HOCHSCHULE MITTWEIDA
UNIVERSITY OF APPLIED SCIENCES

MASTER THESIS

Mr.
Asirifi Boa

Comparison of Generalized Learning Vector Quantization learning dynamic and numerical stability regarding the Crammer-normalization and the Hein-normalization for adversarial robustness

2023

Faculty of **Applied Computer and Bio Sciences**

MASTER THESIS

Comparison of Generalized Learning Vector Quantization learning dynamic and numerical stability regarding the Crammer-normalization and the Hein-normalization for adversarial robustness

Author:
Asirifi Boa

Study Programme:
Applied Mathematics for Network and Data Sciences

Seminar Group:
MA19w1-M

First Referee:
Prof. Dr. Thomas Villmann

Second Referee:
Dr. Marika Kaden

Mittweida, 2023

Acknowledgement

I thank Prof. Dr. Thomas Villmann for his supervision and guidance and Dr. Marika Kaden for her support and comments. I appreciate the German government's support. Thanks to friends and loved ones, especially Nana Abeka Otoo and Shaghik Amirian, for their support. I convey love and thanks to Kofi Nkrumah Asirifi Boa and Nana Yaa Montoa Asirifi Boa. A special thanks to my parents and siblings for their unwavering support. Finally, I thank the Divine for His guidance in my academic journey.

Bibliographic Information

Boa, Asirifi : Comparison of Generalized Learning Vector Quantization learning dynamic and numerical stability regarding the Crammer-normalization and the Hein-normalization for adversarial robustness, 37 pages, 5 figures, Hochschule Mittweida, University of Applied Sciences, Faculty of Applied Computer and Bio Sciences

Master Thesis, 2023

Abstract

Adversarial robustness of a nearest prototype classifier assures safe deployment in sensitive use fields. Much research has been conducted on artificial neural networks regarding their robustness against adversarial attacks, whereas nearest prototype classifiers have not chalked similar successes. This thesis presents the learning dynamics and numerical stability regarding the Crammer-normalization and the Hein-normalization for adversarial robustness of nearest prototype classifiers. Results of conducted experiments are penned down and analyzed to ascertain the bounds given by Saralajew et al. and Hein et al. for adversarial robustness of nearest prototype classifiers.

I. Contents

II. List of Figures

III. List of Tables

1 Introduction

Human activities in recent times have led to the generation of vast amounts of data. An example is the numerous data uploaded on various digital platforms daily [28]. The amount of data is so huge in such a way that analyzing, interpreting, and making predictions using traditional statistics do not carry much insight compared to machine learning methods [28].

Machine learning models learn patterns in data and make intelligent predictions and informed decisions based on the learned patterns given a new input [27, 28]. Machine learning models deliver good generalizations given a new instance when trained well [11]. An Artificial Neural Network is one of the many machine learning models that learn patterns in data and give good predictions given new input. However, the training and testing of artificial neural networks need more in-depth interpretability [10], so deploying it in sensitive areas like medicine and autonomous driving may not be advisable. Nearest Prototype Classifiers are good to consider in sensitive applications since they deliver mathematical precision and are comprehensible [10, 27].

A key requirement of a modern transport pool is autonomous technology [37]. Imagine what will happen to road users and valuable infrastructure when a self-driving truck trained to learn and use the patterns on the road signs and markings is attacked by an adversary who alters the road signs and markings in order for the truck to interpret wrongly the road signs and markings [29]. The consequences will be catastrophic. In medical diagnostic tools and robotic medical setups that are heavily reliant on machine learning [1], when attacked by an adversary who abuses the system will also put the lives of users in severe danger, and the consequences will be devastating. The same is true for naive users of large language models like chatGPT [44] when wrong information is churned out and consumed due to adversarial attacks.

The effects of adversarial attacks and many other reasons have driven researchers to explore how adversaries attack machine learning models and the best defense mechanisms to resist these attacks [32]. Hence, the robustness of a machine learning model against adversarial attacks plays a crucial role in its safe deployment and utilization.

1.1 Motivation

Artificial intelligence, according to [13], is human-like intelligence being mimicked by machines. One branch of artificial intelligence is machine learning, a discipline focused on constructing models that can learn patterns within data and utilize these patterns for making predictions [26–28]. Machine learning encompasses various learning frameworks, including supervised learning, unsupervised learning, and reinforcement learning [28, 40].

In reinforcement learning, an agent engages with its environment over time. The agent makes decisions, taking actions that transition it from one state to the next while receiving rewards or penalties for each action. Again, the model learns through a combination of exploration by experimenting with new strategies and exploitation to improve previously learned techniques. Reinforcement learning maximizes the expected cumulative reward over time through a policy, often modeled as a Markov Decision Process. An example of reinforcement learning is autonomous robotics [25, 28, 40].

Unsupervised learning presents a model with an unlabeled dataset, often denoted as $A = \{\mathbf{a_1}, \ldots, \mathbf{a_N}\}$, a training set and $\mathbf{N}$ represents the number of training instances. Unsupervised learning aims to find meaningful structural properties in the data to learn patterns and derive insights from the unlabeled dataset for decision-making. Examples of unsupervised learning include clustering, where data points are grouped based on similarity, and dimensionality reduction, which seeks to simplify data representation by reducing the number of features while preserving essential information [26, 28].

Supervised learning focuses on establishing a connection from an input $\mathbf{a}$ to an output $\mathbf{y}$. A labeled example dataset guides the supervised learning process, typically represented as $A = \{\mathbf{a_1}, \ldots, \mathbf{a_N}, \mathbf{y_1}, \ldots, \mathbf{y_N}\}$. In this context, A is commonly referred to as the training set, and $\mathbf{N}$ signifies the total number of training examples. If the output variable $\{\mathbf{y_1}, \ldots, \mathbf{y_N}\}$ falls into a finite set of categories or nominal values, expressed as {1, . . . , C}, the specific task is recognized as classification. On the other hand, if $\{\mathbf{y_1}, \ldots, \mathbf{y_N}\}$ assumes real numerical values, the problem is termed regression [17, 18, 28].

This thesis considers classification learning. Moreover, the datasets (MNIST dataset [22] and CIFAR-10 dataset [21]) used for the experiments in chapter four are labeled image datasets. The machine learning algorithms for the classification learning considered in this thesis are nearest prototype classifiers, including Kohonen Learning Vector Quantization [19], Generalized Learning Vector Quantization [35], Generalized Matrix Learning Vector Quantization [6, 36] and Generalized Tangent Learning Vector Quantization [34].

A good machine learning model's vital attributes include good generalization ability. Generalization refers to a machine learning model's ability to predict a given unknown instance [11]. What about the robustness of the machine learning model against noise in data and adversarial attacks? Machine learning has taken center stage in our day-to-day activities, and as a result, the security of its associated algorithms against adversarial attacks is under severe scrutiny as to how reliable and robust the machine learning algorithms are, and their ability to withstand adversarial attacks [32]. Adversarial attacks are carefully crafted inputs designed to mislead a machine learning model during classification [32]. This crafted input can change or alter the output (misclassification) and also can lead the classifier to predict output to a different class as desired by the attacker [32, 33]. An adversarial example of an input $\mathbf{a}$ is defined as a minimum perturbation δ such that the input data is classified wrongly (lies on the decision boundary or is located in another class). The method that gives rise to an adversarial example is known as adversarial attack [33, 41]. Analysis of decision boundaries is crucial in studying learning dynamic and numerical stability of an algorithm. Learning dynamic refers to the performance of a classifier to changes as it is trained on new data or as its hyperparameters are adjusted. On the other hand, the numerical stability of a model refers to the performance of the model in the face of minor errors, perturbations, or uncertainties arising from the inherent limitations of approximation errors during computation. A model is deemed stable when minor alterations in the initial conditions lead to negligible changes in the final solution, signifying its ability to remain accurate and reliable. [2, 7, 32, 42]. Saralajew et al. [32] shows that robustness and generalization are two distinct phenomena and thus must be noted clearly without confusion.

Crammer et al. [11] present bounds of Nearest Prototype Classifiers (NPCs) and prove that the sample margin is the upper bound of the hypothesis margin using Euclidean distance. Saralajew et al. [33] also proved that the sample margin is the upper bound of the hypothesis margin using semi-norms. Hein et al. [47] confirm the findings of [33] using a semi-metric but this time with a derived larger bound for certified robustness accuracy of a NPC. In this thesis, we discuss the generalization ability of NPCs through the analysis of their margin and further extend the discussion to their ability to withstand adversarial perturbation. Certification is key when considering an algorithm's robustness against adversarial attacks; therefore, we discuss how to certify a given NPC and [32, 34] give a lead in the research of adversarial attacks concerning interpretable models.

1.2 Outline of the Thesis

In chapter one, we provided an introduction and outlined the motivation behind the thesis. Chapter two delves into the discussion of Learning Vector Quantization, followed by an exploration of the learning dynamics and numerical stability of nearest prototype classifiers in chapter three. We conducted experiments in chapter four and analyzed our findings. Finally, chapter five encompasses the presentation of conclusions and suggestions.

2 Learning Vector Quantization

Learning Vector Quantization (LVQ) is an interpretable machine learning model introduced as an improvement to the nearest neighbor classifier [11]. For example, the K-nearest neighbor algorithm classifies a given instance based on the majority vote from its k-nearest neighbors within the training dataset. The distances between the input data and all training instances are computed and the algorithm selects the k-closest samples. The new instance is given the class label belonging to the most common class among the k neighbors of the training instances [39, 43]. In the LVQ algorithm, a considerably small dataset termed prototypes are chosen appropriately to represent the entire training data. Compared to the K-nearest neighbor algorithm, LVQ cuts down computational cost and can handle and train reasonably large datasets [11]. In the LVQ algorithm, two important things are requisite: prototypes and a (dis-)similarity measure [33]. The question arises regarding how to distribute prototypes to maximize classification accuracy efficiently. T. Kohonen came out with a heuristic for distributing the prototypes in an iterative way given in [20].

