{w}_{1,i}, \cdots, \boldsymbol{w}_{k,i}],$$

we have $f(\boldsymbol{W}; \boldsymbol{x}) = f(\boldsymbol{W}_i, \boldsymbol{x})$ and hence we get our first claim.

On the other hand, we have

$$\nabla_{\boldsymbol{w}_j} l(\boldsymbol{W}; \{\boldsymbol{x}, y\}) = -\sum_{i \neq y} (v_{y,j} - v_{i,j}) \, \mathbb{1}_{\Omega_{y,i}}(\boldsymbol{x}) \mathbb{1}_{\Omega_{\boldsymbol{w}_j}}(\boldsymbol{x}) \boldsymbol{x},$$

where

$$\Omega_{y,i} = \{\boldsymbol{x} : f_y < f_i + 1\} \quad \text{and} \quad \Omega_{\boldsymbol{w}_j} = \{\boldsymbol{x} : \langle \boldsymbol{w}_j, \boldsymbol{x} \rangle > b_j\}.$$

Now, we see that $\nabla_{\boldsymbol{w}_j} l(\boldsymbol{W}; \{\boldsymbol{x}, y\}) \in V_i$ for almost all $(\boldsymbol{x}, i) \sim \mathcal{D}_i$ so that $\boldsymbol{W}_r' = \boldsymbol{W}_r$ for all $r \neq i$ and hence our desired result follows. □

The optimization problem can be decomposed to n independent problems of the same form i.e. the optimization of l_i's. Therefore, it suffices to consider only one subproblem. Let $\boldsymbol{W}_i = [\boldsymbol{w}_1, \cdots, \boldsymbol{w}_k] \in V_i^k$, where $\boldsymbol{w}_j \in V_i = \mathbb{R}^{d_i}$, the network output for the input data

$\boldsymbol{x} \in V_i = \mathbb{R}^{d_i}$ is given by

$$\tilde{f}^i(\boldsymbol{W}_i;\boldsymbol{x}) = [f_1, \cdots, f_n]^\top = \boldsymbol{V}\sigma\left(\boldsymbol{W}_i^\top \boldsymbol{x}\right). \tag{2.5}$$

Networks (2.1) and (2.5) are different, since the parameters in (2.1) are $\boldsymbol{W} \in \mathbb{R}^{d\times k}$, whereas in (2.5), we have $\boldsymbol{W}_i \in V_i^k \cong \mathbb{R}^{d_i \times k}$. The corresponding input data are also in different intrinsic dimensions. From now on, we just focus on the loss function associated with data of Class i:

$$l_i(\boldsymbol{W}) = \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}_i}\left[\sum_{r\neq i}\max\{0, 1 - f_i + f_r\}\right].$$

2.2.2 Landscape

The following Proposition 2.1 shows that while the loss function is non-convex, any critical point is in fact a global minimum, except for some degenerate cases.

Proposition 2.1. *Consider the neural network in (2.5). Assume $d > 1$, if $\boldsymbol{W}$ is a critical point of $l_i(\boldsymbol{W})$ and there exists some $\boldsymbol{x} \in V_i$ such that $f(\boldsymbol{W};\boldsymbol{x}) \neq \boldsymbol{0}$ then we have $l_i(\boldsymbol{W}) = 0$.*

Proof of Proposition 2.1. For any $j \in [k]$, we have

$$\nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}) = -\sum_{r\neq i}^{n}(v_{i,j} - v_{r,j})\mathbb{E}(\boldsymbol{x}, y) \sim \mathcal{D}_i \mathbb{1}_{\Omega_{i,r}}(\boldsymbol{x})\mathbb{1}_{\Omega_{\boldsymbol{w}_j}}(\boldsymbol{x})\boldsymbol{x} = \boldsymbol{0}.$$

Recall the definition of $\Omega_{\boldsymbol{w}_j}$, we know that each summand is a zero vector. By assumption 2.1, we know that either $v_{i,j} = v_{r,j}$ or

$$\mathbb{E}(\boldsymbol{x}, y) \sim \mathcal{D}_i \mathbb{1}_{\Omega_{i,r}}(\boldsymbol{x})\mathbb{1}_{\Omega_{\boldsymbol{w}_j}}(\boldsymbol{x})\boldsymbol{x} = \boldsymbol{0},$$

where the latter implies $\Omega_{i,r} \cap \Omega_{\boldsymbol{w}_j} = \emptyset$. Observe that

$$f_i - f_r = \sum_{j=1}^{k} (v_{i,j} - v_{r,j})\, \sigma(h_j),$$

we see that if there exists some $(\boldsymbol{x}, y) \sim \mathcal{D}_i$ such that $f_i - f_r < 1$ then there must exist some $\boldsymbol{x} \in \Omega_{i,r}$ but this implies $\boldsymbol{x} \notin \Omega_{\boldsymbol{w}_j}$ for all $j \in [k]$ such that $v_{i,j} \neq v_{r,j}$ which gives $f_i = f_r = 0$. This contradicts with our assumption, so we get $f_i - f_r \geq 1$ for all $\boldsymbol{x} \sim \mathcal{D}_i$ and this implies $l_i(\boldsymbol{W}) = 0$. □

The above result holds only when the global minimum of training loss function exists. The following proposition shows that the loss function has plenty of global minima.

Proposition 2.2. *Consider the network in (2.5). If the convex hull spanned by vertices* $\{\boldsymbol{w}_j : v_{i,j} > 0\}$ *contains a ball centered at the origin with radius* $\max\limits_{j\in[k]} \frac{1+b_j}{m_i}$*, and* $\{\boldsymbol{w}_j : v_{i,j} < 0\}$ *lies in a ball with radius* $\min\limits_{j\in[k]} \frac{b_j}{M_i}$*, then* $l_i(\boldsymbol{W}) = 0$*.*

The above proposition shows that if number of neurons is greater than the dimension of input data, then global minimum exists. Next, we study the smoothness of the loss function. The following proposition shows that as long as weights are bounded away from 0, then the loss function has Lipschitz gradient.

Lemma 2.2. *Consider the network in (2.1) with positive bias* $0 < \sum_{j=1}^{k} b_j < 1$*. The loss function* $l(W)$ *is Lipschitz differentiable, i.e, there exists some constant* $L > 0$ *depending on* k*,* $\boldsymbol{b}, p_{max}$*, ,* M*,* $\boldsymbol{V}$*, such that*

$$|\nabla l(\boldsymbol{W}_1) - \nabla l(\boldsymbol{W}_2)| \leq L\, |\boldsymbol{W}_1 - \boldsymbol{W}_2|\,.$$

Proof of Lemma 2.2. Note that $l(\boldsymbol{W}) = \frac{1}{n}\sum_{i=1}^{n} l_i(\boldsymbol{W})$, it suffice to show that each l_i has

Lipschitz gradient. Note that

$$l_i(\boldsymbol{W}) = \sum_{r\neq i} \mathbb{E}(\boldsymbol{x}, y) \sim \mathcal{D}_i \sigma\left(1 - f_i + f_r\right),$$

it suffice to show each summand has Lipschitz gradient. Now, we write the gradient of the summands as

$$\begin{aligned}
&\nabla_{\boldsymbol{w}_j} \mathbb{E}(\boldsymbol{x}, y) \sim \mathcal{D}_i \sigma\left(1 - f_i + f_r\right)\\
=&\mathbb{E}(\boldsymbol{x}, y) \sim \mathcal{D}_i \nabla_{\boldsymbol{w}_j} \sigma\left(1 - f_i + f_r\right)\\
=&\mathbb{E}(\boldsymbol{x}, y) \sim \mathcal{D}_i \mathbb{1}_{\Omega_{i,r}}(\boldsymbol{x}) \mathbb{1}_{\Omega_{\boldsymbol{w}_j}}(\boldsymbol{x}) \left(v_{r,j} - v_{i,j}\right) \boldsymbol{x}.
\end{aligned}$$

On one hand, if $v_{r,j} = v_{i,j}$, then the formula above is obviously zero and Lipschitz gradient follows. On the other hand, we can without loss of generality assume $|v_{i,j} - v_{r,j}| = 1$. For notation simplicity, we fix i and r and denote

$$\varphi(\boldsymbol{W}) := \mathbb{E}(\boldsymbol{x}, y) \sim \mathcal{D}_i \mathbb{1}_{\Omega_{i,r}}(\boldsymbol{x}) \mathbb{1}_{\Omega_{\boldsymbol{w}_j}}(\boldsymbol{x}) \boldsymbol{x} \quad \text{and} \quad \phi(\boldsymbol{W}; \boldsymbol{x}) = f_i(\boldsymbol{W}; \boldsymbol{x}) - f_r(\boldsymbol{W}; \boldsymbol{x}).$$

Now, our desired result becomes $\varphi(\boldsymbol{W})$ is Lipschitz in $\boldsymbol{W}$. Denote $\Omega_1 = \Omega_{i,r}(\boldsymbol{W}_1)$ and $\Omega_2 = \Omega_{i,r}(\boldsymbol{W}_2)$ where $\boldsymbol{W}_i = [\boldsymbol{w}_1^i, \cdots, \boldsymbol{w}_k^i]$, we only need to show there exists some constant L such that

$$\left\|\varphi(\boldsymbol{W}_1) - \varphi(\boldsymbol{W}_2)\right\| \le L \left\|\boldsymbol{W}_1 - \boldsymbol{W}_2\right\|.$$

Note that

$$\begin{aligned}
&\left\|\varphi(\boldsymbol{W}_1) - \varphi(\boldsymbol{W}_2)\right\|\\
=&\left\|\mathbb{E}\mathbb{1}_{\Omega_1}(\boldsymbol{x}) \mathbb{1}_{\Omega_{\boldsymbol{w}_j^1}}(\boldsymbol{x}) \boldsymbol{x} - \mathbb{E}\mathbb{1}_{\Omega_2}(\boldsymbol{x}) \mathbb{1}_{\Omega_{\boldsymbol{w}_j^2}}(\boldsymbol{x}) \boldsymbol{x}\right\|\\
\le&\underbrace{\mathbb{E}\mathbb{1}_{\Omega_1 \Delta \Omega_2}(\boldsymbol{x}) \left\|\boldsymbol{x}\right\|}_{①} + \underbrace{\mathbb{E}\mathbb{1}_{\Omega_{\boldsymbol{w}_j^1} \Delta \Omega_{\boldsymbol{w}_j^2}} \left\|\boldsymbol{x}\right\|}_{②},
\end{aligned}$$

we can deal with ① and ② respectively.

W.l.o.g, we assume $\epsilon = \|\boldsymbol{W}_1 - \boldsymbol{W}_2\| \leq \frac{1}{2}$. On one hand, if $\boldsymbol{x} \in \Omega_1 \Delta \Omega_2$, then we claim

$$1 - \epsilon \|\boldsymbol{x}\| \leq \phi(\boldsymbol{W}_1; \boldsymbol{x}) \leq 1 + \epsilon \|\boldsymbol{x}\|$$

because

$$\|\phi(\boldsymbol{W}_1; \boldsymbol{x}) - \phi(\boldsymbol{W}_2; \boldsymbol{x})\| = \left\| \sum_{j=1}^{k} (v_{i,j} - v_{r,j}) \left[\sigma(h_j^1) - \sigma(h_j^2) \right] \right\|$$
$$\leq \left\| \sum_{j=1}^{k} \left\langle \boldsymbol{w}_j^1 - \boldsymbol{w}_j^2, \boldsymbol{x} \right\rangle \right\| \leq \left\| \boldsymbol{W}_1^\top \boldsymbol{x} - \boldsymbol{W}_2^\top \boldsymbol{x} \right\| \leq \epsilon \|\boldsymbol{x}\|.$$

Furthermore, we claim for such $\boldsymbol{x}$'s the gradient of ϕ is bounded away from zero. More precisely, with $\boldsymbol{x} = r\omega$ where $r = \|\boldsymbol{x}\|$, we have

$$\frac{d}{dr}\phi(\boldsymbol{W}; \boldsymbol{x}) = \left[\sum_{j=1}^{k} (v_{i,j} - v_{r,j}) \mathbb{1}_{\Omega_{\boldsymbol{w}_j}}(\boldsymbol{x}) \boldsymbol{w}_j \right] \omega$$
$$\geq \left(\phi(\boldsymbol{W}; \boldsymbol{x}) - \sum_{j=1}^{k} b_j \right) / r \geq \frac{1}{M} \left(\frac{1}{2} - \sum_{j=1}^{k} b_j \right) =: C > 0.$$

Now, we have

$$\textcircled{1} \leq \mathbb{E}(\boldsymbol{x}, y) \sim \mathcal{D}_i \mathbb{1}_{1-\epsilon\|\boldsymbol{x}\| \leq \phi(\boldsymbol{W}_1, x) \leq 1+\epsilon\|\boldsymbol{x}\|}(\boldsymbol{x}) \|\boldsymbol{x}\|$$
$$\leq \int_{1-\epsilon\|\boldsymbol{x}\| \leq \phi(\boldsymbol{W}_1, x) \leq 1+\epsilon\|\boldsymbol{x}\|} \|\boldsymbol{x}\| p_{max} \, d\boldsymbol{x} \leq 2 \left(\left\| \mathcal{S}^{d_i - 1} \right\| \frac{M^{d_i} p_{max}}{C} \right) \epsilon.$$

As for $\textcircled{2}$, w.l.o.g. we can assume $\left\|\boldsymbol{w}_j^1\right\| \geq \left\|\boldsymbol{w}_j^2\right\|$. Note that when $\left\|\boldsymbol{w}_j^1\right\| \leq \frac{b_j}{M}$ then $h_j \leq 0$ so that $\nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}) = \boldsymbol{0}$ and this $\textcircled{2} = 0$. Hence, we only need to take care of the case when $\left\|\boldsymbol{w}_j^1\right\| \geq \frac{b_j}{M}$. Note that $\left\|\boldsymbol{w}_j^1 - \boldsymbol{w}_j^2\right\| \leq \epsilon$ we know

$$\sin\theta \leq \frac{\epsilon M}{b_j},$$

where θ denotes the acute angle between $\boldsymbol{w}_j^1$ and $\boldsymbol{w}_j^2$. We have the following estimate

$$\textcircled{2} \leq p_{max} \frac{\epsilon M}{b_j} \left\| \mathcal{S}^{d_i - 1} \right\| .$$

Combine with $\textcircled{1}$, we get our desired result. $\square$

Note that Lipschitz differentiability in Lemma 2.2 does not hold for the case $b_j = 0$, as the gradient might be volatile near the origin.

2.3 Convergence Analysis for Non-Bias Case

With the Lipschitz differentiability shown in Lemma 2.2 in the case $b_j > 0$, it is not hard to prove the convergence result in Theorem 2.1. In this section, we focus on the non-bias case ($b_j = 0$) where the Lipschitz differentiability fails and sketch the convergence analysis.

Lemma 2.3. *Consider the network (2.5), $|\boldsymbol{w}_j^t|$ is non-decreasing in t if $v_{i,j} > 0$. For any $r > 0$, choosing learning rate $\eta < \min\left\{\frac{r}{C_p M_i^2}, \frac{r}{2vnM_i}\right\}$, then $|\boldsymbol{w}_j^t|$ is non-increasing if $v_{i,j} < 0$ and $\left|\boldsymbol{w}_j^t\right| > r$, where C_p is a constant satisfying*

$$C_p = \max_{\boldsymbol{v} \in V_i, a \in \mathbb{R}} \int_{\{\langle \boldsymbol{v}, \boldsymbol{x} \rangle = a\}} p_i(\boldsymbol{x}) \, d\,\boldsymbol{x} = O\left(p_{max} M^{d_i}\right).$$

Proof of Lemma 2.3. We first define

$$C_p = \max_{\boldsymbol{v} \in V_1, a \in \mathbb{R}} \int_{\langle \boldsymbol{v}, \boldsymbol{x} \rangle = a} p_1(\boldsymbol{x}) \, d\,S \leq M^{d-1} p_{\max}.$$

Recall the definition of $\Omega_{\boldsymbol{w}_j}$, for all $\boldsymbol{x} \in \Omega_{\boldsymbol{w}_j}$, we have $\langle \tilde{\boldsymbol{w}}_j, \boldsymbol{x} \rangle > 0$. For $v_{i,j} > 0$,

$$\left\langle \tilde{\boldsymbol{w}}_j, -\nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}) \right\rangle = 2v \sum_{r \neq i} \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{D}_i} \left[\mathbb{1}_{\Omega_{i,r}}(\boldsymbol{x}) \mathbb{1}_{\Omega_{\boldsymbol{w}_j}}(\boldsymbol{x}) \langle \tilde{\boldsymbol{w}}_j, \boldsymbol{x} \rangle \right] \geq 0.$$

Note we have from (2.3) that

$$\boldsymbol{w}_j^{t+1} = \boldsymbol{w}_j^t - \eta \nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t).$$

So,

$$\left|\boldsymbol{w}_j^{t+1}\right| = \left\langle \boldsymbol{w}_j^{t+1}, \tilde{\boldsymbol{w}}_j^{t+1} \right\rangle \geq \left\langle \boldsymbol{w}_j^{t+1}, \tilde{\boldsymbol{w}}_j^{t} \right\rangle \geq \left\langle \boldsymbol{w}_j^{t}, \tilde{\boldsymbol{w}}_j^{t} \right\rangle = \left|\boldsymbol{w}_j^t\right|.$$

Let $\Omega_{i,r}^j = \Omega_{i,r} \cap \Omega_{\boldsymbol{w}_j}$. For $v_{i,j} < 0$, we know

$$\left|\nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W})\right| = 2v \left| \sum_{r \neq i} \mathbb{E}\left[\mathbb{1}_{\Omega_{i,r}^j}(\boldsymbol{x})\boldsymbol{x}\right] \right| \leq 2vM \sum_{r \neq i} \mathbb{P}\left[\Omega_{i,r}^j\right]^2,$$

where we omit the distribution $\boldsymbol{x} \sim \mathcal{D}_i$.

On the other hand, by definition of C_p, we know

$$\left|\left\langle \nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}), \tilde{\boldsymbol{w}}_j \right\rangle\right| = \left| 2v \sum_{r \neq i} \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{D}_i}\left[\mathbb{1}_{\Omega_{i,r}^j}(\boldsymbol{x})\langle \tilde{\boldsymbol{w}}_j, \boldsymbol{x}\rangle\right] \right| \geq \frac{v}{C_p} \sum_{r \neq i} \mathbb{P}\left[\Omega_{i,r}^j\right]^2.$$

When $0 < \eta < \frac{r}{2vnM}$, we have

$$\left\langle \boldsymbol{w}_j - \eta \nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}), \tilde{\boldsymbol{w}}_j \right\rangle = \langle \boldsymbol{w}_j, \tilde{\boldsymbol{w}}_j \rangle - \eta \left\langle \nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}), \tilde{\boldsymbol{w}}_j \right\rangle > r - 2v\eta M \sum_{r \neq i} \mathbb{P}\left[\Omega_{i,r}^j\right] > 0.$$

Now, we decompose $\nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t)$ into two parts,

$$\nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t) = \underbrace{\left\langle \tilde{\boldsymbol{w}}_j^t, \nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t) \right\rangle \tilde{\boldsymbol{w}}_j^t}_{\boldsymbol{n}} + \underbrace{\left(\nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t) - \left\langle \tilde{\boldsymbol{w}}_j^t, \nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t) \right\rangle \tilde{\boldsymbol{w}}_j^t \right)}_{\boldsymbol{\nu}}.$$

So that when $\eta < \frac{r}{M^2 C_p}$, we have

$$
\begin{aligned}
&\left|\boldsymbol{w}_j^{t+1}\right|^2 = \left|\boldsymbol{w}_j^t - \eta \nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t)\right|^2 = \left|\boldsymbol{w}_j^t - \eta \boldsymbol{n}\right|^2 + |\eta \boldsymbol{\nu}|^2 \\
=& \left|\boldsymbol{w}_j^t\right|^2 - \eta \left(2\left|\boldsymbol{w}_j^t\right| |\boldsymbol{n}| - \eta \left(|\boldsymbol{n}|^2 + |\boldsymbol{\nu}|^2\right)\right) \\
\leq& \left|\boldsymbol{w}_j^t\right|^2 - 2v\eta \left(\frac{\left|\boldsymbol{w}_j^t\right|}{C_p} - \eta\, M^2\right) \sum_{r \neq i} \mathrm{P}\Omega_{i,r}^{j}{}^2 \leq \left|\boldsymbol{w}_j^t\right|^2,
\end{aligned}
$$

as long as $|\boldsymbol{w}_j| \geq r$. Now, we get the desired result. □

The above theorem provides the dynamic information of the weights. When we input data with distribution $\mathcal{D}_i$, the weights $\{\boldsymbol{w}_j : v_{i,j} > 0\}$ become more useful for classification as the norm of those $\boldsymbol{w}_j$'s grows larger on every iteration. On the other hand, $\{\boldsymbol{w}_j : v_{i,j} < 0\}$ serve only as noise when $v_{i,j} < 0$. On one hand, when $\boldsymbol{w} \in \{\boldsymbol{w}_j : v_{i,j} < 0\}$ has large magnitude, the learning process guarantees the decreasing of $\|\boldsymbol{w}\|$. On the other hand, when $\boldsymbol{w} \in \{\boldsymbol{w}_j : v_{i,j} < 0\}$ has a small magnitude, it shall be trapped into a small region near the origin and contribute little to classification.

The learning process consists of two phases. In the beginning, since the weights are randomly distributed, there may well be some data that does not activate any neurons. The weights then automatically spread out. We call this process the first (slow learning) phase, during which the learning process is rather slow. The next Lemma lists four equivalent statements of a geometric condition. When the geometrical condition holds, we say that the learning process enters the second (fast learning) phase.

Lemma 2.4. *Let $\tilde{\boldsymbol{w}}_j = \frac{\boldsymbol{w}_j}{|\boldsymbol{w}_j|} \in \mathcal{S}^{d-1}$ be k points on the unit sphere, where $1 \leq j \leq k$. Let Λ_W be the convex hull of $\{\tilde{\boldsymbol{w}}_j\}$ and $\{H_i\}_{i=1}^l$ be the facets of convex hull. The following statements are equivalent:*

1. *For any unit vector $\boldsymbol{n} \in \mathcal{S}^{d-1}$, there exist some j_1 and j_2 such that $\langle \boldsymbol{n}, \tilde{\boldsymbol{w}}_{j_1} \rangle > 0$ and $\langle \boldsymbol{n}, \tilde{\boldsymbol{w}}_{j_2} \rangle < 0$.*

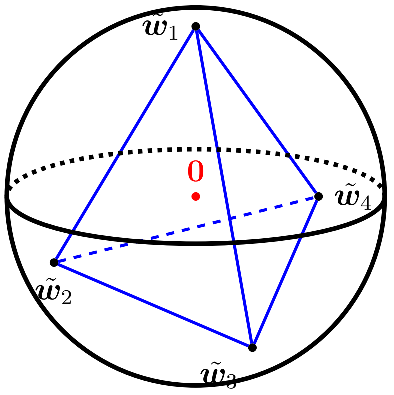

$\tilde{\boldsymbol{w}}_1$
$\mathbf{0}$
$\tilde{\boldsymbol{w}}_4$
$\tilde{\boldsymbol{w}}_2$
$\tilde{\boldsymbol{w}}_3$

Figure 2.1: Geometric Condition in Lemma 2.4 ($d = 3$)

2. *There exists no closed hemisphere that contains all* $\tilde{\boldsymbol{w}}_j$*.*

3. $\mathbf{0}$ *lies in the interior of* Λ_W*.*

4. *For each* $0 \le i \le l$*, we have* $\mathbf{0}$ *and all* $\tilde{\boldsymbol{w}}_j$ *lie on the same side of* H_i*.*

Proof of Lemma 2.4. ($1 \Leftrightarrow 2$) is trivial.

($1 \Rightarrow 3$) Proof by contradiction. Assume $\mathbf{0} \notin \Lambda_W^\circ$. Since Λ_W° is a open convex set, and $\{\mathbf{0}\}$ is a convex set, we know from geometric form of Hahn-Banach Theorem, that there exist a closed hyper-plane that separates $\{\mathbf{0}\}$ and Λ_W°. Hence, there exist a unit vector $\boldsymbol{n} \in \mathcal{S}^{d-1}$, such that $\langle \boldsymbol{n}, \tilde{\boldsymbol{w}}_j \rangle \ge \langle \boldsymbol{n}, \mathbf{0} \rangle = 0$ for all j, but this contradict with our assumptions in 1.

($3 \Rightarrow 4$) Assume that $H_i = [\langle \boldsymbol{v}_i, \boldsymbol{w} \rangle = \alpha_i]$, where $\alpha > 0$. Note that the convex hull Λ_W is a polytope with faces H_i. We know, Λ_W° all lies on one side of H_i. Since $\mathbf{0} \in \Lambda_W^\circ$, we know for all $\boldsymbol{w} \in \Lambda_W^\circ$ we have $\langle \boldsymbol{v}_i, \boldsymbol{w} \rangle < \alpha_i$, and hence $\langle \boldsymbol{v}_i, \tilde{\boldsymbol{w}}_j \rangle \le \alpha_i$ for all j.

($4 \Rightarrow 3$) is trivial.

