CONTROLLING THE COMPLEXITY AND LIPSCHITZ CONSTANT IMPROVES POLYNOMIAL NETS

Anonymous authors
Paper under double-blind review

ABSTRACT

While the class of Polynomial Nets demonstrates comparable performance to neural networks (NN), it currently has neither theoretical generalization characterization nor robustness guarantees. To this end, we derive new complexity bounds for the set of Coupled CP-Decomposition (CCP) and Nested Coupled CP-decomposition (NCP) models of Polynomial Nets in terms of the ℓ_∞-operator-norm and the ℓ_2-operator norm. In addition, we derive bounds on the Lipschitz constant for both models to establish a theoretical certificate for their robustness. The theoretical results enable us to propose a principled regularization scheme that we also evaluate experimentally in six datasets and show that it improves the accuracy as well as the robustness of the models to adversarial perturbations. We showcase how this regularization can be combined with adversarial training, resulting in further improvements.

1 INTRODUCTION

Recently, high-degree Polynomial Nets (PNs) have been demonstrating state-of-the-art performance in a range of challenging tasks like image generation (Karras et al., 2019; Chrysos and Panagakis, 2020), image classification (Wang et al., 2018), reinforcement learning (Jayakumar et al., 2020), non-euclidean representation learning (Chrysos et al., 2020) and sequence models (Su et al., 2020). In particular, in public benchmarks like the *Face verification on MegaFace* task[1] (Kemelmacher-Shlizerman et al., 2016), Polynomial Nets are currently the top performing model.

A major advantage of Polynomial Nets over traditional Neural Networks[2] is that they are compatible with efficient *Leveled Fully Homomorphic Encryption (LFHE)* protocols (Brakerski et al., 2014). Such protocols allow efficient computation on encrypted data, but they only support addition or multiplication operations i.e., polynomials. This has prompted an effort to adapt neural networks by replacing typical activation functions with polynomial approximations (Gilad-Bachrach et al., 2016; Hesamifard et al., 2018). Polynomial Nets do not need any adaptation to work with LFHE.

Without doubt, these arguments motivate further investigation about the inner-workings of Polynomial Nets. Surprisingly, little is known about the theoretical properties of such high-degree polynomial expansions, despite their success. Previous work on PNs (Chrysos et al., 2020; Chrysos and Panagakis, 2020) have focused on developing the foundational structure of the models as well as their training, but do not provide an analysis of their generalization ability or robustness to adversarial perturbations.

In contrast, such type of results are readily available for traditional feed-forward Deep Neural Networks, in the form of high-probability generalization error bounds (Neyshabur et al., 2015; Bartlett et al., 2017; Neyshabur et al., 2017; Golowich et al., 2018) or upper bounds on their Lipschitz constant (Scaman and Virmaux, 2018; Fazlyab et al., 2019; Latorre et al., 2020). Despite their similarity in the compositional structure, theoretical results for Deep Neural Networks[2] do not apply to Polynomial Nets, as they are essentialy two non-overlapping classes of functions.

Why are such results important? First, they provide key theoretical quantities like the sample complexity of a hypothesis class: how many samples are required to succeed at learning in the PAC-framework. Second, they provide certified performance guarantees to adversarial perturbations

[1] https://paperswithcode.com/sota/face-verification-on-megaface
[2] with non-polynomial activation functions.

(Szegedy et al., 2014; Goodfellow et al., 2015) via a worst-case analysis c.f. Scaman and Virmaux (2018). Most importantly, the bounds themselves provide a principled way to regularize the hypothesis class and improve their accuracy or robustness.

For example, Generalization and Lipschitz constant bounds of Deep Neural Networks that depend on the operator-norm of their weight matrices (Bartlett et al., 2017; Neyshabur et al., 2017) have layed out the path for regularization schemes like spectral regularization (Yoshida and Miyato, 2017; Miyato et al., 2018), Lipschitz-margin training (Tsuzuku et al., 2018) and Parseval Networks (Cisse et al., 2017), to name a few.

Indeed, such schemes have been observed in practice to improve the performance of Deep Neural Networks. Unfortunately, similar regularization schemes for Polynomial Nets do not exist due to the lack of analogous bounds. Hence, it is possible that PNs are not yet being used to their fullest potential. We believe that theoretical advances in their understanding might lead to more resilient and accurate models. In this work, we aim to fill the gap in the theoretical understanding of PNs. We summarize our main contributions as follows:

Rademacher Complexity Bounds. We derive bounds on the Rademacher Complexity of the Coupled CP-decomposition model (CCP) and Nested Coupled CP-decomposition model (NCP) of PNs, under the assumption of a unit ℓ_∞-norm bound on the input (Theorems 1 and 3), a natural assumption in image-based applications. Analogous bounds for the ℓ_2-norm are also provided (Appendices D.3 and E.3). Such bounds lead to the first known generalization error bounds for this class of models.

Lipschitz constant Bounds. To complement our understanding of the CCP and NCP models, we derive upper bounds on their ℓ_∞-Lipschitz constants (Theorems 2 and 4), which are directly related to their robustness against ℓ_∞-bounded adversarial perturbations, and provide formal guarantees. Analogous results hold for any ℓ_p-norm (Appendices D.4 and E.4).

Regularization schemes. We identify key quantities that simultaneously control both Rademacher Complexity and Lipschitz constant bounds previously derived. Namely, the operator norms of the weight matrices in the Polynomial Nets. Hence, we propose to regularize the CCP and NCP models by constraining such operator norms. In doing so, our theoretical results indicate that both the generalization and the robustness to adversarial perturbations should improve. We propose a Projected Stochastic Gradient Descent scheme (Algorithm 1), enjoying the same per-iteration complexity as vanilla back-propagation in the ℓ_∞-norm case, and a variant that augments the base algorithm with adversarial traning (Algorithm 2).

Experiments. We conduct experimentation in *five* widely-used datasets on image recognition and on dataset in audio recognition. The experimentation illustrates how the aforementioned regularization schemes impact the accuracy (and the robust accuracy) of both CCP and NCP models, outperforming alternative schemes such as Jacobian regularization and the L_2 weight decay. Indeed, for a grid of regularization parameters we observe that there exists a sweet-spot for the regularization parameter which not only increases the test-accuracy of the model, but also its resilience to adversarial perturbations. Larger values of the regularization parameter also allow a trade-off between accuracy and robustness. The observation is consistent across all datasets and all adversarial attacks demonstrating the efficacy of the proposed regularization scheme.

2 Rademacher Complexity and Lipschitz constant bounds for Polynomial Nets

Notation. The symbol $\circ$ denotes the Hadamard (element-wise) product, the symbol $\bullet$ is the face-splitting product, while the symbol $\star$ denotes a convolutional operator. Matrices are denoted by uppercase letters e.g., V. Due to the space constraints, a detailed notation is deferred to Appendix B.

Assumption on the input distribution. Unless explicitly mentioned otherwise, we assume an ℓ_∞-norm unit bound on the input data i.e., $\|z\|_\infty \leq 1$ for any input z. This is the most common assumption in image-domain applications in contemporary deep learning, i.e., each pixel takes values in $[-1, 1]$ interval. Nevertheless, analogous results for ℓ_2-norm unit bound assumptions are presented in Appendices D.3, D.4, E.3 and E.4.

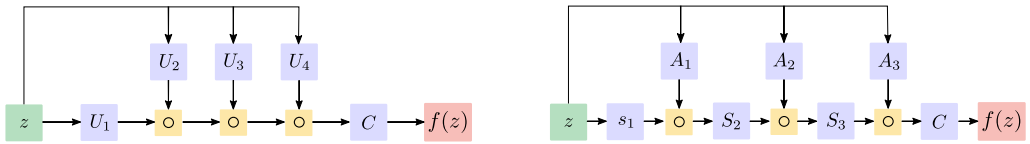


Figure 1: Schematic of CCP model (left) and NCP model (right), where $\circ$ denotes the Hadamard product. Blue boxes correspond to learnable parameters. Green and red boxes denote input and output, respectively. Yellow boxes denote operations.

We now introduce the basic concepts that will be developed throughout the paper i.e., the Rademacher Complexity of a class of functions (Bartlett and Mendelson, 2002) and the Lipschitz constant.

Definition 1 (Empirical Rademacher Complexity)**.** *Let $Z = \{z_1, \dots, z_n\} \subseteq \mathbb{R}^d$ and let $\{\sigma_j : j = 1, \dots, n\}$ be independent Rademacher random variables i.e., taking values uniformly in $\{-1, +1\}$. Let $\mathcal{F}$ be a class of real-valued functions over $\mathbb{R}^d$. The Empirical Rademacher complexity of $\mathcal{F}$ with respect to Z is defined as follows:*

$$\mathcal{R}_Z(\mathcal{F}) \coloneqq \mathbb{E}_\sigma \left[\sup_{f \in \mathcal{F}} \frac{1}{n} \sum_{j=1}^{n} \sigma_j f(z_j) \right]. \tag{1}$$

Definition 2 (Lipschitz constant)**.** *Given two normed spaces $(\mathcal{X}, \|\cdot\|_\mathcal{X})$ and $(\mathcal{Y}, \|\cdot\|_\mathcal{Y})$, a function $f : \mathcal{X} \to \mathcal{Y}$ is called Lipschitz continuous with Lipschitz constant $K \geq 0$ if for all x_1, x_2 in $\mathcal{X}$:*

$$\|f(x_1) - f(x_2)\|_\mathcal{Y} \leq K \|x_1 - x_2\|_\mathcal{X}. \tag{2}$$

2.1 Coupled CP-Decomposition of Polynomial Nets (CCP model)

The Coupled CP-Decomposition (CCP) model of PNs (Chrysos et al., 2020) leverages a coupled CP Tensor decomposition (Kolda and Bader, 2009) to vastly reduce the parameters required to describe a high-degree polynomial, and allows its computation in a compositional fashion, much similar to a feed-forward pass through a traditional neural network. The CCP model was used in Chrysos and Panagakis (2020) to construct a generative model. CCP can be succintly defined as follows:

$$f(z) = C \circ_{i=1}^{k} U_i z, \tag{CCP}$$

where $z \in \mathbb{R}^d$ is the input data, $f(z) \in \mathbb{R}^o$ is the output of the model and $U_n \in \mathbb{R}^{m \times d}, C \in \mathbb{R}^{o \times m}$ are the learnable parameters, where $m \in \mathbb{N}$ is the hidden rank. In Fig. 1 we provide a schematic of the architecture, while in Appendix C.1 we include further details on the original CCP formulation (and how to obtain our equivalent re-parametrization) for the interested reader.

In Theorem 1 we derive an upper bound on the complexity of CCP models with bounded ℓ_∞-operator-norms of the face-splitting product of the weight matrices. Its proof can be found in Appendix D.1. For a given CCP model, we derive an upper bound on its ℓ_∞-Lipschitz constant in Theorem 2 and its proof is given in Appendix D.4.1.

Theorem 1. *Let $Z = \{z_1, \dots, z_n\} \subseteq \mathbb{R}^d$ and suppose that $\|z_j\|_\infty \leq 1$ for all $j = 1, \dots, n$. Let*

$$\mathcal{F}_{CCP}^k \coloneqq \left\{ f(z) = \left\langle c, \circ_{i=1}^k U_i z \right\rangle : \|c\|_1 \leq \mu, \left\| \bullet_{i=1}^k U_i \right\|_\infty \leq \lambda \right\}. \tag{3}$$

The Empirical Rademacher Complexity of CCP$_k$ (k-degree CCP polynomials) with respect to Z is bounded as:

$$\mathcal{R}_Z(\mathcal{F}_{CCP}^k) \leq 2\mu\lambda \sqrt{\frac{2k \log(d)}{n}}. \tag{4}$$

Theorem 2. *The Lipschitz constant (with respect to the ℓ_∞-norm) of the function defined in Eq. (CCP), restricted to the set $\{z \in \mathbb{R}^d : \|z\|_\infty \leq 1\}$ is bounded as:*

$$Lip_\infty(f) \leq k \|C\|_\infty \prod_{i=1}^{k} \|U_i\|_\infty. \tag{5}$$

2.2 NESTED COUPLED CP-DECOMPOSITION (NCP MODEL)

The Nested Coupled CP-Decomposition (NCP) model leverages a joint hierarchical decomposition, which provided strong results in both generative and discriminative tasks in Chrysos et al. (2020). A slight re-parametrization of the model (Appendix C.2) can be expressed with the following recursive relation:

$$x_1 = (A_1 z) \circ (s_1), \qquad x_n = (A_n z) \circ (S_n x_{n-1}), \qquad f(z) = C x_k, \tag{NCP}$$

where $z \in \mathbb{R}^d$ is the input vector and $C \in \mathbb{R}^{o \times m}$, $A_n \in \mathbb{R}^{m \times d}$, $S_n \in \mathbb{R}^{m \times m}$ and $s_1 \in \mathbb{R}^m$ are the learnable parameters. In Fig. 1 we provide a schematic of the architecture.

In Theorem 3 we derive an upper bound on the complexity of NCP models with bounded ℓ_∞-operator-norm of a matrix function of its parameters. Its proof can be found in Appendix E.1. For a given NCP model, we derive an upper bound on its ℓ_∞-Lipschitz constant in Theorem 4 and its proof is given in Appendix E.4.1.