2.1 Kohonen Learning Vector Quantization

Let $A=\{\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_N\}\subseteq\mathbb{R}^n$ and $c(\mathbf{a})\in\mathbb{C}=\{1,2,\ldots,C\}$, be training set and their corresponding class labels respectively. Define $P=\{\mathbf{p}_1,\mathbf{p}_2,\cdots,\mathbf{p}_M\}\subseteq\mathbb{R}^n$ to be the set of prototypes where $\mathbf{p}\in P$ are equipped with corresponding class $c(\mathbf{p})\in\mathbb{C}$. Appropriately each prototype is allocated to one class and at least a prototype is responsible for a class. The goal is to distribute the prototypes given an arbitrary instance such that the classification accuracy is maximized utilizing the nearest prototype principle where the prototype closest to the instance is selected as the competition winner [17, 18, 20]. Thus:

$$F(\mathbf{a})=\arg\min_k\ d(\mathbf{a},\mathbf{p}_k)\ \ \text{with}\ k=\{1,\ldots,M\} \tag{2.1}$$

where d is an appropriate dissimilarity measure usually taken as the Euclidean distance [20] and the winner prototype is $\mathbf{p}_{F(\mathbf{a})}$. After the winner prototype ($\mathbf{p}_{F(\mathbf{a})}$) is determined, it is updated as:

$$\Delta\mathbf{p}_{F(\mathbf{a})}=\gamma\cdot\kappa(\mathbf{a},\ F(\mathbf{a}))\cdot\left(\mathbf{a}-\mathbf{p}_{F(\mathbf{a})}\right);\quad 0<\gamma\ll 1 \tag{2.2}$$

where γ is the learning rate and

$$\kappa(\mathbf{a},\ F(\mathbf{a}))=\begin{cases}+1, & c(\mathbf{a})=c\left(\mathbf{p}_{F(\mathbf{a})}\right)\\ -1, & c(\mathbf{a})\neq c\left(\mathbf{p}_{F(\mathbf{a})}\right)\end{cases} \tag{2.3}$$

LVQ classifier and its variants achieve a scheme known as the attraction and repulsion scheme [20]. During training, the algorithm updates the position of the prototypes iteratively in the feature space to maximize classification accuracy by computing the distance between the input data and the prototypes and then determining which prototype is closest to the input data [20]. The algorithm pulls the prototype closer to the input data (attraction) if the closest prototype belongs to the correct class and vice versa [20]. In this way, the attraction and repulsion scheme is realized. The adaptation of LVQ 1 is heuristically inclined and to enhance the learning dynamics, T. Kohonen came out with other variants like LVQ 2 and LVQ 3, which are also heuristics [20].

2.2 Generalized Learning Vector Quantization

Sato and Yamada [35] came up with a variant of LVQ model. They named it Generalized Learning Vector Quantization (GLVQ) with a differentiable cost function, which addresses the problem of the heuristic adaptation in Kohonen LVQ [20]. The cost function is given by:

$$J = \sum_{\mathbf{a} \in A} f\left(\mu\left(\mathbf{a}\right)\right) \tag{2.4}$$

where

$$\mu\left(\mathbf{a}\right) = \frac{d\left(\mathbf{a}, \mathbf{p}^{+}\right) - d\left(\mathbf{a}, \mathbf{p}^{-}\right)}{d\left(\mathbf{a}, \mathbf{p}^{+}\right) + d\left(\mathbf{a}, \mathbf{p}^{-}\right)} \quad \in [-1, 1] \tag{2.5}$$

is the classifier function [18, 35]. In the classifier function, $\mathbf{p}^{+}$ defines the optimum matching prototype vector belonging to the correct class given that $c\left(\mathbf{a}\right) = c\left(\mathbf{p}_{F(\mathbf{a})}\right)$, $\mathbf{p}^{-}$ is defined as the optimum matching prototype vector belonging to the compliment class of $\mathbf{p}^{+}$ where $c\left(\mathbf{a}\right) \neq c\left(\mathbf{p}_{F(\mathbf{a})}\right)$ and d is an appropriate chosen dissimilarity measure. In the cost function, f defines a monotonically increasing function frequently chosen as the sigmoid function; for example,

$$f\left(v\right) = \frac{1}{1 + exp(-v)} \tag{2.6}$$

or

$$f\left(v\right) = id(v) = v \tag{2.7}$$

the identity function [18]. If there is a correct classification, the dissimilarity measure between the input and the optimum matching prototype vector belonging to the correct class becomes smaller than between the input and the optimum matching prototype vector belonging to the incorrect class. Therefore, the classifier function becomes negative and vice versa [31, 35]. The classifier function $\mu(\mathbf{a})$ together with the sigmoid function is differentiable, and therefore optimization can be achieved using stochastic gradient descent learning (SGDL). The given update rule is as follows:

$$\Delta \mathbf{p}^{\pm} \propto \varepsilon \cdot \frac{\partial J}{\partial \mathbf{p}^{\pm}} \tag{2.8}$$

Taking into account the dissimilarity as the squared Euclidean distance [18, 35], we update both $\mathbf{p}^{+}$ and $\mathbf{p}^{-}$ since they contribute to the determination of the local error. Their derivatives are given by:

$$\begin{aligned} \frac{\partial J}{\partial \mathbf{p}^{\pm}} &= \frac{\partial J}{\partial \mu} \cdot \frac{\partial \mu}{\partial d(\mathbf{a},\mathbf{p}^{\pm})} \cdot \frac{\partial d(\mathbf{a},\mathbf{p}^{\pm})}{\partial \mathbf{p}^{\pm}} \\ &= \pm 4 \cdot \frac{\partial f}{\partial \mu} \cdot \frac{d(\mathbf{a},\mathbf{p}^{\mp})}{\left(d(\mathbf{a},\mathbf{p}^{\pm}) + d(\mathbf{a},\mathbf{p}^{\mp})\right)^2} \cdot \left(\mathbf{a} - \mathbf{p}^{\pm}\right) \end{aligned} \tag{2.9}$$

The update rule then becomes positive for $\mathbf{p}^{+}$ and negative for $\mathbf{p}^{-}$ realizing the attraction and repulsion scheme, which is the main framework of LVQ models [20, 34, 35, 45].

2.3 Generalized Matrix Learning Vector Quantization

In GLVQ, we consider that all the weights assigned to the input vectors are equivalent and that the dissimilarity measure (Euclidean distance) employed is also suitable [16]. Instead of assigning equal weight to each instance, in matrix learning, a relevance matrix is mapped to the instance to learn the weights based on their importance to improve classification accuracy and reduce computational cost. The necessary condition is that the dissimilarity measure is differentiable [6, 36]. A dissimilarity measure for the Generalized Vector Quantization, which uses a full matrix of relevance, is given as:

$$d^{\Lambda}(\mathbf{a},\mathbf{p}) = (\mathbf{a} - \mathbf{p})^{T} \Lambda (\mathbf{a} - \mathbf{p}) \tag{2.10}$$

We require that Λ is a positive definite to get a squared Euclidean distance in an appropriate transformed space [6]. When Λ is positive definite, we can decompose it into:

$$\Lambda = \Omega^{T}\Omega, \quad \Omega \in \mathbb{R}^{m \times n} \tag{2.11}$$

It follows from (2.10) that:

$$\begin{aligned} d^{\Lambda}(\mathbf{a},\mathbf{p}) &= (\mathbf{a} - \mathbf{p})^{T} \Omega^{T} \Omega (\mathbf{a} - \mathbf{p}) \\ d^{\Lambda}(\mathbf{a},\mathbf{p}) &= (\Omega(\mathbf{a} - \mathbf{p}))^{2} \\ d^{\Lambda}(\mathbf{a},\mathbf{p}) &= d_{\Omega}(\mathbf{a},\mathbf{p}) \end{aligned} \tag{2.12}$$

From (2.11), if $m = n$, the resulting dissimilarity takes the form of the squared Euclidean distance of linearly projected data where Ω is interpreted as a projection matrix [46]. The choice of Ω helps to achieve a good classification, and because (2.12) is a squared Euclidean distance, we can adapt the GLVQ cost function in (2.4). The resulting cost function is [6,9] :

$$J_{GMLVQ}(P,\Omega) = \sum_{\mathbf{a}\in A} J(\mathbf{a},P,\Omega) \tag{2.13}$$

with

$$J(\mathbf{a},P,\Omega) = f(\mu(\mathbf{a},P,\Omega))$$

and

$$\mu(\mathbf{a},P,\Omega) = \frac{d_\Omega(\mathbf{a},\mathbf{p}^+) - d_\Omega(\mathbf{a},\mathbf{p}^-)}{d_\Omega(\mathbf{a},\mathbf{p}^+) + d_\Omega(\mathbf{a},\mathbf{p}^-)}, \in [-1,1].$$

$f(\mu(\mathbf{a},P,\Omega))$ depends on both the prototypes P and Ω and therefore we learn both P and Ω by SGDL and (2.13) is optimized as [6,9]:

$$\Delta\Omega \propto -\frac{\partial J(\mathbf{a})}{\partial f}\cdot\frac{\partial f}{\partial\mu}\cdot\left(\frac{\partial\mu}{\partial d_\Omega^+(\mathbf{a})}\cdot\frac{\partial d_\Omega^+(\mathbf{a})}{\partial\Omega} + \frac{\partial\mu}{\partial d_\Omega^-(\mathbf{a})}\cdot\frac{\partial d_\Omega^-(\mathbf{a})}{\partial\Omega}\right) \tag{2.14}$$

$$\Delta\mathbf{p}^\pm \propto -\frac{\partial J(\mathbf{a})}{\partial f}\cdot\frac{\partial f}{\partial\mu}\cdot\frac{\pm 2 d_\Omega^\mp(\mathbf{a})}{\left(d_\Omega^+(\mathbf{a}) + d_\Omega^-(\mathbf{a})\right)^2}\cdot\frac{\partial d_\Omega^\pm(\mathbf{a})}{\partial\mathbf{p}^\pm} \tag{2.15}$$

and $d_\Omega^\pm(\mathbf{a})$ is used as a shorthand form for $d_\Omega(\mathbf{a},\mathbf{p}^\pm)$ [30].

2.4 Generalized Tangent Learning Vector Quantization

We introduce Generalized Tangent Learning Vector Quantization (GTLVQ), a variant of LVQ with some definitions.

Definition 2.1 (Orthogonal matrix): A real matrix A is orthogonal if $A^TA = AA^T = I$. Equivalently $A^T = A^{-1}$ and that the matrix A is non-singular.