($3 \Rightarrow 1$) $\mathbf{0} \in \Lambda_W^\circ$ implies there exist positive numbers λ_j, such that

$$\sum_{j=1}^{k} \lambda_j \tilde{\boldsymbol{w}}_j = \mathbf{0}.$$

Hence for any unit vector $\boldsymbol{n}$, we have

$$\sum_{j=1}^{k} \lambda_j \langle \boldsymbol{n}, \tilde{\boldsymbol{w}}_j \rangle = 0.$$

Since $\tilde{\boldsymbol{w}}_j$ are in general position, so $\langle \boldsymbol{n}, \tilde{\boldsymbol{w}}_j \rangle$ cannot all be 0. Hence, there must be both positive and negative terms. Now, we get the desired result. $\square$

Remark 2.1. *If $\boldsymbol{w}_j$ is initialized such that $\tilde{\boldsymbol{w}}_j$ is uniformly distributed on $\mathcal{S}^{d-1}$, then the probability that the geometric condition (GC) in Lemma 2.4 holds is*

$$\mathrm{P}_{\mathrm{gc}} := \mathrm{Prob}(\textit{GC holds}) = 2^{1-k} \sum_{j=d}^{k-1} \binom{k-1}{j}.$$

In particular, at any fixed feature dimension d,

$$\lim_{k \to \infty} \mathrm{P}_{\mathrm{gc}} = 1.$$

Proof of Remark 2.1. For any $J \in \{\pm 1\}^k$, we let $S_J(\{\tilde{\boldsymbol{w}}_j\}) = \{J_j \tilde{\boldsymbol{w}}_j : 1 \le j \le k\}$. From the choice of $\tilde{\boldsymbol{w}}_j$, we know all S_J have the same distribution, so that

$$\mathrm{Prob}_{\{\tilde{\boldsymbol{w}}_j\}}(\mathrm{GC\,holds}) = \mathrm{Prob}_{S_J}(\mathrm{GC\,holds}).$$

Hence, we can simplify the probability as follows

$$\mathrm{Prob}(\mathrm{GC\,holds}) = \frac{1}{2^n} \mathbb{E}\left[\sum_J \mathbb{1}_{gc}\left(S_J\left(\{\tilde{\boldsymbol{w}}_j\}\right)\right)\right].$$

We claim that $\sum_J \mathbb{1}_{gc}\left(S_J\left(\{\tilde{\boldsymbol{w}}_j\}\right)\right)$ is a constant independent of choice of $\tilde{\boldsymbol{w}}_j$ as long as they are in general position.

Let $\tilde{\boldsymbol{w}}_j \in \mathcal{S}^{d-1}$ where $1 \le j \le k$. Each $\tilde{\boldsymbol{w}}_j$ corresponds to a subspace H_j with codimension

1 in $\mathbb{R}^d$, that is

$$H_j = \left\{ \boldsymbol{x} \in \mathbb{R}^d : \langle \tilde{\boldsymbol{w}}_j, \boldsymbol{x} \rangle = 0 \right\}.$$

Note that any connected region of $\left(\cup_J H_J\right)^c$ corresponds to a choice of J such that the geometric condition fails. More precisely, for any given connected region D of $\left(\cup_J H_J\right)^c$, we know $\langle \tilde{\boldsymbol{w}}_j, \boldsymbol{x} \rangle$ is one sign for all $\boldsymbol{x} \in D$, so with

$$J_j = \operatorname{sign}\left(\langle \tilde{\boldsymbol{w}}_j, \boldsymbol{x} \rangle\right),$$

we know $\boldsymbol{x}$ correspond to $S_J\left(\{\tilde{\boldsymbol{w}}_j\}\right)$ where the geometric condition fails. Hence, the number of connected regions of $\left(\cup_J H_J\right)^c$ is same as $2^n - \sum_J \mathbb{1}_{gc}\left(S_J\left(\{\tilde{\boldsymbol{w}}_j\}\right)\right)$. By [18], we have

$$\sum_J \mathbb{1}_{gc}\left(S_J\left(\{\tilde{\boldsymbol{w}}_j\}\right)\right) = 2^n - 2\sum_{j=0}^{d-1} \binom{k-1}{j} = 2\sum_{j=d}^{k} \binom{k-1}{j}.$$

The desired result follows. $\square$

Per Remark 3, the more neurons the network has, higher possibility the geometric condition in Lemma 2.4 holds upon initialization. As a consequence, the learning process skips the first phase and goes straight to the fast learning phase. This explains why gradient descent for learning a over-parameterized network converges rapidly to a nearby critical point from random initialization.

The following proposition gives an upper bound on the maximum number of iterations for the learning process to enter the second phase.

Proposition 2.3. *Let $b_j = 0$ in (2.5), and assume that $|\boldsymbol{W}^t| \leq R$ for all t. Let T_1 be the set of t such that $\left\{\boldsymbol{w}_j^t : v_{i,j} > 0\right\}$ does not satisfy the geometric condition in Lemma 2.4, then*

$|T_1| \le \frac{C_p R}{v\eta p_R^2}$, *where* p_R *is a positive constant. Also, the following estimate holds for* p_R*:*

$$p_R = \Omega\left(\frac{p_{min}}{\sqrt{d_i}\,(M_i R)^{d_i}}\right)$$

where C_p *is the constant in Lemma 2.3. More precisely,*

$$|T_1| = O\left(\frac{C_p d_i R^{2d_i+1} M_i^{2d_i}}{v\eta p_{min}^2}\right) = O\left(\frac{C_p d R^{2d+1} M^{2d}}{v\eta p_{min}^2}\right).$$

Proof of Proposition 2.3. W.l.o.g, assume $i = 1$ and $v_{1,j} > 0$ if and only if $j \in [k_1]$. Let $t \in T_1$, by Lemma 2.4, we know, for any fixed $t \in T_1$, there exists a $\alpha \in [0, \frac{\pi}{2}]$ and a unit vector $\boldsymbol{v} \in \mathbb{R}^{d_1}$, such that $\langle \boldsymbol{v}, \tilde{\boldsymbol{w}}_j^t \rangle \ge \sin\alpha$ for all $j \in [k_1]$ and $\langle \boldsymbol{v}, \tilde{\boldsymbol{w}}_1^t \rangle = \sin\alpha$. W.l.o.g, we assume $\boldsymbol{v} = (1, 0, \cdots, 0)$ and $\langle \boldsymbol{v}, \tilde{\boldsymbol{w}}_1^t \rangle = \sin\alpha$. For any non-zero $\boldsymbol{x} \in V_1$, we write $\tilde{\boldsymbol{x}} = \frac{\boldsymbol{x}}{|\boldsymbol{x}|} \in \mathcal{S}^{d_1-1}$.

Note that $|\boldsymbol{W}|$ is bounded by R, we know there exist $\beta \in (0, \frac{\pi}{2} - \alpha)$ such that $\sum_{j=1}^k \left|\boldsymbol{w}_j^t\right| \le R = \frac{1}{2vM_1 \sin\beta}$. Now, w.l.o.g, we can assume $\tilde{\boldsymbol{w}}_1^t = (\cos\alpha, \sin\alpha, 0, \cdots, 0)$. Take $\boldsymbol{n} = (-\cos\alpha, \sin\alpha, 0, \cdots, 0)$, we know $\langle \boldsymbol{n}, \tilde{\boldsymbol{w}}_1^t \rangle = 0$. For all $\boldsymbol{x} \in V_1$ such that $\langle \boldsymbol{n}, \tilde{\boldsymbol{x}} \rangle > \cos\beta$, we have

$$\cos\beta < \langle \boldsymbol{n}, \tilde{\boldsymbol{x}} \rangle = -\tilde{x}_1 \cos\alpha + \tilde{x}_2 \sin\alpha \le -\tilde{x}_1 \cos\alpha + \sqrt{1 - \tilde{x}_1^2}\sin\alpha.$$

So, we have

$$\tilde{x}_1 < -\cos(\alpha + \beta).$$

Now, for all $\boldsymbol{x} \in V_1$ such that $\tilde{x}_1 < -\cos(\alpha + \beta)$, we have

$$f_1(\boldsymbol{W}^t; \boldsymbol{x}) - f_r(\boldsymbol{W}^t; \boldsymbol{x}) \le 2vM_1 \sum_{j=1}^k \sigma\left(\langle \boldsymbol{w}_j^t, \tilde{\boldsymbol{x}} \rangle\right) \le 2vM_1 \sum_{j=1}^k |\boldsymbol{w}_j^t| \sin\beta < 1.$$

So, with

$$D^1 := \{\boldsymbol{x} \in V_1 : \langle \boldsymbol{n}, \tilde{\boldsymbol{x}} \rangle > \cos\beta \text{ and } \langle \tilde{\boldsymbol{w}}_1, \tilde{\boldsymbol{x}} \rangle > 0\},$$

we have for any $r > 1$

$$D^1 \subset \Omega_{1,r}.$$

See Fig 2.2 for a intuition of D^1. Note that $\boldsymbol{n}$ and $\tilde{\boldsymbol{w}}_1$ perpendicular to each other, and the

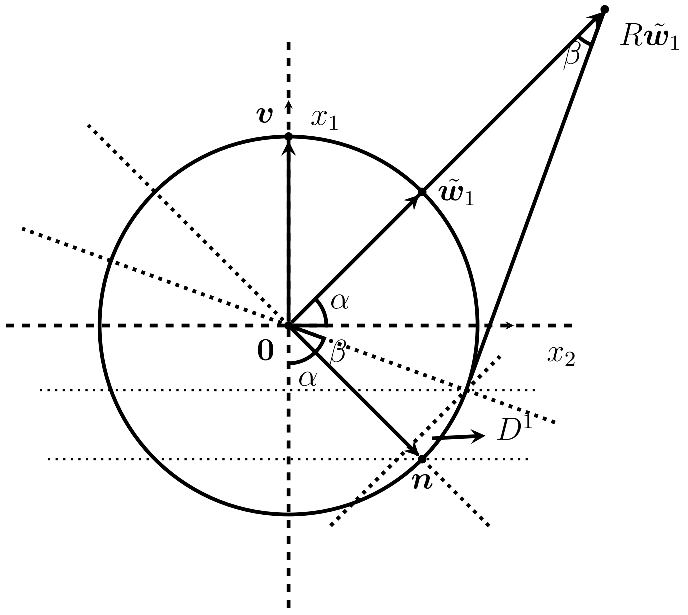


Figure 2.2: 2-dim section of $\mathbb{R}^d$ spanned by $\tilde{\boldsymbol{w}}_1$ and $\boldsymbol{n}$

probability distribution p_1 is bounded from below, so there exists a constant p_R, such that

$$\mathbb{P}\left[\Omega_{1,r} \cap \Omega_{\boldsymbol{w}_j^t}\right] \geq \mathbb{P}\left[D^1\right] = p_R > 0,$$

and we have the following estimate for p_R

$$\begin{aligned} p_R &= \int_{D^1} \left\langle \tilde{\boldsymbol{w}}_1^t, \nabla_{\boldsymbol{w}_1} l(\boldsymbol{W}) \right\rangle p_1(\boldsymbol{x}) \ d\boldsymbol{x} \\ &\geq \frac{p_{\min} \left|\mathcal{S}^{d_1-2}\right|}{\left|\mathcal{S}^{d_1-1}\right|} \int_{\cos\beta}^{1} \left(1-y^2\right)^{\frac{d_1-3}{2}} dy \\ &= \frac{p_{\min} \left|\mathcal{S}^{d_1-2}\right|}{\left|\mathcal{S}^{d_1-1}\right|} \int_0^{\beta} (\sin\theta)^{d_1-2} d\theta \qquad . \\ &= \Omega\left(\frac{p_{\min} (\sin\beta)^{d_1-1}}{(d_1-1)} \frac{\left|\mathcal{S}^{d_1-2}\right|}{\left|\mathcal{S}^{d_1-1}\right|} \right) \\ &= \Omega\left(\frac{p_{\min}}{\sqrt{d_1}\,(M_1 R)^{d_1}} \right). \end{aligned}$$

Now, we know the gradient of $\boldsymbol{w}_1$ on $\tilde{\boldsymbol{w}}_1$ direction is bounded below, with same arguments in proof of Lemma 2.3, we have

$$\left|\left\langle \tilde{\boldsymbol{w}}_1^t, \nabla_{\boldsymbol{w}_1} l_1\left(\boldsymbol{W}^t\right)\right\rangle\right| \geq \frac{v\, p_R^2}{C_p}.$$

Note that

$$\sum_{j=1}^{k_1} \left|\boldsymbol{w}_j^{t+1}\right| - \left|\boldsymbol{w}_j^t\right| \geq \left|\boldsymbol{w}_1^{t+1}\right| - \left|\boldsymbol{w}_1^t\right| \geq \eta \left|\left\langle \tilde{\boldsymbol{w}}_1^t, \nabla_{\boldsymbol{w}_1} l\left(\boldsymbol{W}^t\right)\right\rangle\right| \geq \frac{v\, \eta\, p_R^2}{C_p}$$

and since $\left|\boldsymbol{w}_j^t\right|$ is non-decreasing for $j \in [k_1]$, we have

$$\sum_{t\in T_1} \sum_{j=1}^{k_1} \left|\boldsymbol{w}_j^{t+1}\right| - \left|\boldsymbol{w}_j^t\right| \leq R.$$

Combining the above two equations, it follows $|T_1| \leq \dfrac{C_p R}{v\, \eta\, p_R^2}$. □

At the beginning of the second phase, $\tilde{w}_j$ are already evenly distributed. That is, the convex hull of $\tilde{\boldsymbol{w}}_j$ contains the origin, and any input data must at least activate some of the neurons. As long as an input data contributes to the loss, it also contributes to the gradient. So learning process becomes faster during the second phase. The following proposition shows the loss decays faster than before.

Proposition 2.4. *Let $b_j = 0$ in (2.5). Let T_2 be the set of t such that $\{\boldsymbol{w}_j^t : v_{i,j} > 0\}$ satisfies the geometric condition in Lemma 2.4, and that $|\boldsymbol{W}^t|$ is upper-bounded by R at all t. Then:*

$$\sum_{t\in T_2} l_i\left(\boldsymbol{W}^t\right)^2 \leq 4\eta^{-1} v n^2 C_p R^2 M_i^2 R.$$

Proof of Proposition 2.4. First, we estimate $l_i(\boldsymbol{W}^t)$ as follows

$$
\begin{aligned}
l_i(\boldsymbol{W}^t) &= \sum_{r \neq i} \mathbb{E}(\boldsymbol{x}, y) \sim \mathcal{D}_i \sigma\left(1 - f_i + f_r\right) \\
&\leq \sum_{r \neq i} \left(2vRM_i\right) \mathrm{P}(\boldsymbol{x}, y) \sim \mathcal{D}_i f_i < 1 + f_r \\
&= \left(2vRM_i\right) \sum_{r \neq i} \mathrm{P}(\boldsymbol{x}, y) \sim \mathcal{D}_i \Omega_{i,r}.
\end{aligned}
$$

Now, we have

$$
\sum_{r \neq i} \mathrm{P}(\boldsymbol{x}, y) \sim \mathcal{D}_i \Omega_{i,r} \geq \frac{l_i(\boldsymbol{W}^t)}{2vRM_i}.
$$

Second, we estimate the gradient. Let $\Omega_{i,r}^j = \Omega_{i,r} \cap \Omega_{\boldsymbol{w}_j}$ and assume $v_{i,j} > 0$, we have

$$
\nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t) = \sum_{r \neq i} 2v \mathbb{E}(\boldsymbol{x}, y) \sim \mathcal{D}_i \mathbb{1}_{\Omega_{i,r}^j}(\boldsymbol{x}) \boldsymbol{x}.
$$

By the same arguments in proof of Lemma 2.3, we know

$$
\left\langle \nabla_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t), \tilde{\boldsymbol{w}}_j^t \right\rangle \geq 2v \frac{\mathrm{P}\Omega_{i,r}^{j\ 2}}{2C_p} = \frac{v\mathrm{P}\Omega_{i,r}^{j\ 2}}{C_p}.
$$

Next, we utilize the geometric condition which implies

$$
\sum_{v_{i,j} > 0} \mathrm{P}\Omega_{i,r}^j \geq \mathrm{P}\Omega_{i,r}
$$

and get

$$\sum_{v_{i,j}>0} \|\boldsymbol{w}_j^{t+1}\| - \|\boldsymbol{w}_j^t\| \geq \sum_{\substack{v_{i,j}>0 \\ r\neq i}} \frac{v\eta \mathrm{P}\Omega_{i,r}^{j\ 2}}{C_p} = \frac{v\eta}{C_p} \sum_{\substack{v_{i,j}>0 \\ r\neq i}} \mathrm{P}\Omega_{i,r}^{j\ 2}$$

$$\geq \frac{v\eta}{C_p}\frac{1}{k_i}\sum_{r\neq i} \mathrm{P}\Omega_{i,r}{}^2 \geq \frac{v\eta}{C_p}\frac{1}{k_i(n-1)}\left(\sum_{r\neq i}\mathrm{P}\Omega_{i,r}\right)^2$$

$$\geq \frac{v\eta}{n^2C_p}\frac{l_i(\boldsymbol{W}^t)^2}{4v^2R^2M_i^2} = \frac{\eta}{4vn^2C_pR^2M_i^2} l_i(\boldsymbol{W}^t)^2.$$

where k_i is the number of j's such that $V_{i,j} > 0$.

Finaly, we combine the inequalities above and the assumption $\|\boldsymbol{W}^t\| < R$ and get

$$\sum_{t\in T_2} l_i(\boldsymbol{W}^t)^2 \leq \frac{4vn^2C_pR^2M_i^2}{\eta} \sum_{\substack{v_{i,j}>0 \\ t\in T_2}} \|\boldsymbol{w}_j^{t+1}\| - \|\boldsymbol{w}_j^t\| \leq 4\eta^{-1}vn^2C_pR^2M_i^2R.$$

$\square$

From the above proposition, we see that in order to get $l_i(\boldsymbol{W}^t) < \epsilon$ for some $t \in T_2$, we only need

$$|T_2| \geq \frac{4vn^2C_pR^2M_i^2R}{\eta\epsilon}.$$

Compared with Proposition 2.3, we see that $|T_2|$ does not rely on the dimension d_i, whereas the upper bound of $|T_1|$ is exponential in d_i.

2.4 Main Results

Although (2.4) is a non-convex optimization problem, we show that under mild conditions, the gradient descent algorithm (2.3) converges to a global minimum with zero classification error. Specifically, we consider two different networks with a positive bias $b_j > 0$ (Theorem 2.1) and without a bias (Theorem 2.2), respectively. For both cases, we have the fact that any critical point of problem (2.4) is a global minimum (Proposition 2.1). The key difference between these two cases is that the population loss function has Lipschitz continuous gradient (Lemma 2.2) when $b_j > 0$, whereas this desirable property does not hold otherwise. For the latter case $b_j = 0$, we present a totally different analysis based on a geometric condition proved to emerge during the training process (Proposition 2.3). Under this geometric condition, the objective value converges zero (Proposition 2.4).

Theorem 2.1. *Assume* $0 < \sum_{j=1}^{k} b_j < 1$ *and assumption 2.1 holds in (2.1), and* $\{\boldsymbol{W}^t\}$ *generated by the algorithm (2.3) are bounded uniformly in* t. *If there exists* $(\boldsymbol{x}, i) \sim \mathcal{D}_i$ *and some indices* $j \in [k]$, *such that* $v_{i,j} > 0$ *and* $\left|\left\langle \boldsymbol{w}_j^0, \boldsymbol{x}\right\rangle\right| > b_j$, *then there exists some* $\eta_0\left(v, k, b, p_{max}, n, M\right) > 0$ *such that for the learning rate* $\eta < \eta_0$, $\lim_{t\to\infty} l_i\left(\boldsymbol{W}^t\right) = 0$ *and*

$$\lim_{t\to\infty} \mathop{\mathbb{P}}_{\{\boldsymbol{x},i\}\sim\mathcal{D}_i} \left[\hat{y}\left(\boldsymbol{W}^t; \boldsymbol{x}\right) \neq i\right] = 0, \quad i \in [n], \; \forall\, n \geq 2.$$

Proof of Theorem 2.1. By Lemma 2.1, we only need to prove the convergence of the simplified network (2.5). From Lemma 2.2, we know $l_i(\boldsymbol{W})$ has Lipschitz gradient. We can assume for any $\boldsymbol{W}_1, \boldsymbol{W}_2 \in V_i^k$, we have

$$|\nabla l_i(\boldsymbol{W}_1) - \nabla l_i(\boldsymbol{W}_2)| \leq L\, |\boldsymbol{W}_1 - \boldsymbol{W}_2|\,.$$

As long as we take $\eta < \frac{L}{2}$ in algorithm (2.3), we know

$$l_i(\boldsymbol{W}^{t+1}) \leq l_i(\boldsymbol{W}^t) - \left(\eta - \frac{\eta^2 L}{2}\right) \left|\nabla l_i(\boldsymbol{W}^t)\right|^2 \leq l_i(\boldsymbol{W}^t). \tag{2.6}$$

Hence, $l_i(\boldsymbol{W}^t)$ is monotonically decreasing. Therefore, for any convergent subsequence $\{\boldsymbol{W}^{t_k}\}$ with the limit $\boldsymbol{W}_0$, there exists $l_0 \geq 0$ such that

$$\lim_{t\to\infty} l\left(\boldsymbol{W}^t\right) = \lim_{k\to\infty} l\left(\boldsymbol{W}^{t_k}\right) = l_0.$$

Now, we can take subsequence and limit on both side of equation (2.6), we get

$$l_0 \leq l_0 - \left(\eta - \frac{\eta^2 L}{2}\right) \left|\nabla l_i(\boldsymbol{W}_0)\right|^2.$$

Now, we see that $\nabla l_i(\boldsymbol{W}_0) = \mathbf{0}$. □

Theorem 2.2. *Assume $b_j = 0$ and assumption 2.1 holds in (2.1), and $\{\boldsymbol{W}^t\}$ generated by algorithm (2.3) is bounded by R uniformly in t. If there exist some $(\boldsymbol{x}, i) \sim \mathcal{D}_i$ and some indices $j \in [k]$, such that $v_{i,j} > 0$ and $\langle \boldsymbol{w}_j^0, \boldsymbol{x} \rangle \neq 0$, then $\lim_{t\to\infty} l_i\left(\boldsymbol{W}^t\right) = 0$ and*

$$\lim_{t\to\infty} \mathop{\mathbb{P}}_{\{\boldsymbol{x},i\}\sim\mathcal{D}_i} \left[\hat{y}\left(\boldsymbol{W}^t; \boldsymbol{x}\right) \neq i\right] = 0, \ \ i \in [n], \ \forall\, n \geq 2.$$

Proof of Theorem 2.2. By Proposition 2.3, we know the iterates $\{\boldsymbol{W}^t\}$ stay in the first phase is bounded by $\|T_1\|$. Combining Proposition 2.4 which shows the summation of squared loss values in phase two is bounded, the summation of all loss values in the learning process is bounded

$$\sum_{t\in T_2} l_i\left(\boldsymbol{W}^t\right)^2 < \infty.$$

The desired result follows. □

Remark 2.2. *The assumptions on the initialization $|\langle \boldsymbol{w}_j^0, \boldsymbol{x}\rangle| > b_j$ in both theorems are natural. This assumption guarantees that the neuron $\boldsymbol{w}_j$ is activated by some input data. Without this assumption, the algorithm suffers zero gradient and fails to update.*

Remark 2.3. *Theorem 2.2 does not explicitly require the learning rate η to be small. However, a larger learning rate will implicitly result in a larger bound in Propositions 2.3 and 2.4 which we used to prove Theorem 2.2.*

2.5 Experiments

In this section, we report the results of our experiments on both synthetic and MNIST data. The experiments on synthetic data aim to show that *convergence to global minimum continues to hold if data subspaces form acute angles, going beyond the theoretical orthogonality assumption* under which convergence is observed to be the fastest. Our theoretical results and the geometric conditions are supported by simulations. Experiments on MNIST dataset exhibit subspace structures in data flow and slow-to-fast training dynamics on LeNet-5. These phenomena from our model are worth further study in deep networks.

2.5.1 Synthetic Data

Let $\{\boldsymbol{e}_j\}_{j\in[4]}$ be orthonormal basis of $\mathbb{R}^4$, θ be an acute angle and $\boldsymbol{v}_1 = \boldsymbol{e}_1$, $\boldsymbol{v}_2 = \sin\theta\, \boldsymbol{e}_2 + \cos\theta\, \boldsymbol{e}_3$, $\boldsymbol{v}_3 = \boldsymbol{e}_3$, $\boldsymbol{v}_4 = \boldsymbol{e}_4$. Now, we have two linearly independent subspaces of $\mathbb{R}^4$ namely $V_1 = \operatorname{Span}(\{\boldsymbol{v}_1, \boldsymbol{v}_2\})$ and $V_2 = \operatorname{Span}(\{\boldsymbol{v}_3, \boldsymbol{v}_4\})$. We can easily calculate that the angle between V_1 and V_2 is θ. Next, we define

$$\hat{\mathcal{X}}_1 = \{r\,(\cos\varphi\, \boldsymbol{v}_1 + \sin\varphi\, \boldsymbol{v}_2) : r \in S_r, \varphi \in S_\varphi\},$$

$$\hat{\mathcal{X}}_2 = \{r\,(\cos\varphi\, \boldsymbol{v}_3 + \sin\varphi\, \boldsymbol{v}_4) : r \in S_r, \varphi \in S_\varphi\},$$

where

$$S_r = \left\{\frac{20}{j} : j \in [20] - [9]\right\},\ S_\varphi = \left\{\frac{j\pi}{40} : j \in [80]\right\}.$$

Let $\mathcal{X}_1$ corresponds to label $y = 1$ and $\mathcal{X}_2$ corresponds to label $y = -1$. Since we are considering a binary classification problem, the neural network structure can be simplified as (2.7):

$$\tilde{f}\left(\boldsymbol{W};\boldsymbol{x}\right)=\sum_{j=1}^{k}\sigma\left(h_j\right)-\sum_{j=k+1}^{2k}\sigma\left(h_j\right), \tag{2.7}$$

and the prediction is given by $\hat{y}(\boldsymbol{x})=\operatorname{sign}f\left(\tilde{\boldsymbol{W}};\boldsymbol{x}\right)$. Now, the population loss becomes

$$l_i(\boldsymbol{W}):=\frac{1}{|\hat{\mathcal{X}}_i|}\sum_{\boldsymbol{x}\in\hat{\mathcal{X}}_i}\max\left\{0,1+(-1)^i\tilde{f}\left(\boldsymbol{W};\boldsymbol{x}\right)\right\}.$$

In our first simulation, we set $k=4$ in (2.7) and run gradient descent (2.3) on l_1+l_2 with learning rate $\eta=0.1$. Fig. 2.3 shows the iterations it takes to converge to global minima given θ and a Gaussian noise added to $\mathcal{X}_i$'s. From this simulation, we see that the orthogonal data assumptions are only technically needed and our convergence result holds in more general settings.

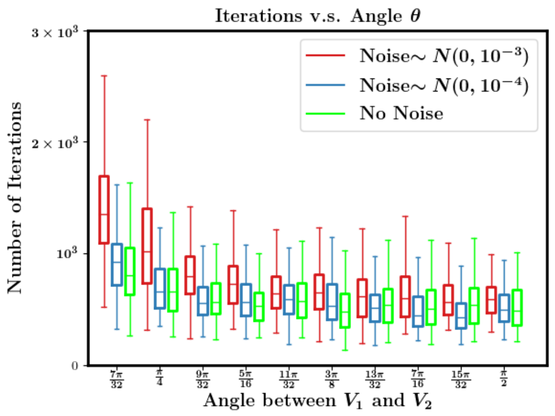


Figure 2.3: Number of iterations to convergence v.s. θ, the anlge between subspaces V_1 and V_2.