Theorem 3. *Let $Z = \{z_1, \dots, z_n\} \subseteq \mathbb{R}^d$ and suppose that $\|z_j\|_\infty \leq 1$ for all $j = 1, \dots, n$. Define the matrix $\Phi(A_1, S_1, \dots, A_n, S_n) \coloneqq (A_k \bullet S_k) \prod_{i=1}^{k-1} I \otimes A_i \bullet S_i$. Consider the class of functions:*

$$\mathcal{F}_{NCP}^k \coloneqq \{f(z) \text{ as in (NCP)} : \|C\|_\infty \leq \mu, \|\Phi(A_1, S_1, \dots, A_k, S_k)\|_\infty \leq \lambda\}, \tag{6}$$

where $C \in \mathbb{R}^{1 \times m}$ (single output case). The Empirical Rademacher Complexity of NCP$_k$ (k-degree NCP polynomials) with respect to Z is bounded as:

$$\mathcal{R}_Z(\mathcal{F}_{NCP}^k) \leq 2\mu\lambda \sqrt{\frac{2k \log(d)}{n}}. \tag{7}$$

Theorem 4. *The Lipschitz constant (with respect to the ℓ_∞-norm) of the function defined in Eq. (NCP), restricted to the set $\{z \in \mathbb{R}^d : \|z\|_\infty \leq 1\}$ is bounded as:*

$$Lip_\infty(f) \leq k\|C\|_\infty \prod_{i=1}^{k} (\|A_i\|_\infty \|S_i\|_\infty). \tag{8}$$

3 ALGORITHMS

By constraining the quantities in the upper bounds on the Rademacher complexity (Theorems 1 and 3), we can regularize the empirical loss minimization objective (Mohri et al., 2018, Theorem 3.3). Such method would prevent overfitting and can lead to an improved accuracy. However, one issue with the quantities involved in Theorems 1 and 3, namely

$$\left\| \bullet_{i=1}^k U_i \right\|_\infty, \qquad \left\| (A_k \bullet S_k) \prod_{i=1}^{k-1} I \otimes A_i \bullet S_i \right\|_\infty$$

is that projecting onto their level sets correspond to a difficult non-convex problem. Nevertheless, we can control an upper bound that depends on the ℓ_∞-operator norm of each weight matrix:

Lemma 1. *It holds that $\left\| \bullet_{i=1}^k U_i \right\|_\infty \leq \prod_{i=1}^k \|U_i\|_\infty$.*

Lemma 2. *It holds that $\left\| (A_k \bullet S_k) \prod_{i=1}^{k-1} I \otimes A_i \bullet S_i \right\|_\infty \leq \prod_{i=1}^k \|A_i\|_\infty \|S_i\|_\infty$.*

The proofs of Lemmas 1 and 2 can be found in Appendix D.2 and Appendix E.2. These results mean that by constraining the operator norms of each weight matrix, we can control the overall complexity of the CCP and NCP models.

Projecting a matrix onto an ℓ_∞-operator norm ball is a simple task that can be achieved by projecting each row of the matrix onto an ℓ_1-norm ball, for example, using the well-known algorithm from Duchi et al. (2008). The final optimization objective for training a regularized CCP is the following:

$$\min_{C, U_1, \dots, U_k} \frac{1}{n} \sum_{i=1}^{n} L(C, U_1, \dots, U_k; x_i, y_i) \qquad \text{subject to } \|C\|_\infty \leq \mu, \|U_i\| \leq \lambda, \tag{9}$$

where $(x_i, y_i)_{i=1}^n$ is the training dataset, L is the loss function (e.g., cross-entropy) and μ, λ are the regularization parameters. We notice that the constraints on the learnable parameters U_i and C have

the effect of simultaneously controlling the Rademacher Complexity and the Lipschitz constant of the CCP model. For the NCP model, an analogous objective function is used.

To solve the optimization problem in Eq. (9) we will use a Projected Stochastic Gradient Descent method Algorithm 1. We also propose a variant that combines Adversarial Training with the projection step (Algorithm 2) with the goal of increasing robustness to adversarial examples.

Algorithm 1: Projected SGD

Input: dataset Z, learning rate $\{\gamma_t > 0\}_{t=0}^{T-1}$, iterations T, hyper-parameters R, f , Loss L.
Output: model with parameters θ.
Initialize θ.
for $t = 0$ to $T-1$ **do**
  Sample (x, y) from Z
  $\theta = \theta - \gamma_t \nabla_\theta L(\theta; x, y)$.
  if $t \mod f = 0$ **then**
    $\theta = \prod_{\{\theta : \|\theta\|_\infty \leq R\}}(\theta)$

Algorithm 2: Projected SGD + Adversarial Training

Input: dataset Z, learning rate $\{\gamma_t > 0\}_{t=0}^{T-1}$, iterations T and n, hyper-parameters R, f , ϵ and α, Loss L
Output: model with parameters θ.
Initialize θ.
for $t = 0$ to $T-1$ **do**
  Sample (x, y) from Z
  for $i = 0$ to $n-1$ **do**
    $x^{\text{adv}} = \prod_{\{x' : ||x' - x||_\infty \leq \epsilon\}} \{x + \alpha \nabla_x L(\theta; x, y)\}$
  $\theta = \theta - \gamma_t \nabla_\theta L(\theta; x^{\text{adv}}, y)$
  if $t \mod f = 0$ **then**
    $\theta = \prod_{\{\theta : \|\theta\|_\infty \leq R\}}(\theta)$

In Algorithms 1 and 2 the parameter f is set in practice to a positive value, so that the projection (denoted by Π) is made only every few iterations. The variable θ represents the weight matrices of the model, and the projection in the last line should be understood as applied independently for every weight matrix. The regularization parameter R corresponds to the variables μ, λ in Eq. (9).

3.1 Convolutional Layers

Frequently, convolutions are employed in the literature, especially in the image-domain. It is important to understand how our previous results extend to this case, and how the proposed algorithms work in that case. Below, we show that the ℓ_∞-operator norm of the convolutional layer (as a linear operator) is related to the ℓ_∞-operator norm of the kernel after a reshaping operation. For simplicity, we consider only convolutions with zero padding.

We study the cases of 1D, 2D and 3D convolutions. For clarity, we mention below the result for the 3D convolution, since this is relevant to our experimental validation, and we defer the other two cases to Appendix F.

Theorem 5. *Let $A \in \mathbb{R}^{n \times m \times r}$ be an input image and let $K \in \mathbb{R}^{h \times h \times r \times o}$ be a convolutional kernel with o output channels. For simplicity assume that $k \leq \min(n, m)$ is odd. Denote by $B = K \star A$ the output of the convolutional layer. Let $U \in \mathbb{R}^{nmo \times nmr}$ be the matrix such that $vec(B) = Uvec(A)$ i.e., U is the matrix representation of the convolution. Let $M(K) \in \mathbb{R}^{o \times hhr}$ be the matricization of K, where each row contains the parameters of a single output channel of the convolution. It holds that: $\|U\|_\infty = \|M(K)\|_\infty$.*

Thus, we can control the ℓ_∞-operator-norm of a convolutional layer during training by controlling that of the reshaping of the kernel, which is done with the same code as for fully connected layers. It can be seen that when the padding is non-zero, the result still holds.

4 Numerical Evidence

The generalization properties and the robustness of PNs are numerically verified in this section. We evaluate the robustness to three widely-used adversarial attacks in Section 4.2. We assess whether the compared regularization schemes can also help in the case of adversarial training in sec. 4.3. Experiments with additional datasets, models (NCP models) and adversarial attacks (APGDT, PGDT) are conducted in Appendix G due to the restricted space. The results exhibit a **consistent** behavior across different adversarial attacks, different datasets and different models. Whenever the results differ, we *explicitly* mention the differences in the main body below.

4.1 EXPERIMENTAL SETUP

The accuracy is reported as as the evaluation metric for every experiment, where a higher accuracy translates to better performance.

Datasets and Benchmark Models: We conduct experiments on the popular datasets of Fashion-MNIST (Xiao et al., 2017) and E-MNIST (Cohen et al., 2017). Each dataset includes $70,000$ grayscale images of resolution 28×28 out of which $60,000$ are for training. Each image is annotated with one out of the ten categories. We use two popular regularization methods from the literature for comparison, i.e., Jacobian regularization (Hoffman et al., 2019) and L_2 regularization (weight decay).

Models: We report results using the following three models: 1) a 4th-degree CCP model named "PN-4", 2) a 10th-degree CCP model referenced as "PN-10" and 3) a 4th-degree Convolutional CCP model called "PN-Conv". In the PN-Conv, we have replaced all the U_i matrices with convolutional kernels. None of the variants contains any activation functions.

Hyper-parameters: Unless mentioned otherwise, all models are trained for 100 epochs with a batch size of 64. The initial value of the learning rate is 0.001. After the first 25 epochs, the learning rate is multiplied by a factor of 0.2 every 50 epochs. The SGD is used to optimize all the models, while the cross-entropy loss is used. In the experiments that include projection or adversarial training, the first 50 epochs are pre-training, i.e., training only with the cross-entropy loss. The projection is performed every ten iterations.

Adversarial Attack Settings: We utilize two widely used attacks: a) Fast Gradient Sign Method (FGSM) and b) Projected Gradient Descent (PGD). In FGSM the hyper-parameter ϵ represents the step size of the adversarial attack. In PGD there is a triple of parameters (ϵ_{total}, n_{iters}, ϵ_{iter}), which represent the maximum step size of the total adversarial attack, the number of steps to perform for a single attack, and the step size of each adversarial attack step respectively. We consider the following hyper-parameters for the attacks: a) FGSM with $\epsilon = 0.1$, b) PGD with parameters (0.1, 20, 0.01), c) PGD with parameters (0.3, 20, 0.03).

4.2 EVALUATION OF THE ROBUSTNESS OF PNS

In the next experiment, we assess the robustness of PNs under adversarial noise. That is, the method is trained on the train set of the respective dataset and the evaluation is performed on the test set perturbed by additive adversarial noise. That is, each image is individually perturbed based on the respective adversarial attack. The proposed method implements Algorithm 1.

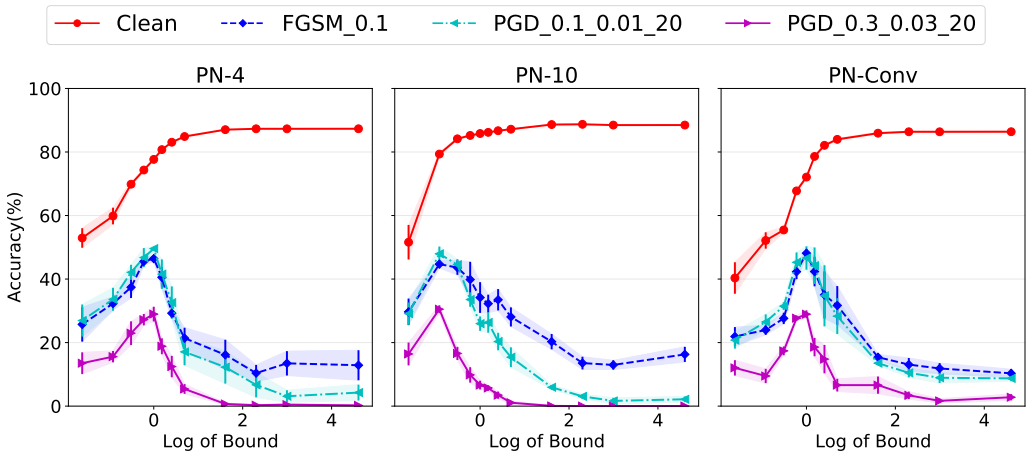
The quantitative results in both Fashion-MNIST and E-MNIST using PNs-4, PNs-10 and PNs-Conv under the three attacks are reported in Table 1. The column 'No-proj' exhibits the plain SGD training (i.e., without regularization), while the remaining columns include the proposed regularization, Jacobian and the L_2 regularization respectively. The results without regularization exhibit a substantial decrease in accuracy for stronger adversarial attacks. The proposed regularization outperforms all methods consistently across different adversarial attacks. Interestingly, the stronger the adversarial attack, the bigger the difference of the proposed regularization scheme with the alternatives of Jacobian and L_2 regularizations.

Next, we learn the networks with varying projection bounds. The results on Fashion-MNIST and E-MNIST are visualized in Fig. 2, where the x-axis is plotted in log-scale. As a reference point, we include the clean accuracy curves, i.e., when there is no adversarial noise. Projection bounds larger than 2 (in the log-axis) leave the accuracy unchanged. As the bounds decrease, the results gradually improve. This can be attributed to the constraints the projection bounds impose into the U_i matrices.

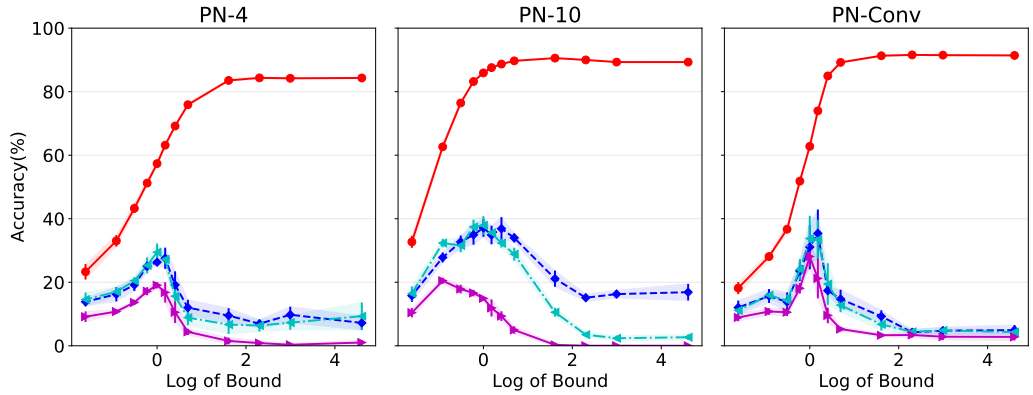
Similar observations can be made when evaluating the clean accuracy (i.e., no adversarial noise in the test set). However, in the case of adversarial attacks a tighter bound performs better, i.e., the best accuracy is exhibited in the region of 0 in the log-axis. The projection bounds can have a substantial improvement on the performance, especially in the case of stronger adversarial attacks, i.e., PGD.