Definition 2.2 (Orthogonal complement): Let V be a finite-dimensional vector space over the field F with an inner product and U a subspace of V. The orthogonal complement of U denoted as $U^\perp$ is the set of all vectors in V that are orthogonal to every vector in U:

$$U^\perp = \{\mathbf{v}\in V \mid \langle\mathbf{u},\mathbf{v}\rangle = 0 \text{ for every } \mathbf{u}\in U\}$$

Definition 2.3 (Orthogonal projection, P_U): Let U be a finite-dimensional subspace of the vector space V. The definition of orthogonal projection of V onto U is the operator

$P_U \in \mathscr{L}(V)$ defined as follows: For $\mathbf{v} \in V$, we write $\mathbf{v} = \mathbf{u} + \mathbf{w}$ where $\mathbf{u} \in U$ and $\mathbf{w} \in U^{\perp}$. Then $P_U \mathbf{v} = \mathbf{u}$. The operator, P_U, has the properties that:

- $P_U \in \mathscr{L}(V)$
- $P_U \mathbf{u} = \mathbf{u}$ for all $\mathbf{u} \in U$
- $P_U \mathbf{w} = 0$ for all $\mathbf{w} \in U^{\perp}$
- ${P_U}^2 = P_U$

Tangent distance learning introduced by [38], applied in machine learning centers on assigning a local geometry of data manifold. Thus, in Generalized Tangent Learning Vector Quantization (GTLVQ), prototypes are not considered as points but rather as a set of points called manifold [34]. The ability of tangent learning to incorporate information about local changes helps models that use tangent distance to be robust against systemic variations (transformation) like rotation and reflections of new samples fed to the classifier [34]. The considerations while using GTLVQ are that the input $\mathbf{a}_i$ is a parameterized data manifold $A_i(\gamma)$ where $\gamma \in \mathbb{R}^s$ is a parameter vector and $\mathbf{a}_i = A_i(O)$ [34]. In the like manner, we define $\mathbf{p_j}$ to be parameterized prototype manifold $P_j(\theta)$ where $\theta \in \mathbb{R}^r$ is a parameter vector and $\mathbf{p_j} = P_j(O)$ with $r, s \ll n$ [34]. The goal is to compute the shortest possible path between the two manifolds proposed in [34] as:

$$\tilde{D}^* \left(A_i, P_j\right) = \min_{\gamma,\theta} \{ d\left(A_i(\gamma), P_j(\theta)\right) \} \tag{2.16}$$

where d is an appropriate dissimilarity measure between the two parameterized manifolds. Computing the shortest possible path between the manifolds is a minimization problem. In this regard, we consider a linear approximation using Taylor approximation of the given manifolds $A_i(\gamma)$ at $\mathbf{a_i}$ [34]. The approximation is given by:

$$A_i(\gamma) \backsimeq \mathbf{a_i} + B_i \gamma \tag{2.17}$$

where

$$B_i = \left. \frac{\partial A_i(\gamma)}{\partial \gamma} \right|_{\gamma=0}$$

is the set of tangent subspace basis vectors at $\mathbf{a_i}$ where $\mathbf{a_i} \in \mathbb{R}^n$, $\gamma \in \mathbb{R}^s$, we obtain $B_i \in \mathbf{a_i}\gamma^T$ with dim(B_i)=s [34]. Similarly, we obtain for prototype manifold $P_j(\theta)$ at $\mathbf{p_j}$ the approximation as [34]:

$$P_j(\theta) \simeq \mathbf{p_j} + C_j \theta \tag{2.18}$$

where

$$C_j = \left. \frac{\partial P_j(\theta)}{\partial \theta} \right|_{\theta=0}$$

is the set of tangent subspace basis vectors at $\mathbf{p_j}$ where $\mathbf{p_j} \in \mathbb{R}^n$, $\theta \in \mathbb{R}^r$, we obtain

$C_j \in \mathbf{p_j}\theta^T$ with dim(C_j)=r [34]. Combining (2.17) and (2.18) yields a two-sided tangent distance defined as:

$$\tilde{D}\left(\mathbf{a_i}, B_i, \mathbf{p_j}, C_j\right) = \min_{\gamma,\theta}\{d\left(\mathbf{a_i} + B_i\gamma, \mathbf{p_j} + C_j\theta\right)\} \tag{2.19}$$

where d defines an appropriate dissimilarity measure for the vectors [34]. It turns out that, for an optimum distance given an input $\mathbf{a}$, we compute the minimum path between the input and the prototype manifold [34]. This distance is termed one-sided tangent distance. The one-sided tangent distance yields an equivalent result as the two-sided tangent [34]. The computation of the one-sided tangent distance is given by:

$$\tilde{D}\left(\mathbf{a_i}, \mathbf{p_j}, C_j\right) = \min_{\theta}\{d\left(\mathbf{a_i}, \mathbf{p_j} + C_j\theta\right)\} \tag{2.20}$$

Taking into consideration the squared Euclidean distance as the dissimilarity d and incorporating it into (2.20) yields:

$$D\left(\mathbf{a_i}, \mathbf{p_j}, C_j\right) = \min_{\theta}\{\left(\mathbf{a_i} - (\mathbf{p_j} + C_j\theta)\right)^2\} \tag{2.21}$$

We optimize (2.21) to find the minimum as:

$$\frac{\partial D\left(\mathbf{a_i}, \mathbf{p_j}, C_j\right)}{\partial\theta} = 0$$

which gives:

$$\theta = C_j^T\left(\mathbf{a_i} - \mathbf{p_j}\right) \tag{2.22}$$

Given that the matrix C_j is orthogonal as defined in (2.1). The second partial derivative is computed to show if the optimum θ is minimum. Therefore, the optimum θ is minimum if the outcome is positive. It holds that:

$$\frac{\partial^2 D\left(\mathbf{a_i}, \mathbf{p_j}, C_j\right)}{\partial\theta\partial\theta} = I$$

Moreover, I, the identity matrix is symmetric and has positive eigenvalues, and therefore the minimum θ is the optimum value [3,34]. Substituting (2.22) into (2.21) gives:

$$D\left(\mathbf{a_i}, \mathbf{p_j}, C_j\right) = \left(\mathbf{a_i} - (\mathbf{p_j} + C_jC_j^T(\mathbf{a_i} - \mathbf{p_j}))\right)^2 \tag{2.23}$$

$$D\left(\mathbf{a_i}, \mathbf{p_j}, C_j\right) = \left((\mathbf{a_i} - \mathbf{p_j}) - (C_jC_j^T(\mathbf{a_i} - \mathbf{p_j}))\right)^2$$

$$D\left(\mathbf{a_i}, \mathbf{p_j}, C_j\right) = \left((\mathbf{a_i} - \mathbf{p_j})(I - C_jC_j^T)\right)^2$$

$$D\left(\mathbf{a_i}, \mathbf{p_j}, C_j\right) = \left(\Omega_j(\mathbf{a_i} - \mathbf{p_j})\right)^2$$

$$D\left(\mathbf{a_i}, \mathbf{p_j}, C_j\right) = d_{\Omega_j}\left(\mathbf{a_i} - \mathbf{p_j}\right) \tag{2.24}$$

Where Ω_j defines an orthogonal projector onto the complement of the linear subspace defined by the basis vectors of C_j and that $D\left(\mathbf{a_i}, \mathbf{p_j}, C_j\right)$ gives the shortest path defined between $\mathbf{a_i}$ and the affine subspace defined by $\mathbf{p_j} + C_j\theta$ taking into consideration, the dissimilarity as the Euclidean squared distance [3,34]. Comparing the GMLVQ equation (2.12) with GTLVQ equation (2.24), if we restrict $\Omega = \Omega_j$ GMLVQ realizes GTLVG. The tangent distance (2.24) is then adapted into the GLVQ classifier function (2.5) and the update of the translation follows the SGDL vividly explained in [34].

3 Learning Dynamics and Numerical Stability of LVQs

In this chapter, we analyze the margins of LVQs and their relevance in determining the learning dynamics, generalization abilities, and numerical stability of the LVQ variants.

3.1 Relevance of Margin Analysis in Determining the Learning Dynamics and Generalization Abilities of LVQs

Analysis of margins is crucial in determining the learning dynamics and generalization abilities of LVQ variants of algorithms. It measures the confidence of a classifier for predictions [11]. Learning dynamics refers to the model's behavior during training regarding the model's convergence and the time it takes to terminate. Generalization refers to the model's ability to predict a given unknown instant. A good classification rule depends on where the decision boundary is [5, 11]. Crammer et al. [11] relate the sample margin to support vector machine and define it as:

Definition 3.1 (Sample margin): The sample margin is the distance between an instance and the decision boundary induced by the classification rule.

The sample margin is difficult to compute in LVQ models. An alternative margin is the hypothesis margin.

Definition 3.2 (Hypothesis margin): The hypothesis margin is the distance a classifier can travel without changing how it labels any sample points.

Definition 3.3 (Semi-metric): A mapping $A \times A \rightarrow \mathbb{R}$ is a semi-metric if the following properties are satisfied for all $\mathbf{a}, \tilde{\mathbf{a}}, \bar{\mathbf{a}} \in A$:

- $d(\mathbf{a}, \tilde{\mathbf{a}}) \geq 0$ (non-negativity)
- $d(\mathbf{a}, \tilde{\mathbf{a}}) = d(\tilde{\mathbf{a}}, \mathbf{a})$ (symmetry)
- $d(\mathbf{a}, \tilde{\mathbf{a}}) \leq d(\mathbf{a}, \bar{\mathbf{a}}) + d(\bar{\mathbf{a}}, \tilde{\mathbf{a}})$ (triangle inequality)

A semi-metric with additional property that $d(\mathbf{a}, \tilde{\mathbf{a}}) = 0 \implies \mathbf{a} = \tilde{\mathbf{a}}$ is a metric.

Definition 3.4 A mapping $A \times A \to \mathbb{R}$ is a semi-norm if the following properties are satisfied for all $\mathbf{a}, \tilde{\mathbf{a}} \in A$:

- $||\mathbf{a}|| \geq 0$ (non-negativity)
- $||\alpha\mathbf{a}|| = \mid \alpha \mid ||\mathbf{a}||$ (absolute homogeneity)
- $||\mathbf{a} + \tilde{\mathbf{a}}|| \leq ||\mathbf{a}|| + ||\tilde{\mathbf{a}}||$ (triangle inequality)

A semi-norm with additional property that $||\mathbf{a}|| = 0 \implies \mathbf{a} = 0$ is a norm.

Problem Setting
Let $\mathbf{p}^*$ be the closest prototype to $\mathbf{a}$ with the same class label and $\mathbf{p}_*$ be the closest prototype to $\mathbf{a}$ with the condition that $c(\mathbf{p}^*) \neq c(\mathbf{p}_*)$ and given a dissimilarity d, then:

$$d(\mathbf{a}, \mathbf{p}^*) - d(\mathbf{a}, \mathbf{p}_*) < 0 \tag{3.1}$$

implies that the point $\mathbf{a}$ is correctly classified, else it is assigned to the wrong class. We consider the following setting and use it to derive bounds for the hypothesis margin [11]. Let $d(\mathbf{a}, \mathbf{p})$ be the Euclidean distance and denote $M_h(A, P)$ as the hypothesis margin defined in (3.2) and $M_s(A, P)$ the sample margin defined in (3.1)

Lemma 3.5 ([11]) *Let $\mathbf{a} \in \mathbb{R}^n$ be an input for the LVQ with prototype set $P = \{\mathbf{p_1}, \ldots, \mathbf{p_k}\}$. Let $\mathbf{p}^*$ be the closest prototype to $\mathbf{a}$ with the same label and $\mathbf{p}_*$ be the closest prototype to $\mathbf{a}$ such that $c(\mathbf{p}^*) \neq c(\mathbf{p}_*)$. Then the hypothesis margin $M_h(A, P)$ of P with respect to $\mathbf{a}$ is :*

$$M_h(\{\mathbf{a}\}, P) = \frac{1}{2}\left(d(\mathbf{a}, \mathbf{p}_*) - d(\mathbf{a}, \mathbf{p}^*)\right) \tag{3.2}$$

Lemma 3.6 ([11]) *Let $A = \{\mathbf{a_1}, \ldots, \mathbf{a_m}\}$ be a sample and $P = \mathbf{p_1}, \ldots, \mathbf{p_k}$ be the set of prototype, then:*

$$M_h(A, P) \leq M_s(A, P) \tag{3.3}$$

Lemma (3.5) and (3.6) follows with a theorem that gives a bound for the generalization error considering the number of prototypes, the size of the sample, the margin and the margin error of the training sample [11].