In our second and remaining simulations of this subsection, we take $\theta=\frac{\pi}{2}$ so that V_1 and V_2 are orthogonal. Lemma 2.1 suggests the learning process of l_1 and l_2 are independent, so we only simulate the training process of l_1 and assume $\boldsymbol{w}_j\in V_1\cong\mathbb{R}^2$. We take entries of $\boldsymbol{W}^0$ to be i.i.d. standard normal i.e. $\boldsymbol{w}_j^0\sim N(\boldsymbol{0},I_2)$. We train the network (2.7) with gradient descent (2.3) in all our simulations, where learning rate $\eta = 0.1$. The left plot of Fig. 2.4 shows how many iterations algorithm (2.3) takes in searching for a global minima from the random initialization mentioned above. For each box, the red mark indicates the median, and the bottom and top of the box indicate the 25th and 75th percentiles, respectively. As we can see from the graph, as number of hidden neurons ($2k$) becomes larger, the algorithm (2.3) tends to need less iterations in searching for a global minima.

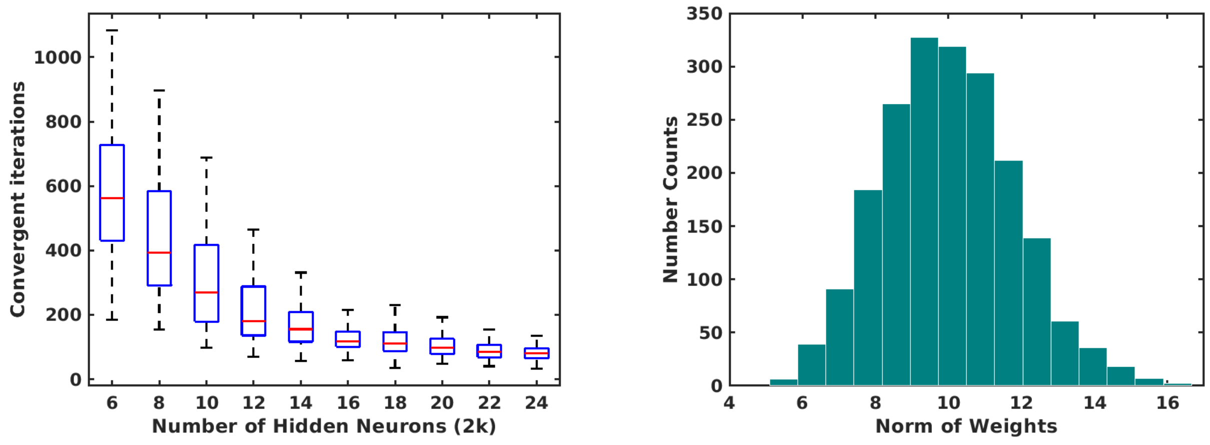


Figure 2.4: Left: convergent iterations vs. number of neurons ($d = 2$). Right: histogram of norm of weights: $\max_t |\boldsymbol{W}^t|$ ($d = 2$ and $k = 4$).

In the third simulation, we compare the convergence speed with and without the geometric condition being satisfied. We introduce two initialization method: random initialization and half space initialization i.e. with $\hat{w}_{j,i} \sim N(0, 1)$, random initialization takes $w^0_{j,i} = \hat{w}_{j,i}$ whereas half space initialization takes $w^0_{j,1} = |\hat{w}_{j,1}|$ and $w^0_{j,2} = \hat{w}_{j,2}$. We run the algorithm for 100 times with different numbers of hidden neurons using initialization methods, and report the means and standard variances of the number of iterations in Table 2.1. We see from Remark 2.1 how the P_{gc} increases when the number of hidden neurons grows. However, the half space initialization never satisfies the geometric condition, as all the weights lie in the same half space. A widely believed explanation on why a neural network can fit all training labels is that the neural network is over-parameterized. Our work explained one of the reasons why over-parameterization helps convergence: it helps the weights to spread more 'evenly' and quickly after initialization. Table 2.1 shows that when we randomly initialize, the iterations for convergence in gradient descent (2.3) come down a lot as the number of hidden neurons increases; much less so in half space initialization.

Table 2.1: Iterations taken (mean $\pm$ std) to convergence with random and half space initializations.

# of Neurons ($2k$)	Random Init.	Half Space Init.
6	578.90±205.43	672.41±226.53
8	423.96±190.91	582.16±200.81
10	313.29±178.67	550.19±180.59
12	242.72±178.94	517.26±172.46
14	183.53±108.60	500.42±215.87
16	141.00±80.66	487.42±220.48
18	126.52±62.07	478.25±202.71
20	102.09±32.32	412.46±195.92
22	90.65±28.01	454.08±203.00
24	82.93±26.76	416.82±216.58

Our fourth simulation take specifically $2k = 8$. With 2000 runs we did a histogram of the maximum norm of $\boldsymbol{W}$ during the training process shown in the right plot of Fig. 2.4. In fact, our third simulation suggests our boundedness assumption on $\boldsymbol{W}$ in Theorem 2.1 and Theorem 2.2 are reasonable.

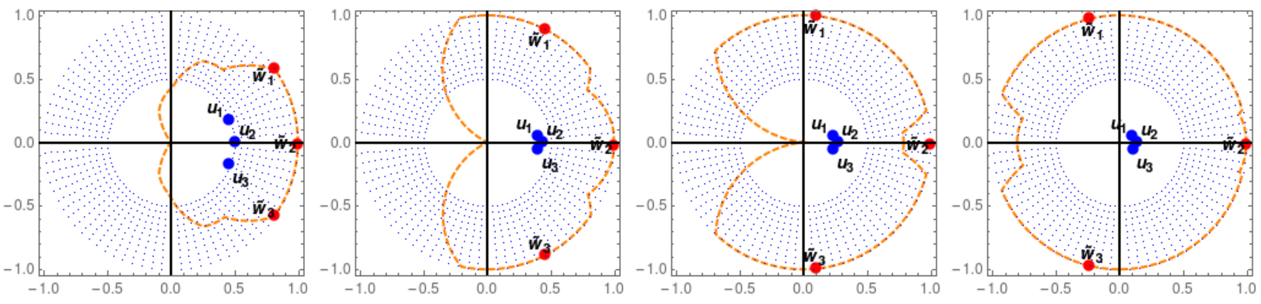


Figure 2.5: Dynamics of weights: $\tilde{\boldsymbol{w}}_j$ and $\boldsymbol{u}_j$

In our last simulation, we take $k = 3$ so that there are in total 6 hidden neurons. For notation simplicity, we denote $\boldsymbol{u}_j = \boldsymbol{w}_{j+3}$. For $j \in [3]$, we plot $\tilde{\boldsymbol{w}}_j$'s and $\boldsymbol{u}_j$'s in Fig. 2.5, where we plot $\tilde{\boldsymbol{w}}_j$'s instead of $\boldsymbol{w}_j$'s since some of $|\boldsymbol{w}_j|$'s are greater than one. Before algorithm (2.3) starts, the parameters in neural network (2.5) are initialized to be

$$\boldsymbol{w}_j^0 = \boldsymbol{u}_j^0 = \frac{3}{4}\left(\cos\frac{(2-j)\,\pi}{6}, \sin\frac{(2-j)\,\pi}{6}\right)$$

for $j \in \{1,2,3\}$. In Fig. 2.5, the tiny blue points are input data under Kelvin transformation: $\boldsymbol{x} \to \boldsymbol{x}^* = \frac{\boldsymbol{x}}{|\boldsymbol{x}|^2}$. Take $\boldsymbol{x} = \frac{1}{r}(\cos\theta, \sin\theta)$ so that under Kelvin transformation $\boldsymbol{x}^* = r(\cos\theta, \sin\theta)$. For convenience, we let $\tilde{\boldsymbol{x}} = r\boldsymbol{x} = (\cos\theta, \sin\theta)$. The orange dashed curve has expression in polar coordinates:

$$\rho(\theta) = \min\left\{1, \sigma\left(\tilde{f}\,(\boldsymbol{W}; \tilde{\boldsymbol{x}})\right)\right\}.$$

Note that we are taking Hinge loss $l(\boldsymbol{W}; \{\boldsymbol{x}, 1\}) = 0$ if and only if $\tilde{f}(\boldsymbol{W}; \boldsymbol{x}) \geq 1$, i.e.

$$\tilde{f}\,(\boldsymbol{W}; \tilde{\boldsymbol{x}}) = r\tilde{f}\,(\boldsymbol{W}; \boldsymbol{x}) \geq r = |\boldsymbol{x}^*|.$$

Here, in our data set $\hat{\mathcal{X}}$, all data point have norm less than one under Kelvin transformation, so $l(\boldsymbol{W}; \{\boldsymbol{x}, 1\}) = 0$ if and only if $\rho(\theta) \geq |\boldsymbol{x}^*|$. This means, the blue points when surrounded by the orange dashed curve provide zero loss. In particular, when $\rho(\theta) = 1$, the population loss is 0.

2.5.2 MNIST Experiments

The two-phase dynamics we proved in our model does appear in deep network training on real (non-synthetic) data sets. In experiments on MNIST, we used a simplified version of LeNet-5 [36] with 2 convolutional layers and two fully-connected (fc) layers; see [1] for the full version where F6 and output layers correspond to our fc layers. The simplified Lenet-5 is trained via stochastic gradient descent (SGD) at constant learning rate 0.01, batch size 1000 and without momentum or regularization. We show in Fig. 2.6 (left) the loss value

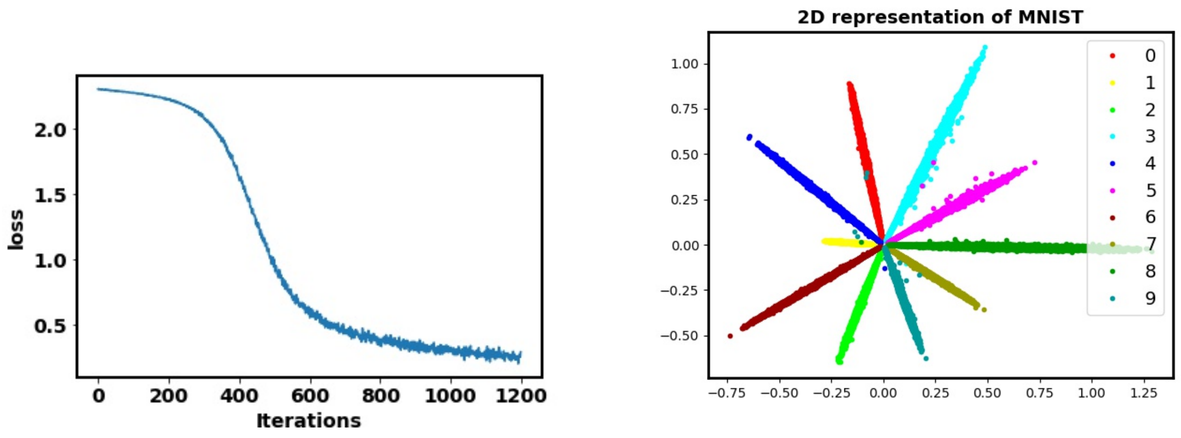


Figure 2.6: **Left**: Slow-to-Fast transition during LeNet [36] training on MNIST dataset. **Right**: 2D projections of MNIST features from a trained convolutional neural network [42]. The 10 classes are color coded, the feature points cluster near linearly independent subspaces.

vs. iterations during training. At the early stage, the loss decays slowly, then the fast phase sets in after 400 iterations. Fig. 2.6 (left) clearly supports our theory on the slow and fast dynamics of gradient descent.

Network visualization helps understand its geometric properties. In Fig. 2.6 (right), we plot 2D projections of feature vectors at input to fc layer extracted by a neural network [42] consisting of 6 convolutional layers followed by an fc layer. Projected features from different classes cluster around linearly independent subspaces. The plot suggests that in trained deep networks, the linearly independent subspace assumption approximately holds for the input to the fully connected layer before classification output. Similar subspace structure of high level feature vectors on CIFAR-10 and enlargement of subspace angles to improve classification accuracy have been studied in [45].

In Fig. 2.7, we show four projections onto unit sphere $\mathcal{S}^2$ (inside randomly selected 3D subspaces) of the weight vectors of the first and second fc layers of LeNet [36]. Visual inspection on these and others (not shown) suggests that our geometric condition holds with high probability.

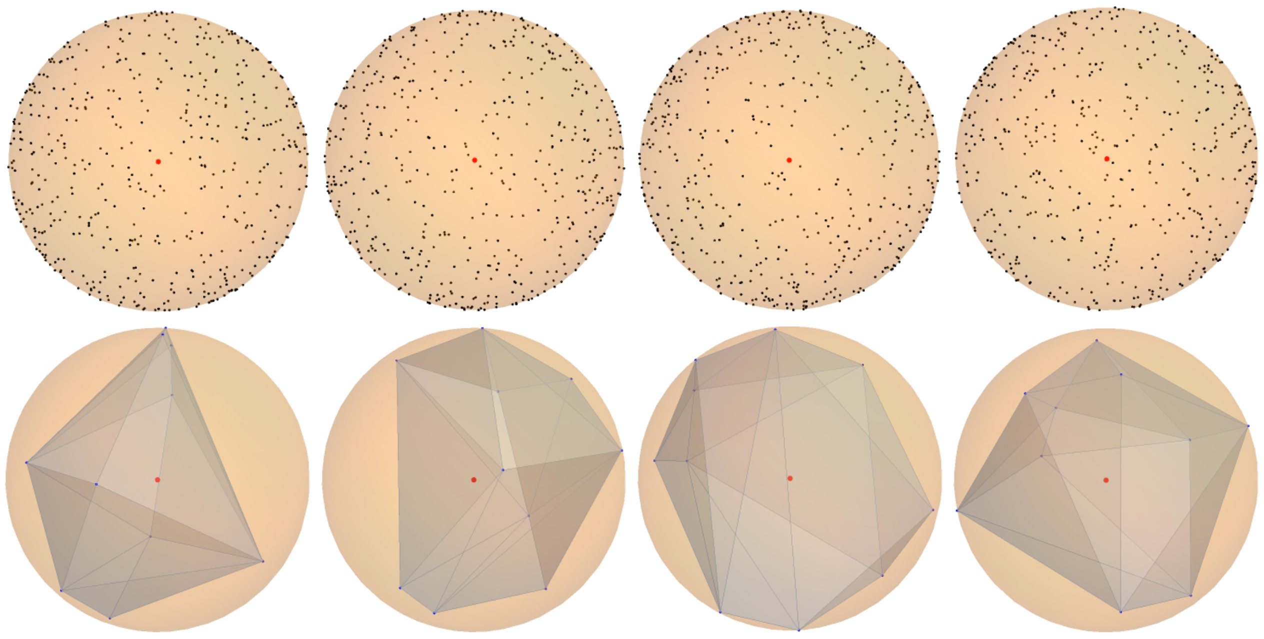


Figure 2.7: **Top row**: Projections onto $\mathcal{S}^2$ (inside randomly selected 3D subspaces) of weight vectors in the first fully connected layer of a trained LeNet. **Bottom row**: Projections onto $\mathcal{S}^2$ (inside randomly selected 3D subspaces) of weight vectors and their convex hull in the second fully connected layer of a trained LeNet.

Chapter 3

Learning Quantized Neural Nets by Coarse Gradient Method for Non-Linear Classification

3.1 Problem Setup

3.1.1 Data Assumptions

In this section, we consider the n-ary classification problem in the d-dimensional space $\mathcal{X} = \mathbb{R}^d$. Let $\mathcal{Y} = [n]$ be the set of labels, and for $i \in [n]$ let $\mathcal{D}_i$ be probabilistic distributions over $\mathcal{X} \times \mathcal{Y}$. Throughout this chapter, we make the following assumptions on the data:

1. **(Separability)** There are n orthogonal sub-spaces $V_i \subseteq \mathcal{X}$, $i \in [n]$ where $\dim V_i = d_i$, such that

$$\mathop{\mathbb{P}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i} [\boldsymbol{x} \in \mathcal{V}_i \text{ and } y = i] = 1, \text{ for all } i \in [n].$$

2. **(Boundedness of data)** There exist positive constants m and M, such that

$$\mathbb{P}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}[m < |\boldsymbol{x}| < M] = 1, \text{ for all } i \in [n].$$

3. **(Boundedness of p.d.f.)** For $i \in [n]$, let p_i be the marginal probability distribution function of $\mathcal{D}_i$ on $\mathcal{V}_i$. For any $\boldsymbol{x} \in \mathcal{V}_i$ with $m < |\boldsymbol{x}| < M$, it holds that

$$0 < p_i(\boldsymbol{x}) < p_{\max} < \infty.$$

Later on, we denote $\mathcal{D}$ to be the evenly mixed distribution of $\mathcal{D}_i$ for $i \in [n]$.

We have Table 3.1 for notations used in this chapter.

Table 3.1: Frequently Used Notations

Symbols	**Definitions**
$[n]$	$\{1, 2, \cdots, n\}$
$\mathbb{1}_{\mathcal{S}}(x)$	indicator function which take value 1 for $x \in \mathcal{S}$ and 0 for $x \notin \mathcal{S}$
$\lvert\boldsymbol{x}\rvert$	ℓ_2-norm of vector $\boldsymbol{x}$
$\lvert\boldsymbol{W}\rvert$	the collumn-wise ℓ_2-norm sum for a matrix $\boldsymbol{W}$. For $\boldsymbol{W} := [\boldsymbol{w}_1, \cdots, \boldsymbol{w}_k]$, $\lvert\boldsymbol{W}\rvert = \sum_{j=1}^{k} \lvert\boldsymbol{w}_j\rvert$
$\mathcal{H}^d$	d-dimensional Hausdorff measure
$\tilde{\boldsymbol{x}}$	the unit vector in the direction of $\boldsymbol{x}$, i.e., $\tilde{\boldsymbol{x}} := \frac{\boldsymbol{x}}{\lvert\boldsymbol{x}\rvert}$. Additionally, $\tilde{\mathbf{0}} := \mathbf{0}$.
σ	quantized ReLU function
$\Omega_{\boldsymbol{W}}$	$\{\boldsymbol{x} \in \mathcal{X} : l(\boldsymbol{W}; \{\boldsymbol{x}, y\}) > 0\}$
$\Omega_{\boldsymbol{v}}^{a}$	$\{\boldsymbol{x} \in \mathcal{X} : \langle \boldsymbol{v}, \boldsymbol{x} \rangle > a\}$
$\Omega_{\boldsymbol{W}}^{j}$	$\Omega_{\boldsymbol{W}} \cap \Omega_{\boldsymbol{w}_j}^{0}$

Remark 3.1. *The orthogonality of subspaces $\mathcal{V}_i$'s in the data assumption (1) above is technically needed for our proof here. However, the convergence in Theorem 3.1 to a perfect classification with random initialization is observed in more general settings when $\mathcal{V}_i$'s form acute angles and contain a certain level of noise. We refer to section 3.7.1 for supporting experimental results.*

Remark 3.2. *Assumption (3) can be relaxed to the following, while the proof remains basically the same.*

$\mathcal{D}_i$ is a mixture of n_i distributions namely $\mathcal{D}_{i,j}$ for $j \in [n_i]$. There exists a linear decomposition of $\mathcal{V}_i = \bigoplus_{j=1}^{n_i} \mathcal{V}_{i,j}$ and $\mathcal{D}_{i,j}$ each has a marginal probability distribution function $p_{i,j}$ on $\mathcal{V}_{i,j}$. For any $\boldsymbol{x} \in \mathcal{V}_{i,j}$ and $< m < |\boldsymbol{x}| < M$, it holds that

$$0 < p_{i,j}(\boldsymbol{x}) \leq p_{max} < \infty.$$

3.1.2 Network Architecture

We consider a two-layer neural architecture with k hidden neurons. Denote by $\boldsymbol{W} = [\boldsymbol{w}_1, \cdots, \boldsymbol{w}_k] \in \mathbb{R}^{d \times k}$ the weight matrix in the hidden layer. Let

$$h_j = \langle \boldsymbol{w}_j, \boldsymbol{x} \rangle$$

the input to the activation function, or the so-called pre-activation. **Throughout this chapter, we make the following assumptions:**

Assumption 3.1. *The weight matrix in the second layer $\boldsymbol{V} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n]$ is fixed and known in the training process and satisfies:*

1. *For any $i \in [n]$, there exists some $j \in [k]$ such that $v_{i,j} > 0$.*
2. *If $v_{i,j} > 0$, then for any $r \in [n]$ and $r \neq i$, we have $v_{r,j} = 0$.*
3. *For any $i \in [n]$ and $j \in [k]$ we have $v_{i,j} < 1$.*

One can easily show that as long as $k \geq n$, such a matrix $\boldsymbol{V} = (v_{i,j})$ is ubiquitous.

For any input data $\boldsymbol{x} \in \mathcal{X} = \mathbb{R}^d$, the neural net output is

$$f(\boldsymbol{W}; \boldsymbol{x}) = [o_1, \cdots, o_n], \tag{3.1}$$

where

$$o_i = \langle \boldsymbol{v}_i, \sigma(\boldsymbol{h}) \rangle = \sum_{j=1}^{k} v_{i,j} \sigma(h_j).$$

The $\sigma(\cdot)$ is the quantized ReLU function acting element-wise; see Fig. 1.1 for examples of binary and ternary activation functions. More general quantized ReLU function of the bit-width b can be defined as follows:

$$\sigma(x) = \begin{cases} 0 & \text{if} \quad x \leq 0, \\ \text{ceil}(x) & \text{if} \quad 0 < x < 2^b - 1, \\ 2^b - 1 & \text{if} \quad x \geq 2^b - 1. \end{cases}$$

The prediction is given by the network output label

$$\hat{y}(\boldsymbol{W}, \boldsymbol{x}) = \underset{r \in [n]}{\text{argmax}}\, o_r,$$

ideally $\hat{y}(\boldsymbol{x}) = i$ for all $\boldsymbol{x} \in \mathcal{V}_i$. The classification accuracy in percentage is the frequency that this event occurs (when network output label $\hat{y}$ matches the true label) on a validation data set.

Given the data sample $\{\boldsymbol{x}, y\}$, the associated hinge loss function reads

$$l(\boldsymbol{W}; \{\boldsymbol{x}, y\}) := \max\{0, 1 - f_y\} := \max\left\{0, 1 - \left(o_y - \max_{i \neq y} o_i\right)\right\}. \tag{3.2}$$

To train the network with quantized activation σ, we consider the following population loss minimization problem

$$\min_{\boldsymbol{W}\in\mathbb{R}^{d\times k}} l\left(\boldsymbol{W}\right) := \mathop{\mathbb{E}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}}\left[l\left(\boldsymbol{W};\{\boldsymbol{x},y\}\right)\right], \tag{3.3}$$

where the sample loss $l\left(\boldsymbol{W};\{\boldsymbol{x},y\}\right)$ is defined in (3.2). Let l_i be the population loss function of class i with the label $y=i$, $i\in[n]$. More precisely,

$$\begin{aligned} l_i(\boldsymbol{W}) &= \mathop{\mathbb{E}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}\left[\max\left\{0, 1-f_i\right\}\right] \\ &= \mathop{\mathbb{E}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}\left[\max\left\{0, 1-\left(o_i-\max_{r\neq i} o_r\right)\right\}\right]. \end{aligned}$$

Thus, we can rewrite the loss function as

$$l(\boldsymbol{W}) = \frac{1}{n}\sum_{i=1}^{n} l_i(\boldsymbol{W}).$$

Note that the population loss

$$l_i(\boldsymbol{W}) = \mathop{\mathbb{E}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}\left[l(\boldsymbol{W};\{\boldsymbol{x},y\})\right]$$

fails to have simple closed-form solution even if p_i are constant functions on their supports. We do not have closed-form formula at hand to analyze the learning process, which makes our analysis challenging.

For notational convenience, we define:

$$\Omega_{\boldsymbol{W}} = \left\{\boldsymbol{x}\in\mathcal{X} : l(\boldsymbol{W};\{\boldsymbol{x},y\}) > 0\right\},$$

$$\Omega_{\boldsymbol{v}}^{a} = \left\{\boldsymbol{x}\in\mathcal{X} : \langle\boldsymbol{v},\boldsymbol{x}\rangle > a\right\},$$

and

$$\Omega_{\boldsymbol{W}}^{j} = \Omega_{\boldsymbol{W}}\cap\Omega_{\boldsymbol{w}_j}^{0}.$$

3.1.3 Coarse Gradient Methods

We see that derivative of quantized ReLU function σ is a.e. zero, which gives a trivial gradient of sample loss function with respect to (w.r.t.) $\boldsymbol{w}_j$. Indeed, differentiating the sample loss function with respect to $\boldsymbol{w}_j$, we have

$$\nabla_{\boldsymbol{w}_j} l(\boldsymbol{W}; \{\boldsymbol{x}, y\}) = -\left(v_{y,j} - v_{\xi,j}\right) \mathbb{1}_{\Omega_{\boldsymbol{W}}}(\boldsymbol{x})\, \sigma'\left(h_j\right) \boldsymbol{x} = \boldsymbol{0}, \text{ a.e.}, \quad 1 \leq j \leq k$$

where $\xi = \operatorname{argmax}_{i \neq y} o_i$.

The partial coarse gradient w.r.t. $\boldsymbol{w}_j$ associated with the sample $\{\boldsymbol{x}, y\}$ is given by replacing σ' with a straight through estimator (STE) which is the derivative of function g, namely,

$$\tilde{\nabla}_{\boldsymbol{w}_j} l(\boldsymbol{W}; \{\boldsymbol{x}, y\}) := -\left(v_{y,j} - v_{\xi,j}\right) \mathbb{1}_{\Omega_{\boldsymbol{W}}}(\boldsymbol{x})\, g'(h_j) \boldsymbol{x}. \tag{3.4}$$

The sample coarse gradient $\tilde{\nabla} l(\boldsymbol{W}; \{\boldsymbol{x}, y\})$ is just the concatenation of $\tilde{\nabla}_{\boldsymbol{w}_j} l(\boldsymbol{W}; \{\boldsymbol{x}, y\})$'s. It is worth noting that coarse gradient is not an actual gradient, but some biased first-order oracle which depends on the choice of g.