4.3 ADVERSARIAL TRAINING (AT) ON PNS

Adversarial training has been used as a strong defence against adversarial attacks. In this experiment we evaluate whether different regularization methods can work in conjunction with adversarial training



(a) Fashion-MNIST



(b) E-MNIST

Figure 2: Adversarial attacks during testing on (a) Fashion-MNIST (top), (b) E-MNIST (bottom) with the x-axis is plotted in log-scale. Note that intermediate values of projection bounds yield the highest accuracy. The patterns are consistent in both datasets and across adversarial attacks.

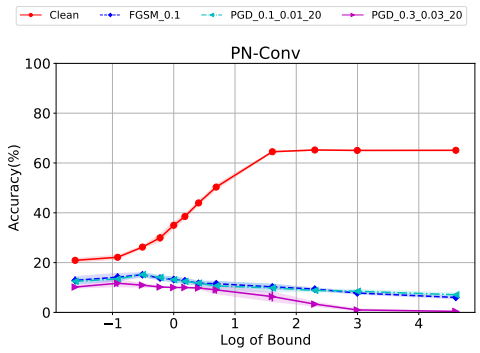


Figure 3: Adversarial attacks during testing on Cifar10.

Model	PN-Conv	
Projection	No-proj	Proj
Clean accuracy	$65.09 \pm 0.14\%$	$\mathbf{65.22 \pm 0.13}\%$
FGSM-0.1	$6.00 \pm 0.53\%$	$\mathbf{15.13 \pm 0.81}\%$
PGD-(0.1, 20, 0.01)	$7.08 \pm 0.68\%$	$\mathbf{15.17 \pm 0.88}\%$
PGD-(0.3, 20, 0.03)	$0.41 \pm 0.09\%$	$\mathbf{11.71 \pm 1.11}\%$

Table 2: Evaluation of the robustness of PN models on Cifar10. Each line refers to a different adversarial attack. The projection offers an improvement in the accuracy in each case; in PGD attacks projection improves the accuracy by a remarkable margin.

	Method	No proj.	Our method	Jacobian	L_2
			Fashion-MNIST		
PN-4	Clean	$87.28 \pm 0.18\%$	$\mathbf{87.32 \pm 0.14}\%$	$86.24 \pm 0.14\%$	$87.31 \pm 0.13\%$
	FGSM-0.1	$12.92 \pm 2.74\%$	$\mathbf{46.43 \pm 0.95}\%$	$17.90 \pm 6.51\%$	$13.80 \pm 3.65\%$
	PGD-(0.1, 20, 0.01)	$5.64 \pm 1.76\%$	$\mathbf{49.58 \pm 0.59}\%$	$12.23 \pm 5.63\%$	$5.01 \pm 2.44\%$
	PGD-(0.3, 20, 0.03)	$0.18 \pm 0.16\%$	$\mathbf{28.96 \pm 2.31}\%$	$1.27 \pm 1.29\%$	$0.28 \pm 0.18\%$
PN-10	Clean	$88.48 \pm 0.17\%$	$\mathbf{88.72 \pm 0.12}\%$	$88.12 \pm 0.11\%$	$88.46 \pm 0.19\%$
	FGSM-0.1	$15.96 \pm 1.00\%$	$\mathbf{44.71 \pm 1.24}\%$	$19.52 \pm 1.14\%$	$16.51 \pm 2.33\%$
	PGD-(0.1, 20, 0.01)	$1.94 \pm 0.82\%$	$\mathbf{47.94 \pm 2.29}\%$	$5.44 \pm 0.81\%$	$2.16 \pm 0.95\%$
	PGD-(0.3, 20, 0.03)	$0.02 \pm 0.03\%$	$\mathbf{30.51 \pm 1.22}\%$	$0.05 \pm 0.02\%$	$0.01 \pm 0.02\%$
PN-Conv	Clean	$86.36 \pm 0.21\%$	$86.38 \pm 0.26\%$	$84.69 \pm 0.44\%$	$\mathbf{86.45 \pm 0.21}\%$
	FGSM-0.1	$10.80 \pm 1.82\%$	$\mathbf{48.15 \pm 1.23}\%$	$10.62 \pm 0.77\%$	$10.73 \pm 1.58\%$
	PGD-(0.1, 20, 0.01)	$9.37 \pm 1.00\%$	$\mathbf{46.63 \pm 3.68}\%$	$10.20 \pm 0.32\%$	$8.96 \pm 0.83\%$
	PGD-(0.3, 20, 0.03)	$1.75 \pm 0.83\%$	$\mathbf{28.94 \pm 1.20}\%$	$8.26 \pm 1.05\%$	$2.03 \pm 0.99\%$
			E-MNIST		
PN-4	Clean	$84.27 \pm 0.26\%$	$\mathbf{84.34 \pm 0.31}\%$	$81.99 \pm 0.33\%$	$84.22 \pm 0.33\%$
	FGSM-0.1	$8.92 \pm 1.99\%$	$\mathbf{27.56 \pm 3.32}\%$	$14.96 \pm 1.32\%$	$8.18 \pm 3.48\%$
	PGD-(0.1, 20, 0.01)	$6.24 \pm 1.43\%$	$\mathbf{29.46 \pm 2.73}\%$	$6.75 \pm 2.92\%$	$5.93 \pm 1.97\%$
	PGD-(0.3, 20, 0.03)	$1.22 \pm 0.85\%$	$\mathbf{19.07 \pm 0.98}\%$	$3.06 \pm 0.53\%$	$1.00 \pm 0.76\%$
PN-10	Clean	$89.31 \pm 0.09\%$	$\mathbf{90.56 \pm 0.10}\%$	$89.19 \pm 0.07\%$	$89.23 \pm 0.13\%$
	FGSM-0.1	$15.56 \pm 1.16\%$	$\mathbf{37.11 \pm 2.81}\%$	$24.21 \pm 1.89\%$	$16.30 \pm 1.82\%$
	PGD-(0.1, 20, 0.01)	$2.63 \pm 0.65\%$	$\mathbf{37.89 \pm 2.91}\%$	$9.18 \pm 1.09\%$	$2.33 \pm 0.43\%$
	PGD-(0.3, 20, 0.03)	$0.00 \pm 0.00\%$	$\mathbf{20.47 \pm 0.96}\%$	$0.11 \pm 0.08\%$	$0.02 \pm 0.03\%$
PN-Conv	Clean	$91.49 \pm 0.29\%$	$\mathbf{91.57 \pm 0.19}\%$	$90.38 \pm 0.13\%$	$91.41 \pm 0.18\%$
	FGSM-0.1	$4.28 \pm 0.55\%$	$\mathbf{35.39 \pm 7.51}\%$	$3.88 \pm 0.04\%$	$4.13 \pm 0.41\%$
	PGD-(0.1, 20, 0.01)	$3.98 \pm 0.82\%$	$\mathbf{33.75 \pm 7.17}\%$	$3.86 \pm 0.01\%$	$4.83 \pm 0.87\%$
	PGD-(0.3, 20, 0.03)	$3.24 \pm 0.76\%$	$\mathbf{28.10 \pm 3.27}\%$	$3.84 \pm 0.01\%$	$2.76 \pm 0.65\%$

Table 1: Comparison of regularization techniques on Fashion-MNIST (top) and E-MNIST (bottom). In each dataset, the base networks are PN-4, i.e., a 4th degree polynomial, on the top four rows, PN-10, i.e., a 10th degree polynomial, on the middle four rows and PN-Conv, i.e., a 4th degree polynomial with convolutions, on the bottom four rows. Our projection method exhibits the best performance in all three attacks, with the difference on accuracy to stronger attacks being substantial.

that is widely used as a defence method. Since multi-step adversarial attacks are computationally intensive, we utilize the FGSM attack during training, while we evaluate the trained model in all four adversarial attacks. For this experiment we select PN-10 as the base model. The proposed model implements Algorithm 2.

The accuracy is reported in Table 3 with Fashion-MNIST on the top and E-MNIST on the bottom. In the FGSM attack, the difference of the compared methods is smaller, which is expected since similar attack is used for the training. However, for stronger attacks the difference becomes pronounced with the proposed regularization method outperforming both the Jacobian and the L_2 regularization methods.

Method	AT	Our method + AT	Jacobian + AT	L_2 + AT
		Adversarial training (AT) with PN-10 on *Fashion-MNIST*		
FGSM-0.1	$65.33 \pm 0.46\%$	$\mathbf{65.64 \pm 0.35}\%$	$62.04 \pm 0.22\%$	$65.62 \pm 0.15\%$
PGD-(0.1, 20, 0.01)	$57.45 \pm 0.35\%$	$\mathbf{59.89 \pm 0.22}\%$	$57.42 \pm 0.24\%$	$57.40 \pm 0.36\%$
PGD-(0.3, 20, 0.03)	$24.46 \pm 0.45\%$	$\mathbf{39.79 \pm 1.40}\%$	$25.59 \pm 0.20\%$	$24.99 \pm 0.57\%$
		Adversarial training (AT) with PN-10 on *E-MNIST*		
FGSM-0.1	$78.30 \pm 0.18\%$	$\mathbf{78.61 \pm 0.58}\%$	$70.11 \pm 0.18\%$	$78.31 \pm 0.32\%$
PGD-(0.1, 20, 0.01)	$68.40 \pm 0.32\%$	$\mathbf{68.51 \pm 0.19}\%$	$64.61 \pm 0.16\%$	$68.41 \pm 0.37\%$
PGD-(0.3, 20, 0.03)	$35.58 \pm 0.33\%$	$\mathbf{42.22 \pm 0.60}\%$	$39.83 \pm 0.24\%$	$35.17 \pm 0.46\%$

Table 3: Comparison of regularization techniques on (a) Fashion-MNIST (top) and (b) E-MNIST (bottom) along with adversarial training (AT). The base network is a PN-10, i.e., 10th degree polynomial. Our projection method exhibits the best performance in all four attacks.

The limitations of the proposed work are threefold. Firstly, Theorem 1 relies on the ℓ_∞-operator norm of the face-splitting product of the weight matrices, which in practice we relax in Lemma 1 for performing the projection. In the future, we aim to study if it is feasible to compute the non-convex projection onto the set of PNs with bounded ℓ_∞-norm of the face-splitting product of the weight matrices. This would allow us to let go off the relaxation argument and directly optimize the original tighter Rademacher Complexity bound (Theorem 1).

Secondly, the regularization effect of the projection differs across datasets and adversarial attacks, a topic that is worth investigating in the future.

Thirdly, our bounds do not take into account the algorithm used, which corresponds to a variant of the Stochastic Projected Gradient Descent, and hence any improved generalization properties due to possible uniform stability (Bousquet and Elisseeff, 2002) of the algorithm or implicit regularization properties (Neyshabur, 2017), do not play a role in our analysis.

5 Related Work

Rademacher Complexity: Known bounds for the class of polynomials are a consequence of more general result for kernel methods (Mohri et al., 2018, Theorem 6.12). Support Vector Machines (SVMs) with a polynomial kernel of degree k effectively correspond to a general polynomial with the same degree. In contrast, our bound is tailored to the parametric definition of the CCP and the NCP models, which are a subset of the class of all polynomials. Hence, they are tighter than the general kernel complexity bounds.

Bounds for the class of neural networks were stablished in (Bartlett et al., 2017; Neyshabur et al., 2017), but they require a long and technical proof, and in particular it assumes an ℓ_2-bound on the input, which is incompatible with image-based applications. This bound also depend on the product of spectral norms of each layer. In contrast, our bounds are more similar in spirit to the path-norm-based complexity bounds (Neyshabur et al., 2015), as they depend on interactions between neurons at different layers. This interaction precisely corresponds to the face-splitting product between weight matrices that appears in Theorem 1.

Lipschitz constant: A variety of methods have been proposed for estimating the Lipschitz constant of neural networks. For example, Scaman and Virmaux (2018) (SVD), Fazlyab et al. (2019) (Semidefinite programming) and Latorre et al. (2020) (Polynomial Optimization) are expensive optimization methods to compute tighter bounds on such constant. These methods are unusable in our case as they would require a non-trivial adaptation to work with Polynomial Nets. In contrast we find an upper bound that applies to such family of models, and it can be controlled with efficient ℓ_∞-operator-norm projections. However, our bounds might not be the tightest. Developing tighter methods to bound and control the Lipschitz constant for Polynomial Nets is a promising direction of future research.

6 Conclusion

In this work, we explore the generalization properties of the Coupled CP-decomposition (CCP) and nested coupled CP-decomposition (NCP) models that belong in the class of Polynomial Nets (PNs). We derive bounds for the Rademacher complexity and the Lipschitz constant of the CCP and the NCP models. We utilize the computed bounds as a regularization during training. The regularization terms have also a substantial effect on the robustness of the model, i.e., when adversarial noise is added to the test set. Along with the recent empirical results on PNs, our derived bounds can further explain the benefits and drawbacks of using PNs.

References

P. Bartlett, D. J. Foster, and M. Telgarsky. Spectrally-normalized margin bounds for neural networks, 2017.

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

O. Bousquet and A. Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

Z. Brakerski, C. Gentry, and V. Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):1–36, 2014.

G. Chrysos, S. Moschoglou, G. Bouritsas, Y. Panagakis, J. Deng, and S. Zafeiriou. $\pi-$nets: Deep polynomial neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

G. G. Chrysos and Y. Panagakis. NAPS: Non-adversarial polynomial synthesis. *Pattern Recognit. Lett.*, 140:318–324, 2020.

M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier. Parseval networks: Improving robustness to adversarial examples. In *International Conference on Machine Learning*, pages 854–863. PMLR, 2017.

T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha. Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*, 2018.

G. Cohen, S. Afshar, J. Tapson, and A. van Schaik. Emnist: an extension of mnist to handwritten letters. *arXiv preprint arXiv:1908.06571*, 2017.