Theorem 3.7 ([11]) *Let $A = \{\mathbf{a_1}, \ldots, \mathbf{a_m}, \mathbf{y_1}, \ldots, \mathbf{y_m}\} \in \{\mathbb{R}^n \times \mathbb{Y}\}^m$ be the training set with given labels drawn from D denoting an underlying data distribution.*

- *for all samples, $||\mathbf{a_i}|| \leq R$ for some constant $R \geq 0$*
- *let P be the set of prototypes with k prototype from each class*

- *let* $\theta \in (0, \frac{1}{2})$
- *let* $M_h^\theta(A,P) = \frac{1}{m} \mid \{i : M_h(\{\mathbf{a_i}\},P)\} < \theta \mid$
- *let* $e_D(P)$ *be the generalization error:* $e_D(P) = Pr_{(\mathbf{a},c(\mathbf{y}))\sim D}[P(\mathbf{a}) \neq y]$
- *let* $\varepsilon > 0$ *, with probability* $1-\varepsilon$ *taking into account the choices fo data:*

for all P*:*

$$e_D \leq M_h^\theta(A,P) + \sqrt{\frac{8}{m}\left(d \log^2 \frac{32m}{\theta^2} + \log \frac{4}{\varepsilon}\right)} \tag{3.4}$$

with

$$d = \min\left(n+1, \frac{64R^2}{\theta^2}\right) 2k^{|Y|} \log ek^2 \tag{3.5}$$

is called Vapnik–Chervonenkis (VC) dimension.We note the following from theorem (3.7): First of all, the bound for the generalization error does not depend on the input dimension (n) directly [11]. Second, while the cardinality of prototypes (P) per given class grows, the (VC) dimension grows. Consequently, using too many prototypes per class may lead to low performance [11]. Hence, we choose a nontrivial optimum number of prototypes. After selecting a desirable number of prototypes, we find a margin threshold θ such that the empirical margin error M_h^θ is small for a large margin threshold θ [11]. However, a large margin threshold θ leads to a high margin error. This contrast can be dealt with by introducing an appropriate loss function [11]. We can use the procedure in [4] to define a loss function for a LVQ that maximizes the hypothesis margin.

One of the primary aims of learning a classifier is to achieve a good generalization, and LVQ that optimizes the hypothesis margin together with a loss function when trained well delivers a good generalization [7]. Amongst the several LVQ variants, GLVQ optimizes the hypothesis margin. LVQ performance drops if an appropriate dissimilarity measure is not employed. Therefore, LVQ variants (GRLVQ, GMLVQ, GTLVQ) that incorporate dissimilarity measures that adapt the GLVQ dissimilarity are also margin optimizers [7, 15]. The question arises if all LVQ variants that optimize the hypothesis margin are also robust against adversarial attacks. In the next section, we analyze LVQ variants that optimize the hypothesis margin and their robustness against adversarial attacks.

3.2 Numerical Stability of LVQ Algorithms

In the introduction, information on adversarial examples is given. The method that gives rise to an adversarial example is known as an adversarial attack. There are two fundamental categories of adversarial attacks: black-box and white-box attacks, as discussed in [29, 32]. In a black-box attack, the adversary lacks information about the model's parameters and has no prior knowledge of the input data [29, 32]. For example, the boundary attack is a notable instance of a black-box attack [8]. However, in a white-box attack, the adversary possesses prior knowledge of both the model's input and its parameters [29, 32]. White-box attacks encompass techniques such as the fast gradient sign method [14] and the momentum iterative method [12]. This chapter considers an adversarial attack characterized by a specified bound $\varepsilon > 0$, ensuring that the attack generates an adversarial example, δ with a maximum perturbation magnitude of $\|\delta\|$. This form of adversarial attack is called an ε-limited adversarial attack [24, 33]. Again, we discuss in this chapter the ability of an NPC to withstand adversarial attacks; thus, the NPC's numerical stability and robustness are presented.

The question of whether all LVQ variants that optimize the hypothesis margin are also robust against adversarial attacks is answered by Saralajew et al. [32]. For a LVQ to be robust against adversarial attacks, the LVQ variant in question should optimize the hypothesis margin in an appropriate space [32]. Therefore, GLVQ and GTLVQ are robust against adversarial attacks while GMLVQ is not, and LVQ variants that maximize the hypothesis margin do not necessarily guarantee robustness [32]. Therefore, robustness and generalization should be studied as two different concepts. Again Saralajew et al. [33] opine that adversaries are related to the analysis of margin and use the semi-norm defined in definition (3.4) to derive bounds for the hypothesis margin similar to equations (3.2) and (3.3).

Theorem 3.8 ([33]) *Let the data space $\mathbb{A}$ be a vector space defined over the field of real or complex numbers, $d(\mathbf{a},\mathbf{p}) = \|\mathbf{a}-\mathbf{p}\|$ be a dissimilarity induced by a semi-norm $\|.\|$ and A be a set of inputs. Then, the hypothesis margin of P with respect to A yields a lower bound on the sample margin of P with respect to A:*

$$M_h(A,P) \leq M_s(A,P) \tag{3.6}$$

Corollary 3.9 ([33]) *Given the labeled data point $(\mathbf{a}, c(\mathbf{a}))$ that is correctly classified by an NPC according to theorem (3.8) and a corresponding perturbation δ that changes the assigned class label, then, the following inequality is true and has tight bounds (existence of data points for which the equality is true)*

$$M_h(\{\mathbf{a}\},P) \leq M_s(\{\mathbf{a}\} \leq \|\delta\|. \tag{3.7}$$

Hein et al. in [47] provide a bound for the adversarial perturbation using the semi-metric given in definition (3.3) as dissimilarity.

Definition 3.10 Let $\mathbf{a} \in \mathbb{R}^n$ be an input for a Nearest Prototype Classifier with prototype set $P = \{\mathbf{p_1}, \ldots, \mathbf{p_k}\}$. Let $\mathbf{p}^*$ be the closest prototype to $\mathbf{a}$ with the same label and $\mathbf{p}_*$ be the closest prototype to $\mathbf{a}$ such that $c(\mathbf{p}^*) \neq c(\mathbf{p}_*)$. The minimal adversarial perturbation $\delta_d(\mathbf{a})$ of $\mathbf{a} \in A$ of a NPC taking the dissimilarity d as a semi-metric is defined as:

$$\delta_d(\mathbf{a}) = \min\{r \mid \max_{a \in B_d(a,r)} (d(\mathbf{a},\mathbf{p}^*) - d(\mathbf{a},\mathbf{p}_*))\} \geq 0 \tag{3.8}$$

where

$$B_d(a,r) = \{\tilde{\mathbf{a}} \in A \mid d(\tilde{\mathbf{a}},\mathbf{a}) \leq r\} \tag{3.9}$$

and in equation (3.8) r is the radius of the smallest ball around $\mathbf{a}$ such that at least a point in $B_d(a,r)$ is classified differently other than its intended class. If $\mathbf{a}$ is not classified correctly, we set $\delta_d(\mathbf{a})$ to 0. Thus

$$d(\mathbf{a},\mathbf{p}^*) - d(\mathbf{a},\mathbf{p}_*) \geq 0 \text{ ; set } \delta_d(\mathbf{a}) = 0 \tag{3.10}$$

Theorem 3.11 ([47]) *Let d be a dissimilarity defined in (3.3), and $\mathbf{a} \in A$ an instance correctly classified by an NPC. Let $\mathbf{p}^*$ be the closest prototype to $\mathbf{a}$ with the same label and $\mathbf{p}_*$ be the closest prototype to $\mathbf{a}$ such that $c(\mathbf{p}^*) \neq c(\mathbf{p}_*)$. Then it holds for the minimal adversarial perturbation $\delta_d(\mathbf{a})$ of $\mathbf{a} \in A$:*

$$\delta_d(\mathbf{a}) \geq \max\{0, \psi\} \tag{3.11}$$

where

$$\psi = \frac{1}{2}(d(\mathbf{a},\mathbf{p}_*) - d(\mathbf{a},\mathbf{p}^*)) \tag{3.12}$$

We reiterate that in equation (3.11) we take the maximum of equation (3.10) or (3.12). Equation (3.12) is the hypothesis margin derived by Crammer et al. [11] and that if we choose the dissimilarity d as the semi-norm defined (3.4) we get the hypothesis margin derived as the lower bound of the sample margin by Saralajew et al. [33].
We define next the minimal adversarial perturbation given the dissimilarity $l_q \in \{1,2,\infty\}$ for measuring the perturbation and a corresponding dissimilarity $l_t \in \{1,2,\infty\}$ for the NPC [47].

Let I be the index set of prototypes P and I_y be the index set of prototypes $(\mathbf{p_i})$ that correctly assigns a given instance to a class and I_y^c be the index set of prototypes $(\mathbf{p_j})$ such that:

$$I = I_y \cup I_y^c \text{ and } I_y \cap I_y^c = \emptyset$$

Again, we let $\mathbf{p}^*$ be the closest prototype to $\mathbf{a}$ with the same label and $\mathbf{p}_*$ be the closest prototype to $\mathbf{a}$ such that $c(\mathbf{p}^*) \neq c(\mathbf{p}_*)$.