Throughout this chapter, we consider a class of surrogate functions during the backward pass with the following properties:

Assumption 3.2. *$g : \mathbb{R} \to \mathbb{R}$ satisfies*

1. *$g(x) = 0$ for all $x \leq 0$.*
2. *$g'(x) \in [\delta, \tilde{\delta}]$ for all $x > 0$ with some constants $0 < \delta < \tilde{\delta} < \infty$.*

Such a g is ubiquitous in quantized deep networks training; see Fig.3.1 for examples of $g(x)$ satisfying Assumption 3.2. Typical examples include the classical ReLU $g(x) = \max(x, 0)$

and log-tailed ReLU [7]:

$$g(x) = \begin{cases} 0 & \text{if} \quad x \leq 0 \\ x & \text{if} \quad 0 < x \leq q_b \\ q_b + \log(x - q_b + 1) & \text{if} \quad x > q_b \end{cases}$$

where $q_b := 2^b - 1$ is the maximum quantization level. In addition, if the input of the activation function is bounded by a constant, one also can use $g(x) = \max\{0, q_b(1 - e^{-x/q_b})\}$, which we call *reverse exponential STE*.

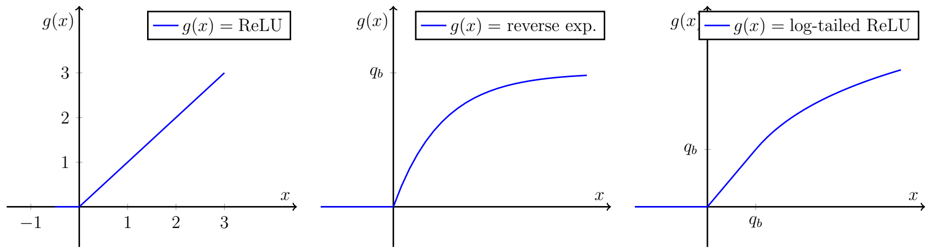


Figure 3.1: Different choices of $g(x)$ for the straight-through estimator.

To train the network with quantized activation σ, we use the expectation of coarse gradient over training samples:

$$\tilde{\nabla} l(\boldsymbol{W}) := \mathop{\mathbb{E}}_{\{\boldsymbol{x}, y\} \sim \mathcal{D}} \tilde{\nabla} l(\boldsymbol{W}; \{\boldsymbol{x}, y\})$$

where $\tilde{\nabla} l(\boldsymbol{W}; \{\boldsymbol{x}, y\})$ is given by (3.4). In this chapter, we study the convergence of coarse gradient algorithm for solving the minimization problem (3.3), which takes the following iteration with some learning rate $\eta > 0$:

$$\boldsymbol{W}^{t+1} = \boldsymbol{W}^t - \eta \, \tilde{\nabla} l(\boldsymbol{W}^t) \tag{3.5}$$

3.2 Main Result and Outline of Proof

We show that if the iterates $\{\boldsymbol{W}^t\}$ are uniformly bounded in t, coarse gradient decent with the proxy function g under Assumption 3.2 converges to a global minimizer of the population loss, resulting in a perfect classification.

Theorem 3.1. *Suppose data assumptions (1)-(3) and STE assumptions 3.1-3.2 hold. If the network initialization satisfies $\mathbf{w}_{j,i}^0 \neq 0$ for all $j \in [k]$ and $i \in [n]$ and $\boldsymbol{W}^t$ is uniformly bounded by R in t, then for all $v_{i,j} > 0$ we have*

$$\lim_{t \to \infty} \left| \tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t) \right| = 0.$$

Furthermore, if $\boldsymbol{W}^\infty$ is an accumulation point of $\{\boldsymbol{W}^t\}$ and all non-zero unit vectors $\tilde{\boldsymbol{w}}_{j,i}^\infty$'s are distinct for all $j \in [k]$ and $i \in [n]$, then

$$\mathbb{P}_{\{\boldsymbol{x},y\} \sim \mathcal{D}} \left(\hat{y} \left(\boldsymbol{W}^\infty, \boldsymbol{x} \right) \neq y \right) = 0.$$

We outline the major steps in the proof below.

Step 1: Decompose the population loss into n components. Recall the definition of l_i which is population loss functions for $\{\boldsymbol{x}, y\} \sim \mathcal{D}_i$. In Section 4, we show under certain decomposition of $\boldsymbol{W}$, the coarse gradient decent of each one of them only affects a corresponding component of $\boldsymbol{W}$.

Step 2: Bound the total increment of weight norm from above. Show that for all $v_{i,j} > 0$ we have $|\boldsymbol{w}_{j,i}|$'s are monotonically increasing under coarse gradient descent. Based on boundedness on $\boldsymbol{W}$, we further give an upper bound on the total increment of all $|\boldsymbol{w}_j|$'s, from which the convergence of coarse gradient descent follows.

Step 3: Show that when the coarse gradient vanishes, so does the population loss. In section 6, we show that when the coarse gradient vanishes towards the end of training, the population loss is zero which implies a perfect classification.

3.3 Space Decomposition

With $\mathcal{V} = \bigoplus_{i=1}^{n} \mathcal{V}_i$, we have the orthogonal complement of $\mathcal{V}$ in $\mathcal{X} = \mathbb{R}^d$, namely $\mathcal{V}_{n+1}$. Now, we can decompose $\mathcal{X} = \mathbb{R}^d$ into $n+1$ linearly independent parts:

$$\mathbb{R}^d = \mathcal{V} \bigoplus \mathcal{V}_{n+1} = \bigoplus_{i=1}^{n+1} \mathcal{V}_i$$

and for any vector $\boldsymbol{w}_j \in \mathbb{R}^d$, we have a unique decomposition of $\boldsymbol{w}_j$:

$$\boldsymbol{w}_j = \sum_{i=1}^{n+1} \boldsymbol{w}_{j,i},$$

where $\boldsymbol{w}_{j,i} \in \mathcal{V}_i$ for $i \in [n+1]$. To simply notation, we let

$$\boldsymbol{W}_i = \left[\boldsymbol{w}_{1,i}, \cdots, \boldsymbol{w}_{k,i}\right].$$

Lemma 3.1. *For any $\boldsymbol{W} \in \mathbb{R}^{k\times d}$ and $i \in [n]$, we have*

$$l_i\left(\boldsymbol{W}\right) = l_i\left(\sum_{r=1}^{n} \boldsymbol{W}_r\right) = l_i(\boldsymbol{W}_i).$$

Proof. Note that for any $\boldsymbol{x} \in \mathcal{V}_i$ and $j \in [k]$, we have $\boldsymbol{x} \in \mathcal{V}$, so

$$\langle \boldsymbol{w}_{j,n+1}, \boldsymbol{x} \rangle = 0$$

and

$$h_j = \langle \boldsymbol{w}_j, \boldsymbol{x} \rangle = \left\langle \sum_{j=1}^{k} \boldsymbol{w}_{j,i}, \boldsymbol{x} \right\rangle = \langle \boldsymbol{w}_{j,i}, \boldsymbol{x} \rangle .$$

Hence

$$f(\boldsymbol{W}; \boldsymbol{x}) = f\left(\sum_{j=1}^{k} \boldsymbol{W}_i; \boldsymbol{x} \right) = f(\boldsymbol{W}_i)$$

for all $\boldsymbol{W} \in \mathbb{R}^{d \times k}$, $\boldsymbol{x} \in \mathcal{V}_i$. The desired result follows. □

Lemma 3.2. *Running the algorithm (3.5) on l_i only does not change the value of $\boldsymbol{W}_r$ for all $r \neq i$. More precisely, for any $\boldsymbol{W} \in \mathbb{R}^{d \times k}$, let*

$$\boldsymbol{W}' = \boldsymbol{W} - \eta \tilde{\nabla} l_i(\boldsymbol{W}),$$

then for any $r \in [n]$ and $r \neq i$

$$\boldsymbol{W}'_r = \boldsymbol{W}_r.$$

Proof of Lemma 3.2. Assume $i, r \in [n]$ and $i \neq r$. Note that

$$\boldsymbol{w}'_j = \boldsymbol{w}_j - \eta \tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W})$$

and

$$\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}) = - \mathop{\mathbb{E}}_{\{\boldsymbol{x}, y\} \sim \mathcal{D}_i} \left[(v_{y,j} - v_{\xi,j}) \, \mathbb{1}_{\Omega_{\boldsymbol{W}}}(\boldsymbol{x}) \, g'(h_j) \boldsymbol{x} \right] \in V_i.$$

Since $\mathcal{V}_i$'s are linearly independent, we have

$$\boldsymbol{w}'_{j,i} = \boldsymbol{w}_{j,i} - \eta \tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W})$$

and

$$\boldsymbol{w}'_{j,r} = \boldsymbol{w}_{j,r}.$$

□

By the above result, we know (3.5) is equivalent to

$$\boldsymbol{W}_i^{t+1} = \boldsymbol{W}_i^t - \frac{\eta}{n}\tilde{\nabla} l_i\left(\boldsymbol{W}^t\right). \tag{3.6}$$

3.4 Learning Dynamics

In this section, we show that some components of the weight iterates have strictly increasing magnitude whenever coarse gradient does not vanish, and it quantifies the increment during each iteration.

Lemma 3.3. *Assume*

$$\hat{v}_j = \max_{i_1, i_2 \in [n]} v_{i_1,j} - v_{i_2,j}\,,$$

we have the following estimate:

$$\mathop{\mathbb{P}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}\left(\Omega_W^j\right) \geq \frac{1}{\hat{v}_j\tilde{\delta}M}\left|\tilde{\nabla}_{\boldsymbol{w}_j} l_i\left(\boldsymbol{W}\right)\right|.$$

Proof of Lemma 3.3.

$$\begin{aligned}\left|\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W})\right| =& \left|\mathop{\mathbb{E}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}\left[(v_{y,j} - v_{\xi,j})\ \mathbb{1}_{\Omega_{\boldsymbol{W}}}(\boldsymbol{x})\, g'(h_j)\boldsymbol{x}\right]\right| \\ \leq& \hat{v}_j\tilde{\delta}M \mathop{\mathbb{E}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}\left[\mathbb{1}_{\Omega_{\boldsymbol{W}}^j}(\boldsymbol{x})\right] \\ =& \hat{v}_j\tilde{\delta}M \mathop{\mathbb{P}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}\left(\Omega_{\boldsymbol{W}}^j\right)\end{aligned}$$

□

Lemma 3.4. *For any $j \in [k]$ if*

$$\tilde{v}_{i,j} := v_{i,j} - \max_{r\neq i} v_{r,j} > 0$$

we have

$$\left\langle \tilde{\boldsymbol{w}}_{j,i}, -\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}) \right\rangle \geq \frac{\tilde{v}_{i,j}\delta}{2C_p} \underset{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}{\mathbb{P}} \left(\Omega_W^j\right)^2,$$

where

$$C_p = \max_{\boldsymbol{v}\in V_i, a\in\mathbb{R}} \int_{\langle \boldsymbol{v},\boldsymbol{x}\rangle = a} p_i(\boldsymbol{x})\, d\,\mathcal{H}^{d_i-1}(\boldsymbol{x}).$$

Proof of Lemma 3.4. First, we prove an inequality which will be used later. Recall that $|\boldsymbol{x}| \leq M$, and that $\tilde{\nabla}_{\boldsymbol{w}_j} l(\boldsymbol{W}, \{\boldsymbol{x}, y\}) \neq 0$ only when $\boldsymbol{x} \in \Omega_{\boldsymbol{W}}^j$. Hence, we have $\langle \tilde{\boldsymbol{w}}_{j,i}, \boldsymbol{x}\rangle > 0$. We have

$$\begin{aligned}
\underset{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}{\mathbb{P}} \left(\Omega_{\boldsymbol{W}}^j \cap \{\boldsymbol{x} : \langle \tilde{\boldsymbol{w}}_{j,i}, \boldsymbol{x}\rangle < t\}\right) &= \int_{\Omega_{\boldsymbol{W}}^j} \mathbb{1}_{\{\langle \tilde{\boldsymbol{w}}_{j,i}, \boldsymbol{x}\rangle < t\}}(\boldsymbol{x}) p_i(\boldsymbol{x})\, d\,\boldsymbol{x} \\
&= \int_0^t \int_{\langle \tilde{\boldsymbol{w}}_{j,i}, \boldsymbol{x}\rangle = s} p_i(\boldsymbol{x})\, d\,\mathcal{H}^{d_i-1}(\boldsymbol{x})\, d\,s \\
&\leq t\; C_p.
\end{aligned}$$

Now, we use Fubini's Theorem to simplify the inner product:

$$\begin{aligned}
\left\langle \tilde{\boldsymbol{w}}_{j,i}, -\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}) \right\rangle &= \underset{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}{\mathbb{E}} \left[(v_{y,j} - v_{\xi,j})\, \mathbb{1}_{\Omega_{\boldsymbol{W}}^j}(\boldsymbol{x})\, g'(h_j)\, \langle \tilde{\boldsymbol{w}}_{j,i}, \boldsymbol{x}\rangle\right] \\
&\geq \tilde{v}_{i,j}\, \delta \int_{\Omega_{\boldsymbol{W}}^j \cap V_i} \langle \tilde{\boldsymbol{w}}_{j,i}, \boldsymbol{x}\rangle p_i(\boldsymbol{x})\, d\,\boldsymbol{x} \\
&= \tilde{v}_{i,j}\, \delta \int_{\Omega_{\boldsymbol{W}}^j \cap V_i} \int_0^\infty \mathbb{1}_{\{\langle \tilde{\boldsymbol{w}}_{j,i}, \boldsymbol{x}\rangle > t\}}\, d\,t\; p_i(\boldsymbol{x})\, d\,\boldsymbol{x} \\
&= \tilde{v}_{i,j}\, \delta \int_0^\infty \int_{\Omega_{\boldsymbol{W}}^j \cap V_i} \mathbb{1}_{\{\langle \tilde{\boldsymbol{w}}_{j,i}, \boldsymbol{x}\rangle > t\}}\; p_i(\boldsymbol{x})\, d\,\boldsymbol{x}\, d\,t \\
&= \tilde{v}_{i,j}\, \delta \int_0^\infty \underset{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}{\mathbb{P}} \left(\Omega_{\boldsymbol{W}}^j \cap \{\boldsymbol{x} : \langle \tilde{\boldsymbol{w}}_{j,i}, \boldsymbol{x}\rangle > t\}\right) d\,t.
\end{aligned}$$

Now using the inequality just proved above, we have

$$\begin{aligned}
&\mathop{\mathbb{P}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}\left(\Omega_{\boldsymbol{W}}^j \cap \{\boldsymbol{x} : \langle \tilde{\boldsymbol{w}}_{j,i}, \boldsymbol{x}\rangle > t\}\right) \\
=&\mathop{\mathbb{P}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}\left(\Omega_{\boldsymbol{W}}^j\right) - \mathop{\mathbb{P}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}\left(\Omega_{\boldsymbol{W}}^j \cap \{\boldsymbol{x} : \langle \tilde{\boldsymbol{w}}_{j,i}, \boldsymbol{x}\rangle < t\}\right) \\
\geq& \max\left\{\mathop{\mathbb{P}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}\left(\Omega_{\boldsymbol{W}}^j\right) - t\, C_p, 0\right\}.
\end{aligned}$$

Combining the above two inequalities, we have

$$\begin{aligned}
\left\langle \tilde{\boldsymbol{w}}_{j,i}, -\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W})\right\rangle \geq& \tilde{v}_{i,j}\, \delta \int_0^\infty \max\left\{\mathop{\mathbb{P}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}\left(\Omega_{\boldsymbol{W}}^j\right) - t\, C_p, 0\right\} dt \\
\geq& \frac{\tilde{v}_{i,j}\, \delta}{2C_p} \mathop{\mathbb{P}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}\left(\Omega_{\boldsymbol{W}}^j\right)^2.
\end{aligned}$$

□

Lemma 3.5. *If $\tilde{v}_{i,j} > 0$ in Lemma 3.4, then $\{|\boldsymbol{w}_{j,i}^t|\}$ in Equation (3.1) is non-decreasing with coarse gradient decent (3.5). Moreover, under the same assumption, we have*

$$\left|\boldsymbol{w}_{j,i}^{t+1}\right| - \left|\boldsymbol{w}_{j,i}^t\right| \geq \frac{\eta \tilde{v}_{i,j}\delta}{2nC_p \hat{v}_j^2 \tilde{\delta}^2 M^2}\left|\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t)\right|^2,$$

where C_p is defined as in Lemma 3.4 and $\hat{v}_j$ as in Lemma 3.3.

Proof of Lemma 3.5. Since $\boldsymbol{w}_{j,i}^{t+1} = \boldsymbol{w}_{j,i}^t - \frac{\eta}{n}\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t)$, we have

$$\left|\boldsymbol{w}_{j,i}^{t+1}\right| - \left|\boldsymbol{w}_{j,i}^t\right| \geq \left\langle \boldsymbol{w}_{j,i}^{t+1} - \boldsymbol{w}_{j,i}^t, \tilde{\boldsymbol{w}}_{j,i}^t\right\rangle = \left\langle -\frac{\eta}{n}\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t), \tilde{\boldsymbol{w}}_{j,i}^t\right\rangle.$$

Hence, it follows from Lemma 3.3 and Lemma 3.4 that

$$\left|\boldsymbol{w}_{j,i}^{t+1}\right| - \left|\boldsymbol{w}_{j,i}^t\right| \geq \frac{\eta \tilde{v}_{i,j}\delta}{2nC_p \hat{v}_j^2 \tilde{\delta}^2 M^2}\left|\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t)\right|^2, \tag{3.7}$$

which is the desired result. □

Note that one component of $\boldsymbol{w}_j$ is increasing but the weights are bounded by assumption, hence, summation of the increments over all steps should also be bounded. This gives the following proposition:

Proposition 3.1. *Assume $\{|\boldsymbol{w}_j^t|\}$ is bounded by R, then if $\tilde{v}_{i,j} > 0$ in Lemma 3.4, then*

$$\sum_{t=1}^{\infty} \left| \tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t) \right|^2 \leq \frac{2nC_p \hat{v}_j^2 \tilde{\delta}^2 M^2 R}{\eta \tilde{v}_{i,j} \delta} < \infty,$$

where C_p is as defined in Lemma 3.4 and $\hat{v}_j$ defined in Lemma 3.3. This implies that

$$\lim_{t\to\infty} \left| \tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t) \right| = 0$$

as long as $\tilde{v}_{i,j} > 0$.

Remark 3.3. *Lemmas 3.3, 3.4, 3.5 and Proposition 3.1 were proved without Assumption 3.1. Under Assumption 3.1, we have $\hat{v}_j = \max_{i\in[n]} v_{i,j}$ in Lemma 3.3 and $\tilde{v}_{i,j} = \hat{v}_j$ if $v_{i,j} > 0$ and $\tilde{v}_{i,j} = -\hat{v}_j$ if $v_{i,j} = 0$ in Lemma 3.4.*

3.5 Landscape Properties

We have shown that under boundedness assumptions, the algorithm will converge to some point where the coarse gradient vanishes. However, this does not immediately indicate the convergence to a valid point because coarse gradient is a fake gradient. We will need the following lemma to prove Proposition 3.2, which confirms that the points with zero coarse gradient are indeed global minima.

Lemma 3.6. *Let $\Omega = \{\boldsymbol{x} \in \mathbb{R}^l : m < |\boldsymbol{x}| < M\}$, where $0 < m < M < \infty$. For $j \in [k]$, let $\Omega_j = \{\boldsymbol{x} : \langle \boldsymbol{w}_j, \boldsymbol{x} \rangle > a\}$, where $a \geq 0$ and $\Omega_i \neq \Omega_j$ for all $i \neq j$. If for $i \in [k]$ and $\boldsymbol{x} \in \Omega_i \cap \Omega$,* *there exists some $j \neq i$ such that $\boldsymbol{x} \in \Omega_j$, then*

$$\left(\bigcup_{j=1}^{k} \Omega_j\right) \cap \Omega = \emptyset \quad or \quad \Omega.$$

Proof of Lemma 3.6. Define $\tilde{\Omega} = \bigcup_{j=1}^{k} \Omega_j$, by De Morgan's law, we have

$$\tilde{\Omega}^c = \left(\bigcup_{j=1}^{k} \Omega_j\right)^c = \bigcap_{j=1}^{k} \Omega_j^c.$$

Note that k is finite and $\mathbf{0} \in \Omega_j^c$ for all $j \in [k]$, we know $\tilde{\Omega}^c$ is a generalized polyherdon and hence either

$$\left(\partial\tilde{\Omega}\right) \cap \Omega = \emptyset$$

or

$$\mathcal{H}^{l-1}\left(\left(\partial\tilde{\Omega}\right) \cap \Omega\right) > 0.$$

The first case is trivial. We show that the second case contradicts our assumption. Note that

$$\partial\tilde{\Omega} = \partial\left(\bigcup_{j=1}^{k} \Omega_j\right) \subseteq \bigcup_{j=1}^{k} \partial\Omega_j,$$

we know there exists some $j^\star \in [k]$ such that $\mathcal{H}^{l-1}\left(\partial\Omega_{j^\star} \cap \Omega\right) > 0$. It follows from our assumption that $\tilde{\Omega} = \cup_{j=1}^{k} \Omega_j = \cup_{j \neq j^\star} \Omega_j$, and hence

$$\mathcal{H}^{l-1}\left(\partial\Omega_{j^\star} \cap \partial\Omega_j\right) > 0.$$

Note that $\partial\Omega_j$'s are hyperplanes. Therefore, $\Omega_j = \Omega_{j^\star}$, contradicting with our assumption that all Ω_j's are distinct. □

The following result shows that the coarse gradient vanishes only at a global minimizer with zero loss, except for some degenerate cases.

Proposition 3.2. *Under Assumption 3.1, if $\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}) = \mathbf{0}$ for all $\tilde{v}_{i,j} > 0$ and $\tilde{\boldsymbol{w}}_{j,i}$'s are* *distinct, then* $l_i(\boldsymbol{W}) = 0$.

Proof of Proposition 3.2. For quantized ReLU function, let $q_b := \max_{x\in\mathbb{R}} \sigma(x)$ be the maximum quantization level, so that

$$\sigma(x) = \sum_{a=0}^{q_b-1} \mathbb{1}_{\{x>a\}}(x).$$

Note that

$$f_i\left(\boldsymbol{W};\boldsymbol{x}\right) = o_i - o_\xi = \sum_{j=1}^{k} \left(v_{i,j} - v_{\xi,j}\right)\sigma(h_j) = \sum_{j=1}^{k} \left(v_{i,j} - v_{\xi,j}\right)\sum_{a=0}^{q_b} \mathbb{1}_{\Omega^a_{\boldsymbol{w}_j}}(\boldsymbol{x}).$$

By assumption, $\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}) = \boldsymbol{0}$ for all $\tilde{v}_{i,j} > 0$ which implies $\mathbb{1}_{\Omega_{\boldsymbol{W}}}(\boldsymbol{x})\mathbb{1}_{\Omega^a_{\boldsymbol{w}_j}}(\boldsymbol{x}) = 0$ for all $\tilde{v}_{i,j} > 0$ and $a \in [n]$ almost surely. Now, for any $\boldsymbol{x} \in \Omega^a_{\boldsymbol{w}_j}$ we have $\boldsymbol{x} \notin \Omega_{\boldsymbol{W}}$. Note that $\boldsymbol{x} \in \Omega_{\boldsymbol{W}}$ if and only if $o_i - o_\xi \geq 1$, then for any $\boldsymbol{x} \in \Omega^a_{\boldsymbol{w}_j}$, since $v_{i,j} - v_{\xi,j} < 1$, there exist $j' \neq j$ and $a' \in [n]$ such that $v_{i,j'} > 0$ and $\boldsymbol{x} \in \Omega^{a'}_{\boldsymbol{w}_{j'}}$. By Lemma 3.6, $\mathbb{P}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i}\left[\Omega_{\boldsymbol{W}}\right] = 0$ is empty, and thus $l_i(\boldsymbol{W}) = 0$. □

The following lemma shows that the expected coarse gradient is continuous except at $\boldsymbol{w}_{j,i} = \boldsymbol{0}$ for some $j \in [k]$.