F. Croce and M. Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the l 1-ball for learning in high dimensions. In *Proceedings of the 25th international conference on Machine learning*, pages 272–279, 2008.

J. Engel, C. Resnick, A. Roberts, S. Dieleman, D. Eck, K. Simonyan, and M. Norouzi. Neural audio synthesis of musical notes with wavenet autoencoders, 2017.

M. Fazlyab, A. Robey, H. Hassani, M. Morari, and G. J. Pappas. Efficient and accurate estimation of lipschitz constants for deep neural networks. In *Advances in neural information processing systems (NeurIPS)*, 2019.

H. Federer. *Geometric measure theory*. Springer, 2014.

R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 201–210, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/gilad-bachrach16.html.

N. Golowich, A. Rakhlin, and O. Shamir. Size-independent sample complexity of neural networks. In *COLT*, 2018.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015.

E. Hesamifard, H. Takabi, M. Ghasemi, and R. N. Wright. Privacy-preserving machine learning as a service. *Proc. Priv. Enhancing Technol.*, 2018(3):123–142, 2018.

J. Hoffman, D. A. Roberts, and S. Yaida. Robust learning with jacobian regularization, 2019.

S. M. Jayakumar, W. M. Czarnecki, J. Menick, J. Schwarz, J. Rae, S. Osindero, Y. W. Teh, T. Harley, and R. Pascanu. Multiplicative interactions and where to find them. In *International Conference on Learning Representations (ICLR)*, 2020.

T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

I. Kemelmacher-Shlizerman, S. M. Seitz, D. Miller, and E. Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4873–4882, 2016.

T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

A. Krizhevsky, V. Nair, and G. Hinton. The cifar-10 dataset. *online: http://www. cs. toronto. edu/kriz/cifar. html*, 55, 2014.

F. Latorre, P. Rolland, and V. Cevher. Lipschitz constant estimation of neural networks via sparse polynomial optimization. In *International Conference on Learning Representations (ICLR)*, 2020.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1QRgziT-.

M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2 edition, 2018. ISBN 978-0-262-03940-6.

B. Neyshabur. Implicit regularization in deep learning. *arXiv preprint arXiv:1709.01953*, 2017.

B. Neyshabur, R. Tomioka, and N. Srebro. Norm-based capacity control in neural networks. In *Conference on Learning Theory*, pages 1376–1401. PMLR, 2015.

B. Neyshabur, S. Bhojanapalli, and N. Srebro. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. *arXiv preprint arXiv:1707.09564*, 2017.

K. Scaman and A. Virmaux. Lipschitz regularity of deep neural networks: analysis and efficient estimation. In *Advances in neural information processing systems (NeurIPS)*, 2018.

V. Slyusar. A family of face products of matrices and its properties. *Cybernetics and Systems Analysis*, 35(3):379–384, 1999.

J. Su, W. Byeon, J. Kossaifi, F. Huang, J. Kautz, and A. Anandkumar. Convolutional tensor-train lstm for spatio-temporal learning. *Advances in neural information processing systems (NeurIPS)*, 2020.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations (ICLR)*, 2014.

Y. Tsuzuku, I. Sato, and M. Sugiyama. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/485843481a7edacbfce101ecb1e4d2a8-Paper.pdf.

X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7794–7803, 2018.

H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Y. Yoshida and T. Miyato. Spectral Norm Regularization for Improving the Generalizability of Deep Learning. *arXiv e-prints*, art. arXiv:1705.10941, May 2017.

H. Zhang, Y. Yu, J. Jiao, E. Xing, L. E. Ghaoui, and M. Jordan. Theoretically principled trade-off between robustness and accuracy. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

A APPENDIX INTRODUCTION

The Appendix is organized as follows:

- In Appendix B the notation and the core Lemmas from the literature are described.
- Further details on the Polynomial Nets are provided in Appendix C.
- The proofs on the CCP and the NCP models are added in Appendix D and Appendix E respectively.
- The extensions of the theorems for convolutional layers and their proofs are detailed in Appendix F.
- Additional experiments are included in Appendix G.

B BACKGROUND

Below, we develop a detailed notation in Appendix B.1, we include related definitions in Appendix B.2 and Lemmas required for our proofs in Appendix B.3. The goal of this section is to cover many of the required information for following the proofs and the notation we follow in this work. Readers familiar with the different matrix/vector products, e.g., Khatri-Rao or face-splitting product, and with basic inequalities, e.g., Hölder's inequality, can skip to the next section.

B.1 NOTATION

Table 4: Symbols for various matrix products. The precise definitions of the products are included in Appendix B.2 for completion.

Symbol	Definition
$\circ$	Hadamard (element-wise) product.
$*$	Column-wise Khatri–Rao product.
$\bullet$	Face-splitting product.
$\otimes$	Kronecker product.
$\star$	Convolution.

Different matrix products and their associated symbols are referenced in Table 4, while matrix operations on a matrix A are defined on Table 5. Every matrix product, e.g., Hadamard product, can be used in two ways: a) $A \circ B$, which translates to Hadamard product of matrices A and B, b) $\circ_{i=1}^{N} A_i$ abbreviates the Hadamard products $\underbrace{A_1 \circ A_2 \circ \ldots A_N}_{N \text{ products}}$.

Table 5: Operations and symbols on a matrix A.

Symbol	Definition
$\|A\|_\infty$	ℓ_∞-operator-norm; corresponds to the maximum ℓ_1-norm of its rows.
A^i	i^{th} row of A.
$a_{i,j}$	$(i,j)^{\text{th}}$ element of A.
$A_{i,j}$	$(i,j)^{\text{th}}$ block of a block-matrix A
A_i	The i-th matrix in a set of matrices $\{A_1, \cdots, A_N\}$.

The symbol x_i^j refers to j^{th} element of vector x_i.

B.2 DEFINITIONS

For thoroughness, we include the definitions of the core products that we use in this work. Specifically, the definitions of the Hadamard product (Definition 3), Kronecker product (Definition 4), the Khatri-Rao product (Definition 5), column-wise Khatri-Rao product (Definition 6) and the face-splitting product (Definition 7) are included.

Definition 3 (Hadamard product)**.** *For two matrices A and B of the same dimension $m \times n$, the Hadamard product $A \circ B$ is a matrix of the same dimension as the operands, with elements given by*

$$(a \circ b)_{i,j} = a_{i,j} b_{i,j} \tag{10}$$

Definition 4 (Kronecker product)**.** *If A is an $m \times n$ matrix and B is a $p \times q$ matrix, then the Kronecker product $A \otimes B$ is the $pm \times qn$ block matrix, given as follows:*

$$A \otimes B = \begin{bmatrix} a_{1,1}B & \cdots & a_{1,n}B \\ \vdots & \ddots & \vdots \\ a_{m,1}B & \cdots & a_{m,n}B \end{bmatrix}. \tag{11}$$

Example: the Kronecker product of the matrices $A \in \mathbb{R}^{2\times 2}$ and $B \in \mathbb{R}^{2\times 2}$ is computed below:

$$\underbrace{\begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix}}_{A} \otimes \underbrace{\begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix}}_{B} = \underbrace{\begin{bmatrix} a_{1,1}b_{1,1} & a_{1,1}b_{1,2} & a_{1,2}b_{1,1} & a_{1,2}b_{1,2} \\ a_{1,1}b_{2,1} & a_{1,1}b_{2,2} & a_{1,2}b_{2,1} & a_{1,2}b_{2,2} \\ a_{2,1}b_{1,1} & a_{2,1}b_{1,2} & a_{2,2}b_{1,1} & a_{2,2}b_{1,2} \\ a_{2,1}b_{2,1} & a_{2,1}b_{2,2} & a_{2,2}b_{2,1} & a_{2,2}b_{2,2} \end{bmatrix}}_{A\otimes B}.$$

Definition 5 (Khatri–Rao product)**.** *The Khatri–Rao product is defined as:*

$$A * B = (A_{i,j} \otimes B_{i,j})_{i,j}, \tag{12}$$

in which the (i, j)-th block is the $m_i p_i \times n_j q_j$ sized Kronecker product of the corresponding blocks of A and B, assuming the number of row and column partitions of both matrices is equal. The size of the product is then $(\sum_i m_i p_i) \times (\sum_i n_j q_j)$.

Example: if A and B both are 2×2 partitioned matrices e.g.:

$$A = \left[\begin{array}{c|c} A_{1,1} & A_{1,2} \\ \hline A_{2,1} & A_{2,2} \end{array}\right] = \left[\begin{array}{cc|c} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ \hline a_{3,1} & a_{3,2} & a_{3,3} \end{array}\right], \tag{13}$$

$$B = \left[\begin{array}{c|c} B_{1,1} & B_{1,2} \\ \hline B_{2,1} & B_{2,2} \end{array}\right] = \left[\begin{array}{c|cc} b_{1,1} & b_{1,2} & b_{1,3} \\ \hline b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,1} & b_{3,2} & b_{3,3} \end{array}\right], \tag{14}$$

then we obtain the following:

$$A * B = \left[\begin{array}{c|c} A_{1,1} \otimes B_{1,1} & A_{1,2} \otimes B_{1,2} \\ \hline A_{2,1} \otimes B_{2,1} & A_{2,2} \otimes B_{2,2} \end{array}\right] = \left[\begin{array}{cc|cc} a_{1,1}b_{1,1} & a_{1,2}b_{1,1} & a_{1,3}b_{1,2} & a_{1,3}b_{1,3} \\ a_{2,1}b_{1,1} & a_{2,2}b_{1,1} & a_{2,3}b_{1,2} & a_{2,3}b_{1,3} \\ \hline a_{3,1}b_{2,1} & a_{3,2}b_{2,1} & a_{3,3}b_{2,2} & a_{3,3}b_{2,3} \\ a_{3,1}b_{3,1} & a_{3,2}b_{3,1} & a_{3,3}b_{3,2} & a_{3,3}b_{3,3} \end{array}\right]. \tag{15}$$

Definition 6 (Column-wise Khatri–Rao product)**.** *A column-wise Kronecker product of two matrices may also be called the Khatri–Rao product. This product assumes the partitions of the matrices are their columns. In this case $m_1 = m$, $p_1 = p$, $n = q$ and for each j: $n_j = p_j = 1$. The resulting product is a $mp \times n$ matrix of which each column is the Kronecker product of the corresponding columns of A and B.*

Example: the Column-wise Khatri–Rao product of the matrices $A \in \mathbb{R}^{2\times 3}$ and $B \in \mathbb{R}^{3\times 3}$ is computed below:

$$\underbrace{\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \end{bmatrix}}_{A} * \underbrace{\begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,1} & b_{3,2} & b_{3,3} \end{bmatrix}}_{B} = \underbrace{\begin{bmatrix} a_{1,1}b_{1,1} & a_{1,2}b_{1,2} & a_{1,3}b_{1,3} \\ a_{1,1}b_{2,1} & a_{1,2}b_{2,2} & a_{1,3}b_{2,3} \\ a_{1,1}b_{3,1} & a_{1,2}b_{3,2} & a_{1,3}b_{3,3} \\ a_{2,1}b_{1,1} & a_{2,2}b_{1,2} & a_{2,3}b_{1,3} \\ a_{2,1}b_{2,1} & a_{2,2}b_{2,2} & a_{2,3}b_{2,3} \\ a_{2,1}b_{3,1} & a_{2,2}b_{3,2} & a_{2,3}b_{3,3} \end{bmatrix}}_{A*B}.$$

From here on, all $*$ refer to Column-wise Khatri–Rao product.

Definition 7 (Face-splitting product)**.** *The alternative concept of the matrix product, which uses row-wise splitting of matrices with a given quantity of rows. This matrix operation was named the* face-splitting product *of matrices or the* transposed Khatri–Rao product. *This type of operation is based on row-by-row Kronecker products of two matrices.*

Example: the Face-splitting product of the matrices $A \in \mathbb{R}^{3\times 2}$ and $B \in \mathbb{R}^{3\times 2}$ is computed below:

$$\underbrace{\begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{3,2} \\ a_{3,1} & a_{3,2} \end{bmatrix}}_{A} \bullet \underbrace{\begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \\ b_{3,1} & b_{3,2} \end{bmatrix}}_{B} = \underbrace{\begin{bmatrix} a_{1,1}b_{1,1} & a_{1,2}b_{1,1} & a_{1,1}b_{1,2} & a_{1,2}b_{1,2} \\ a_{2,1}b_{2,1} & a_{2,2}b_{2,1} & a_{2,1}b_{2,2} & a_{2,2}b_{2,2} \\ a_{3,1}b_{3,1} & a_{3,2}b_{3,1} & a_{3,1}b_{3,2} & a_{3,2}b_{3,2} \end{bmatrix}}_{A\bullet B}.$$

B.3 Related Lemmas

In this section, we provide the details on the Lemmas required for our proofs along with their proofs or the corresponding citations where the Lemmas can be found as well.

Lemma 3. *(Federer, 2014) Let g, h be two composable Lipschitz functions. Then $g \circ h$ is also Lipschitz with Lip$(g \circ h) \leq$ Lip(g)Lip(h). Here and only here $\circ$ represents function composition.*

Lemma 4. *(Federer, 2014) Let $f : \mathcal{X} \subseteq \mathbb{R}^n \to \mathbb{R}^m$ be differentiable and Lipschitz continuous. Let $J_f(x)$ denote its total derivative (Jacobian) at x. Then Lip$_p(f) = \sup_{x\in\mathcal{X}} \|J_f(x)\|_p$ where $\|J_f(x)\|_p$ is the induced operator norm on $J_f(x)$.*

Lemma 5 (Hölder's inequality)**.** *Let (S, Σ, μ) be a measure space and let $p, q \in [1, \infty]$ with $\frac{1}{p} + \frac{1}{q} = 1$. Then, for all measurable real-valued functions f and g on S, it holds that:*

$$\|fg\|_1 \leq \|f\|_p \|g\|_q \tag{16}$$

Lemma 6 (Mixed Product Property 1)**.** *(Slyusar, 1999) The following holds:*

$$(A_1B_1) \circ (A_2B_2) = (A_1 \bullet A_2)(B_1 * B_2). \tag{17}$$

Lemma 7 (Mixed Product Property 2)**.** *The following holds:*

$$\circ_{i=1}^{N}(A_iB_i) = \bullet_{i=1}^{N}(A_i) *_{i=1}^{N} (B_i). \tag{18}$$

Proof. We prove this lemma by induction on N.