Definition 3.12 The minimal adversarial perturbation $\delta_t^q(\mathbf{a})$ of $\tilde{\mathbf{a}} \in A$ of a Nearest Prototype Classifier taking the dissimilarity l_t- metric for measuring the NPC and an l_q-metric as a dissimilarity for measuring the perturbation where $t,q \in \{1,2,\infty\}$ is defined as:

$$\delta_t^q(\mathbf{a})_j = \min_{r\in\mathbb{R},\tilde{\mathbf{a}}\in A} r \tag{3.13}$$

$$\text{with the constraint that } d_t(\tilde{a},\mathbf{p_i}) - d_t(\tilde{\mathbf{a}},\mathbf{p_j}) \geq 0: \ \tilde{\mathbf{a}} \in B_q(a,r) \tag{3.14}$$

$$\text{If } d_t(\tilde{\mathbf{a}},\mathbf{p}^*) - d_t(\tilde{\mathbf{a}},\mathbf{p}_*) > 0 \text{ ; set } \delta_t^q(\mathbf{a}) = 0 \tag{3.15}$$

The following theorem states how to compute $\delta_t^q(\mathbf{a})$

Theorem 3.13 ([47]) *Let $\mathbf{a} \in A \subset \mathbb{R}^n$ be an input and denote I_y the index set of prototypes that correctly assign a given instance to a class and I_y^c be its complement.Then define for every $j \in I_y^c$:*

$$\delta_t^q(\mathbf{a}) = \min_{j\in I_y^c} r_t^q(\mathbf{a})_j \tag{3.16}$$

where

$$r_t^q(\mathbf{a})_j = \min_{\tilde{\mathbf{a}}\in\mathbb{R}^n} d_q(\tilde{\mathbf{a}},\mathbf{a}) \tag{3.17}$$

subject to:

$$d_t(\tilde{\mathbf{a}},\mathbf{p_i}) - d_t(\tilde{\mathbf{a}},\mathbf{p_j}) \geq 0 \textit{ for all } i \in I_y; \ \tilde{\mathbf{a}} \in A$$

Lower bound is derived for $\delta_t^q(\mathbf{a})$ by relaxing the optimization problem in (3.17) to get $\rho_t^q(\mathbf{a})$. Let $i \in I_y$ and $j \in I_y^c$, we the compute:

$$\rho_t^q(\mathbf{a})_{i,j} = \min_{\tilde{\mathbf{a}}\in\mathbb{R}^n} d_q(\tilde{\mathbf{a}},\mathbf{a}) \tag{3.18}$$

with constraint:

$$d_t(\tilde{a},\mathbf{p_i}) - d_t(\tilde{a},\mathbf{p_j}) \geq 0; \ \tilde{a} \in A \tag{3.19}$$

The next theorem (3.14) shows for a given $A \in [0,1]^n$ the time complexity required to optimise $r_t^q(\mathbf{a})_j$.

Theorem 3.14 ([47]) *The time complexity for computing $r_t^q(\mathbf{a})_j$ as an optimization problem in (3.17) given that $t,q \in \{1,2,\infty\}$ for a given $A \in [0,1]^n$ is summarize in table (3.1)*

		l_q- threat model		
		l_1	l_2	l_∞
l_p - distance	l_1	NP-hard	NP-hard	Polynomial
	l_2	Polynomial	Polynomial	Polynomial
	l_∞	NP-hard	NP-hard	NP-hard

Table (3.1) Time complexity of $r_t^q(\mathbf{a})$ and $\delta_t^q(\mathbf{a})$

It is clear from table (3.1) that the computation and optimization problem in (3.17) is difficult to compute for l_∞-NPC for all threat models (l_1, l_2, l_∞). The l_2-NPC is certified for all threat models (l_1, l_2, l_∞) and l_1-NPC are certified for l_∞-thread model.

Theorem (3.15) shows for a given $A = \mathbb{R}^n$ the time complexity required to optimise $\rho_t^q(\mathbf{a})_{i,j}$.

Theorem 3.15 ([47]) *The time complexity for computing $\rho_t^q(\mathbf{a})_{i,j}$ as an optimization problem in (3.18) given that $t, q \in \{1, 2, \infty\}$ for a given $A \in \mathbb{R}^n$ is summarize in table (3.2)*

		l_q- threat model		
		l_1	l_2	l_∞
l_p - distance	l_1	NP-hard	NP-hard	$\mathcal{O}(d\log d)$
	l_2	$\Theta(d)$	$\Theta(d)$	$\Theta(d)$
	l_∞	$\Theta(d)$	$\mathcal{O}(d\log d)$	$\Theta(d)$

Table (3.2) Time complexity of $\rho_t^q(\mathbf{a})_{i,j}$

Table (3.2) depict that, l_2 and l_∞-NPC can be certified for all threat model (l_1, l_2, l_∞) and l_1-NPC can be certified for (l_∞) threat model. It follows from theorems (3.14) and (3.15) that we can compute a lower bound for $\delta_t^q(\mathbf{a})$.

Lemma 3.16 ([47]) *It holds that*

$$\min_{j \in I_y^c} \max_{i \in I_y} \rho_t^q(\mathbf{a})_{i,j} \leq \delta_t^q(\mathbf{a})$$

Again, let $(i^*, j^*) \in I_y \times I_y^c$ be the prototype pair that gives the lower bound and let $\tilde{a}^*$ be the minimizer of $\rho_t^q(\mathbf{a})_{i,j}$. Then if $\tilde{a}^*$ satisfies

$$d_t(\tilde{\mathbf{a}^*}, \mathbf{p_i}) - d_t(\tilde{\mathbf{a}^*}, \mathbf{p_j}) \geq 0 \text{ for all } i \in I_y$$

$$\text{then } \min_{j \in I_y^c} \max_{i \in I_y} \rho_t^q(\mathbf{a})_{i,j} = \delta_t^q(\mathbf{a})$$

We finally state the bound for $\delta_t^t(\mathbf{a})$ in theorem (3.17) given by Hein et al. [47].

Theorem 3.17 ([47]) *The lower bound on $\delta_t^t(\mathbf{a})$ of lemma (3.16) is at least as good as the one of theorem (3.11). That is:*

$$\max\{0,\psi\} \leq \min_{j\in I_y^c} \rho_t^t(\mathbf{a})_{i^*,j} \leq \min_{j\in I_y^c} \max_{i\in I_y} \rho_t^t(\mathbf{a})_{i,j}$$

where

$$\psi = \frac{1}{2}\left(d(\mathbf{a},\mathbf{p}_*) - d(\mathbf{a},\mathbf{p}^*)\right)$$

and

$$i^* = arg\min_{i\in I_y} d_t(\mathbf{a},\mathbf{p_i}) \tag{3.20}$$

3.3 Certification of a NPC

In [33], Saralajew et al. certify a given NPC by optimizing a signed version of the hypothesis margin, termed the signed hypothesis margin.

Definition 3.18 (Signed hypothesis margin): Given a labeled instance $(\mathbf{a},c(\mathbf{a}))$ and prototype set P. The signed hypothesis margin of an instance is given by:

$$M_h^c(\mathbf{a},c(\mathbf{a}),P)) = \begin{cases} M_h(\{\mathbf{a}\},P) & \text{if } \mathbf{a} \text{ is correctly classified} \\ -M_h(\{\mathbf{a}\},P) & \text{if } \mathbf{a} \text{ is wrongly classified} \end{cases} \tag{3.21}$$

An absolute distance difference $\Delta d(\mathbf{a})$ is defined as:

$$\Delta d(\mathbf{a}) = d\left(\mathbf{a},\mathbf{p}^-\right) - d\left(\mathbf{a},\mathbf{p}^+\right) \tag{3.22}$$

where $\mathbf{p}^+$ and $\mathbf{p}^-$ denote the optimum matching prototype vector belonging to the correct class, respectively, the optimum matching prototype vector belonging to the incorrect class as defined in (2.5). From the definition of the hypothesis margin in [33], we have:

$$\Delta d(\mathbf{a}) = d\left(\mathbf{a},\mathbf{p}^-\right) - d\left(\mathbf{a},\mathbf{p}^+\right) \leq d(\mathbf{a},\mathbf{p}_*) - d(\mathbf{a},\mathbf{p}^*)$$

$$\Delta d(\mathbf{a}) \leq 2.M_h^c(\mathbf{a},c(\mathbf{a}),P)) \tag{3.23}$$

and equality holds if

$$\mathbf{p}^+ = \mathbf{p}^* \text{ or } \mathbf{p}^+ = \mathbf{p}_*$$

Let A be labeled training set and we denote $\#A$ as the cardinality of A. Let also P denote a set of prototypes with labels. From (3.23), an upper bound exists for the robust test error under ε-limited adversarial attack and is expressed as [33]:

$$\text{error}_{\varepsilon}(A,P) = \frac{1}{\#A} \cdot \#\{(\mathbf{a}, c(\mathbf{a})) \in A \mid \Delta d(\mathbf{a}) \leq 2\varepsilon\} \tag{3.24}$$

Hence, maximizing $\Delta d(\mathbf{a})$ or minimizing $-\Delta d(\mathbf{a})$ leads to maximizing the adversarial robustness and $-\Delta d(\mathbf{a})$ is termed as the *triplet loss* [33]. For adversarial attacks not up to ε-limited attack, optimize:

$$\frac{1}{\#A} \sum_{(\mathbf{a},c(\mathbf{a}))\in A} \text{ReLU}(2\varepsilon - \Delta d(\mathbf{a})) \tag{3.25}$$

To avoid the computation of square roots, one might employ the squared semi-norms computed as:

$$d^2\left(\mathbf{a},\mathbf{p}^+\right) - d^2\left(\mathbf{a},\mathbf{p}^-\right) = -\Delta d(\mathbf{a}) \cdot \left(d\left(\mathbf{a},\mathbf{p}^+\right) + d\left(\mathbf{a},\mathbf{p}^-\right)\right) \tag{3.26}$$

Hence, a NPC becomes robust against adversarial attacks if optimization uses the triplet loss [33]. However, optimizing a NPC using the triplet loss may cause training instability. To curb this problem of instability in training, employ normalization techniques as in (2.5) [33].

Hein et al. [47] certify a NPC using algorithm (1). Comparing theorems (3.11) and (3.17), both have the bound on $\delta_t^q(\mathbf{a})$ existing for $t = q$ and the same computational complexity of $\mid I_y \mid + \mid I_y^c \mid$ operations [47]. A lower bound with a considerably low computational cost exists if the box constraint ($A = [0,1]^n$) is incorporated [47]. The lower bound is realized by first certifying $A = \mathbb{R}^n$ which has a closed form, and thus the lower bound for $A = \mathbb{R}^n$ also satisfies that of the restricted case $A = [0,1]^n$ [47]. Thus:

$$s_j = \rho_t^q(\mathbf{a})_{i^*,j} \;\text{ where } i^* \text{ is fixed} \tag{3.27}$$

$$\text{Let } (\lambda, j^*) = \min_{j\in I_y^c} \rho_t^q(\mathbf{a})_{i^*,j} \text{ be the minimum and minimizer} \tag{3.28}$$

$$\text{Let } \kappa_t^q(\mathbf{a})_{i^*,j} \text{ be the corresponding element given that } A = [0,1]^n \text{ but not } A = \mathbb{R}^n \tag{3.29}$$

$$\text{Solve for } \kappa_t^q(\mathbf{a})_{i^*,j} \text{ only if } \kappa_t^q(\mathbf{a})_{i^*,j^*} \geq \rho_t^q(\mathbf{a})_{i^*,j} \tag{3.30}$$

The function to optimize is:

$$\max_{(P_i)_{i\in I}} \frac{1}{k} \sum_{r=1}^{k} \min\left\{ \min_{j\in I_y^c} \max_{i\in I_y} \rho_t^q(\mathbf{a}_r)_{i,j},\ \mathscr{R} \right\} \tag{3.31}$$

where

$$\rho_t^q(\mathbf{a})_{i,j} = \min_{\tilde{\mathbf{a}}\in\mathbb{R}^n} d_q(\tilde{\mathbf{a}},\mathbf{a})$$

with constraint:

$$d_t(\tilde{a}, \mathbf{p_i}) - d_t(\tilde{a}, \mathbf{p_j}) \geq 0;\ \tilde{a} \in A$$

and $\mathscr{R}$ denotes the cap on the margin to be enforced.