Lemma 3.7. *Consider the network in (3.1).* $\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W})$ *is continuous on*

$$\left\{\boldsymbol{W} \in \mathbb{R}^{k\times d} : |\boldsymbol{w}_{j,i}| > 0 \textit{ for all } j \in [k], i \in [n]\right\}.$$

Proof of Lemma 3.7. It suffices to prove the result for $j \in [k]$. Note that

$$\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}) = \mathop{\mathbb{E}}_{\{\boldsymbol{x},y\}\sim\mathcal{D}_i} \left[-\left(v_{y,j} - v_{\xi,j}\right) \mathbb{1}_{\Omega_{\boldsymbol{W}}}(\boldsymbol{x})\, g'(h_j)\boldsymbol{x}\right]$$

For any $\boldsymbol{W}^0$ satisfying our assumption, we know

$$\lim_{\boldsymbol{W}\to\boldsymbol{W}^0} \mathbb{1}_{\Omega_{\boldsymbol{W}}}(\boldsymbol{x})g'(h_j) = \mathbb{1}_{\Omega_{\boldsymbol{W}^0}}(\boldsymbol{x})g'(h_j^0), \text{ a.e.}$$

The desired result follows from the Dominant Convergence Theorem. □

3.6 Proof of Main Results

Equipped with the technical lemmas, we present:

Proof of Theorem 3.1. It is easily noticed from Assumption 3.1 that $v_{i,j} > 0$ if and only if $\tilde{v}_{i,j} > 0$. By Lemma 3.5, if $v_{i,j} > 0$ and $|\boldsymbol{w}_{j,i}^0| > 0$, then $|\boldsymbol{w}_{j,i}^t| > 0$ for all t. Since $\boldsymbol{W}$ is randomly initialized, we can ignore the possibility that $\boldsymbol{w}_{j,i}^0 = \mathbf{0}$ for some $j \in [k]$ and $i \in [n]$. Moreover, Proposition 3.1 and Equation (3.5) imply for all $v_{i,j} > 0$

$$\lim_{t\to\infty} \left| \tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^t) \right| = 0.$$

Suppose $\boldsymbol{W}^\infty$ is an accumulation point and $\boldsymbol{w}_{j,r}^\infty \neq \mathbf{0}$ for all $j \in [k]$ and $r \in [n]$, we know for all $v_{i,j} > 0$

$$\tilde{\nabla}_{\boldsymbol{w}_j} l_i\left(\boldsymbol{W}^\infty\right) = \mathbf{0}.$$

Next, we consider the case when $\boldsymbol{w}_{j,r} = \mathbf{0}$ for some $j \in [k]$ and $r \in [n]$. Lemma 3.4 implies $v_{r,j} = 0$. We construct a new sequence

$$\hat{\boldsymbol{w}}_{j,r}^t = \begin{cases} \boldsymbol{w}_{j,r}^t & \text{if } \boldsymbol{w}_{j,r}^\infty \neq 0 \\ \mathbf{0} & \text{if } \boldsymbol{w}_{j,r}^\infty = 0 \end{cases}$$

and

$$\hat{\boldsymbol{W}}_r^t = \left[\hat{\boldsymbol{w}}_{1,r}^t, \cdots, \hat{\boldsymbol{w}}_{k,r}^t\right].$$

With

$$\hat{o}_r = \sum_{j=1}^{k} v_{r,j}\sigma(\hat{h}_j) = \sum_{j=1}^{k} v_{r,j}\sigma\left(\langle \hat{\boldsymbol{w}}_{j,r}, \boldsymbol{x}\rangle\right),$$

we know $\hat{o}_r = o_r$ for all $r \in [n]$. Hence, we have

$$l\left(\hat{\boldsymbol{W}}^t, \{\boldsymbol{x}, i\}\right) = \operatorname{ReLU}\left(1 - \hat{o}_i + \hat{o}_\xi\right) = l\left(\boldsymbol{W}^t, \{\boldsymbol{x}, i\}\right).$$

This implies that $\Omega_{\hat{\boldsymbol{W}}^t} = \Omega_{\boldsymbol{W}^t}$, so we have for all $j \in [k]$,

$$\left|\left\langle \tilde{\nabla}_{\boldsymbol{w}_j} l_i(\hat{\boldsymbol{W}}_1^t), \tilde{\boldsymbol{w}}_{j,i}^t \right\rangle\right| \le \left|\left\langle \tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}_i^t), \tilde{\boldsymbol{w}}_{j,i}^t \right\rangle\right| \le \left|\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}_i^t)\right|.$$

Letting t go to infinity on both side, we get

$$\left|\left\langle \tilde{\nabla}_{\boldsymbol{w}_j} l_i(\hat{\boldsymbol{W}}^\infty), \tilde{\boldsymbol{w}}_{j,i}^\infty \right\rangle\right| = 0.$$

By Lemma 3.3 and Lemma 3.4, we know

$$\tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}^\infty) = \tilde{\nabla}_{\boldsymbol{w}_j} l_i(\boldsymbol{W}_i^\infty) = 0,$$

so $\tilde{\nabla}_{\boldsymbol{W}} l_i(\boldsymbol{W}^\infty) = 0$. By Proposition 3.2, $l_i(\boldsymbol{W}^t) = 0$, which completes the proof. □

3.7 Experiments

In this section, we conduct experiments on both synthetic and MNIST data to verify and complement our theoretical findings. Experiments on larger networks and data sets will left for a future work.

3.7.1 Synthetic Data

Let $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3, \boldsymbol{e}_4\}$ be orthonormal basis of $\mathbb{R}^4$, θ be an acute angle and $\boldsymbol{v}_1 = \boldsymbol{e}_1$, $\boldsymbol{v}_2 = \sin\theta\, \boldsymbol{e}_2 + \cos\theta\, \boldsymbol{e}_3$, $\boldsymbol{v}_3 = \boldsymbol{e}_3$, $\boldsymbol{v}_4 = \boldsymbol{e}_4$. Now, we have two linearly independent subspaces of $\mathbb{R}^4$ namely $\mathcal{V}_1 = \operatorname{Span}(\{\boldsymbol{v}_1, \boldsymbol{v}_2\})$ and $\mathcal{V}_2 = \operatorname{Span}(\{\boldsymbol{v}_3, \boldsymbol{v}_4\})$. We can easily calculate that the angle between $\mathcal{V}_1$ and $\mathcal{V}_2$ is θ. Next, with

$$S_r = \left\{\frac{j}{10} : j \in [20] - [9]\right\},\ S_\varphi = \left\{\frac{j\pi}{40} : j \in [80]\right\},$$

we define

$$\hat{\mathcal{X}}_1 = \{r\,(\cos\varphi\, \boldsymbol{v}_1 + \sin\varphi\, \boldsymbol{v}_2) : r \in S_r, \varphi \in S_\varphi\}$$

and

$$\hat{\mathcal{X}}_2 = \{r\,(\cos\varphi\, \boldsymbol{v}_3 + \sin\varphi\, \boldsymbol{v}_4) : r \in S_r, \varphi \in S_\varphi\}.$$

Let $\hat{\mathcal{D}}_i$ be uniform distributed on $\hat{\mathcal{X}}_i \times \{i\}$ and $\hat{\mathcal{D}}$ be a mixture of $\hat{\mathcal{D}}_1$ and $\hat{\mathcal{D}}_2$. Let $\hat{\mathcal{X}} = \hat{\mathcal{X}}_1 \cup \hat{\mathcal{X}}_2$. The activation function σ is 4-bit quantized ReLU:

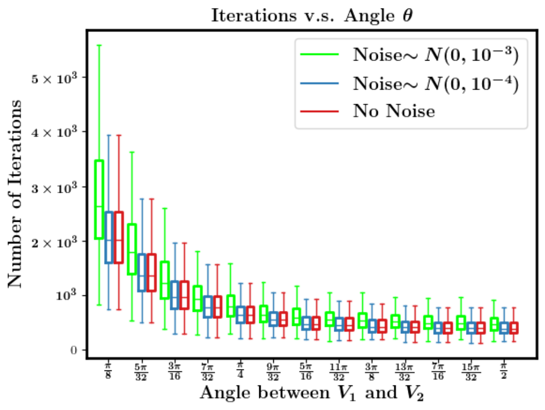
$$\sigma(x) = \begin{cases} 0 & \text{if} \quad x < 0, \\ \operatorname{ceil}(x) & \text{if} \quad 0 \le x < 15, \\ 15 & \text{if} \quad x \ge 15. \end{cases}$$

For simplicity, we take $k = 24$ and $v_{i,j} = \frac{1}{2}$ if $j - 12(i-1) \in [12]$ for $i \in [2]$ and $j \in [24]$ and 0 otherwise. Now, our neural network becomes

$$f_i = \frac{(-1)^{i-1}}{2}\left[\sum_{j=1}^{12}\sigma(h_j) - \sum_{j=1}^{12}\sigma(h_{j+12})\right]$$

where $h_j = \langle \boldsymbol{w}_j, \boldsymbol{x}\rangle$ and $\boldsymbol{x} \in \mathbb{R}^4$. The population loss is given by

$$l(\boldsymbol{W}) = \mathop{\mathbb{E}}_{\{\boldsymbol{x},y\}\sim\hat{\mathcal{D}}}[l(\boldsymbol{W}; \{\boldsymbol{x}, y\})] = \mathop{\mathbb{E}}_{\{\boldsymbol{x},y\}\sim\hat{\mathcal{D}}}[\max\{1 - f_i\}].$$



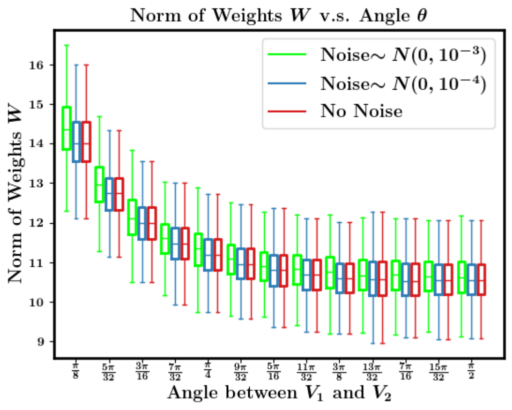


Figure 3.2: **Left**: Iterations to convergence v.s. θ, **Right**: Norm of weights v.s. θ.

We choose the ReLU STE (i.e., $g(x) = \max\{0, x\}$) and use the coarse gradient

$$\tilde{\nabla}_{\boldsymbol{W}} l(\boldsymbol{W}) = \mathop{\mathbb{E}}_{\{\boldsymbol{x},y\}\sim\hat{\mathcal{D}}}\left[\tilde{\nabla}_{\boldsymbol{W}} l\left(\boldsymbol{W}, \{\boldsymbol{x}, y\}\right)\right]$$
$$= \frac{1}{|\hat{\mathcal{X}}|}\left[\sum_{\boldsymbol{x}\in\hat{\mathcal{X}}_1} \tilde{\nabla}_{\boldsymbol{W}} l\left(\boldsymbol{W}; \{\boldsymbol{x}, 1\}\right) + \sum_{\boldsymbol{x}\in\hat{\mathcal{X}}_2} \tilde{\nabla}_{\boldsymbol{W}} l\left(\boldsymbol{W}; \{\boldsymbol{x}, 2\}\right)\right].$$

Taking learning rate $\eta = 1$, we have equation 3.5 becomes

$$\boldsymbol{W}^{t+1} = \boldsymbol{W}^t - \tilde{\nabla}_{\boldsymbol{W}} l\left(\boldsymbol{W}^t\right).$$

We find that the coarse gradient method converges to a global minimum with zero loss. As shown in box plots of Fig. 3.2, the *convergence still holds when the sub-spaces $\mathcal{V}_1$ and $\mathcal{V}_2$ form an acute angle*, and even when the data come from two levels of Gaussian noise perturbations of $\mathcal{V}_1$ and $\mathcal{V}_2$. The *convergence is faster* and with a smaller weight norm *when θ increases towards $\frac{\pi}{2}$ or $\mathcal{V}_2$ are orthogonal to each other.* This observation clearly supports the robustness of Theorem 1 beyond the regime of orthogonal classes.

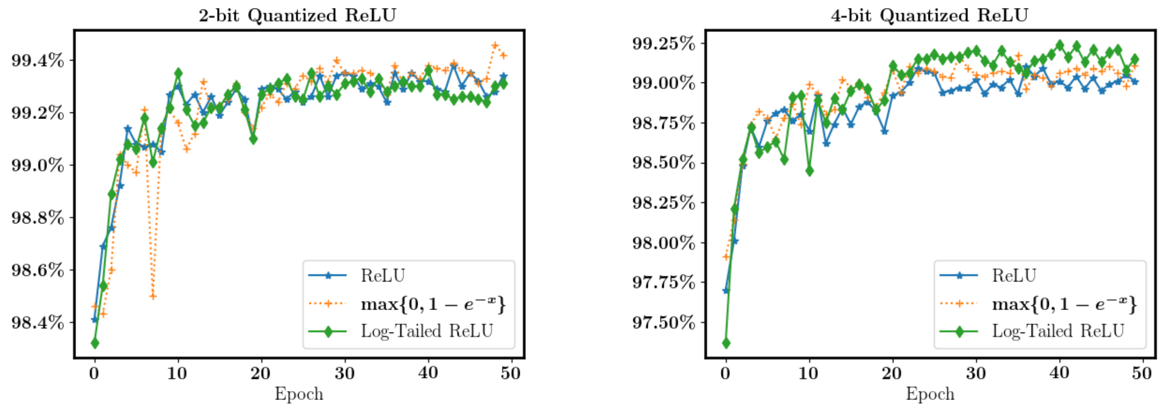


Figure 3.3: Validation accuracies in training LeNet-5 with quantized (2-bit and 4-bit) ReLU activation.

3.7.2 MNIST Experiments

Our theory works for a board range of STEs, while their empirical performances on deeper networks may differ. In this subsection, we compare the performances of the three type of STEs in Fig. 2.

As in [7], we resort to a modified batch normalization layer [23] and add it before each activation layer. As such, the inputs to quantized activation layers always follow unit Gaussian distribution. Then the scaling factor τ applied to the output of quantized activation layers can be pre-computed via k-means approach and get fixed during the whole training process. The optimizer we use to train quantized LeNet-5 is the (stochastic) coarse gradient method with momentum = 0.9. The batch size is 64, and learning rate is initialized to be 0.1 and then decays by a factor of 10 after every 20 epochs. The three backward pass substitutions g for the straight through estimator are (1) ReLU $g(x) = \max\{x, 0\}$, (2) reverse exponential $g(x) = \max\{0, q_b(1 - e^{-x/q_b})\}$ (3) log-tailed ReLU. The validation accuracy for each epoch is shown in Fig. 3.3. The validation accuracies at bit-widths 2 and 4 are listed in Table. 3.2. Our results show that these STEs all perform very well and give satisfactory accuracy. Specifically, reverse exponental and log-tailed STEs are comparable, both of which are

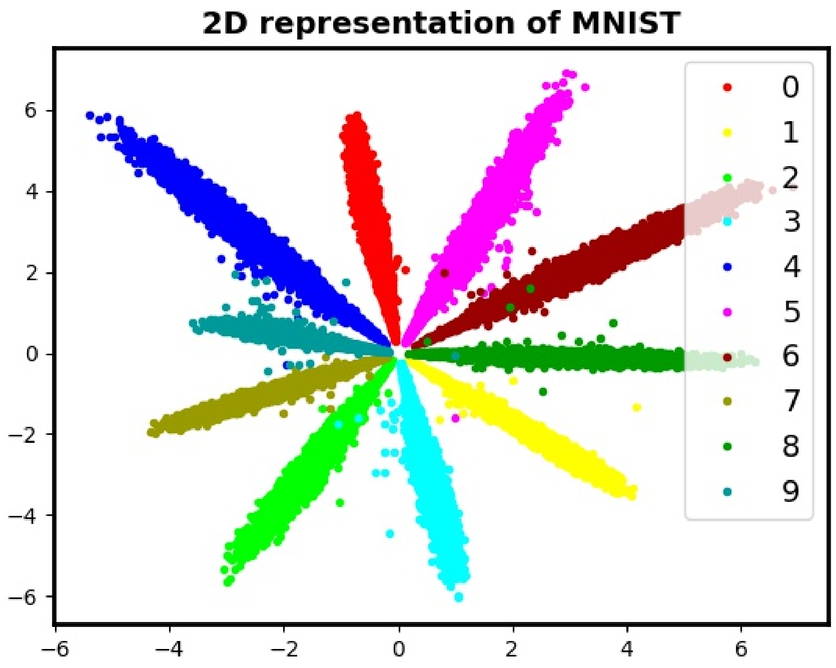


Figure 3.4: 2D projections of MNIST features from a trained convolutional neural network [42] with quantized activation function. The 10 classes are color coded, the feature points cluster near linearly independent subspaces.

slightly better than ReLU STE. In Fig. 3.4, we show 2D projections of MNIST features at the end of 100 epoch training of a 7 layer convolutional neural network [42] with quantized activation. The features are extracted from input to the last fully connected layer. The data points cluster near linearly independent subspaces. Together with subsection 8.1, we have numerical evidence that the linearly independent subspace data structure (working as an extension of subspace orthogonality) occurs for high level features in a deep network for a nearly perfect classification, rendering support to the realism of our theoretical study. Enlarging angles between linear subspaces can improve classification accuracy, see [45] for such an effort on MNIST and CIFAR-10 data sets via linear feature transform.

Table 3.2: Validation Accuracy (%) on MNIST with LeNet5.

$g(x)$	bit-width (b)	valid. accuracy
	32	99.45
ReLU	2	99.10
	4	99.38
reverse exp.	2	99.17
	4	99.46
log-tailed ReLU	2	99.24
	4	99.36

3.7.3 CIFAR-10 Experiments

In this experiment, we train VGG-11/ResNet-20 with 4-bit activation function on CIFAR-10 data set to numerically validate the boundedness assumption upon the ℓ_2-norm of weight. The optimizer is momentum SGD with no weight decay. We used initial learning rate = 0.1, with a decay factor of 0.1 at the 80-th and 140-th epoch.

we see from Fig. 3.5 that the ℓ_2 norm of weights is bounded during the training process. This figure also shows that the norm of weights is generally increasing in epochs which coincides with our theoretical finding shown in Lemma 3.5.

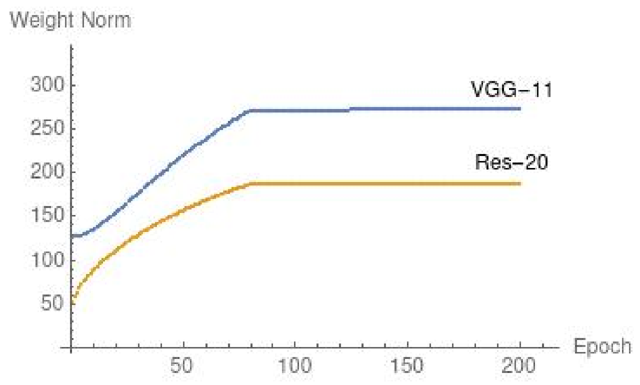


Figure 3.5: CIFAR-10 experiments for VGG-11 and ResNet-20: weight ℓ_2-norm vs epoch.

Chapter 4

Recurrence of Optimum for Training Weight and Activation Quantized Networks

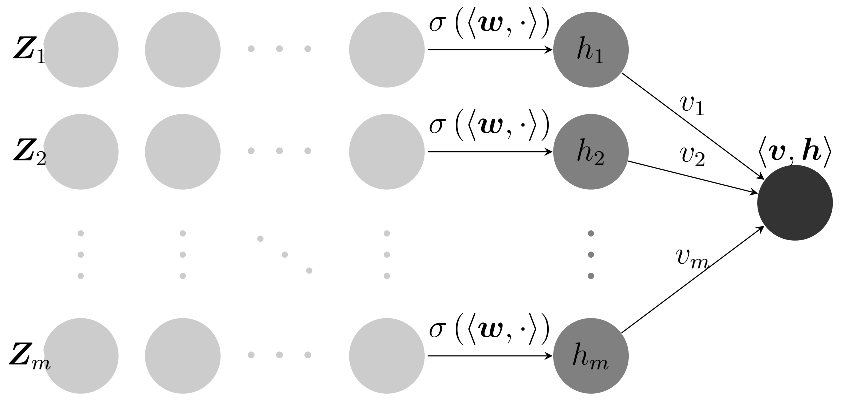
4.1 Preliminaries

4.1.1 Problem Setup

We consider a one-hidden-layer model that outputs the prediction for an input $\boldsymbol{Z} \in \mathbb{R}^{m\times n}$:

$$y(\boldsymbol{Z};\boldsymbol{w}) := \sum_{i=1}^{m} v_i \sigma\left(\boldsymbol{Z}_i^\top \boldsymbol{w}\right) = \boldsymbol{v}^\top \sigma\left(\boldsymbol{Z}\boldsymbol{w}\right) \tag{4.1}$$

where $\boldsymbol{Z}_i^\top$ denotes the i-th row vector of $\boldsymbol{Z}$; $\boldsymbol{w} \in \mathbb{R}^n$ is the trainable weights in the first linear layer, and $\boldsymbol{v} \in \mathbb{R}^m$ the weights in the second linear layer which are assumed to be known and fixed during the training process; the activation function $\sigma(x) = \mathbb{1}_{\{x>0\}}$ is *binary*, acting component-wise on the vector $\boldsymbol{Zw}$. The label is generated according to $y^*_{\boldsymbol{Z}} := y(\boldsymbol{Z}; \boldsymbol{w}^*)$ for some unknown teacher (real-valued) parameters $\boldsymbol{w}^* \in \mathbb{R}^n$.



$\boldsymbol{Z}_1$
$\boldsymbol{Z}_2$
$\boldsymbol{Z}_m$
$\sigma(\langle \boldsymbol{w}, \cdot \rangle)$
$\sigma(\langle \boldsymbol{w}, \cdot \rangle)$
$\sigma(\langle \boldsymbol{w}, \cdot \rangle)$
h_1
h_2
h_m
v_1
v_2
v_m
$\langle \boldsymbol{v}, \boldsymbol{h} \rangle$

Figure 4.1: One-hidden-layer neural network. The first linear layer resembles a convolutional layer with each $\boldsymbol{Z}_i$ being a patch of size n and $\boldsymbol{w}$ being the shared weights or filter. The second linear layer serves as the classifier.

We fit the described model with quantized weights $\boldsymbol{w} \in \mathcal{Q}$ and binary activation function $\sigma(x) = \mathbb{1}_{\{x>0\}}$ on the i.i.d. Gaussian data $\{(\boldsymbol{Z}, y^*_{\boldsymbol{Z}})\}_{\boldsymbol{Z}\sim\mathcal{N}(\boldsymbol{0},\mathbf{I})}$. In this chapter, we will focus on the cases of binary and ternary weights. In the binary case, every quantized weight in $\boldsymbol{w}$ is either α or $-\alpha$ for some universal real-valued constant $\alpha > 0$, or equivalently, $\mathcal{Q} = \mathbb{R}_+ \times \{\pm 1\}^n$; this setup of binary weights is widely adopted in the literature; for example, [34]. Similarly in the ternary case, we take $\mathcal{Q} = \mathbb{R}_+ \times \{0, \pm 1\}^n$; see [26, 49] for examples.

We use the squared loss to measure the discrepancy between the model output and label:

$$\begin{aligned} \ell(\boldsymbol{w}; \boldsymbol{Z}) &:= \frac{1}{2}\left(y(\boldsymbol{Z}; \boldsymbol{w}) - y^*_{\boldsymbol{Z}}\right)^2 \\ &= \frac{1}{2}\left(\boldsymbol{v}^\top \sigma(\boldsymbol{Zw}) - \boldsymbol{v}^\top \sigma(\boldsymbol{Zw}^*)\right)^2. \end{aligned} \tag{4.2}$$

We cast the learning task as the following population loss minimization problem:

$$\min_{\boldsymbol{w}\in\mathbb{R}^n} f(\boldsymbol{w}) := \mathbb{E}_{\boldsymbol{Z}\sim\mathcal{N}(\boldsymbol{0},\mathbf{I})}\left[\ell(\boldsymbol{w}; \boldsymbol{Z})\right] \quad \text{subject to} \quad \boldsymbol{w} \in \mathcal{Q} \tag{4.3}$$

where the sample loss function $\ell(\boldsymbol{w}; \boldsymbol{Z})$ is given in (4.2).

In the rest of the chapter, we study the convergence behavior of QUANT described below in Algorithm 1 for solving optimization problem (4.3), in which $\tilde{\nabla} f$ standards for an unusual gradient of f called coarse gradient [44], so as to side-step the vanished gradient issue. Since the loss function is scale-invariant, i.e., $\ell(\boldsymbol{Z}; \boldsymbol{w}) = \ell(\boldsymbol{Z}; \boldsymbol{w}/c)$ for any scalar $c > 0$, without loss of generality, we assume that $\|\boldsymbol{w}^*\| = 1$ is unit-normed.

Algorithm 1: QUANT algorithm for solving (4.3)

```
Input: number of iterations $T$, learning rate $\eta_t$, weight bits $b$;
Initialize: auxiliary real-valued weights $\boldsymbol{y}^0 \in \mathbb{R}^n$, step number $t = 1$;
while $t \leq T$ do
    $\boldsymbol{y}^t = \boldsymbol{y}^{t-1} - \eta_t \tilde{\nabla} f(\boldsymbol{w}^{t-1})$;
    $\boldsymbol{w}^t = \mathrm{proj}_{\mathcal{Q}}(\boldsymbol{y}^t)$ ;
    $t = t + 1$;
end
```

Throughout this chapter we assume the following on the learning rate $\eta_t > 0$:

1. $\sum_{t=1}^{\infty} \eta_t = \infty$.

2. η_t is upper bounded by some positive constant η.

4.1.2 Characterization of Optimal Solutions

To study the convergence of Algorithm 1, we first obtain the closed-form expression of the objective function for the optimization problem (4.2), which only depends on the angle between quantized weight vector $\boldsymbol{w}$ and the true weight vector $\boldsymbol{w}^*$. This helps us find the expression of global minimum to (1).

Lemma 4.1. *Let $\boldsymbol{w} \neq \boldsymbol{0}$ be nonzero vector.*

- *the training loss in (4.3) is given by*

$$f(\boldsymbol{w}) = \frac{\|\boldsymbol{v}\|^2}{2\pi} \arccos\left(\frac{\boldsymbol{w}^\top \boldsymbol{w}^*}{\|\boldsymbol{w}\|}\right)$$

- *For any* $\delta > 0$*,* $\boldsymbol{w} = \delta \cdot \mathrm{proj}_{\mathcal{Q}}(\boldsymbol{w}^*)$ *is a global optimum of quantization problem (4.3).*

The above result can be easily derived from Lemma 1 of [44], so we omit the proof. Lemma 4.1 states that the optimal quantized weights is just the projection of $\boldsymbol{w}^*$ onto $\mathcal{Q}$, i.e., the direct quantization of teacher parameters $\boldsymbol{w}^*$. Note that the projection/quantization may not be unique, we refer to $\mathrm{proj}_{\mathcal{Q}}(\boldsymbol{y})$ as any choice of the projection of $\boldsymbol{y}$ onto $\mathcal{Q}$.