Base case ($N = 1$): $A_1B_1 = A_1B_1$.

Inductive step: Assume that the induction hypothesis holds for a particular k, i.e., the case $N = k$ holds. That can be expressed as:

$$\circ_{i=1}^{k}(A_iB_i) = \bullet_{i=1}^{k}(A_i) *_{i=1}^{k} (B_i). \tag{19}$$

Then we will prove that it holds for $N = k + 1$:

$$\begin{aligned}
&\circ_{i=1}^{k+1} (A_iB_i) \\
&= [\circ_{i=1}^{k}(A_iB_i)] \circ (A_{k+1}B_{k+1}) \\
&= [\bullet_{i=1}^{k}(A_i) *_{i=1}^{k} (B_i)] \circ (A_{k+1}B_{k+1}) \quad \text{use inductive step (Eq. (19))} \\
&= [\bullet_{i=1}^{k}(A_i) \bullet A_{k+1}][*_{i=1}^{k}(B_i) * B_{k+1}] \quad \text{Lemma 6 [Mixed product property 1]} \\
&= \bullet_{i=1}^{k+1}(A_i) *_{i=1}^{k+1} (B_i)
\end{aligned} \tag{20}$$

That is, the case $N = k + 1$ also holds true, establishing the inductive step. □

Lemma 8 (Massart Lemma)**.** *Let $\mathcal{A} = \{\mathbf{a}_1, \cdots, \mathbf{a}_N\}$ be a finite set of vectors in $\mathbb{R}^m$. Define $\bar{\mathbf{a}} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{a}_i$. Then:*

$$\mathcal{R}(\mathcal{A}) \leq \max_{\mathbf{a}\in A} \|\mathbf{a} - \bar{\mathbf{a}}\| \frac{\sqrt{2\log N}}{m}. \tag{21}$$

Definition 8 (Consistency of a matrix norm)**.** *A matrix norm is called consistent on* $\mathbb{C}^{n,n}$*, if*

$$\|AB\| \leq \|A\| \, \|B\| \tag{22}$$

holds for $A, B \in \mathbb{C}^{n,n}$*.*

Lemma 9 (Consistency of the operator norm)**.** *The operator norm is consistent if the vector norm* $\|\cdot\|_\alpha$ *is defined for all* $m \in \mathbb{N}$ *and* $\|\cdot\|_\beta = \|\cdot\|_\alpha$

Proof.

$$\begin{aligned} \|AB\| &= \max_{Bx \neq 0} \frac{\|ABx\|_\alpha}{\|x\|_\alpha} = \max_{Bx \neq 0} \frac{\|ABx\|_\alpha}{\|Bx\|_\alpha} \frac{\|Bx\|_\alpha}{\|x\|_\alpha} \\ &\leq \max_{y \neq 0} \frac{\|Ay\|_\alpha}{\|y\|_\alpha} \max_{x \neq 0} \frac{\|Bx\|_\alpha}{\|x\|_\alpha} = \|A\| \, \|B\| \, . \end{aligned} \tag{23}$$

□

Lemma 10.

$$(AC) * (BD) = (A \otimes B)(C * D) \tag{24}$$

C Details on polynomial networks

In this section, we provide further details on the two most prominent parametrizations proposed in Chrysos et al. (2020). This re-parametrization creates equivalent models in terms of models, but enables us to absorb the bias terms into the input terms. Firstly, we provide the re-parametrization of the CCP model in Appendix C.1 and then we create the re-parametrization of the NCP model in Appendix C.2.

C.1 Re-parametrization of CCP model

The Coupled CP-Decomposition (CCP) model of PNs (Chrysos et al., 2020) leverages a coupled CP Tensor decomposition. A *k-degree CCP model* $f(\zeta)$ can be succinctly described by the following recursive relations:

$$y_1 = V_1 \zeta, \qquad y_n = (V_n \zeta) \circ y_{n-1} + y_{n-1}, \qquad f(\zeta) = Q y_k + \beta, \tag{25}$$

where $\zeta \in \mathbb{R}^\delta$ is the input data with $\delta \in \mathbb{N}$, $f(\zeta) \in \mathbb{R}^o$ is the output of the model and $V_n \in \mathbb{R}^{\mu \times \delta}$, $Q \in \mathbb{R}^{o \times \mu}$ and $\beta \in \mathbb{R}^o$ are the learnable parameters, where $\mu \in \mathbb{N}$ is the hidden rank. In order to simplify the bias terms in the model, we will introduce a minor re-parametrization in Lemma 11 that we will use to present our results in the subsequent sections.

Lemma 11. *Let* $z = [\zeta^T, 1]^T \in \mathbb{R}^d$*,* $x_n = [y_n^T, 1]^T \in \mathbb{R}^m$*,* $C = [Q, \beta] \in \mathbb{R}^{o \times m}, d = \delta + 1, m = \mu + 1$*. Define:*

$$U_1 = \begin{bmatrix} V_1 & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \in \mathbb{R}^{m \times d}, \qquad U_i = \begin{bmatrix} V_i & \mathbf{1} \\ \mathbf{0}^T & 1 \end{bmatrix} \in \mathbb{R}^{m \times d} \qquad (i > 1) \tag{26}$$

where the boldface numbers $\mathbf{0}$ *and* $\mathbf{1}$ *denote all-zeros and all-ones column vectors of appropriate size, respectively. The CCP model in Eq. (25) can be rewritten as* $f(z) = C \circ_{i=1}^k U_i z$*, which is the one used in Eq. (CCP).*

As a reminder before providing the proof, the core symbols for this proof are summarized in Table 6.

Proof. By definition, we have:

$$x_1 = [y_1^T, 1]^T = \begin{bmatrix} y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} V_1 \zeta \\ 1 \end{bmatrix} = \begin{bmatrix} V_1 & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \zeta \\ 1 \end{bmatrix} = \begin{bmatrix} V_1 & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} [\zeta^T, 1]^T = U_1 z. \tag{27}$$

Table 6: Core symbols in the proof of Lemma 11.

Symbol	Dimensions	Definition
$\circ$	-	Hadamard (element-wise) product.
ζ	$\mathbb{R}^{\delta}$	Input of the polynomial expansion.
$f(\zeta)$	$\mathbb{R}^{o}$	Output of the polynomial expansion.
k	$\mathbb{N}$	Degree of polynomial expansion.
m	$\mathbb{N}$	Hidden rank of the expansion.
V_n	$\mathbb{R}^{\mu\times\delta}$	Learnable matrices of the expansion.
Q	$\mathbb{R}^{o\times\mu}$	Learnable matrix of the expansion.
β	$\mathbb{R}^{o}$	Bias of the expansion.
z	$\mathbb{R}^{d}$	Re-parametrization of the input.
C	$\mathbb{R}^{o\times m}$	$C = (Q, \beta)$.

$$
\begin{aligned}
x_n = [y_n^T, 1]^T = \begin{bmatrix} y_n \\ 1 \end{bmatrix} = \begin{bmatrix} (V_n\zeta)\circ y_{n-1} + y_{n-1} \\ 1 \end{bmatrix} = \begin{bmatrix} V_n\zeta + \mathbf{1} \\ 1 \end{bmatrix} \circ \begin{bmatrix} y_{n-1} \\ 1 \end{bmatrix} \\
= \begin{bmatrix} V_n & \mathbf{1} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \zeta \\ 1 \end{bmatrix} \circ \begin{bmatrix} y_{n-1} \\ 1 \end{bmatrix} = \begin{bmatrix} V_n & \mathbf{1} \\ \mathbf{0}^T & 1 \end{bmatrix} [\zeta^T, 1]^T \circ [y_{n-1}^T, 1]^T = U_n z \circ x_{n-1}.
\end{aligned} \tag{28}
$$

Hence, it holds that:

$$
\begin{aligned}
f(z) = Qy_k + \beta = (Q, \beta) \begin{bmatrix} y_k \\ 1 \end{bmatrix} = Cx_k \\
&= C\, U_k z \circ x_{k-1} \\
&= C\, U_k z \circ (U_{k-1} z) \circ x_{k-2} \\
&= \cdots \\
&= C\, U_k z \circ (U_{k-1} z) \circ \cdots \circ (U_2 z) \circ x_1 \\
&= C\, U_k z \circ (U_{k-1} z) \circ \cdots \circ (U_2 z) \circ (U_1 z) \\
&= C \circ_{i=1}^{k} (U_i z)
\end{aligned} \tag{29}
$$

□

C.2 Reparametrization of the NCP model

The nested coupled CP decomposition (NCP) model of PNs (Chrysos et al., 2020) leverages a joint hierarchical decomposition. A *k-degree NCP model* $f(\zeta)$ is expressed with the following recursive relations:

$$
y_1 = (V_1\zeta)\circ(b_1), \qquad y_n = (V_n\zeta)\circ(U_n y_{n-1} + b_n), \qquad f(\zeta) = Qy_k + \beta \tag{30}
$$

where $\zeta \in \mathbb{R}^{\delta}$ is the input data with $\delta \in \mathbb{N}$, $f(\zeta) \in \mathbb{R}^{o}$ is the output of the model and $V_n \in \mathbb{R}^{\mu\times\delta}$, $b_n \in \mathbb{R}^{\mu}$, $U_n \in \mathbb{R}^{\mu\times\mu}$, $Q \in \mathbb{R}^{o\times\mu}$ and $\beta \in \mathbb{R}^{o}$ are the learnable parameters, where $\mu \in \mathbb{N}$ is the hidden rank. In order to simplify the bias terms in the model, we will introduce a minor re-parametrization in Lemma 12 that we will use to present our results in the subsequent sections.

Lemma 12. *Let $z = [\zeta^T, 1]^T \in \mathbb{R}^d$, $x_n = [y_n^T, 1]^T \in \mathbb{R}^m$, $C = [Q, \beta] \in \mathbb{R}^{o\times m}, d = \delta + 1, m = \mu + 1$. Let:*

$$
s_1 = [b_1^T, 1]^T \in \mathbb{R}^m, \quad S_i = \begin{bmatrix} U_i & b_i \\ \mathbf{0}^T & 1 \end{bmatrix} \in \mathbb{R}^{m\times m} (i > 1), \quad A_i = \begin{bmatrix} V_i & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \in \mathbb{R}^{m\times d} \tag{31}
$$

where the boldface numbers $\mathbf{0}$ and $\mathbf{1}$ denote all-zeros and all-ones column vectors of appropriate size, respectively. The NCP model in Eq. (30) can be rewritten as

$$
x_1 = (A_1 z)\circ(s_1), \qquad x_n = (A_n z)\circ(S_n x_{n-1}), \qquad f(z) = Cx_k. \tag{32}
$$

In the aforementioned Eq. (32), we have written S_n even for $n = 1$, when s_1 is technically a vector, but this is done for convenience only and does not change the end result.

D RESULT OF THE CCP MODEL

D.1 PROOF OF THEOREM 1: RADEMACHER COMPLEXITY BOUND OF CCP UNDER ℓ_∞ NORM

To facilitate the proof below, we include the related symbols in Table 7. Below, to avoid cluttering the notation, we consider that the expectation is over σ and omit the brackets as well.

Table 7: Core symbols for proof of Theorem 1.

Symbol	Dimensions	Definition
$\circ$	-	Hadamard (element-wise) product.
$\bullet$	-	Face-splitting product.
$*$	-	Column-wise Khatri–Rao product.
z	$\mathbb{R}^d$	Input of the polynomial expansion.
$f(z)$	$\mathbb{R}$	Output of the polynomial expansion.
k	$\mathbb{N}$	Degree of polynomial expansion.
m	$\mathbb{N}$	Hidden rank of the expansion.
U_i	$\mathbb{R}^{m\times d}$	Learnable matrices.
c	$\mathbb{R}^{1\times m}$	Learnable matrix.
μ	$\mathbb{R}$	$\|c\|_1 \leq \mu$.
λ	$\mathbb{R}$	$\left\|\bullet_{i=1}^k (U_i)\right\|_\infty \leq \lambda$.