Algorithm 1 [47] Certification algorithm for correctly classified given point $\mathbf{a}$

```
1: **Compute:** $\lambda$ as lower bound on $\delta_t^q(\mathbf{a})$ using equation $(3.16)$
2: **Compute:** $i^*$, $s_j$ using equations $(3.20)$ and $(3.27)$ respectively.
3: **Determine:** $(\lambda, j^*)$ using equation $(3.28)$
4: **if** minimizer $\tilde{a}^*$of $\rho_t^q(\mathbf{a})_{i^*,j^*}$ is feasible for $r_t^q(\mathbf{a})_{j^*}$ **then**
5:    $\delta_t^q(\mathbf{a}) = \lambda$ and return
6: **else**
7:    $\lambda$ is lower bound on $\delta_t^q(\mathbf{a})$
8: **end if**
9: **Compute:** $\delta_t^q(\mathbf{a})$
10: **Initialize:** Default parameters $(t = 2)$ or $(t, q) = (1, \infty)$
11: **compute:** $\mu = r_t^q(\mathbf{a})_{j^*}$ and check if $\delta_t^q(\mathbf{a}) \leq \mu$
12: **for** $j = 1$ **to** $\mid I_y^c \mid$ **do**
13:    **if** $s_j \leq \mu$ **then**
14:       compute $r_t^q(\mathbf{a})_j$
15:       **if** $r_t^q(\mathbf{a})_j < \mu$ **then**
16:          $\mu = r_t^q(\mathbf{a})_j$
17:       **end if**
18:    **end if**
19:    $\delta_t^q(\mathbf{a}) = \mu$
20: **end for**
```

4 Experimental Results

4.1 Dataset

This chapter presents experiments to certify a given NPC based on the bounds presented in [33] and [47]. We conducted experiments using the MNIST dataset and then the CIFAR-10 dataset. The MNIST dataset [22] is a real image dataset consisting of 70000 handwritten digits with 28×28 gray-scale pixels grouped into ten classes from 0 – 9. The MNIST dataset has 10000 instances for testing and the rest for training.

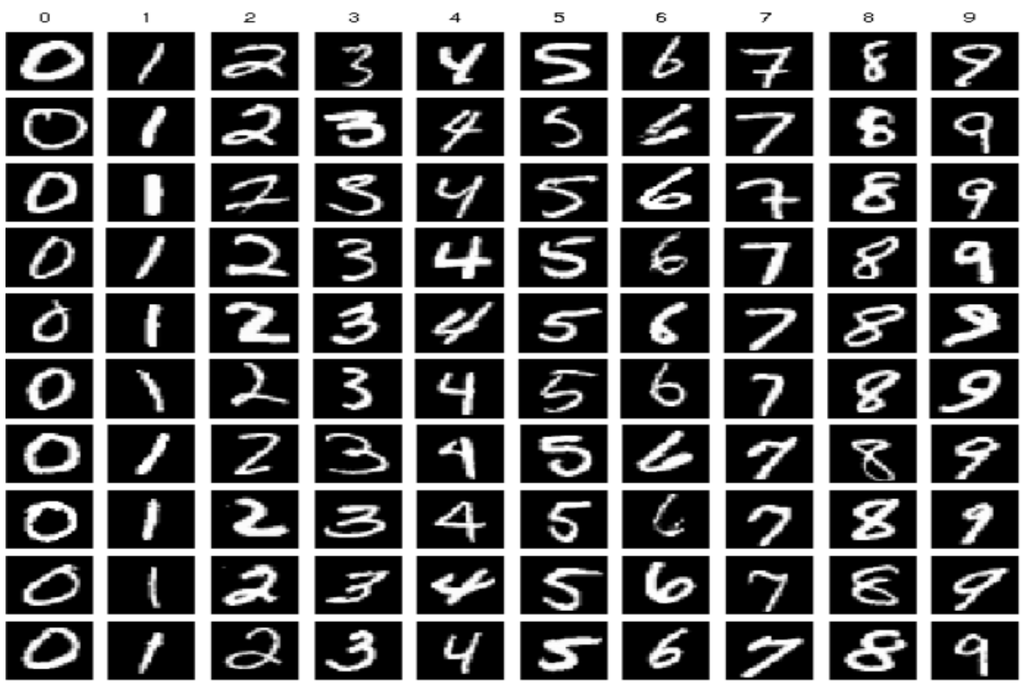

Figure (4.1) [23] MNIST dataset images sample

The CIFAR-10 dataset is a thirty-two by thirty-two (32×32) color image dataset consisting of 60000 samples, out of which 10000 instances are for testing and the rest for training. The dataset has ten classes: cat, airplane, frog, automobile, deer, ship, dog, truck, bird and horse, with each class consisting of 6000 images [21].

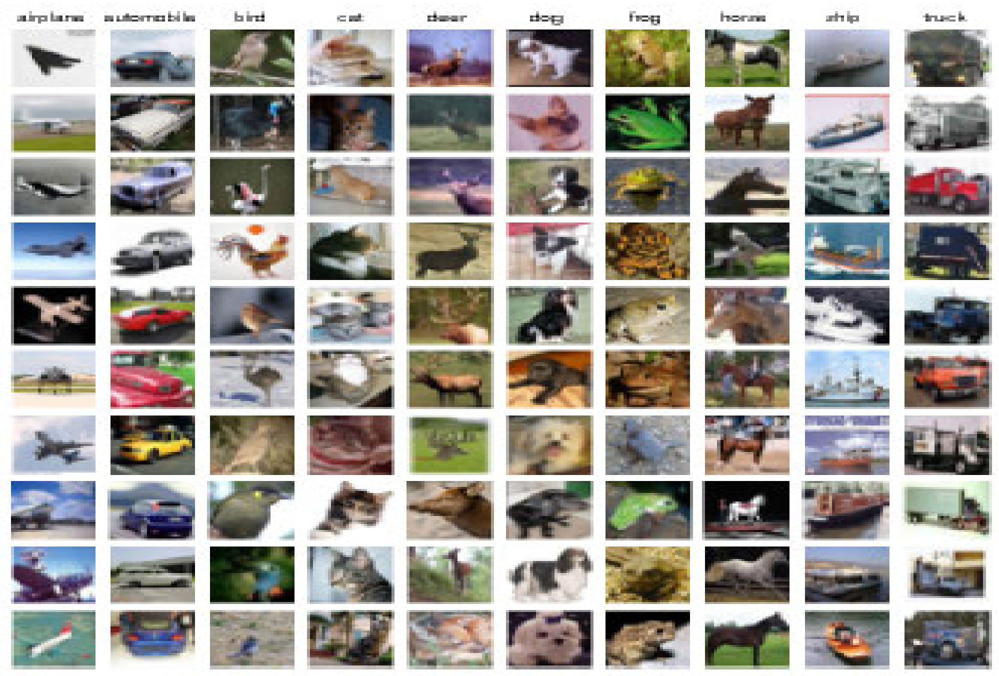

Figure (4.2) [23] CIFAR-10 dataset samples images

4.2 Experimentation to Certify a NPC

In this thesis, we denote NPCs trained by the triplet loss [33] as FNPC and the NPCs trained in [47] as PNPC. The experiments to get the exact certified values for FNPC and PNPC require special hyperparameter tuning. To get the reported values in the respective papers ([33] and [47]), one might run the exact replicate flags from the GitHub repositories provided by [33] and [47]. We used the same codes from the respective GitHub repositories for the simulation in this thesis. Different hyperparameters are chosen in the experiments to ascertain the behavior of both bounds given by FNPC and PNPC and report accordingly.

4.2.1 Certification of FNPC using the MNIST dataset

Emphasizing, an upper bound exists for the robust test error under ε-limited adversarial attacks for FNPC and is computed using the equation (3.24). To avoid the computation of square roots, one might employ the squared seminorms using the equation (3.25).

Certification of FNPC requires foolbox for implementation. The prototype initialization is done using the k-means algorithm. 128 prototypes per class were selected for training. Batch size of 128 and 1000 epochs with 0.001 learning rates were chosen for FNPC-GLVQ and FNPC-GTLVQ to train and certify the MNIST dataset. Lower Bound Robustness Test Accuracy (LRTA) and Upper Bound Robustness Test Accuracy (URTA) values for FNPC are simulated and the summary of the results are tabulated in (4.1) for FNPC-GLVQ using the GLVQ loss and FNPC-GTLVQ using the ReLU loss.

Dataset	ε	Model	Loss	LRTA [%]	URTA [%]
MNIST	0.001	GLVQ	GLVQ	93.61	94.29
		GTLVQ	ReLU	93.26	93.98
	0.01	GLVQ	GLVQ	83.90	94.08
		GTLVQ	ReLU	83.79	93.41
	0.1	GLVQ	GLVQ	0.00	91.68
		GTLVQ	ReLU	0.001	90.74
	0.3	GLVQ	GLVQ	0.00	82.78
		GTLVQ	ReLU	0.00	83.25
	0.5	GLVQ	GLVQ	0.00	69.44
		GTLVQ	ReLU	0.00	72.10

Table (4.1) FNPC-GLVQ and FNPC-GTLVQ trained with l_2 norm on the MNIST dataset

Figure (4.3) shows FNPC-GTLVQ model trained with l_2 norm on the MNIST dataset.