4.1.3 Coarse Gradient

In this part, we specify the coarse gradient $\tilde{\nabla} f(\boldsymbol{w})$ in Algorithm 1. The standard back-propagation gives the gradient of $\ell(\boldsymbol{w}; \boldsymbol{Z})$ w.r.t. $\boldsymbol{w}$ by

$$\nabla_{\boldsymbol{w}} \ell(\boldsymbol{w}; \boldsymbol{Z}) = \boldsymbol{Z}^\top \left(\sigma'(\boldsymbol{Z}\boldsymbol{w}) \odot \boldsymbol{v}\right) \ell(\boldsymbol{w}; \boldsymbol{Z}).$$

Note that σ' is zero a.e., which makes $\nabla_{\boldsymbol{w}} \ell(\boldsymbol{w}; \boldsymbol{Z})$ inapplicable to the training. The sample coarse gradient w.r.t. $\boldsymbol{w}$ associated with the sample $(\boldsymbol{Z}, y^*_{\boldsymbol{Z}})$ is given by replacing σ' with a surrogate derivative, known as straight-through estimator (STE) [3, 44]. Here we consider the derivative of ReLU function $\mu(x) = \max\{x, 0\}$ which is a widely used STE for quantization, namely, we modify the original gradient $\nabla_{\boldsymbol{w}} \ell(\boldsymbol{w}; \boldsymbol{Z})$ as follows:

$$\tilde{\nabla}_{\boldsymbol{w}} \ell(\boldsymbol{w}; \boldsymbol{Z}) = \boldsymbol{Z}^\top \left(\mu'(\boldsymbol{Z}\boldsymbol{w}) \odot \boldsymbol{v}\right) \ell(\boldsymbol{w}; \boldsymbol{Z}).$$

The coarse gradient induced by ReLU STE μ' is just the expectation of $\tilde{\nabla}_{\boldsymbol{w}} \ell(\boldsymbol{w}; \boldsymbol{Z})$ over $\boldsymbol{Z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. We evaluate the coarse gradient $\tilde{\nabla} f(\boldsymbol{w})$ used in Algorithm 1:

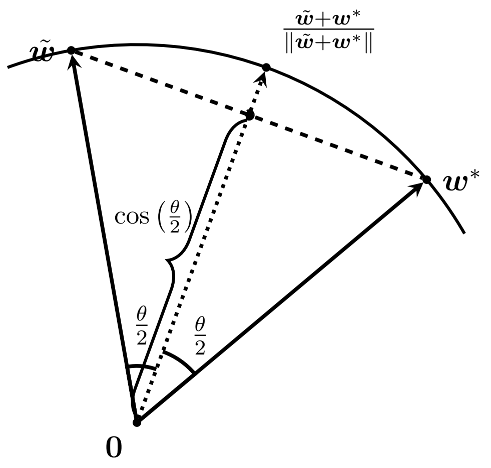
Lemma 4.2. *The expected coarse gradient of $\ell(\boldsymbol{w};\boldsymbol{Z})$ w.r.t. $\boldsymbol{w}$ is*

$$\begin{aligned}\tilde{\nabla} f(\boldsymbol{w}) :=& \mathbb{E}_{\boldsymbol{Z}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}[\tilde{\nabla}_{\boldsymbol{w}}\ell(\boldsymbol{w};\boldsymbol{Z})] \\ =& \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}\left(\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}-\boldsymbol{w}^*\right).\end{aligned} \tag{4.4}$$

Proof or Lemma 4.2. [44] gives

$$\tilde{\nabla} f(\boldsymbol{w}) = \frac{\|\boldsymbol{v}^*\|^2}{\sqrt{2\pi}}\left(\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}-\cos\left(\frac{\theta}{2}\right)\frac{\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}+\boldsymbol{w}^*}{\left\|\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}+\boldsymbol{w}^*\right\|}\right).$$

Let $\tilde{\boldsymbol{w}} = \frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}$, we can easily see from Fig. 4.2 that the coarse gradient can be further



$\tilde{\boldsymbol{w}}$
$\frac{\tilde{\boldsymbol{w}}+\boldsymbol{w}^*}{\|\tilde{\boldsymbol{w}}+\boldsymbol{w}^*\|}$
$\boldsymbol{w}^*$
$\cos\left(\frac{\theta}{2}\right)$
$\frac{\theta}{2}$
$\frac{\theta}{2}$
$\mathbf{0}$

Figure 4.2: 2-dim section of $\mathbb{R}^n$ spanned by $\tilde{\boldsymbol{w}}$ and $\boldsymbol{w}^*$

simplified as (4.4) $\square$

4.1.4 Weight Quantization Step

The following two lemmas give the closed-form formulas of the projection/quantization $\text{proj}_{\mathcal{Q}}(\cdot)$ in Algorithm 1 in the binary and ternary cases, respectively.

Lemma 4.3 (Binary Case)**.** *For any non-zero $\boldsymbol{y} \in \mathbb{R}^n$, the projection of $\boldsymbol{y}$ onto $\mathcal{Q}$ =*

$\mathbb{R}_+ \times \{\pm 1\}^n$ *is*

$$\mathrm{proj}_{\mathcal{Q}}(\boldsymbol{y}) = \frac{\|\boldsymbol{y}\|_1}{n} \widetilde{\mathrm{sign}}(\boldsymbol{y}),$$

where the sign function acts element-wise

$$\widetilde{\mathrm{sign}}(\boldsymbol{y})_i = \begin{cases} 1 & \text{if } y_i \geq 0 \\ -1 & \text{if } y_i < 0. \end{cases}$$

The above lemma is due to [34]. In the ternary case, [49] gives the following result:

Lemma 4.4 (Ternary Case)**.** *For any non-zero $\boldsymbol{y} \in \mathbb{R}^n$, the projection of $\boldsymbol{y}$ on $\mathcal{Q} = \mathbb{R}_+ \times \{0, \pm 1\}^n$ is*

$$\mathrm{proj}_{\mathcal{Q}}(\boldsymbol{y}) = \frac{\left\|\boldsymbol{y}_{[j^*]}\right\|_1}{j^*} \mathrm{sign}\left(\boldsymbol{y}_{[j^*]}\right)$$

where $j^* = \arg\max_{1 \leq j \leq n} \frac{\left\|\boldsymbol{y}_{[j]}\right\|_1^2}{j}$, and $\boldsymbol{y}_{[j]} \in \mathbb{R}^n$ extracts the first j largest entries in magnitude of $\boldsymbol{y}$ and enforces 0 elsewhere. Here,

$$\mathrm{sign}(\boldsymbol{y})_i = \begin{cases} 1 & \text{if } y_i > 0 \\ 0 & \text{if } y_i = 0 \\ -1 & \text{if } y_i < 0. \end{cases}$$

4.2 Binary Weight

The binary case is rather simple. We show that part of the coordinates is stable while others have oscillating sign. We further prove that the set of oscillating coordinates is not empty as long as $\boldsymbol{w}^* \notin \mathcal{Q} = \mathbb{R}_+ \times \{\pm 1\}^n$ is not quantized. In view of Lemmas 4.1 and 4.3, we have that the normalized optimum of (4.3) is $\frac{1}{\sqrt{n}} \widetilde{\mathrm{sign}}(\boldsymbol{w}^*)$. The Lemma below shows that some coordinates of $\boldsymbol{w}^t$ generated by Algorithm 1 have oscillating signs.

Lemma 4.5. *Let $\boldsymbol{w}^t$ be any infinite sequence generated by Algorithm 1. If $|w_j^*| < \frac{1}{\sqrt{n}}$, then there exist infinitely many t_1 and t_2 such that $w_j^{t_1} = \frac{1}{\sqrt{n}}$ and $w_j^{t_2} = -\frac{1}{\sqrt{n}}$.*

Proof of Lemma 4.5. For notational simplicity, since $\left\|w_j^*\right\| < \frac{1}{\sqrt{n}}$, we have

$$\alpha := \frac{1}{\sqrt{n}} - w_j^* > 0 \text{ and } \beta := \frac{1}{\sqrt{n}} + w_j^* > 0.$$

Using Lemma 4.3 in Algorithm 1, we see that

$$\begin{aligned} y_j^{t+1} &= y_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}(w_j^* - w_j^t) \\ &= y_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}\left(w_j^* - \frac{1}{\sqrt{n}}\widetilde{\operatorname{sign}}\left(y_j^t\right)\right), \end{aligned}$$

and thus

$$y_j^{t+1} = \begin{cases} y_j^t - \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}\alpha & \text{if } y_j^t \geq 0 \\ y_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}\beta & \text{if } y_j^t < 0 \end{cases}$$

Since y_j^t is bounded for each fixed $t \geq 0$ and $j \in [n]$, our desired result follows from our assumptions on learning rate η_t. □

The above proposition clearly implies that $\boldsymbol{w}^t$ does not converge, as long as $\boldsymbol{w}^* \notin \mathcal{Q}$.

Corollary 4.1. *If $\boldsymbol{w}^* \notin \mathcal{Q}$, then any sequence $\{\boldsymbol{w}^t\}$ generated by Algorithm 1 does not converge.*

Proof of Corollary 4.1. Since $\boldsymbol{w}^* \notin \tilde{\mathcal{Q}}_1^n$, we know there must exist some $j \in [n]$ such that $|w_j^*| < \frac{1}{\sqrt{n}}$ and Proposition 4.5 gives our desired result. □

Since Algorithm 1 does not have a limit unless the weights in the network are already quantized, we ask a natural question: Can we guarantee the optimum to be visited infinitely many times? The general answer is no. We have the following example *demonstrating that the optimum may never be achieved.* We refer the proof of the following example to the appendix.

Example 4.1. *Let* $\boldsymbol{w}^* = \left(\frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{2}\sqrt{\frac{11}{3}}\right)$ *so that the best the optimum* $\widetilde{\text{proj}}_{\mathcal{Q}}\boldsymbol{w}^* = \left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$. *Let* $\eta_t = \eta$, $\lambda = \frac{\eta\|\boldsymbol{v}\|^2}{6\sqrt{2\pi}}$ *and*

$$\begin{cases} y_1^0 \in (-\lambda, 0) \\ y_2^0 \in (0, \lambda) \\ y_3^0 \in (\lambda, 2\lambda) \\ y_4^0 \in (0, \infty) \end{cases}$$

the sequence $\{\boldsymbol{w}^t\}$ *generated by Algorithm 1 with initialization* $\boldsymbol{y}^0$ *satisfies* $\boldsymbol{w}^{t+3} = \boldsymbol{w}^t$ *and* $\boldsymbol{w}^t \neq \widetilde{\text{proj}}_{\mathcal{Q}}\boldsymbol{w}^*$ *for all* t.

Proof of Example 4.1. In order to show the periodicity, it suffices to show $w_j^{t+3} = w_j^t$. Note that $\tilde{\partial}_{w_4} f(\boldsymbol{w}) < 0$ we have $y_4^t > 0$ for all t since $w_4^0 > 0$. It follows that $w_4^t = w_4^0 = \frac{1}{2}$. Next, we would like to show the periodicity of w_j^t for $j \in [3]$. Note that

$$y_j^{t+1} = \begin{cases} y_j^t + \dfrac{\eta\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}\left(w_j^* + \dfrac{1}{2}\right) & \text{if } y_j^t < 0 \\ y_j^t + \dfrac{\eta\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}\left(-w_j^* + \dfrac{1}{2}\right) & \text{if } y_j^t \geq 0 \end{cases}$$

we choose $\boldsymbol{w}_j^* = \frac{1}{6}$ so that with

$$\lambda = \frac{\eta\|\boldsymbol{v}\|^2}{6\sqrt{2\pi}}$$

we have

$$y_j^{t+1} = \begin{cases} y_j^t + 2\lambda & \text{if } y_j^t < 0 \\ y_j^t - \lambda & \text{if } y_j^t \geq 0 \end{cases}$$

Hence, we have

$$\boldsymbol{w}^t = \begin{cases} \left(-\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right) & \text{if } t \equiv 0 (\text{mod } 3) \\ \left(\frac{1}{2}, -\frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right) & \text{if } t \equiv 1 (\text{mod } 3) \\ \left(\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, \frac{1}{2}\right) & \text{if } t \equiv 2 (\text{mod } 3) \end{cases}$$

□

In the following, we give a sufficient condition for the optimum to be recurrent. The condition requires $\boldsymbol{w}^*$ to be close to $\mathcal{Q}$.

Theorem 4.1. *If the optimum* $\hat{\boldsymbol{w}} := \widetilde{\text{proj}}_{\mathcal{Q}}(\boldsymbol{w}^*) = \frac{1}{\sqrt{n}}\widetilde{\text{sign}}(\boldsymbol{w}^*)$ *of (4.3) satisfies* $0 < \sum_{|w_j^*| < \frac{1}{\sqrt{n}}} |w_j^* - \hat{w}_j| < \frac{2}{\sqrt{n}}$ *then there exist infinitely many* t *values for any sequence* $\{\boldsymbol{w}^t\}$ *generated by Algorithm 1 such that* $\boldsymbol{w}^t = \widetilde{\text{proj}}_{\mathcal{Q}}(\boldsymbol{w}^*)$.

Proof of Theorem 4.1. Without loss of generality, we can assume $w_j^* \geq 0$ for all $j \in [n]$ so that $\hat{w}_j = \frac{1}{\sqrt{n}}$ for all j.

Firstly, if $w_j^* > \frac{1}{\sqrt{n}}$, we know

$$y_j^{t+1} = y_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left(w_j^* - w_j^t\right) \geq w_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left(w_j^* - \frac{1}{\sqrt{n}}\right),$$

so that

$$y_j^t \geq y_j^0 + \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left(\sum_{s=0}^{t-1} \eta_s\right) \left(w_j^* - \frac{1}{\sqrt{n}}\right)$$

where the right hand side goes to infinity and thus $w_j^t = \hat{w}_j$ for all but finitely many t values.

Secondly, if $w_j^* = \frac{1}{\sqrt{n}}$, we know when $w_j^t < 0$:

$$y_j^{t+1} = y_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left(w_j^* - w_j^t\right) = y_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \frac{2}{\sqrt{n}}$$

holds so that there must exist some t such that $y_j^t > 0$. Once $y_j^t > 0$ we have $w_j^* = w_j^t$ so that $y_j^{t+1} = y_j^t$ and hence $w_j^t = \hat{w}_j$ for all but finitely many t values.

Third, if $w_j^* < \frac{1}{\sqrt{n}}$, we have $y_j^t \cdot \tilde{\partial}_j f(\boldsymbol{w}^t) > 0$ so that y_j^t is increasing when $y_j^t < 0$ and decreasing when $y_k^t > 0$. This tells us y_j^t is bounded uniformly in t. Furthermore,

$$y_j^t = y_j^0 + \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}\Bigg[\left(\sum_{s=0}^{t-1} \mathbb{1}_{\left\{w_j^s>0\right\}}\eta_s\right)\left(w_j^* - \frac{1}{\sqrt{n}}\right) + \left(\sum_{s=0}^{t-1} \mathbb{1}_{\left\{w_j^s<0\right\}}\eta_s\right)\left(w_j^* + \frac{1}{\sqrt{n}}\right)\Bigg].$$

For notation simplicity, we let

$$\alpha_j = \frac{1}{\sqrt{n}} - w_j^* > 0 \quad\text{and}\quad \beta_j = w_j^* + \frac{1}{\sqrt{n}} > 0,$$

$$a_j^t = \frac{1}{t}\sum_{s=0}^{t-1} \mathbb{1}_{\left\{w_j^s>0\right\}}\eta_s \quad\text{and}\quad b_j^t = \frac{1}{t}\sum_{s=0}^{t-1} \mathbb{1}_{\left\{w_j^s<0\right\}}\eta_s.$$

Now, we have

$$\frac{y_j^t - y_j^0}{t} = \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}\left(-\alpha_j a_j^t + \beta_j b_j^t\right).$$

Since y_j^t is bounded for all $w_j^* < \frac{1}{\sqrt{n}}$, we let $t \to \infty$ so that left hand side vanishes and

$$\lim_{t\to\infty} \frac{b_j^t}{a_j^t + b_j^t} = \frac{\alpha_j}{\alpha_j + \beta_j}.$$

By assumption, we have

$$\lim_{t\to\infty} \sum_{j=1}^{n} \frac{b_j^t}{a_j^t + b_j^t} = \sum_{j=1}^{n} \frac{\alpha_j}{\alpha_j + \beta_j} < 1.$$

Hence, we know

$$\lim_{t\to\infty}\sum_{s=0}^{t-1}\mathbb{1}_{\{\boldsymbol{w}^s=\hat{\boldsymbol{w}}^*\}}\eta_s \geq \lim_{t\to\infty}\left[\left(1-\sum_{j=1}^{n}\frac{b_j^t}{a_j^t+b_j^t}\right)\sum_{s=0}^{t-1}\eta_s\right]=\infty,$$

where we used the assumption $\sum_{t=0}^{\infty}\eta_t=\infty$. Now, the desired result follows. □

4.3 Ternary Weight

The proof of the ternary case follows the following steps. Our first step shows the sequence $\boldsymbol{y}^t$ generated by Algorithm 1 is bounded away from the origin for all but finitely many t values. Then, our second step shows each coordinate of $\boldsymbol{y}^t$ is of the same sign of $\boldsymbol{w}^*$ for all but finitely many t values. This forces $\boldsymbol{y}^t$ to stay in the same orthant to which $\boldsymbol{w}^*$ belongs. As a matter of fact, an n-dimensional space has in total 2^n orthants, which means $\boldsymbol{y}^t$ can only stay in a small region near $\boldsymbol{w}^*$. After that, our third step furthermore cuts the orthant into $n!$ congruent cones and argue $\boldsymbol{y}^t$ must stay in the same cone where $\boldsymbol{w}^*$ is for all but finitely many t values. In the last step, we prove the ternary case of Theorem 4.2, which asserts that as long as the underlying true parameter $\boldsymbol{w}^*$ is close to quantized state $\mathcal{Q}=\mathbb{R}_+\times\{0,\pm1\}^n$, i.e., any vertex of the cone it belongs to, the optimum is guaranteed to be recurrent.

The first result shows that $\boldsymbol{w}^t$ generated by Algorithm 1 is generally divergent, and it converges only when the true parameters $\boldsymbol{w}^*\in\mathcal{Q}=\mathbb{R}_+\times\{0,\pm1\}^n$.

Proposition 4.1 (Ternary Case)**.** *Let $\{\boldsymbol{w}^t\}$ be any sequence generated by Algorithm 1. If $\boldsymbol{w}^*\notin\mathcal{Q}=\mathbb{R}_+\times\{0,\pm1\}^n$, then $\{\boldsymbol{w}^t\}$ is not a convergent sequence.*

Proof of Proposition 4.1. We prove by contradiction. Observe that $\mathcal{Q}\cap\mathcal{S}^{n-1}$ is a finite set, we know $\boldsymbol{w}^t$ converges to $\boldsymbol{w}^\infty$ is equivalent to $\boldsymbol{w}^t=\boldsymbol{w}^\infty$ for all but finitely many t values. Assume $\boldsymbol{w}^t=\boldsymbol{w}^\infty$ for all but finitely many t values, we know there exists some $T\geq 0$ such that $\boldsymbol{w}^t = \boldsymbol{w}^\infty$ for all $t \geq T$. Thus,

$$\begin{aligned}
\boldsymbol{y}^{T+t} &= \boldsymbol{y}^T - \sum_{s=0}^{t-1} \eta_{T+s} \tilde{\nabla} f\left(\boldsymbol{w}^{T+s}\right) \\
&= \boldsymbol{y}^T - \left(\sum_{s=0}^{t-1} \eta_{T+s}\right) \tilde{\nabla} f\left(\boldsymbol{w}^\infty\right) \\
&= \boldsymbol{y}^t + \left(\sum_{s=0}^{t-1} \eta_{T+s}\right) \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left(\boldsymbol{w}^* - \boldsymbol{w}^\infty\right).
\end{aligned}$$

Now, we have

$$\left\langle \boldsymbol{y}^{T+t}, \boldsymbol{w}^\infty \right\rangle = \left\langle \boldsymbol{y}^T, \boldsymbol{w}^\infty \right\rangle + \left(\sum_{s=0}^{t-1} \eta_{T+s}\right) \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left\langle \boldsymbol{w}^* - \boldsymbol{w}^\infty, \boldsymbol{w}^\infty \right\rangle$$

where

$$\left\langle \boldsymbol{w}^* - \boldsymbol{w}^\infty, \boldsymbol{w}^\infty \right\rangle = \left\langle \boldsymbol{w}^*, \boldsymbol{w}^\infty \right\rangle - 1 < 0.$$

Note that $\sum_{s=0}^\infty \eta_{T+s} = \infty$, there exists some $T_1(T)$, such that for all $t > T_1(T)$

$$\left\langle \boldsymbol{y}^t, \boldsymbol{w}^\infty \right\rangle < 0.$$

This contradicts Lemma 4.4 and our desired result follows. □

In what follows, we detail the proof of convergence behavior of Algorithm 1.

Our first step is to rule out an exceptional case that the direction of $\boldsymbol{y}^t$ changes significantly in only one iteration. As shown in Lemma 4.2, the coarse gradient is bounded by a constant depending only on the fixed weight vector $\boldsymbol{v}$. So it suffices to show that $\|\boldsymbol{y}^t\|$ is bounded away from zero for all but finitely many t values.

Lemma 4.6. *Let $\{\boldsymbol{y}^t\}$ be any auxiliary sequence generated by Algorithm 1. If $\boldsymbol{w}^* \notin \mathcal{Q}$, then $\|\boldsymbol{y}^t\|_1$ converges to infinity as t increases.*

Proof of Lemma 4.6. $\mathcal{Q} \cap \mathcal{S}^{n-1}$ is a compact set because it is finite. Also, since $\mathcal{Q}$ is symmetric, $\boldsymbol{w}^* \notin \mathcal{Q}$ also implies $-\boldsymbol{w}^* \notin \mathcal{Q}$. It follows that

$$\alpha := \inf_{\boldsymbol{w} \in \mathcal{Q} \cap \mathcal{S}^{n-1}} \theta\left(\boldsymbol{w}^*, \boldsymbol{w}\right) \in (0, \pi).$$

Hence, for any $\boldsymbol{w} \in \mathcal{Q} \cap \mathcal{S}^{n-1}$ we have

$$\left\langle -\tilde{\nabla} f(\boldsymbol{w}), \boldsymbol{w}^* \right\rangle = \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left\langle \boldsymbol{w}^* - \boldsymbol{w}, \boldsymbol{w}^* \right\rangle \geq \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left(1 - \cos\alpha\right).$$

Now, we know

$$\begin{aligned} \left\langle \boldsymbol{y}^T, \boldsymbol{w}^* \right\rangle &= \left\langle \boldsymbol{y}^0, \boldsymbol{w}^* \right\rangle + \sum_{t=0}^{T-1} \eta_t \left\langle -\tilde{\nabla} f\left(\boldsymbol{w}^t\right), \boldsymbol{w}^* \right\rangle \\ &\geq \left\langle \boldsymbol{y}^0, \boldsymbol{w}^* \right\rangle + \left(\sum_{t=0}^{T-1} \eta_t \right) \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \cdot \left(1 - \cos\alpha\right). \end{aligned}$$

Let $T \to \infty$, we see that $\lim_{t\to\infty} \|\boldsymbol{y}^t\| = \infty$ which is equivalent to $\lim_{t\to\infty} \|\boldsymbol{y}^t\|_1 = \infty$. $\square$

Lemma 4.6 shows that for any positive constant $c > 0$, we have $\|\boldsymbol{y}^t\|_1 > c$ for all but finitely many t values.

Since Lemma 4.6 guarantees that the direction of $\boldsymbol{y}^t$ will not change significantly, we cut down the region that $\boldsymbol{y}^t$ can belong to in two steps. To describe our first cut down, we need the following definition to make our statement precise.

Definition 4.1. *For any $\boldsymbol{x} \in \mathbb{R}^n$, we define the orthant of $\boldsymbol{x}$ as*

$$\boldsymbol{O}(\boldsymbol{x}) := \left\{ \boldsymbol{y} \in \mathbb{R}^n : \operatorname{sign}\left(\boldsymbol{y}\right) = \operatorname{sign}\left(\boldsymbol{x}\right) \right\},$$

where $\operatorname{sign}(\cdot)$ *acts coordinate-wise. Furthermore, we say $\boldsymbol{O}(\boldsymbol{x})$ is regular if any coordinate of $\boldsymbol{x}$ is not zero.*

We state some basic properties of the defined orthant.

Proposition 4.2. *For any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, the following statements are true:*

1. *Either $\boldsymbol{O}(\boldsymbol{x}) = \boldsymbol{O}(\boldsymbol{y})$ or $\boldsymbol{O}(\boldsymbol{x}) \cap \boldsymbol{O}(\boldsymbol{y}) = \emptyset$.*

2. *$\boldsymbol{x} \in \boldsymbol{O}(\boldsymbol{x})$.*

3. *$\cup_{\boldsymbol{x}\in\mathbb{R}^n} \boldsymbol{O}(\boldsymbol{x}) = \mathbb{R}^n$.*

4. *There are in total 3^n orthants.*

5. *There are in total 2^n regular orthants.*

Lemma 4.7. *Let $\boldsymbol{w} = \operatorname{proj}_{\mathcal{Q}}(\boldsymbol{y})$, then $|y_j| < \frac{1}{5n} \|\boldsymbol{y}\|_1$ implies $w_j = 0$.*

Proof of Lemma 4.7. Without loss of generality, we assume $y_i \geq 0$ for all $i \in [n]$ and $y_j < \frac{1}{5n} \|\boldsymbol{y}\|_1$ for a fixed $j \in [n]$. Let $\delta = \frac{1}{5n} \|\boldsymbol{y}\|_1$ and

$$j_\delta := | \{ i \in [n] : |y_i| \geq \delta \} |$$

we know $j_\delta \geq 1$ by the principle of drawer. Now, with

$$j^* = \arg\max \frac{\left\|\boldsymbol{y}_{[j]}\right\|_1^2}{j}$$

for any $1 \leq k \leq n - j_\delta$

$$\begin{aligned}
&\frac{\left\|\boldsymbol{y}_{[j^*]}\right\|_1^2}{j^*} - \frac{\left\|\boldsymbol{y}_{[j_\delta+k]}\right\|_1^2}{j_\delta + k} \\
\geq &\frac{\left\|\boldsymbol{y}_{[j_\delta]}\right\|_1^2}{j_\delta} - \frac{\left\|\boldsymbol{y}_{[j_\delta+k]}\right\|_1^2}{j_\delta + k} \\
= &\frac{(j_\delta + k)\left\|\boldsymbol{y}_{[j_\delta]}\right\|_1^2 - j_\delta \left\|\boldsymbol{y}_{[j_\delta+k]}\right\|_1^2}{j_\delta (j_\delta + k)},
\end{aligned}$$

where the numerator is

$$\begin{aligned}
&k\left\|\boldsymbol{y}_{[j_\delta]}\right\|_1^2 - j_\delta \left(\left\|\boldsymbol{y}_{[j_\delta+k]}\right\|_1^2 - \left\|\boldsymbol{y}_{[j_\delta]}\right\|_1^2 \right) \\
\geq &k\left[\left\|\boldsymbol{y}_{[j_\delta]}\right\|_1^2 - j_\delta \delta \left(\left\|\boldsymbol{y}_{[j_\delta+k]}\right\|_1 + \left\|\boldsymbol{y}_{[j_\delta]}\right\|_1 \right) \right].
\end{aligned}$$

With $\tau = \frac{\left\|\boldsymbol{y}_{[j_\delta+k]}\right\|_1}{n\delta}$, we have

$$\begin{aligned}
& k\left[\left\|\boldsymbol{y}_{[j_\delta]}\right\|_1^2 - j_\delta\delta\left(\left\|\boldsymbol{y}_{[j_\delta+k]}\right\|_1 + \left\|\boldsymbol{y}_{[j_\delta]}\right\|_1\right)\right] \\
\geq & k\left[\left(\left\|\boldsymbol{y}_{[j_\delta+k]}\right\|_1 - k\delta\right)^2 - 2n\delta\left\|\boldsymbol{y}_{[j_\delta+k]}\right\|_1\right] \\
= & k\left(n\delta\right)^2\left(\tau^2 - 4\tau + 1\right).
\end{aligned}$$

Note that

$$\tau = \frac{\left\|\boldsymbol{y}_{[j_\delta+k]}\right\|_1}{n\delta} \geq \frac{\|\boldsymbol{y}\|_1 - n\delta}{n\delta} \geq 4,$$

we conclude that

$$\frac{\left\|\boldsymbol{y}_{[j^*]}\right\|_1^2}{j^*} > \frac{\left\|\boldsymbol{y}_{[j_\delta+k]}\right\|_1^2}{j_\delta + k}$$

and hence $j^* \leq j_\delta$. Now, Lemma 4.4 gives $w_j = 0$. $\square$

Lemma 4.8. *Let $\{\boldsymbol{w}^t\}$ and $\{\boldsymbol{y}^t\}$ be the sequence and the auxiliary sequence generated by algorithm 1. Assume $\boldsymbol{w}^* \notin \mathcal{Q}$, the following statements hold.*

- *If $w_j^* = 0$, then y_j^t is bounded and $w_j^t = 0$ for all but finitely many t values.*
- *If $w_j^* \neq 0$, then $\operatorname{sign}\left(y_j^t\right) = \operatorname{sign}\left(w_j^*\right)$ for all but finitely many t values.*

Proof of Lemma 4.8. On the one hand, we consider the case $w_j^* = 0$, so that

$$y_j^{t+1} = y_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}\left(w_j^* - w_j^t\right) = y_j^t - \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} w_j^t.$$

Note that Lemma 4.4 shows y_j^t and w_j^t are of the same sign if $w_j^t \neq 0$, we know y_j^t is bounded by $C_j := \max\left\{|y_j^0|, \eta\frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}\right\}$. Moreover Lemma 4.6 shows $\|\boldsymbol{y}^t\|_1 > 5nC_j$ for all but finitely many t values. Finally, we see from Lemma 4.7 that $w_j^t = 0$ for all but finitely many t values.