Proof.

$$
\begin{aligned}
\mathcal{R}_Z(\mathcal{F}^k_{\text{CCP}}) &= \mathbb{E} \sup_{f\in\mathcal{F}^k_{\text{CCP}}} \frac{1}{n}\sum_{j=1}^n \sigma_j f(z_j) \\
&= \mathbb{E} \sup_{f\in\mathcal{F}^k_{\text{CCP}}} \frac{1}{n}\sum_{j=1}^n \left(\sigma_j \left\langle c, \circ_{i=1}^k (U_i z_j)\right\rangle\right) \\
&= \mathbb{E} \sup_{f\in\mathcal{F}^k_{\text{CCP}}} \frac{1}{n}\left\langle c, \sum_{j=1}^n [\sigma_j \circ_{i=1}^k (U_i z_j)]\right\rangle \\
&\leq \mathbb{E} \sup_{f\in\mathcal{F}^k_{\text{CCP}}} \frac{1}{n}\|c\|_1 \left\|\sum_{j=1}^n [\sigma_j \circ_{i=1}^k (U_i z_j)]\right\|_\infty && \text{Lemma 5 [Hölder's inequality]} \\
&= \mathbb{E} \sup_{f\in\mathcal{F}^k_{\text{CCP}}} \frac{1}{n}\|c\|_1 \left\|\sum_{j=1}^n [\sigma_j \bullet_{i=1}^k (U_i) *_{i=1}^k (z_j)]\right\|_\infty && \text{Lemma 7 [Mixed product property]} \\
&= \mathbb{E} \sup_{f\in\mathcal{F}^k_{\text{CCP}}} \frac{1}{n}\|c\|_1 \left\|\bullet_{i=1}^k (U_i) \sum_{j=1}^n [\sigma_j *_{i=1}^k (z_j)]\right\|_\infty \\
&\leq \mathbb{E} \sup_{f\in\mathcal{F}^k_{\text{CCP}}} \frac{1}{n}\|c\|_1 \left\|\sum_{j=1}^n [\sigma_j *_{i=1}^k (z_j)]\right\|_\infty \left\|\bullet_{i=1}^k (U_i)\right\|_\infty \\
&\leq \frac{\mu\lambda}{n}\mathbb{E}\left\|\sum_{j=1}^n [\sigma_j *_{i=1}^k (z_j)]\right\|_\infty
\end{aligned}
\tag{33}
$$

Next, we compute the bound of $\mathbb{E}\left\|\sum_{j=1}^n [\sigma_j *_{i=1}^k (z_j)]\right\|_\infty$.

Let $Z_j = *_{i=1}^k(z_j) \in \mathbb{R}^{d^k}$. For each $l \in [d^k]$, let $v_l = (Z_1^l, \ldots, Z_n^l) \in \mathbb{R}^n$. Note that $\|v_l\|_2 \leq \sqrt{n}\ \max_j \|Z_j\|_\infty$. Let $V = \{v_1, \ldots, v_n\}$. Then, it is true that:

$$\mathbb{E}\left\|\sum_{j=1}^n [\sigma_j *_{i=1}^k (z_j)]\right\|_\infty = \mathbb{E}\left\|\sum_{j=1}^n \sigma_j Z_j\right\|_\infty = \mathbb{E}\max_{l=1}^{2^k}\left|\sum_{j=1}^n \sigma_j (v_l)_j\right| = n\mathcal{R}(V). \tag{34}$$

Using Lemma 8 [Massart Lemma] we have that:

$$\mathcal{R}(V) \leq 2\max_j \|Z_j\|_\infty \sqrt{2\log(d^k)/n}. \tag{35}$$

Then, it holds that:

$$\begin{aligned}
\mathcal{R}_Z(\mathcal{F}_{\text{CCP}}^k) &= \mathbb{E}\sup_{f\in\mathcal{F}_{\text{CCP}}^k} \frac{1}{n}\sum_{j=1}^n \sigma_j f(z_j) \\
&\leq \frac{\mu\lambda}{n}\mathbb{E}\left\|\sum_{j=1}^n [\sigma_j *_{i=1}^k (z_j)]\right\|_\infty && \text{Eq. (33)} \\
&= \frac{\mu\lambda}{n} n\mathcal{R}(V) && \text{Eq. (34)} \\
&\leq 2\mu\lambda \max_j \|Z_j\|_\infty \sqrt{2\log(d^k)/n} && \text{Eq. (35)} \\
&= 2\mu\lambda \max_j \left\|*_{i=1}^k (z_j)\right\|_\infty \sqrt{2\log(d^k)/n} \\
&\leq 2\mu\lambda (\max_j \|z_j\|_\infty)^k \sqrt{2\log(d^k)/n} \\
&\leq 2\mu\lambda\sqrt{2k\log(d)/n}
\end{aligned} \tag{36}$$

□

D.2 Proof of Lemma 1

Table 8: Core symbols in the proof of Lemma 1.

Symbol	Dimensions	Definition
$\otimes$	-	Kronecker product.
$\bullet$	-	Face-splitting product.
z	$\mathbb{R}^d$	Input of the polynomial expansion.
$f(z)$	$\mathbb{R}$	Output of the polynomial expansion.
k	$\mathbb{N}$	Degree of polynomial expansion.
m	$\mathbb{N}$	Hidden rank of the expansion.
U_i	$\mathbb{R}^{m\times d}$	Learnable matrices.
U_i^j	$\mathbb{R}^d$	j^{th} row of U_i.
λ_i	$\mathbb{R}$	$\|U_i\|_\infty \leq \lambda_i$ for $i = 1, 2, \ldots, k$.

Proof.

$$
\begin{aligned}
\left\|\bullet_{i=1}^{k}(U_i)\right\|_\infty &= \max_{j=1}^{m} \left\|[\bullet_{i=1}^{k}(U_i)]^j\right\|_1 \\
&= \max_{j=1}^{m} \left\|\otimes_{i=1}^{k}[U_i^j]\right\|_1 \quad \text{Definition of Face-splitting product} \\
&= \max_{j=1}^{m} \left[\prod_{i=1}^{k}\left\|U_i^j\right\|_1\right] \quad \text{Multiplicativity of absolute value} \\
&\leq \prod_{i=1}^{k}\left[\max_{j=1}^{m}\left\|U_i^j\right\|_1\right] \\
&= \prod_{i=1}^{k}\|U_i\|_\infty
\end{aligned}
\tag{37}
$$

□

D.3 Rademacher Complexity bound under ℓ_2 norm

Theorem 6. *Let $Z = \{z_1, \ldots, z_n\} \subseteq \mathbb{R}^d$ and suppose that $\|z_j\|_\infty \leq 1$ for all $j = 1, \ldots, n$. Let*

$$
\mathcal{F}_{CCP}^{k} \coloneqq \left\{f(z) = \left\langle c, \circ_{i=1}^{k} U_i z\right\rangle : \|c\|_2 \leq \mu, \left\|\bullet_{i=1}^{k} U_i\right\|_2 \leq \lambda\right\} \tag{38}
$$

The Empirical Rademacher Complexity of CCP$_k$ (k-degree CCP polynomials) with respect to Z is bounded as:

$$
\mathcal{R}_Z(\mathcal{F}_{CCP}^{k}) \leq \frac{\mu\lambda}{\sqrt{n}} \tag{39}
$$

To facilitate the proof below, we include the related symbols in Table 9. Below, to avoid cluttering the notation, we consider that the expectation is over σ and omit the brackets as well.

Table 9: Core symbols for proof of Theorem 6.

Symbol	Dimensions	Definition
$\circ$	-	Hadamard (element-wise) product.
$\bullet$	-	Face-splitting product.
$*$	-	Column-wise Khatri–Rao product.
z	$\mathbb{R}^d$	Input of the polynomial expansion.
$f(z)$	$\mathbb{R}$	Output of the polynomial expansion.
k	$\mathbb{N}$	Degree of polynomial expansion.
m	$\mathbb{N}$	Hidden rank of the expansion.
U_i	$\mathbb{R}^{m\times d}$	Learnable matrices.
c	$\mathbb{R}^{1\times m}$	Learnable matrix.
μ	$\mathbb{R}$	$\|c\|_2 \leq \mu$.
λ	$\mathbb{R}$	$\left\|\bullet_{i=1}^{k}(U_i)\right\|_2 \leq \lambda$.

Proof.

$$
\begin{aligned}
\mathcal{R}_Z(\mathcal{F}^k_{\text{CCP}}) &= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{CCP}}} \frac{1}{n} \sum_{j=1}^{n} \sigma_j f(z_j) \\
&= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{CCP}}} \frac{1}{n} \sum_{j=1}^{n} \left(\sigma_j \left\langle c, \circ_{i=1}^k (U_i z_j) \right\rangle\right) \\
&= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{CCP}}} \frac{1}{n} \left\langle c, \sum_{j=1}^{n} [\sigma_j \circ_{i=1}^k (U_i z_j)] \right\rangle \\
&\leq \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{CCP}}} \frac{1}{n} \|c\|_2 \left\| \sum_{j=1}^{n} [\sigma_j \circ_{i=1}^k (U_i z_j)] \right\|_2 && \text{Lemma 5 [Hölder's inequality]} \\
&= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{CCP}}} \frac{1}{n} \|c\|_2 \left\| \sum_{j=1}^{n} [\sigma_j \bullet_{i=1}^k (U_i) *_{i=1}^k (z_j)] \right\|_2 && \text{Lemma 7 [Mixed product property]} \\
&= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{CCP}}} \frac{1}{n} \|c\|_2 \left\| \bullet_{i=1}^k (U_i) \sum_{j=1}^{n} [\sigma_j *_{i=1}^k (z_j)] \right\|_2 \\
&\leq \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{CCP}}} \frac{1}{n} \|c\|_2 \left\| \sum_{j=1}^{n} [\sigma_j *_{i=1}^k (z_j)] \right\|_2 \left\| \bullet_{i=1}^k (U_i) \right\|_2
\end{aligned}
\tag{40}
$$

$$
\begin{aligned}
\mathbb{E} \left\| \sum_{j=1}^{n} [\sigma_j *_{i=1}^k (z_j)] \right\|_2 &= \mathbb{E} \sqrt{\left\| \sum_{j=1}^{n} [\sigma_j *_{i=1}^k (z_j)] \right\|_2^2} \\
&\leq \sqrt{\mathbb{E} \left\| \sum_{j=1}^{n} [\sigma_j *_{i=1}^k (z_j)] \right\|_2^2} && \text{Jensen's inequality} \\
&= \sqrt{\mathbb{E} \sum_{s,j}^{n} [\sigma_s \sigma_j < *_{i=1}^k (z_s), *_{i=1}^k (z_j) >]} \\
&= \sqrt{\sum_{j=1}^{n} [\left\| *_{i=1}^k (z_j) \right\|_2^2]} \\
&= \sqrt{\sum_{j=1}^{n} (\prod_{i=1}^{k} \|z_j\|_2^2)} \\
&\leq \sqrt{n}
\end{aligned}
\tag{41}
$$

So:

$$
\begin{aligned}
\mathcal{R}_Z(\mathcal{F}^k_{\text{CCP}}) &\leq \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{CCP}}} \frac{1}{n} \|c\|_2 \left\| \sum_{j=1}^{n} [\sigma_j *_{i=1}^k (z_j)] \right\|_2 \left\| \bullet_{i=1}^k (U_i) \right\|_2 \\
&\leq \frac{\sup_{f \in \mathcal{F}^k_{\text{CCP}}} \|c\|_2 \left\| \bullet_{i=1}^k (U_i) \right\|_2}{\sqrt{n}} \\
&\leq \frac{\mu \lambda}{\sqrt{n}}
\end{aligned}
\tag{42}
$$

□

D.4 LIPSCHITZ CONSTANT BOUND OF THE CCP MODEL

We will first prove a more general result about the ℓ_p-Lipschitz constant of the CCP model.

Theorem 7. *The Lipschitz constant (with respect to the ℓ_p-norm) of the function defined in Eq. (CCP), restricted to the set $\{z \in \mathbb{R}^d : \|z\|_p \leq 1\}$ is bounded as:*

$$\mathit{Lip}_p(f) \leq k\|C\|_p \prod_{i=1}^{k} \|U_i\|_p \tag{43}$$

Proof. Let $g(x) = Cx$ and $h(z) = \circ_{i=1}^{k}(U_i z)$. Then it holds that $f(z) = g(h(z))$. By Lemma 3, we have: $\text{Lip}_p(f) \leq \text{Lip}_p(g)\text{Lip}_p(h)$. We will compute an upper bound of each function individually.

Let us first consider the function $g(x) = Cx$. By Lemma 4, because g is a linear map represented by a matrix C, its Jacobian is $J_g(x) = C$. So:

$$\text{Lip}_p(g) = \|C\|_p := \sup_{\|x\|_p=1} \|Cx\|_p \tag{44}$$

where $\|C\|_p$ is the operator norm on matrices induced by the vector p-norm.