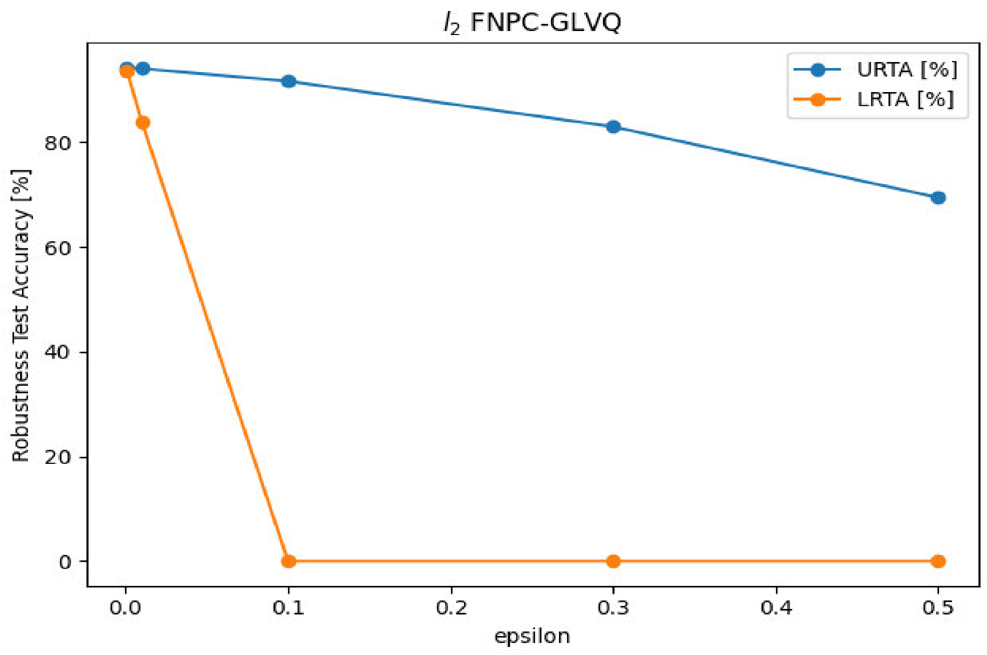

Figure (4.3) FNPC-GLVQ model trained with l_2 norm on the MNIST dataset

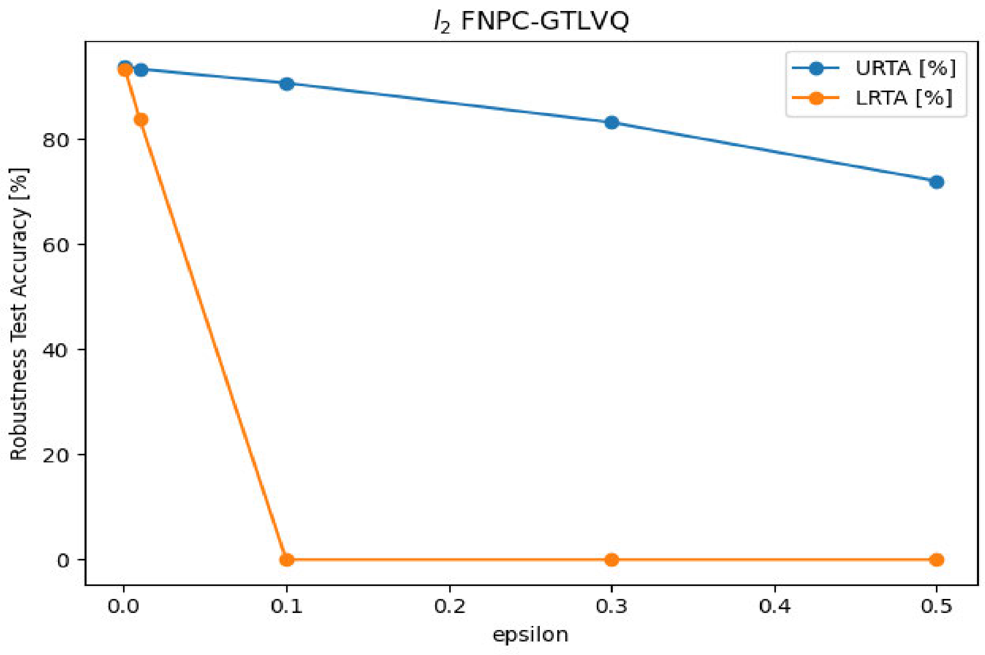

Figure (4.4) FNPC-GTLVQ model trained with l_2 norm on the MNIST dataset

Experimental results of Upper Bound Robustness Test Accuracy (URTA) and Lower Bound Robustness Test Accuracy (LRTA) values are populated in table (4.1). The values obtained from URTA and LRTA are graphed for FNPC-GLVQ and FNPC-GTLVQ in figures (4.3) respectively (4.4). It is evident that as perturbation increases, accuracy reduces for both LRTA and URTA until they stabilize or reach zero. URTA gives a better robustness accuracy than the LRTA. FNPC-GTLVQ provides a better robustness accuracy compared to the FNPC-GLVQ classifier.

Table (4.2) contains values for FNPC-GLVQ and FNPC-GTLVQ trained with l_∞ norm on the MNIST dataset.

Dataset	ε	Model	Loss	LRTA [%]	URTA [%]
MNIST	0.001	GLVQ	GLVQ	86.97	86.97
		GTLVQ	ReLU	94.34	94.34
	0.01	GLVQ	GLVQ	76.75	76.75
		GTLVQ	ReLU	93.82	93.82
	0.1	GLVQ	GLVQ	10.65	10.65
		GTLVQ	ReLU	93.60	93.60
	0.3	GLVQ	GLVQ	0.00	0.00
		GTLVQ	ReLU	93.51	93.51
	0.5	GLVQ	GLVQ	0.00	0.00
		GTLVQ	ReLU	92.52	92.52

Table (4.2) FNPC-GLVQ and FNPC-GTLVQ trained with l_∞ norm on the MNIST dataset

Similar experiments are performed for FNPC trained with the l_∞ norm and certified robustness values for URTA and LRTA are herein tabulated in (4.2). Outcome from table (4.2) shows a similar pattern as in Table (4.1). However, the FNPC-GTLVQ model is resilient and provides good accuracy results even as we increase the perturbation. Notably, the URTA and LRTA coincide for FNPC-GLVQ and FNPC-GTLVQ models.

4.2.2 Certification of PNPC using MNIST dataset

Experimentation to test PNPC requires torchvision, perceptual-advex, jax and jaxlib. Batch size of 128 and 10 epochs with 128 prototypes per class were chosen to train and certify the MNIST dataset using the PNPC. Table (4.3) shows the experiment's outcome.

Dataset	ε	Model	LRTA [%]	URTA [%]
MNIST	0.001	l_2 PNPC	96.16	96.17
		l_∞ PNPC	26.48	26.49
	0.01	l_2 PNPC	96.05	96.18
		l_∞ PNPC	15.44	15.49
	0.1	l_2 PNPC	95.91	96.93
		l_∞ PNPC	28.13	28.67
	0.3	l_2 PNPC	93.40	97.25
		l_∞ PNPC	9.97	10.59
	0.5	l_2 PNPC	90.29	97.64
		l_∞ PNPC	17.53	19.99

Table (4.3) PNPC trained with l_2 and l_∞ metric for certified robustness accuracy on the MNIST dataset

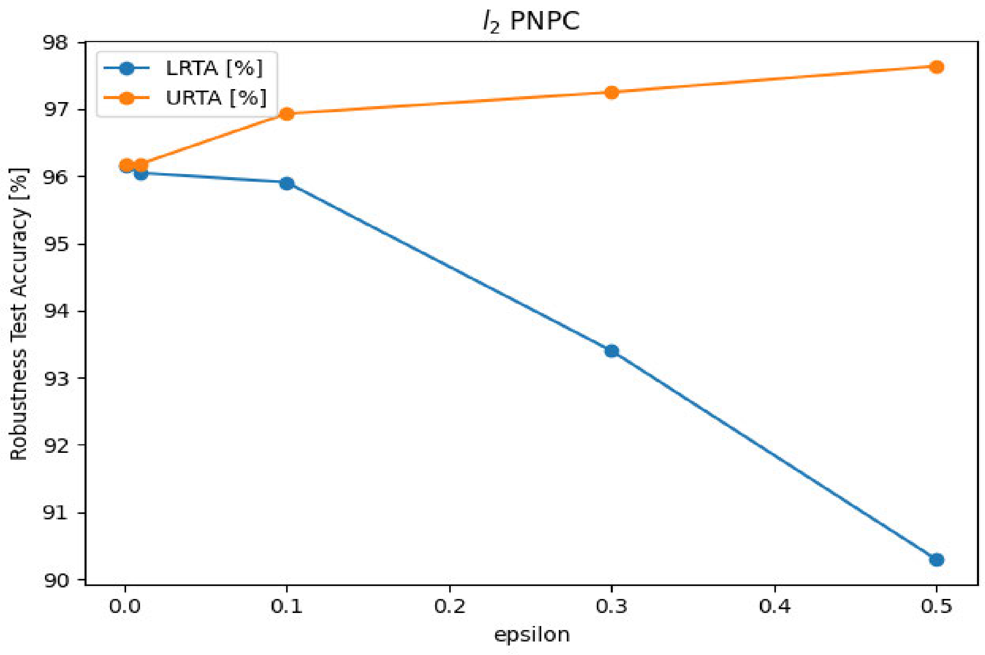

Figure (4.5) PNPC trained with l_2 metric on the MNIST dataset

Table (4.3) shows high robustness values for both LRTA and URTA and figure (4.5) shows the visualization. As the perturbation (ε) increases, the URTA values increase while the LRTA values decrease. The result in table (4.3) for PNPC is therefore in contrast to FNPC values obtained in table (4.1), figure (4.3) whose values for both LRTA and URTA begins with relatively high robustness values and both values reduces to zero or stabilizers. Values obtained from PNPC trained with the l_∞ norm from table (4.3) depict values for both LRTA and URTA almost coinciding in contrast to those obtained for FNPC in table (4.2) whose values for both LRTA and URTA coincided. In conclusion, PNPC has a larger radius for certification than FNPC.

4.2.3 Certification of NPC using the CIFAR-10 dataset

Experiments are carried out to certify the adversarial robustness of FNPC and PNPC using the CIFAR-10 dataset. For FNPC-GLVQ, one prototype per class with a 0.001 learning rate and 100 epochs are chosen. Also, batch size of 128 and the GLVQ loss were employed in the training.

The same parameters used to certify the FNPC-GLVQ were used to certify FNPC-GTLVQ, except for the number of epochs, which were 10 instead of 100, and ReLU loss used instead of GLVQ loss.

The experiment to certify PNPC was performed with 64 prototypes per class, 10 epochs, and batch size of 128. The outcome of the results are presented in table (4.4) ‘

Dataset	ε	Model	LRTA [%]	URTA [%]
CIFAR-10	$\frac{2}{255}$	l_2 PNPC	24.87	25.02
		l_2 FNPC-GLVQ	32.49	32.80
		l_2 FNPC-GTLVQ	39.06	39.58
		l_∞ PNPC	10.00	10.00
		l_∞ FNPC-GLVQ	21.05	21.05
		l_∞ FNPC-GTLVQ	37.34	37.34
	$\frac{8}{255}$	l_2 PNPC	10.00	10.00
		l_2 FNPC-GLVQ	31.35	32.69
		l_2 FNPC-GTLVQ	39.37	41.32
		l_∞ PNPC	10.00	10.00
		l_∞ FNPC-GLVQ	21.42	21.42
		l_∞ FNPC-GTLVQ	38.48	38.48
	$\frac{36}{255}$	l_2 PNPC	10.00	10.00
		l_2 FNPC-GLVQ	26.00	32.61
		l_2 FNPC-GTLVQ	32.43	40.34
		l_∞ PNPC	10.00	10.00
		l_∞ FNPC-GLVQ	20.36	20.36
		l_∞ FNPC-GTLVQ	38.87	38.87

Table (4.4) NPCs trained with l_2 and l_∞ dissimilarities for certified robustness accuracy on the CIFAR-10 dataset

The observations from table (4.4) show that l_2 FNPC-GLVQ LRTA and URTA decrease as ε values increase, but the rate of decay is slow. The l_∞ FNPC-GLVQ LRTA and URTA values are equal for a given ε and thus lie on each other. The l_2 FNPC-GTLVQ LRTA and URTA values slightly increased and decreased as ε values increased. Again, the l_∞ FNPC-GTLVQ LRTA and URTA attained equal values as ε increases. The remark is that there is a consistent pattern and behavior in both LRTA and URTA values when FNPCs are certified using both the MNIST and CIFAR-10 datasets.