On the other hand, consider the case $w_j^* \neq 0$. Without loss of generality, we can assume

$w_j^* > 0$. Note that whenever $y_j^t \leq 0$, we also have $w_j^t \leq 0$ so that

$$y_j^{t+1} = y_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left(w_j^* - w_j^t\right) \geq y_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} w_j^*.$$

From the above inequality, we see that y_j^t is increasing where the increment is bounded from below by $\eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} w_j^* > 0$ where $\sum \eta_t = \infty$, so that there must exist some $T_j > 0$ such that $y_j^{T_j} > 0$. With Lemma 4.6, we can without loss of generality assume that $\|\boldsymbol{y}^t\|_1 \geq 5n\eta \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}$ for all $t \geq T_j$. For ease of notation, we let $\delta = \eta \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}$ so that $\|\boldsymbol{y}^t\|_1 \geq 5n\delta$ for all $t \geq T_j$. We shall next prove that $y_j^t \geq 0$ for all $t \geq T_j$. We prove by induction, assume $y_j^t > 0$ for some $t > T_j$ and show $y_j^{t+1} > 0$.

1. If $y_j^t > \delta$,
$$y_j^{t+1} = y_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left(w_j^* - w_j^t\right) \geq y_j^t - \delta > 0.$$

2. If $0 < y_j^t \leq \delta$, since $\|\boldsymbol{y}^t\|_1 \geq 5n\delta$, Lemma 4.7 shows $w_j^t = 0$ so that
$$y_j^{t+1} = y_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left(w_j^* - w_j^t\right) = y_j^t + \frac{\eta_t \delta}{\eta} w_j^* > y_j^t > 0.$$

Combining the above two cases, we get our desired result. □

Lemma 4.9. *Let $\{\boldsymbol{y}^t\}$ be any auxiliary sequence generated by Algorithm 1. If $\boldsymbol{w}^* \notin \mathcal{Q}^n$, then any subsequential limit of $\tilde{\boldsymbol{y}}^t := \frac{\boldsymbol{y}^t}{\|\boldsymbol{y}^t\|}$ belongs to the closure of $\boldsymbol{O}(\boldsymbol{w}^*)$. Furthermore, if $\boldsymbol{O}(\boldsymbol{w}^*)$ is regular, then $\boldsymbol{y}^t$ lies in $\boldsymbol{O}(\boldsymbol{w}^*)$ for all but finitely many t values.*

Proof of Lemma 4.9. By Lemma 4.8, we see that $\operatorname{sign}\left(y_j^t\right) = \operatorname{sign}\left(w_j^*\right)$ for all $\boldsymbol{w}_j^* \neq 0$. We only need to prove $w_j^* = 0$ implies $\lim_{t\to\infty} \tilde{y}_j^t = 0$. Indeed, by Lemma 4.8, we know that y_j^t is bounded by C_j while Lemma 4.6 tells us $\|\boldsymbol{y}^t\|$ goes to infinity. Thus, $\lim_{t\to\infty} \tilde{y}_j^t = \frac{y_j^t}{\|\boldsymbol{y}\|} = 0$. □

In our previous step, we have partitioned $\mathbb{R}^n$ into orthants and showed that $\boldsymbol{y}^t$ enter into a small neighborhood of the orthant where $\boldsymbol{w}^*$ stays. Now, we prove a stronger result based on the conclusion of our previous step. We would like to cut each orthant into several congruent cones which we shall define later and argue $\boldsymbol{y}^t$ will move and stay in close neighborhood of the cone where $\boldsymbol{w}^*$ stays. This step makes a stronger statement because we manage to shrink the size of the region where $\boldsymbol{y}^t$ can stay.

Definition 4.2. *For any non-zero vector $\boldsymbol{x} \in \mathbb{R}^n$, we define the cone of $\boldsymbol{x}$ to be*

$$Cone(\boldsymbol{x}) := \left\{ \boldsymbol{y} \in \boldsymbol{O}(\boldsymbol{x}) : \operatorname{sign}\left(|y_j| - |y_i|\right) = \operatorname{sign}\left(|x_j| - |x_i|\right) \text{ for } \forall i, j \in [n] \right\}.$$

Moreover, we say $Cone(\boldsymbol{x})$ is regular if $\boldsymbol{O}(\boldsymbol{x})$ is regular and any $|x_j| \neq |x_i|$ for all $j \neq i$.

Proposition 4.3. *For any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, the following statements are true:*

1. *Either $Cone(\boldsymbol{x}) = Cone(\boldsymbol{y})$ or $Cone(\boldsymbol{x}) \cap Cone(\boldsymbol{y}) = \emptyset$.*
2. *$\boldsymbol{x} \in Cone(\boldsymbol{x})$.*
3. *If $\boldsymbol{y} \in Cone(\boldsymbol{x})$, then $Cone(\boldsymbol{y}) = Cone(\boldsymbol{x})$.*
4. *$\cup_{\boldsymbol{y} \in \boldsymbol{O}(\boldsymbol{x})} Cone(\boldsymbol{y}) = \boldsymbol{O}(\boldsymbol{x})$.*
5. *Any regular orthant contains $n!$ regular cones.*

Lemma 4.10. *Let $\{\boldsymbol{w}^*\}$ and $\{\boldsymbol{y}^t\}$ be any sequence and auxiliary sequence generated by Algorithm 1. Assuming that $\boldsymbol{w}^* \notin \mathcal{Q}_2^n$, we have the following fact.*

1. *If $|w_j^*| > |w_i^*|$, then $|y_j^t| > |y_i^t|$ for all but finitely many t values.*
2. *If $|w_j^*| = |w_i^*|$, then $\big||y_j^t| - |y_i^t|\big|$ is bounded and $|w_j^t| = |w_i^t|$ for all but finitely many t values.*

Proof of Lemma 4.10. Without loss of generality, we can assume $w_1^* \geq w_2^* \geq \cdots \geq w_n^* \geq 0$.

For the first statement, we only need to show that $w_j^* > w_{j+1}^*$ implies $y_j^t > y_{j+1}^t$ for all but finitely many t values. Note that whenever $y_j^t < y_{j+1}^t$, then Lemma 4.4 implies $w_j^t \le w_{j+1}^t$, hence

$$\begin{aligned}
&y_j^{t+1} - y_{j+1}^{t+1} \\
&= \left(y_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left(w_j^* - w_j^t \right) \right) - \left(y_{j+1}^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left(w_{j+1}^* - w_{j+1}^t \right) \right) \\
&= \left(y_j^t - y_{j+1}^t \right) + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left[\left(w_j^* - w_{j+1}^* \right) + \left(w_{j+1}^t - w_j^t \right) \right] \\
&\ge \left(y_j^t - y_{j+1}^t \right) + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left(w_j^* - w_{j+1}^* \right).
\end{aligned}$$

Now that we know $y_j^t - y_{j+1}^t$ is increasing as long as it is negative and $\sum \eta_t = \infty$. Therefore, we conclude that there exist infinitely many t values such that $y_j^t - y_{j+1}^t > 0$. We can therefore assume $y_j^T - y_{j+1}^T > 0$, where T is the constant in Lemma 4.6 such that $\|\boldsymbol{y}^t\|_1 \ge 5n\sqrt{2\epsilon}$ for all $t \ge T$ where we set $\epsilon = \frac{\eta\|\boldsymbol{v}^*\|^2}{\sqrt{2\pi n}}$. Next, we would like to show $y_j^t - y_{j+1}^t > 0$ for all $t \ge T$ by induction.

Next, assuming $y_j^t - y_{j+1}^t > 0$, we want to show $y_j^{t+1} - y_{j+1}^{t+1} > 0$.

On the one hand, if $y_j^t - y_{j+1}^t \ge \epsilon$, we have

$$\begin{aligned}
&y_j^{t+1} - y_{j+1}^{t+1} \\
&= \left(y_j^t - y_{j+1}^t \right) + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left[\left(w_j^* - w_{j+1}^* \right) + \left(w_{j+1}^t - w_j^t \right) \right] \\
&> \left(y_j^t - y_{j+1}^t \right) + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left(w_{j+1}^t - w_j^t \right) \\
&\ge \left(y_j^t - y_{j+1}^t \right) - \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \frac{1}{\sqrt{n}} \ge \left(y_j^t - y_{j+1}^t \right) - \epsilon \ge 0.
\end{aligned}$$

On the other hand, if $y_j^t - y_{j+1}^t < \epsilon$, we still have

$$y_j^{t+1} - y_{j+1}^{t+1} > \left(y_j^t - y_{j+1}^t \right) - \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left(w_j^t - w_{j+1}^t \right),$$

so that it suffices to show $w^t_j = w^t_{j+1}$. From Lemma 4.4, we see that with

$$j^* = \arg\max_{j\in[n]} \frac{\left\|\boldsymbol{y}^t_{[j]}\right\|_1^2}{j}, \tag{4.5}$$

we only need to show $j \neq j^*$. We prove by contradiction, assuming $j = j^*$ so that $w^t_j > 0$ and $w^t_{j+1} = 0$. Lemma 4.7 shows $y^t_j \geq \frac{1}{5n}\left\|\boldsymbol{y}^t\right\|_1$. Also, (4.5) gives

$$\frac{\left\|\boldsymbol{y}^t_{[j-1]}\right\|_1^2}{j-1} \leq \frac{\left\|\boldsymbol{y}^t_{[j]}\right\|_1^2}{j} = \frac{\left(\left\|\boldsymbol{y}^t_{[j-1]}\right\|_1 + y^t_j\right)^2}{j}. \tag{4.6}$$

Simplifying the above inequality, we get

$$\left(\frac{\left\|\boldsymbol{y}^t_{[j-1]}\right\|_1}{y^t_j}\right)^2 - 2(j-1)\left(\frac{\left\|\boldsymbol{y}^t_{[j-1]}\right\|_1}{y^t_j}\right) - (j-1) \leq 0.$$

Left hand side is a quadratic function of $\left(\frac{\left\|\boldsymbol{y}^t_{[j-1]}\right\|_1}{y^t_j}\right)$, we know

$$\frac{\left\|\boldsymbol{y}^t_{[j-1]}\right\|_1}{y^t_j} \leq j-1+\sqrt{j(j-1)} \leq n-1+\sqrt{n(n-1)} < 2n. \tag{4.7}$$

We write equation (4.6) in a different way and get

$$j \geq \frac{\left(\left\|\boldsymbol{y}^t_{[j-1]}\right\|_1 + y^t_j\right)^2}{y^t_j\left(2\left\|\boldsymbol{y}^t_{[j-1]}\right\|_1 + y^t_j\right)}. \tag{4.8}$$

Now, we use $j = j^*$ again, to get

$$\frac{\left\|\boldsymbol{y}^t_{[j]}\right\|_1^2}{j} \leq \frac{\left\|\boldsymbol{y}^t_{[j+1]}\right\|_1^2}{j+1}. \tag{4.9}$$

Rewriting the above inequality, we get

$$j \leq \frac{\left(\left\|\boldsymbol{y}^t_{[j-1]}\right\|_1 + y^t_j\right)^2}{y^t_{j+1}\left(2\left\|\boldsymbol{y}^t_{[j-1]}\right\|_1 + 2y^t_j + y^t_{j+1}\right)}. \tag{4.10}$$

Combining (4.8) and (4.10), we get

$$\left(y^t_j - y^t_{j+1}\right)^2 - \left(2\left\|\boldsymbol{y}^t_{[j-1]}\right\|_1 + 4y^t_j\right)\left(y^t_j - y^t_{j+1}\right) + 2\left(y^t_j\right)^2 \leq 0.$$

Solving the above inequality, we get

$$\begin{aligned} y^t_j - y^t_{j+1} \geq{} & \left\|\boldsymbol{y}^t_{[j-1]}\right\|_1 + 2y^t_j \\ & - \sqrt{\left\|\boldsymbol{y}^t_{[j-1]}\right\|_1^2 + 4\left\|\boldsymbol{y}^t_{[j-1]}\right\|_1 y^t_j + 2(y^t_j)^2}. \end{aligned} \tag{4.11}$$

Combining (4.7) and (4.11), we get

$$y^t_j - y^t_{j+1} \geq (y^t_j)^2\left(2n + 2 - \sqrt{4n^2 + 4n + 2}\right) > \frac{(y^t_j)^2}{2}. \tag{4.12}$$

Recalling that $y^t_j \geq \frac{1}{5n}\left\|\boldsymbol{y}^t\right\|_1 \geq \sqrt{2\epsilon}$, we have

$$y^t_j - y^t_{j+1} > \frac{1}{2}\left(\frac{\|\boldsymbol{y}^t\|_1}{5n}\right)^2 > \epsilon.$$

This contradiction shows $j \neq j^*$, and hence $w^t_j = w^t_{j+1}$ and it follows that $y^{t+1}_j > y^{t+1}_{j+1}$. Now, we have proved our first statement.

For the second statement, since $w_j^* = w_i^*$, we have

$$\begin{aligned}
&y_j^{t+1} - y_i^{t+1} \\
=&\left(y_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}\left(w_j^* - w_j^t\right)\right) - \left(y_i^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}\left(w_i^* - w_i^t\right)\right) \\
=&y_j^t - y_i^t - \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}\left(w_j^t - w_i^t\right) = y_j^t - y_i^t - 2\frac{\eta_t \epsilon}{\eta}\left(w_j^t - w_i^t\right).
\end{aligned}$$

Hence, we know that $|y_j^t - y_i^t|$ is bounded by

$$C_{i,j} := \max\left\{|y_j^0 - y_i^0|, \frac{\eta\|\boldsymbol{v}^*\|^2}{\sqrt{2\pi}}\right\}.$$

Without loss of generality, we can assume $j < i$ and $\min\left\{y_j^t, y_i^t\right\} \geq 0$ by Lemma 4.8. Recalling (4.12), we have $w_j^t \neq w_i^t$ implying that

$$|y_j^t - y_i^t| > \frac{\max\left\{y_j^t, y_i^t\right\}^2}{2} \geq \frac{1}{2}\left(\frac{\|\boldsymbol{y}^t\|}{5n}\right)^2$$

where the right hand side goes to infinity. This contradicts the boundedness of $|y_j^t - y_i^t|$ if there are infinitely many t values such that $w_j^t \neq w_i^t$. □

Lemma 4.11. *Let $\{\boldsymbol{y}^t\}$ be any auxiliary real-valued sequence generated by Algorithm 1. If $\boldsymbol{w}^* \notin \mathcal{Q}$, then any sub-sequential limit of $\tilde{\boldsymbol{y}}^t := \frac{\boldsymbol{y}^t}{\|\boldsymbol{y}^t\|}$ belongs to the closure of Cone$(\boldsymbol{w}^*)$. Moreover, if Cone$(\boldsymbol{w}^*)$ is regular, then $\boldsymbol{y}^t \in$ Cone$(\boldsymbol{w}^*)$ for all but finitely many t values.*

Proof of Lemma 4.11. Note that we already have Lemma 4.9, we only need to show for any sub-sequential limit $\boldsymbol{y}$ of $\tilde{\boldsymbol{y}}^t$, we have $\operatorname{sign}(|y_j| - |y_i|) = \operatorname{sign}\left(|w_j^*| - |w_i^*|\right)$. The first statement of Lemma 4.10 tells us that it is true for all $\operatorname{sign}\left(|w_j^*| - |w_i^*|\right) \neq 0$. Thus, it suffices to show that $|w_j^*| = |w_i^*|$ implies $|y_j| = |y_i|$.

Note that the second statement of Lemma 4.10 says that $||y_j| - |y_i||$ is bounded by $C_{i,j}$, while Lemma 4.6 gives $\lim_{t\to\infty} \|\boldsymbol{y}^t\| = \infty$, we see that

$$|\tilde{y}_j| = \lim_{k\to\infty} \frac{|y_j^{t_k}|}{\|\boldsymbol{y}^{t_k}\|} = \lim_{k\to\infty} \frac{|y_i^{t_k}|}{\|\boldsymbol{y}^{t_k}\|} = |\tilde{y}_i|.$$

□

The auxiliary weight vector $\boldsymbol{y}^t$ can only stay in a small region around $\boldsymbol{w}^*$ for large t values.

Definition 4.3. *For any point $\boldsymbol{x} \in \mathbb{R}^n$, assume $(j_1, j_2, \cdots, j_n)$ is a permutation of $[n]$ such that*

$$|x_{j_1}| \geq |x_{j_2}| \geq \cdots \geq |x_{j_n}|$$

We define the set of vertexes of $\boldsymbol{x}$ to be

$$\Lambda(\boldsymbol{x}) := \left\{ \frac{1}{\sqrt{k}} \sum_{i=1}^{k} \operatorname{sign}(x_{j_i})\, \boldsymbol{e}_{j_i} : x_{j_{k+1}} \neq x_{j_k} \text{ are nonzeros} \right\}.$$

Below are some basic facts about connection between vertexes and cones.

Proposition 4.4. *For any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ let $k := |\Lambda(\boldsymbol{x})|$, the following statements are true:*

1. $0 \leq k \leq n$.
2. *$\Lambda(\boldsymbol{x})$ is empty if and only if $\boldsymbol{x} = \boldsymbol{0}$.*
3. *$\Lambda(\boldsymbol{x})$ is a subset of the boundary of $Cone(\boldsymbol{x})$.*
4. *$Cone(\boldsymbol{x}) = Cone(\boldsymbol{y})$ if and only if $\Lambda(\boldsymbol{x}) = \Lambda(\boldsymbol{y})$.*
5. *$\widetilde{\operatorname{proj}}_{\mathcal{Q}}(\boldsymbol{x}) \in \Lambda(\boldsymbol{x})$.*
6. *$\boldsymbol{y}$ lies in $Cone(\boldsymbol{x})$ if and only if there exists k positive numbers $\{\mu_{\boldsymbol{z}}(\boldsymbol{y}) : \boldsymbol{z} \in \Lambda(\boldsymbol{x})\}$ such that $\boldsymbol{y} = \sum_{\boldsymbol{z}\in\Lambda(\boldsymbol{x})} \mu_{\boldsymbol{z}}(\boldsymbol{y})\boldsymbol{z}$.*

7. *$\boldsymbol{y}$ lies in the closure of $Cone(\boldsymbol{x})$ if and only if there exists k non-negative numbers $\{\mu_{\boldsymbol{z}}(\boldsymbol{y}) : \boldsymbol{z} \in \Lambda(\boldsymbol{x})\}$ such that $\boldsymbol{y} = \sum_{\boldsymbol{z}\in\Lambda(\boldsymbol{x})} \mu_{\boldsymbol{z}}(\boldsymbol{y})\boldsymbol{z}$.*

8. *$\cup_{\boldsymbol{x}\in\mathbb{R}^n} \Lambda(\boldsymbol{x}) = \{\boldsymbol{x} \in \mathcal{Q} : \|\boldsymbol{x}\| = 1\}$.*

Lemma 4.12. *Let $\{\boldsymbol{w}^t\}$ be the sequence generated by Algorithm 1. If $\boldsymbol{w}^* \notin \mathcal{Q} = \mathbb{R}_+ \times \{0, \pm 1\}^n$, then $\boldsymbol{w}^t \in \Lambda(\boldsymbol{w}^*)$ for all but finitely many t values.*

Proof of Lemma 4.12. First, by Proposition 4.3, $\boldsymbol{y}^t \in Cone(\boldsymbol{w}^*)$ implies $\widetilde{\text{proj}}_{\mathcal{Q}}(\boldsymbol{y}^t) \in \Lambda(\boldsymbol{w}^*)$.

Second, let $\tilde{\partial} Cone(\boldsymbol{w}^*) = \overline{Cone(\boldsymbol{w}^*)} - Cone(\boldsymbol{w}^*)$. Now, a non-zero $\boldsymbol{y}^t \in \tilde{\partial} Cone(\boldsymbol{w}^*)$ implies $Cone(\boldsymbol{y}^t) \subset \tilde{\partial} Cone(\boldsymbol{w}^*)$ so that we also have $\widetilde{\text{proj}}_{\mathcal{Q}}(\boldsymbol{y}^t) \in \Lambda(\boldsymbol{y}^t) \subset \Lambda(\boldsymbol{w}^*)$.

Third, by compactness of $\overline{Cone(\boldsymbol{w}^*)} \cap \mathcal{S}^{n-1}$, we know there exists some $\epsilon > 0$ such that $\tilde{\boldsymbol{y}}^t := \frac{\boldsymbol{y}^t}{\|\boldsymbol{y}^t\|}$ lies in ϵ-neighborhood of $Cone(\boldsymbol{w}^*) \cap \mathcal{S}^{n-1}$ implying $\widetilde{\text{proj}}_{\mathcal{Q}}(\boldsymbol{y}^t) \in \Lambda(\boldsymbol{w}^*)$.

Finally, Lemma 4.11 suggests $\tilde{\boldsymbol{y}}^t$ lies in ϵ-neighborhood of $Cone(\boldsymbol{w}^*)$ for all but finitely many t values. We get our desired result. □

In the following, we give a sufficient condition for the optimum to be recurrent. The condition requires $\boldsymbol{w}^*$ to be close to $\mathcal{Q}$.

Theorem 4.2. *[Ternary Case] Let $\{\boldsymbol{z}_j\}_{j=1}^k = \Lambda(\boldsymbol{w}^*)$ where $\boldsymbol{z}_1 = \widetilde{\text{proj}}_{\mathcal{Q}}\boldsymbol{w}^*$ is the optimum and $\boldsymbol{w}^* = \sum_{j=1}^k \lambda_j \boldsymbol{z}_j$. If $0 < \sum_{j=2}^k \lambda_j < 1$, we have $\boldsymbol{w}^t = \widetilde{\text{proj}}_{\mathcal{Q}}\boldsymbol{w}^*$ for infinitely many t values, where $\boldsymbol{w}^t$ is any infinite sequence generated by Algorithm 1 with any initialization.*

Proof of Theorem 4.2 (Ternary Case). Note that Lemma 4.11 suggests $\tilde{\boldsymbol{y}}^t = \frac{\boldsymbol{y}^t}{\|\boldsymbol{y}^t\|}$ lies in ϵ-neighborhood of $Cone(\boldsymbol{w}^*)$ for all but finitely many t values. Let $\Lambda(\boldsymbol{w}^*) = \{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_k\}$ and define μ_j^t be the constants such that

$$\boldsymbol{y}^t = \sum_{j=1}^k \mu_j^t \boldsymbol{z}_j$$

which is determined uniquely by $\boldsymbol{y}^t$.

Let $\boldsymbol{w}^t = \boldsymbol{z}_{j_t}$, we know from Algorithm 1 that

$$\boldsymbol{y}^{t+1} - \boldsymbol{y}^t = \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}}(\boldsymbol{w}^* - \boldsymbol{z}_{j_t}).$$

Thus

$$\sum_{j=2}^{k} \mu_j^{t+1} = \sum_{j=2}^{k} \mu_j^t + \eta_t \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left[\left(\sum_{j=2}^{k} \lambda_j \right) - 1 \right].$$

It follows that

$$\sum_{j=2}^{k} \mu_j^t = \text{Constant} + \left(\sum_{s=0}^{t-1} \eta_s \right) \frac{\|\boldsymbol{v}\|^2}{2\sqrt{2\pi}} \left[\left(\sum_{j=2}^{k} \lambda_j \right) - 1 \right] < 0,$$

for large t's. Now we see that when t is large enough, $\tilde{\boldsymbol{y}}^t$ is bounded away from $Cone(\boldsymbol{w}^*)$ which contradicts Lemma 4.11 and our desired result follows. $\square$

Intuitively, the parameter λ_j in Theorem 4.2 stands for the proportion of time that $\{\boldsymbol{w}^t\}$ stays at $\boldsymbol{z}_j$. For instance, if $\lambda_j \approx 1$, then most of $\{\boldsymbol{w}^t\}$ stay at $\boldsymbol{z}_j$ so that the oscillation has a longer 'period' and is harder to observe. On the contrary, if all λ_j's are almost the same then $\{\boldsymbol{w}^t\}$ behaves like uniform distribution and oscillation becomes more obvious. Beside λ_j's, a smaller learning rate can render $\boldsymbol{y}^t$ moves slower which can also slow down the oscillation. Although there are ways to stabilize the training process, both our theorem and the experiments in the next section suggests the oscillation behavior is inevitable.