Now, let us consider the function $h(z) = \circ_{i=1}^{k} U_i z$. Its Jacobian is given by:

$$\frac{dh}{dz} = \sum_{i=1}^{k} \text{diag}(\circ_{j\neq i} U_j z) U_i \tag{45}$$

Using Lemma 4 we have:

$$\begin{aligned}
\text{Lip}_p(h) &\leq \sup_{z:\|z\|_p\leq 1} \left\| \sum_{i=1}^{k} [\text{diag}(\circ_{j\neq i}(U_j z))U_i] \right\|_p \\
&\leq \sup_{z:\|z\|_p\leq 1} \sum_{i=1}^{k} \|\text{diag}(\circ_{j\neq i}(U_j z))U_i\|_p && \text{Triangle inequality} \\
&\leq \sup_{z:\|z\|_p\leq 1} \sum_{i=1}^{k} \|\text{diag}(\circ_{j\neq i}(U_j z))\|_p \|U_i\|_p && \text{Lemma 9 [consistency]} \\
&\leq \sup_{z:\|z\|_p\leq 1} \sum_{i=1}^{k} \|\circ_{j\neq i}(U_j z)\|_p \|U_i\|_p \\
&\leq \sup_{z:\|z\|_p\leq 1} \sum_{i=1}^{k} \prod_{j\neq i} (\|U_j z\|_p) \|U_i\|_p \\
&\leq \sup_{z:\|z\|_p\leq 1} \sum_{i=1}^{k} \prod_{j\neq i} (\|U_j\|_p \|z\|_p) \|U_i\|_p \\
&= \sup_{z:\|z\|_p\leq 1} \sum_{i=1}^{k} \prod_{j=1}^{k} (\|U_j\|_p) \\
&= k \prod_{j=1}^{k} \|U_j\|_p
\end{aligned} \tag{46}$$

So:

$$\begin{aligned} Lip_p(\mathcal{F}_L) &\leq Lip_p(g) Lip_p(h) \\ &\leq k\|C\|_p \prod_{i=1}^{k} \|U_i\|_p \end{aligned} \tag{47}$$

□

D.4.1 Proof of Theorem 2

Proof. This is a particular case of Theorem 7 when $p = \infty$. □

E Result of the NCP model

E.1 Proof of Theorem 3: Rademacher Complexity of NCP under ℓ_∞ norm

Proof.

$$\begin{aligned} \mathcal{R}_Z(\mathcal{F}^k_{\text{NCP}}) &= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \sum_{j=1}^{n} \sigma_j f(z_j) \\ &= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \sum_{j=1}^{n} (\sigma_j \langle c, x_k(z_j) \rangle) \\ &= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \left\langle c, \sum_{j=1}^{n} [\sigma_j x_k(z_j)] \right\rangle \\ &\leq \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \|c\|_1 \left\| \sum_{j=1}^{n} [\sigma_j x_k(z_j)] \right\|_\infty && \text{Lemma 5 [Hölder's inequality]} \\ &= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \|c\|_1 \left\| \sum_{j=1}^{n} [\sigma_j ((A_k z_j) \circ (S_k x_{k-1}(z_j)))] \right\|_\infty \\ &= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \|c\|_1 \left\| \sum_{j=1}^{n} [\sigma_j ((A_k \bullet S_k)(z_j * x_{k-1}(z_j)))] \right\|_\infty && \text{Lemma 7 [Mixed product property]} \\ &= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \|c\|_1 \left\| (A_k \bullet S_k) \sum_{j=1}^{n} [\sigma_j (z_j * x_{k-1}(z_j))] \right\|_\infty \end{aligned} \tag{48}$$

Now, because of the recursive definition of the Eq. (NCP), we obtain:

$$\begin{aligned} \sum_{j=1}^{n} \sigma_j (z_j * x_{k-1}(z_j)) &= \sum_{j=1}^{n} \sigma_j (z_j * (A_{k-1} z_j) \circ (S_{k-1} x_{k-2}(z_j))) \\ &= \sum_{j=1}^{n} \sigma_j (z_j * ((A_{k-1} \bullet S_{k-1})(z_j * x_{k-2}(z_j)))) && \text{Lemma 7} \\ &= \sum_{j=1}^{n} \sigma_j (I \otimes (A_{k-1} \bullet S_{k-1}))(z_j * (z_j * x_{k-2}(z_j))) && \text{Lemma 10} \\ &= I \otimes (A_{k-1} \bullet S_{k-1}) \sum_{j=1}^{n} [\sigma_j (z_j * (z_j * x_{k-2}(z_j))) \end{aligned} \tag{49}$$

recursively applying this argument we have:

$$\sum_{j=1}^{n} \sigma_j (z_j * x_{k-1}(z_j)) = \left(\prod_{i=1}^{k-1} I \otimes A_i \bullet S_i \right) \sum_{j=1}^{n} \sigma_j *_{i=1}^{k} (z_j) \tag{50}$$

Combining the two previous equations (Eqs. (49) and (50)) inside Eq. (48) we finally obtain

$$\begin{aligned}
\mathcal{R}_Z(\mathcal{F}^k_{\text{NCP}}) &\leq \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \|c\|_1 \left\| (A_k \bullet S_k) \left(\prod_{i=1}^{k-1} I \otimes A_i \bullet S_i \right) \sum_{j=1}^{n} \sigma_j *_{i=1}^{k} (z_j) \right\|_\infty \\
&\leq \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \|c\|_1 \left\| (A_k \bullet S_k) \left(\prod_{i=1}^{k-1} I \otimes A_i \bullet S_i \right) \right\|_\infty \left\| \sum_{j=1}^{n} \sigma_j *_{i=1}^{k} (z_j) \right\|_\infty \\
&\leq \frac{\mu\lambda}{n} \mathbb{E} \left\| \sum_{j=1}^{n} \sigma_j *_{i=1}^{k} (z_j) \right\|_\infty \\
&= \frac{\mu\lambda}{n} n \mathcal{R}(V) = \mu\lambda \mathcal{R}(V) && \text{Eq. (34)}
\end{aligned} \tag{51}$$

following the same arguments as in Eq. (36) it follows that:

$$\mathcal{R}_Z(\mathcal{F}^k_{\text{NCP}}) \leq 2\mu\lambda \sqrt{\frac{2k \log(d)}{n}} \tag{52}$$

□

E.2 Proof of Lemma 2

Proof.

$$\begin{aligned}
\left\| (A_k \bullet S_k) \prod_{i=1}^{k-1} I \otimes A_i \bullet S_i \right\|_\infty &\leq \|A_k \bullet S_k\|_\infty \prod_{i=1}^{k-1} \|I \otimes A_i \bullet S_i\|_\infty && \text{Lemma 9 [consistent]} \\
&= \prod_{i=1}^{k} \|A_i \bullet S_i\|_\infty \\
&\leq \prod_{i=1}^{k} \|A_i\|_\infty \|S_i\|_\infty && \text{Lemma 1}
\end{aligned} \tag{53}$$

□

E.3 Rademacher Complexity under ℓ_2 norm

Theorem 8. *Let $Z = \{z_1, \ldots, z_n\} \subseteq \mathbb{R}^d$ and suppose that $\|z_j\|_\infty \leq 1$ for all $j = 1, \ldots, n$. Define the matrix $\Phi(A_1, S_1, \ldots, A_n, S_n) := (A_k \bullet S_k) \prod_{i=1}^{k-1} I \otimes A_i \bullet S_i$. Consider the class of functions:*

$$\mathcal{F}^k_{NCP} := \{ f(z) \text{ as in (NCP)} : \|C\|_2 \leq \mu, \|\Phi(A_1, S_1, \ldots, A_k, S_k)\|_2 \leq \lambda \} \tag{54}$$

where $C \in \mathbb{R}^{1 \times m}$ (single output case). The Empirical Rademacher Complexity of NCP$_k$ (k-degree NCP polynomials) with respect to Z is bounded as:

$$\mathcal{R}_Z(\mathcal{F}^k_{NCP}) \leq \frac{\mu\lambda}{\sqrt{n}} \tag{55}$$

Proof.

$$
\begin{aligned}
\mathcal{R}_Z(\mathcal{F}^k_{\text{NCP}}) &= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \sum_{j=1}^{n} \sigma_j f(z_j) \\
&= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \sum_{j=1}^{n} \left(\sigma_j \left\langle c, x_k(z_j) \right\rangle\right) \\
&= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \left\langle c, \sum_{j=1}^{n} [\sigma_j x_k(z_j)] \right\rangle \\
&\leq \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \|c\|_2 \left\| \sum_{j=1}^{n} [\sigma_j x_k(z_j)] \right\|_2 && \text{Lemma 5 [Hölder's inequality]} \\
&. = \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \|c\|_2 \left\| \sum_{j=1}^{n} [\sigma_j ((A_k z_j) \circ (S_k x_{k-1}(z_j)))] \right\|_2 \\
&= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \|c\|_2 \left\| \sum_{j=1}^{n} [\sigma_j ((A_k \bullet S_k)(z_j * x_{k-1}(z_j)))] \right\|_2 && \text{Lemma 7 [Mixed product property]} \\
&= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \|c\|_2 \left\| (A_k \bullet S_k) \sum_{j=1}^{n} [\sigma_j (z_j * x_{k-1}(z_j))] \right\|_2 \\
&= \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \|c\|_2 \left\| (A_k \bullet S_k) \left(\prod_{i=1}^{k-1} I \otimes A_i \bullet S_i \right) \sum_{j=1}^{n} \sigma_j *_{i=1}^{k} (z_j)] \right\|_2 && \text{Eq. (50)} \\
&\leq \mathbb{E} \sup_{f \in \mathcal{F}^k_{\text{NCP}}} \frac{1}{n} \|c\|_2 \left\| (A_k \bullet S_k) \left(\prod_{i=1}^{k-1} I \otimes A_i \bullet S_i \right) \right\|_2 \left\| \sum_{j=1}^{n} \sigma_j *_{i=1}^{k} (z_j)] \right\|_2 \\
&\leq \frac{\mu\lambda}{n} \mathbb{E} \left\| \sum_{j=1}^{n} \sigma_j *_{i=1}^{k} (z_j) \right\|_2 \\
&\leq \frac{\mu\lambda}{\sqrt{n}} && \text{Eq. (41)}
\end{aligned}
\tag{56}
$$

□

E.4 Lipschitz constant bound of the NCP model

Theorem 9. *Let $\mathcal{F}_L$ be the class of functions defined as*

$$
\mathcal{F}_L \coloneqq \{x_1 = (A_1 z) \circ (S_1), x_n = (A_n z) \circ (S_n x_{n-1}), f(z) = C x_k : \|C\|_p \leq \mu, \|A_i\|_p \leq \lambda_i, \|S_i\|_p \leq \rho_i, \|z\|_p \leq 1\}. \tag{57}
$$

The Lipschitz Constant of $\mathcal{F}_{\mathcal{L}}$ (k-degree NCP polynomial) under ℓ_p norm restrictions is bounded as:

$$
Lip_p(\mathcal{F}_L) \leq k\mu \prod_{i=1}^{k} (\lambda_i \rho_i). \tag{58}
$$

Proof. Let $g(x) = Cx$, $h(z) = (A_n z) \circ (S_n x_{n-1}(z))$. Then it holds that $f(z) = g(h(z))$.

By Lemma 3, we have: $\text{Lip}(f) \leq \text{Lip}(g)\text{Lip}(h)$. This enables us to compute an upper bound of each function (i.e., g, h) individually.

Let us first consider the function $g(x) = Cx$. By Lemma 4, because g is a linear map represented by a matrix C, its Jacobian is $J_g(x) = C$. So:

$$\text{Lip}_p(g) = \|C\|_p := \sup_{\|x\|_p=1} \|Cx\|_p = \begin{cases} \sigma_{\max}(C) & \text{if } p = 2 \\ \max_i \sum_j |C_{(i,j)}| & \text{if } p = \infty \end{cases} \tag{59}$$

where $\|C\|_p$ is the operator norm on matrices induced by the vector p-norm, and $\sigma_{\max}(C)$ is the largest singular value of C.

Now, let us consider the function $x_n(z) = h(z) = (A_n z) \circ (S_n x_{n-1}(z))$. Its Jacobian is given by:

$$J_{x_n} = \text{diag}(A_n z) S_n J_{x_{n-1}} + \text{diag}(S_n x_{n-1}) A_n, \qquad J_{x_1} = \text{diag}(S_1) A_1 \tag{60}$$

$$\begin{aligned}
Lip_p(h) &= \sup_{z:\|z\|_p \le 1} \|J_{x_n}\|_p \\
&= \sup_{z:\|z\|_p \le 1} \|\text{diag}(A_n z) S_n J_{x_{n-1}} + \text{diag}(S_n x_{n-1}) A_n\|_p \\
&\le \sup_{z:\|z\|_p \le 1} \|\text{diag}(A_n z) S_n J_{x_{n-1}}\|_p + \|\text{diag}(S_n x_{n-1}) A_n\|_p \qquad \text{Triangle inequality} \\
&\le \sup_{z:\|z\|_p \le 1} \|\text{diag}(A_n z)\|_p \|S_n\|_p \|J_{x_{n-1}}\|_p + \|\text{diag}(S_n x_{n-1})\|_p \|A_n\|_p \qquad \text{Lemma 9 [consistent]} \\
&\le \sup_{z:\|z\|_p \le 1} \|A_n z\|_p \|S_n\|_p \|J_{x_{n-1}}\|_p + \|S_n x_{n-1}\|_p \|A_n\|_p \\
&\le \sup_{z:\|z\|_p \le 1} \|A_n\|_p \|z\|_p \|S_n\|_p \|J_{x_{n-1}}\|_p + \|S_n\|_p \|x_{n-1}\|_p \|A_n\|_p \\
&= \sup_{z:\|z\|_p \le 1} \|A_n\|_p \|z\|_p \|S_n\|_p \|J_{x_{n-1}}\|_p + \|S_n\|_p \|(A_{n-1} z) \circ (S_{n-1} x_{n-2})\|_p \|A_n\|_p \\
&\le \sup_{z:\|z\|_p \le 1} \|A_n\|_p \|z\|_p \|S_n\|_p \|J_{x_{n-1}}\|_p + \|S_n\|_p \|A_{n-1} z\|_p \|S_{n-1} x_{n-2}\|_p \|A_n\|_p \\
&\le \sup_{z:\|z\|_p \le 1} \|A_n\|_p \|z\|_p \|S_n\|_p \|J_{x_{n-1}}\|_p + \|S_n\|_p \|A_{n-1}\|_p \|z\|_p \|S_{n-1}\|_p \|x_{n-2}\|_p \|A_n\|_p \\
&= \sup_{z:\|z\|_p \le 1} \|A_n\|_p \|z\|_p \|S_n\|_p (\|J_{x_{n-1}}\|_p + \|A_{n-1}\|_p \|S_{n-1}\|_p \|x_{n-2}\|_p) \\
&\le \sup_{z:\|z\|_p \le 1} \|A_n\|_p \|z\|_p \|S_n\|_p (\|J_{x_{n-1}}\|_p + \prod_{i=1}^{n-1} (\|S_i\|_p \|A_i\|_p) \|z\|_p^{n-2})
\end{aligned} \tag{61}$$

Then we proof the result by induction.