The certification of l_2 PNPC using the CIFAR-10 dataset values from table (4.4) shows that as ε is increased, the LRTA values decrease and the URTA values gained a minor increase and decreased afterward. The l_∞ values for both LRTA and URTA are equal to a constant. The remark is that the behavior obtained for PNPC is equivalent to that of the FNPC while certifying using the CIFAR-10 dataset.

5 Discussion of Results, Conclusions and Suggestions

This chapter examines the findings from the experiments conducted in chapter 4. Additionally, we provide an overview of the conclusions drawn from these findings and offer relevant suggestions for future directions.

5.1 Discussion of Results

It is a fact that (semi-)norms induce (semi-)metrics and (semi-)norms have more restrictions than (semi-)metrics. Hein et al. use semi-metric in PNPC and Saralajew et al. employ a semi-norm in FNPC. The bound provided by PNPC for certification of adversarial robustness using the semi-metric might be larger and more general than the bound presented by Saralajew et al. in FNPC for adversarial robustness using the semi-norm. The values in tables (4.1), (4.2) and (4.3) confirm that PNPC provides a larger bound than FNPC when both PNPC and FNPC were experimented with, using the MNIST dataset.

What about the robustness of FNPC and PNPC against adversarial attacks? Every instance certified by FNPC is expected to be certified by PNPC because of the expected larger margin given by PNPC due to the use of semi-metric. However, the converse may not necessarily be true and it is expected that FNPC yields higher robustness against adversarial attacks than PNPC for numerical stability.

Values obtained for FNPC-GLVQ trained with the l_∞ norm on the MNIST dataset from table (4.2) were low. Similar low outcomes were reported for PNPC values presented in table (4.3) when trained with the l_∞ norm. Again, NPCs trained with l_2 and l_∞ norms for certified robustness accuracy on the CIFAR-10 dataset yielded low values. The reason for the low performance may be due to hyperparameter tuning. For example, the number of prototypes selected per class in this thesis were considerably lower than those selected in FNPC and PNPC for training and certification for adversarial robustness. However, the values obtained in this thesis are consistent with those obtained in FNPC and PNPC, not necessarily in the performance but in behavior. Optimal performance requires special hyperparameter tuning, as FNPC and PNPC publications reported.

5.2 Conclusions and Suggestions

This thesis compared NPCs learning dynamic and numerical stability regarding the Crammer-normalization and the Hein-normalization for adversarial robustness. The bounds given by Saralajew et al. for adversarial robustness were discussed. Hein et al. confirmed the bounds given by Saralajew et al. and further gave new bounds in their PNPC, incorporating the box constraint in their work. Whereas Saralajew et al. optimized the triplet loss and normalized at every training, Hein et al. enforced an upper bound in their objective function to prevent individual training instances from attaining extremely high margins. Experimental outcomes from the use of the MNIST and the CIFAR-10 real datasets indicate that even though the Hein normalization in PNPC offers a larger bound and a more generalization of robustness certification for NPCs, Saralajew et al. FNPC is more robust against adversarial attacks than PNPC due to the restrictions introduced by the use of semi-norm as compared to the semi-metric utilized in PNPC by Hein et al. Extensive work should be geared towards this area since it offers applications in sensitive fields. Future work will look into adversarial robustness for PNPC and FNPC bounds when both are trained and certified using the same dissimilarity measure (either semi-norm or semi-metric). Again we suggest that the adversarial robustness of soft NPCs could be explored.

6 Appendix

The code implementation of Provably Adversarially Robust Nearest Prototype Classifiers referred to in this thesis as PNPC can be found at `https://github.com/asirifiboa/Provably-Adversarially-Robust-Nearest-Prototype-Classifiers` and that of Fast Adversarial Robustness Certification of Nearest Prototype Classifiers for Arbitrary Seminorms also referred to FNPC in this thesis can be found at `https://github.com/asirifiboa/robust_NPCs`

Bibliography

[1] Muhammad Aurangzeb Ahmad, Carly Eckert, and Ankur Teredesai. Interpretable machine learning in healthcare. In *Proceedings of the 2018 ACM international conference on bioinformatics, computational biology, and health informatics*, pages 559–560, 2018.

[2] Edward Allen. *Modeling with Itô stochastic differential equations*, volume 22. Springer Science & Business Media, 2007.

[3] Sheldon Axler. *Linear algebra done right*. Springer Science & Business Media, 1997.

[4] Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

[5] James C Bezdek and Ludmila I Kuncheva. Nearest prototype classifier designs: An experimental study. *International journal of Intelligent systems*, 16(12):1445–1473, 2001.

[6] Michael Biehl. Matrix learning in learning vector quantization. 2006.

[7] Michael Biehl, Anarta Ghosh, and Barbara Hammer. Dynamics and generalization ability of lvq algorithms. *Journal of Machine Learning Research*, 8(2), 2007.

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

[9] Kerstin Bunte, Petra Schneider, Barbara Hammer, Frank-Michael Schleif, Thomas Villmann, and Michael Biehl. Limited rank matrix learning, discriminative dimension reduction and visualization. *Neural Networks*, 26:159–173, 2012.

[10] Nadia Burkart and Marco F Huber. A survey on the explainability of supervised machine learning. *Journal of Artificial Intelligence Research*, 70:245–317, 2021.

[11] Koby Crammer, Ran Gilad-Bachrach, Amir Navot, and Naftali Tishby. Margin analysis of the lvq algorithm. *Advances in neural information processing systems*, 15, 2002.

[12] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and

Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018.

[13] James H Fetzer and James H Fetzer. *What is Artificial Intelligence?* Springer, 1990.

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

[15] Barbara Hammer, Marc Strickert, and Thomas Villmann. On the generalization ability of grlvq networks. *Neural Processing Letters*, 21:109–120, 2005.

[16] Barbara Hammer and Thomas Villmann. Generalized relevance learning vector quantization. *Neural Networks*, 15(8-9):1059–1068, 2002.

[17] Marika Kaden, Mandy Lange, David Nebel, Martin Riedel, Tina Geweniger, and Thomas Villmann. Aspects in classification learning-review of recent developments in learning vector quantization. *Foundations of Computing and Decision Sciences*, 39(2):79–105, 2014.

[18] Marika Kaden, Martin Riedel, Wieland Hermann, and Thomas Villmann. Border-sensitive learning in generalized learning vector quantization: an alternative to support vector machines. *Soft Computing*, 19:2423–2434, 2015.

[19] T. Kohonen. Improved versions of learning vector quantization. In *1990 IJCNN International Joint Conference on Neural Networks*, pages 545–550 vol.1, 1990.

[20] T Kohonen. Self-organizing maps, springer. 3th. 2001.

[21] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[22] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[23] Seung-Hwan Lim, Steven R Young, and Robert M Patton. An analysis of image storage systems for scalable training of deep neural networks. *system*, 5(7):11, 2016.

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

[25] Thomas M Moerland, Joost Broekens, and Catholijn M Jonker. A framework for reinforcement learning and planning. *arXiv preprint arXiv:2006.15009*, 127, 2020.

[26] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.

[27] Christoph Molnar. *Interpretable machine learning*. Lulu. com, 2020.

[28] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

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

[30] Jensun Ravichandran, Marika Kaden, Sascha Saralajew, and Thomas Villmann. Variants of dropconnect in learning vector quantization networks for evaluation of classification stability. *Neurocomputing*, 403:121–132, 2020.

[31] Sascha Saralajew. New prototype concepts in classification learning. 2020.

[32] Sascha Saralajew, Lars Holdijk, Maike Rees, and Thomas Villmann. Robustness of generalized learning vector quantization models against adversarial attacks. In *Advances in Self-Organizing Maps, Learning Vector Quantization, Clustering and Data Visualization: Proceedings of the 13th International Workshop, WSOM+ 2019, Barcelona, Spain, June 26-28, 2019 13*, pages 189–199. Springer, 2020.

[33] Sascha Saralajew, Lars Holdijk, and Thomas Villmann. Fast adversarial robustness certification of nearest prototype classifiers for arbitrary seminorms. *Advances in Neural Information Processing Systems*, 33:13635–13650, 2020.

[34] Sascha Saralajew and Thomas Villmann. Adaptive tangent distances in generalized learning vector quantization for transformation and distortion invariant classification learning. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 2672–2679. IEEE, 2016.

[35] Atsushi Sato and Keiji Yamada. Generalized learning vector quantization. *Advances in neural information processing systems*, 8, 1995.

[36] Petra Schneider, Michael Biehl, and Barbara Hammer. Adaptive relevance matrices in learning vector quantization. *Neural computation*, 21(12):3532–3561, 2009.

[37] Wilko Schwarting, Javier Alonso-Mora, and Daniela Rus. Planning and decision-

making for autonomous vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:187–210, 2018.

[38] Patrice Simard, Yann LeCun, and John Denker. Efficient pattern recognition using a new transformation distance. *Advances in neural information processing systems*, 5, 1992.

[39] Bo Sun, Junping Du, and Tian Gao. Study on the improvement of k-nearest-neighbor algorithm. In *2009 International Conference on Artificial Intelligence and Computational Intelligence*, volume 4, pages 390–393, 2009.

[40] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

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

[42] Remi Tachet, Mohammad Pezeshki, Samira Shabanian, Aaron Courville, and Yoshua Bengio. On the learning dynamics of deep neural networks. *arXiv preprint arXiv:1809.06848*, 2018.

[43] Kashvi Taunk, Sanjukta De, Srishti Verma, and Aleena Swetapadma. A brief review of nearest neighbor algorithm for learning and classification. In *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, pages 1255–1260, 2019.

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[45] Thomas Villmann, Andrea Bohnsack, and Marika Kaden. Can learning vector quantization be an alternative to svm and deep learning?-recent trends and advanced variants of learning vector quantization for classification learning. *Journal of Artificial Intelligence and Soft Computing Research*, 7(1):65–81, 2017.

[46] Thomas Villmann, Andrea Bohnsack, and Marika Kaden. Can learning vector quantization be an alternative to svm and deep learning?-recent trends and advanced variants of learning vector quantization for classification learning. *Journal of Artificial Intelligence and Soft Computing Research*, 7(1):65–81, 2017.

[47] Václav Voráček and Matthias Hein. Provably adversarially robust nearest prototype

classifiers. In *International Conference on Machine Learning*, pages 22361–22383. PMLR, 2022.

Erklärung

Hiermit erkläre ich, dass ich meine Arbeit selbstständig verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die Arbeit noch nicht anderweitig für Prüfungszwecke vorgelegt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Mittweida, im 2023 [redacted]