4.4 Experiments

In this section, we implement QUANT algorithm on both synthetic data and MNIST/CIFAR image data. Our goals are (1) to validate our theoretical findings and (2) to show the appearance of the oscillation behavior in more complicated setups. **With that said, we emphasize that we did not extensively tune the hyper-parameters or use ad-hoc tricks to achieve the best possible validation accuracy.** More comprehensive experimental results for QUANT-based approaches can be found in, for examples, [7, 8, 22, 51]. Here we report the validation accuracies on MNIST and CIFAR-10 for fully quantized networks in Table 4.1. For both synthetic and image data sets, we observed the oscillation behavior.

4.4.1 Synthetic Data

We take $m = 4$, $n = 8$ in (4.1) and construct $\boldsymbol{v} \sim N(\boldsymbol{0}, \boldsymbol{I}_m)$ and $\boldsymbol{w}^* \sim N(\boldsymbol{0}, \boldsymbol{I}_n)$ be random vectors. For each run, we fix $\boldsymbol{v}$ and $\boldsymbol{w}^*$ and train the neural network (4.1) by algorithm (1) for 200 iterations with a learning rate being 0.1. Fig. 4.3 show the evolution of binary/ternary weight of $\boldsymbol{w}^t$ in the last 100 iterations. Each block of size 8×100 corresponds to the evolution of $\boldsymbol{w}^t$ during the 100 iterations. The (quantized) global minimum $\text{proj}_{\mathcal{Q}} \boldsymbol{w}^*$ for each run is shown on the right side of the corresponding subplot in Fig. 4.3.

4.4.2 MNIST

We train LeNet-5 with binary/ternary weights and 4-bit activations using QUANT algorithm. For deep networks, the (quantized) global optimum is generally unknown, we instead show the oscillating behavior around local optimum. Note that Fig. 4.5 shows the training loss no longer drop significantly during the last 30 epochs (50 in total). This suggests the network parameters have reached a local valley. However, Fig. 4.4 shows the iterating sequence of model parameters still have oscillating signs towards the end of training.

Fig. 4.4 shows the evolution of the quantized weights of one convolution filter in the first

	float	binary	ternary
LeNet-5	99.37	99.33	99.34
ResNet-20	92.33	89.42	90.86
VGG-11	92.15	89.47	90.91

Table 4.1: Validation Accuracy of LeNet-5 on MNIST and ResNet-20/VGG-11 on CIFAR-10.

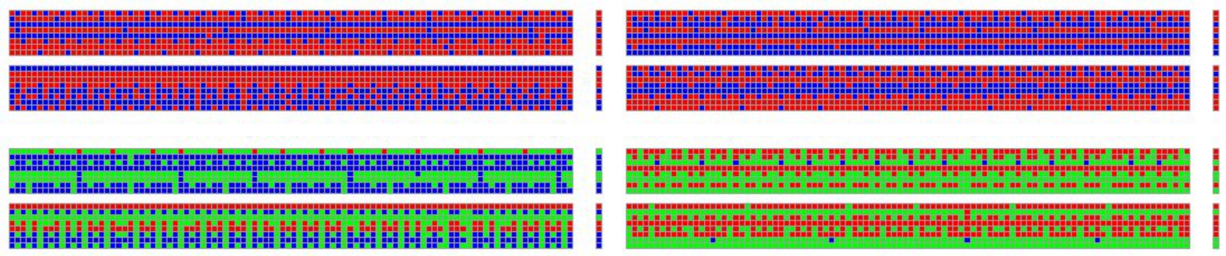


Figure 4.3: **Evolution of Weight signs of synthetic network described in (4.1).** Each of the 8 large blocks is a colored display of weight sign values via 8×100 matrix (i.e., 8 filter weight signs evolved over the last 100 iterations). The bars to the right of blocks are the corresponding optima. **Top two rows**: Binary weight signs, red /blue for $1/-1$. **Bottom two rows**: Ternary weight signs, red/green/blue for $1/0/-1$.

convolution layer during the last 600 iterations. To visualize the weights, each quantized filter is reshaped into a 25-dimensional column vector. Each block (3 in a group) of size 25×200 corresponds to the evolution of the one filter during 200 iterations. As we can see from these two figures, a proportion of the weights do not converge to a limit but rather have oscillating signs.

4.4.3 CIFAR-10

We repeat the experiments on CIFAR-10 [24] with ResNet-20/VGG-11. We train ResNet-20 [16]/VGG-11 [41] with binary/ternary weights and 4-bits activation using QUANT for 200 epochs. We refer to the appendix for some figures that show similar oscillation behavior. Towards the end of training, although there has been no noticeable decay of training loss, we can see a clearer pattern of the oscillating signs of the weights.

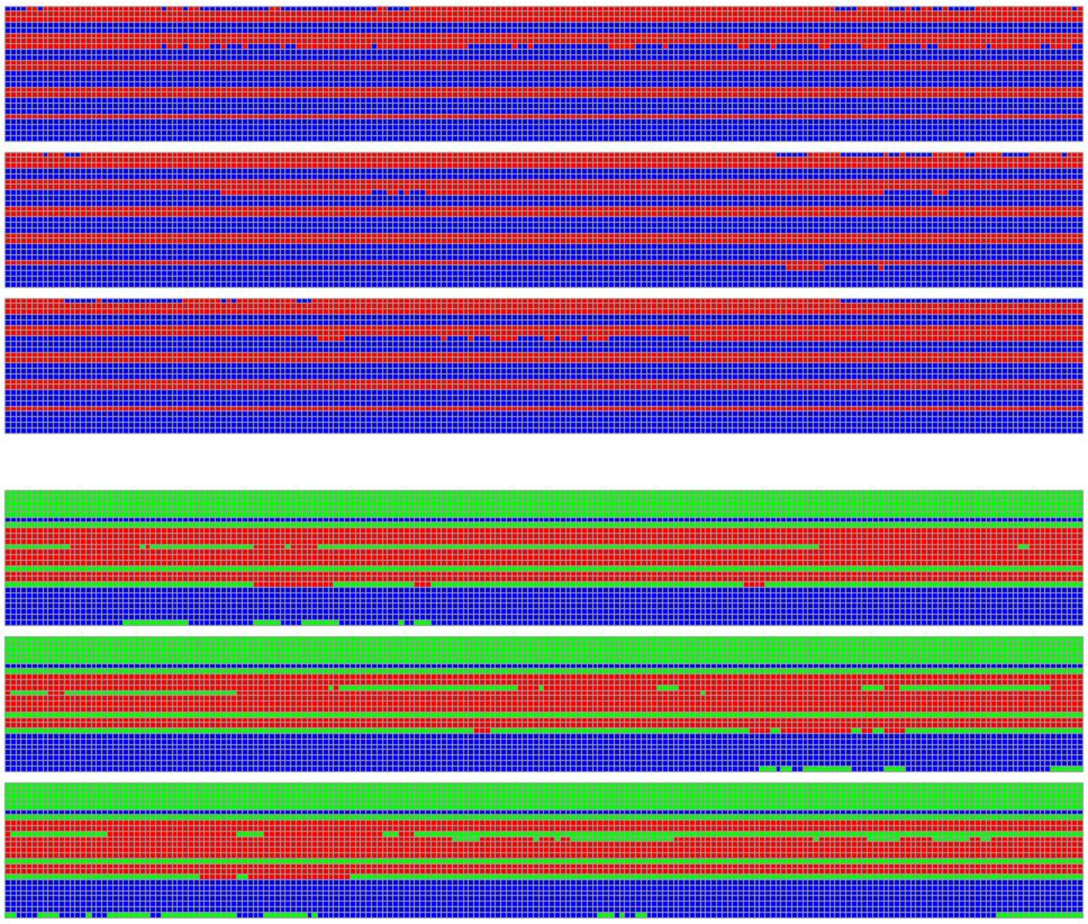


Figure 4.4: **Evolution of signs of weight filters in the last training epoch (or 600 iterations) of LeNet-5.** Each of the six 25×200 blocks corresponds to evolution of the 5×5 convolutional filter over 200 iterations. **Top three rows**: Binary weights over the last 600 iterations of training, red/blue for sign values $1/-1$. **Bottom three rows**: Ternary weights over the last 600 iterations of training, red/green/blue for sign values $1/0/-1$.

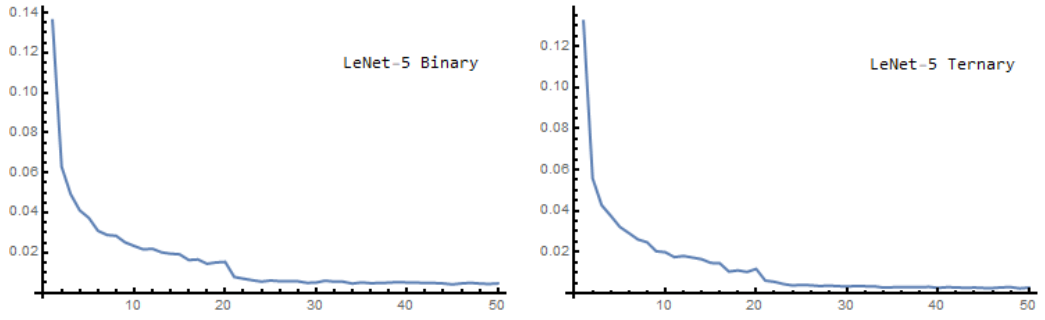


Figure 4.5: LeNet-5 Training Loss v.s. Epoch. **Left**: Binary weights. **Bottom**: Ternary weights.

Bibliography

[1] Lenet-5 architecture.

[2] Z. Allen-Zhu, Y. Li, and Z. Song. A convergence theory for deep learning via over-parameterization. *CoRR*, abs/1811.03962, 2018.

[3] Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[4] A. Brutzkus and A. Globerson. Globally optimal gradient descent for a convnet with gaussian inputs. *arXiv preprint arXiv:1702.07966*, 2017.

[5] A. Brutzkus and A. Globerson. Over-parameterization improves generalization in the XOR detection problem. *CoRR*, abs/1810.03037, 2018.

[6] A. Brutzkus, A. Globerson, E. Malach, and S. Shalev-Shwartz. SGD learns over-parameterized networks that provably generalize on linearly separable data. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[7] Z. Cai, X. He, J. Sun, and N. Vasconcelos. Deep learning with low precision by half-wave gaussian quantization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[8] J. Choi, Z. Wang, S. Venkataramani, P. I.-J. Chuang, V. Srinivasan, and K. Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018.

[9] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *International Conference on Machine Learning*, pages 160–167. ACM, 2008.

[10] M. Courbariaux, Y. Bengio, and J.-P. David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Advances in Neural Information Processing Systems*, pages 3123–3131, 2015.

[11] R. T. des Combes, M. Pezeshki, S. Shabanian, A. Courville, and Y. Bengio. Convergence properties of deep neural networks on separable data, 2019.

[12] Y. Ding, J. Liu, J. Xiong, and Y. Shi. On the universal approximability and complexity bounds of quantized relu neural networks. *arXiv preprint arXiv:1802.03646*, 2018.

[13] S. S. Du, X. Zhai, B. Póczos, and A. Singh. Gradient descent provably optimizes over-parameterized neural networks. *CoRR*, abs/1810.02054, 2018.

[14] Y. Freund and R. E. Schapire. Large margin classification using the perceptron algorithm. *Machine learning*, 37(3):277–296, 1999.

[15] J. He, L. Li, J. Xu, and C. Zheng. ReLU deep neural networks and linear finite elements. *Journal of Computational Mathematics*, 38:502–527, 2020.

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[17] G. Hinton. Neural networks for machine learning, coursera. *Coursera, video lectures*, 2012.

[18] C. Ho and S. Zimmerman. On the number of regions in an m-dimensional space cut by n hyperplanes. *The Australian Mathematical Society Gazette*, 33, 01 2006.

[19] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[20] L. Hou and J. T. Kwok. Loss-aware weight quantization of deep networks. In *International Conference on Learning Representations*, 2018.

[21] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio. Binarized neural networks. In *Advances in Neural Information Processing Systems*, 2016.

[22] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *Journal of Machine Learning Research*, 18:1–30, 2018.

[23] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[24] A. Krizhevsky. Learning multiple layers of features from tiny images. *Tech Report*, 2009.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[26] F. Li, B. Zhang, and B. Liu. Ternary weight networks. *arXiv preprint arXiv:1605.04711*, 2016.

[27] H. Li, S. De, Z. Xu, C. Studer, H. Samet, and T. Goldstein. Training quantized nets: A deeper understanding. In *Advances in Neural Information Processing Systems*, pages 5811–5821, 2017.

[28] Y. Li and Y. Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. *CoRR*, abs/1808.01204, 2018.

[29] S. Liang, R. Sun, Y. Li, and R. Srikant. Understanding the loss surface of neural networks for binary classification. *CoRR*, abs/1803.00909, 2018.

[30] C. Louizos, M. Reisser, T. Blankevoort, E. Gavves, and M. Welling. Relaxed quantization for discretized neural networks. In *International Conference on Learning Representations*, 2019.

[31] V. Mnih, 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.

[32] B. Neyshabur, Z. Li, S. Bhojanapalli, Y. LeCun, and N. Srebro. Towards understanding the role of over-parametrization in generalization of neural networks. *CoRR*, abs/1805.12076, 2018.

[33] Q. Nguyen, M. C. Mukkamala, and M. Hein. On the loss landscape of a class of deep neural networks with no bad local valleys. *CoRR*, abs/1809.10749, 2018.

[34] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, pages 525–542. Springer, 2016.

[35] S. Ren, 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*, pages 91–99, 2015.

[36] A. Rosebrock. Lenet – convolutional neural network in python, 2016.

[37] F. Rosenblatt. *The perceptron, a perceiving and recognizing automaton Project Para.* Cornell Aeronautical Laboratory, 1957.

[38] F. Rosenblatt. *Principles of neurodynamics.* Spartan Book, 1962.

[39] Z. Shen, H. Yang, and S. Zhang. Deep network approximation with discrepancy being reciprocal of width to power of depth. *arXiv preprint arXiv:2006.12231*, 2020.

[40] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484, 2016.

[41] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[42] H. Wang, Y. Wang, Z. Zhou, X. Ji, Z. Li, D. Gong, J. Zhou, and W. Liu. Cosface: Large margin cosine loss for deep face recognition. *CVPR 2018. DOI: 10.1109/CVPR.2018.00552. CoRR*, abs/1801.09414, 2018.

[43] B. Widrow and M. A. Lehr. 30 years of adaptive neural networks: perceptron, madaline, and backpropagation. *Proceedings of the IEEE*, 78(9):1415–1442, 1990.

[44] P. Yin, J. Lyu, S. Zhang, S. J. Osher, Y. Qi, and J. Xin. Understanding straight-through estimator in training activation quantized neural nets. In *International Conference on Learning Representations*, 2019.

[45] P. Yin, J. Xin, and Y. Qi. Linear feature transform and enhancement of classification on deep neural network. *J. Sci. Computing*, 76(3):1396–1406, 2018.

[46] P. Yin, S. Zhang, J. Lyu, S. Osher, Y. Qi, and J. Xin. Binaryrelax: A relaxation approach for training deep neural networks with quantized weights. *SIAM Journal on Imaging Sciences*, 11(4):2205–2223, 2018.

[47] P. Yin, S. Zhang, J. Lyu, S. Osher, Y. Qi, and J. Xin. Blended coarse gradient descent for full quantization of deep neural networks. *Research in the Mathematical Sciences*, 6(14), 2019.

[48] P. Yin, S. Zhang, Y. Qi, and J. Xin. Quantization and training of low bit-width convolutional neural networks for object detection. *Journal of Computational Mathematics*, 37:349–359, 2019.

[49] P. Yin, S. Zhang, J. Xin, and Y. Qi. Training ternary neural networks with exact proximal operator. *ArXiv*, abs/1612.06052, 2016.

[50] A. Zhou, A. Yao, Y. Guo, L. Xu, and Y. Chen. Incremental network quantization: Towards lossless CNNs with low-precision weights. *arXiv preprint arXiv:1702.03044*, 2017.

[51] S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016.

[52] C. Zhu, S. Han, H. Mao, and W. J. Dally. Trained ternary quantization. *arXiv preprint arXiv:1612.01064*, 2016.

Appendix A

Supplementary Figures for Chapter 4

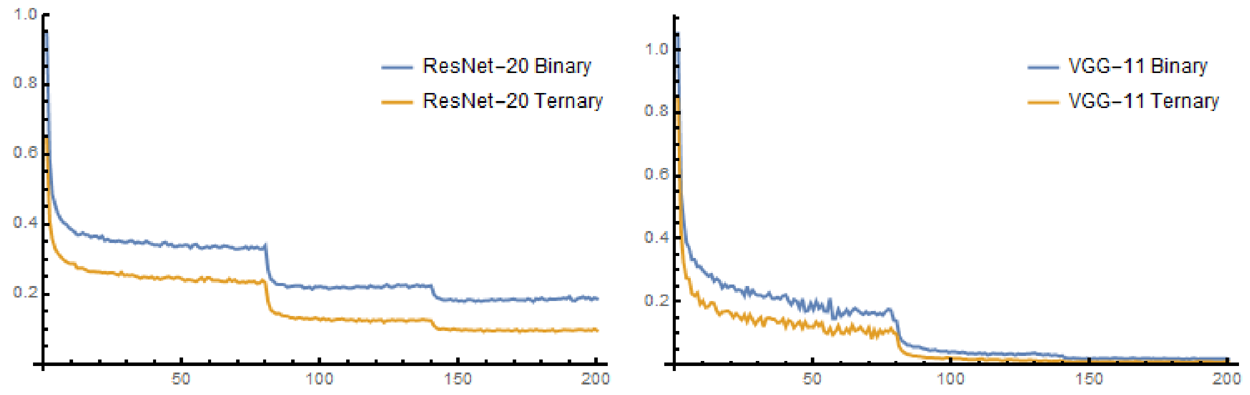


Figure A.1: Training Loss of CIFAR-10. **Left:** Binary/Ternary weight ResNet-20. **Right:** Binary/Ternary weight VGG-11.

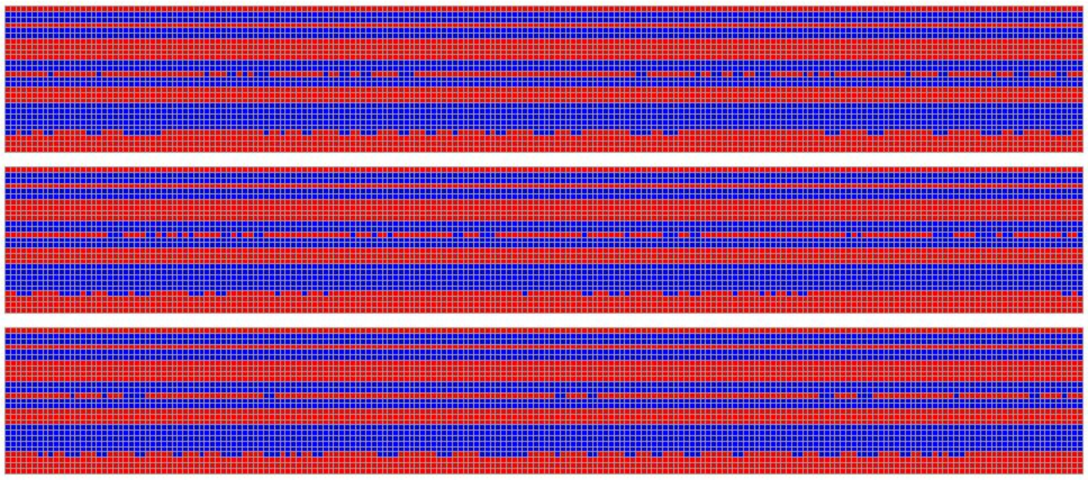


Figure A.2: **Evolution of signs of weight filters in the last training epoch (or 600 iterations) of ResNet-20.** Each of the three 27×200 blocks corresponds to evolution of the $3 \times 3 \times 3$ convolutional filter over 200 iterations. Binary weights over the last 600 iterations of training, red/blue for sign values $1/-1$.

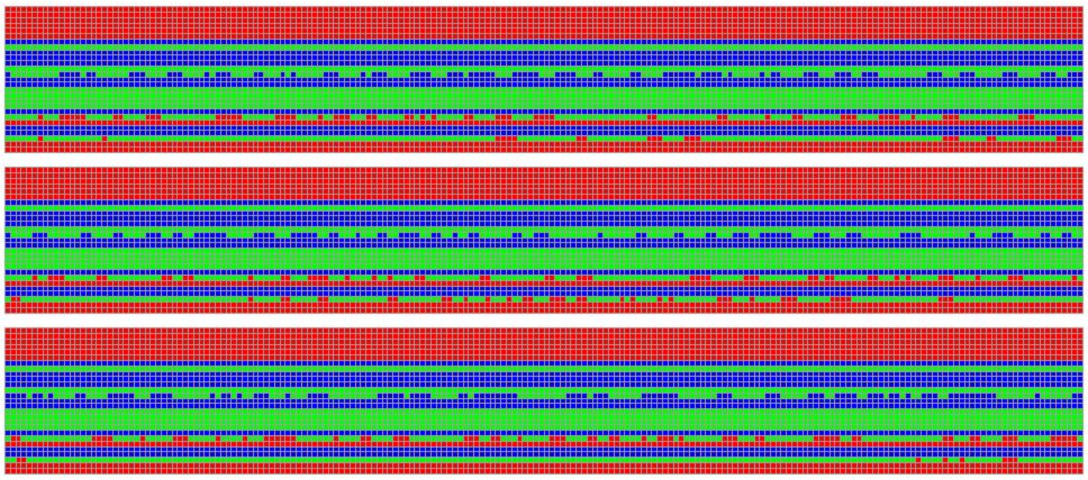


Figure A.3: **Evolution of signs of weight filters in the last training epoch (or 600 iterations) of ResNet-20.** Each of the three 27×200 blocks corresponds to evolution of the $3 \times 3 \times 3$ convolutional filter over 200 iterations. Ternary weights over the last 600 iterations of training, red/green/blue for sign values $1/0/-1$.

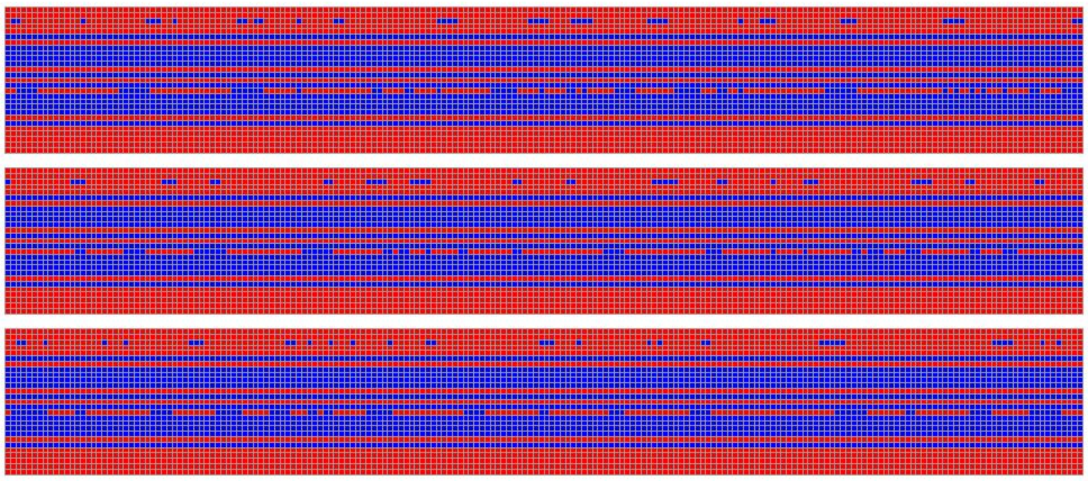


Figure A.4: **Evolution of signs of weight filters in the last training epoch (or 600 iterations) of VGG-11.** Each of the three 27×200 blocks corresponds to evolution of the $3 \times 3 \times 3$ convolutional filter over 200 iterations. Binary weights over the last 600 iterations of training, red/blue for sign values $1/-1$.

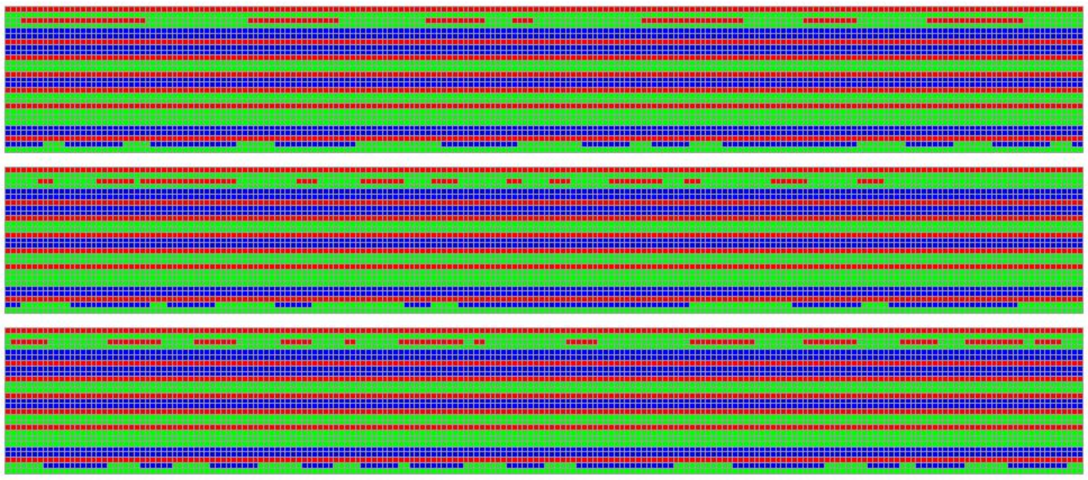


Figure A.5: **Evolution of signs of weight filters in the last training epoch (or 600 iterations) of VGG-11.** Each of the three 27×200 blocks corresponds to evolution of the $3 \times 3 \times 3$ convolutional filter over 200 iterations. Ternary weights over the last 600 iterations of training, red/green/blue for sign values $1/0/-1$.