Inductive hypothesis:

$$\sup_{z:\|z\|_p \le 1} \|J_{x_n}\|_p \le n \prod_{i=1}^{n} (\|S_i\|_p \|A_i\|_p) \tag{62}$$

Case $k = 1$:

$$\begin{aligned}
Lip_p(h) &= \sup_{z:\|z\|_p \le 1} \|J_{x_1}\|_p \\
&= \|\text{diag}(S_1) A_1\|_p \\
&\le \|\text{diag}(S_1)\|_p \|A_1\|_p \\
&\le \|S_1\|_p \|A_1\|_p
\end{aligned} \tag{63}$$

Case $k = n$:

$$
\begin{aligned}
Lip_p(h) &= \sup_{z:\|z\|_p \leq 1} \|J_{x_n}\|_p \\
&\leq \sup_{z:\|z\|_p \leq 1} \|A_n\|_p \|z\|_p \|S_n\|_p (\|J_{x_{n-1}}\|_p + \prod_{i=1}^{n-1} (\|S_i\|_p \|A_i\|_p) \|z\|_p^{n-2}) \\
&\leq \sup_{z:\|z\|_p \leq 1} \|A_n\|_p \|z\|_p \|S_n\|_p ((n-1) \prod_{i=1}^{n-1} (\|S_i\|_p \|A_i\|_p) + \prod_{i=1}^{n-1} (\|S_i\|_p \|A_i\|_p) \|z\|_p^{n-2}) \\
&\leq n \prod_{i=1}^{n} (\|S_i\|_p \|A_i\|_p)
\end{aligned}
\tag{64}
$$

So:

$$
\begin{aligned}
Lip_p(\mathcal{F}_L) &\leq Lip_p(g) Lip_p(h) \\
&\leq k \|C\|_p \prod_{i=1}^{k} (\|S_i\|_p \|A_i\|_p)
\end{aligned}
\tag{65}
$$

□

E.4.1 PROOF OF THEOREM 4

Proof. This is particular case of Theorem 9 with $p = \infty$. □

F RELATIONSHIP BETWEEN A CONVOLUTIONAL LAYER AND A FULLY CONNECTED LAYER

In this section we discuss various cases of input/output types depending on the dimensionality of the input tensor and the output tensor. We also provide the proof of Theorem 5.

Theorem 10. *Let $A \in \mathbb{R}^n$. Let $K \in \mathbb{R}^h$ be a 1-D convolutional kernel. For simplicity, we assume h is odd and $h \leq n$. Let $B \in \mathbb{R}^n$, $B = K \star A$ be the output of the convolution. Let U be the convolutional operator i.e., the linear operator (matrix) $U \in \mathbb{R}^{n \times n}$ such that $B = K \star A = UA$. It holds that $\|U\|_\infty = \|K\|_1$.*

Theorem 11. *Let $A \in \mathbb{R}^{n \times m}$, and let $K \in \mathbb{R}^{h \times h}$ be a 2-D convolutional kernel. For simplicity assume h is odd number and $h \leq \min(n, m)$. Let $B \in \mathbb{R}^{n \times m}$, $B = K \star A$ be the output of the convolution. Let U be the convolutional operator i.e., the linear operator (matrix) $U \in \mathbb{R}^{nm \times nm}$ such that $vec(B) = Uvec(A)$. It holds that $\|U\|_\infty = \|vec(K)\|_1$.*

F.1 PROOF OF THEOREM 10

Proof. From, $B = K \star A = UA$ we can obtain the following:

$$
\begin{pmatrix}
u_{1,1} & u_{1,2} & \cdots & u_{1,n} \\
u_{2,1} & u_{2,2} & \cdots & u_{2,n} \\
\vdots & \vdots & \ddots & \vdots \\
u_{n,1} & u_{n,2} & \cdots & u_{n,n}
\end{pmatrix}
(A^1, \cdots, A^n)^T = (K^1, \cdots, K^h)^T \star (A^1, \cdots, A^n)^T. \tag{66}
$$

We observe that:

$$
u_{i,j} = \begin{cases} K^{\frac{h+1}{2}+j-i} & \text{if } |i-j| \leq \frac{h-1}{2} \\ 0 & \text{if } |i-j| > \frac{h-1}{2} \end{cases}. \tag{67}
$$

Then, it holds that:

$$\|U\|_\infty = \max_{i=1}^{n} \sum_{j=1}^{n} |u_{i,j}| \leq \max_{i=1}^{n} \sum_{j=1}^{h} \left|K^j\right| = \|K\|_1 . \tag{68}$$

□

F.2 PROOF OF THEOREM 11

Proof. We partition U into $n \times n$ partition matrices of shape $m \times m$. Then the $(i,j)^{\text{th}}$ partition matrix $U_{(i,j)}$ describes the relationship between the B^i and the A^j. So, $U_{(i,j)}$ is also similar to the Toeplitz matrix in the previous result.

$$U_{(i,j)} = \begin{cases} M_{\frac{h+1}{2}+j-i} & \text{if } |i-j| \leq \frac{h-1}{2}; \\ \mathbf{0} & \text{if } |i-j| > \frac{h-1}{2}, \end{cases} \tag{69}$$

Meanwhile, the matrix M satisfies:

$$m_{i(s,l)} = \begin{cases} k_{(i,\frac{h+1}{2}+l-s)} & \text{if } |s-l| \leq \frac{h-1}{2}; \\ 0 & \text{if } |s-l| > \frac{h-1}{2}. \end{cases} \tag{70}$$

Then, we have the following:

$$\|U\|_\infty = \max_{i=1}^{n\times m} \sum_{j=1}^{n\times m} |u_{i,j}| \leq \max_{i=1}^{n} \sum_{j=1}^{n} \|U_{i,j}\|_\infty \leq \max_{i=1}^{n} \sum_{j=1}^{h} \left\|K^j\right\|_1 = \sum_{i=1}^{h} \left\|K^i\right\|_1 = \|\text{vec}(K)\|_1 \tag{71}$$

In addition, by $h \leq n$, $h \leq m$ we have:

$$\left\|U^{\frac{h+1}{2}+M(\frac{h-1}{2})}\right\|_1 = \sum_{i=1}^{h} \left\|K^i\right\|_1 . \tag{72}$$

Then, it holds that $\|U\|_\infty = \sum_{i=1}^{h} \left\|K^i\right\|_1$. □

F.3 PROOF OF THEOREM 5

Proof. We partition U into $o \times r$ partition matrices of shape $nm \times nm$. Then the $(i,j)^{\text{th}}$ partition of the matrix $U_{(i,j)}$ describes the relationship between the ith channel of B and the jth channel of A. Then, the following holds: $\left\|U_{(i,j)}\right\|_\infty = \sum_{i=1}^{h} \left\|K_{ij}^i\right\|_1$, where K_{ij} means the two-dimensional

tensor obtained by the third dimension of K takes j and the fourth dimension of K takes i.

$$
\begin{aligned}
\|U\|_\infty &= max_{i=1}^{n\times m\times o} \sum_{j=1}^{n\times m\times r} \left|u_{(i,j)}\right| \\
&= \max_{l=0}^{o-1} \max_{i=1}^{n\times m} \sum_{s=0}^{r-1} \sum_{j=1}^{n\times m} \left|u_{(i+nml,j+nms)}\right| \\
&\leq \max_{l=0}^{o-1} \sum_{s=0}^{r-1} (\max_{i=1}^{n\times m} \sum_{j=1}^{n\times m} \left|u_{(i+nml,j+nms)}\right|) \\
&= \max_{l=0}^{o-1} \sum_{s=0}^{r-1} \left\|U_{(l+1,s+1)}\right\|_\infty \\
&= \max_{l=1}^{o} \sum_{s=1}^{r} \left\|U_{(l,s)}\right\|_\infty \\
&= \max_{l=1}^{o} \sum_{s=1}^{r} \sum_{i=1}^{h} \left\|K_{ls}^{i}\right\|_1 \\
&\leq \max_{l=1}^{o} \left\|\hat{K}^{l}\right\|_1 \\
&= \left\|\hat{K}\right\|_\infty
\end{aligned}
\tag{73}
$$

Similar to Eq. (72): for every nm rows, we choose $\frac{k+1}{2}^{\text{th}}$ row. Then its 1-norm is equal to this nm rows of the $\hat{K}$'s ∞-norm. So the equation holds. □

G AUXILIARY NUMERICAL EVIDENCE

A number of additional experiments are conducted in this section. Unless explicitly mentioned otherwise, the experimental setup remains similar to the one in the main paper. The following experiments are conducted below:

1. Evolution of the bound during training is conducted in Appendix G.1.
2. An ablation study on the hidden size is conducted in Appendix G.2.
3. An ablation study is conducted on the effect of adversarial steps in Appendix G.3.
4. We evaluate the effect of the proposed projection into the testset performance in Appendix G.4.
5. We conduct experiments on four new datasets, i.e., MNIST, K-MNIST, E-MNIST-BY, NSYNTH in Appendix G.5.
6. In Appendix G.6 experiments on three additional adversarial attacks, i.e., FGSM-0.01, APGDT and TPGD, are performed.
7. We conduct an experiment using the NCP model in Appendix G.7.

G.1 EVOLUTION OF THE BOUND DURING TRAINING

Figure 4 shows the change of bound during training with no projection. It can be seen that as training progress, bound increases linearly (different slopes correspond to different learning rates). It confirms that the generalization error will become larger and larger as the training progress.

Figure 5 shows the model trained under different projections. It can be seen that the bound of the model is almost unchanged when the projection constraints are very loose. When the constraints of the projection are tightened, the bound of the model decrease significantly. It confirms that our algorithm can successfully control the generalization error in the training process.

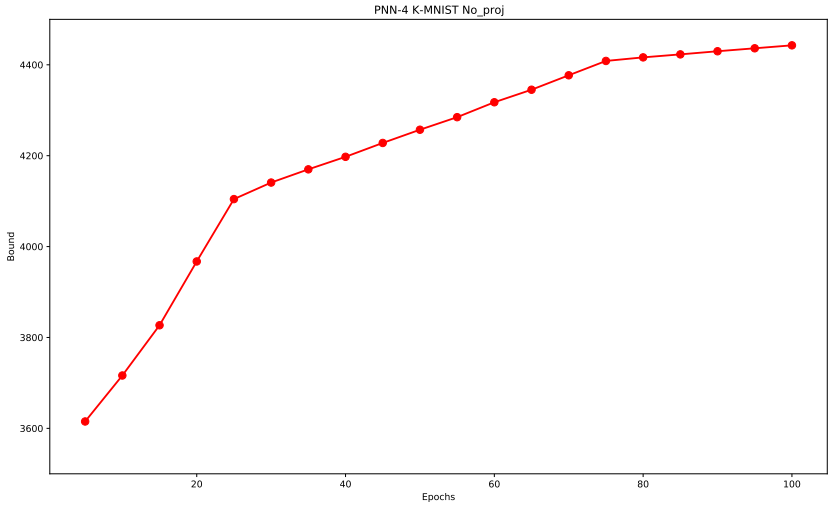


Figure 4: Changes in the bound of Rademacher complexity during training

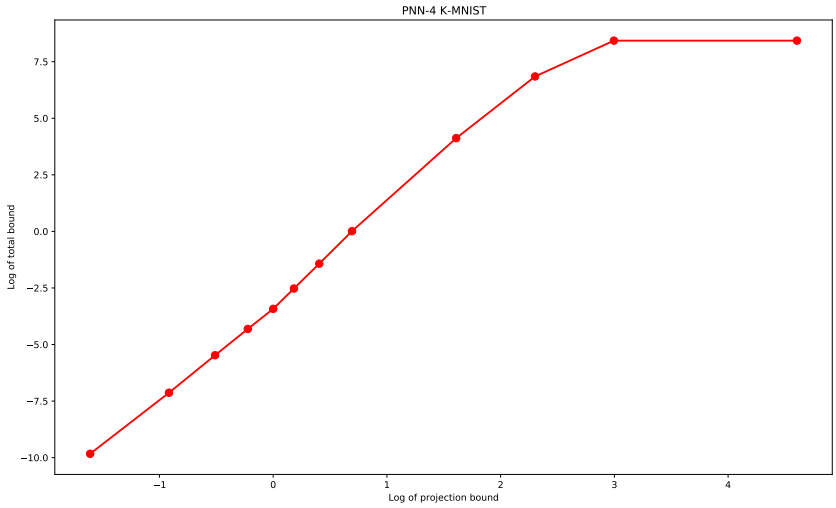


Figure 5: Rademacher complexity bound by using different intensity constraint projection training

G.2 ABLATION STUDY ON THE HIDDEN SIZE

Initially, we explore the effect of the hidden rank of PN-4 and PN-10 on Fashion-MNIST. Fig. 6 exhibits the accuracy on both the training and the test-set for both models. We observe that PN-10 has a better accuracy on the training set, however the accuracy on the test set is the same in the two models. We also note that increasing the hidden rank improves the accuracy on the training set, but not on the test set.

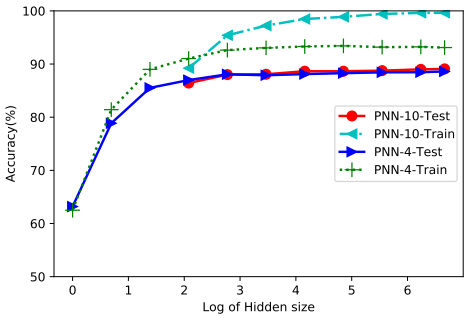


Figure 6: Accuracy of PN-4 and PN-10 when the hidden rank varies (plotted in log-scale).

G.3 ABLATION STUDY ON THE EFFECT OF ADVERSARIAL STEPS

Our next experiment scrutinizes the effect of the number of adversarial steps on the robust accuracy. We consider in all cases a projection bound of 1, which provides the best empirical results. We vary the number of adversarial steps and report the accuracy in Fig. 7. The results exhibit a similar performance both in terms of the dataset (i.e., Fashion-MNIST and K-MNIST) and in terms of the network (PN-4 and PN-Conv). Notice that when the adversarial attack has more than 10 steps the performance does not vary significantly from the performance at 10 steps, indicating that the projection bound is effective for stronger adversarial attacks.

G.4 EVALUATION OF THE ACCURACY OF PNS

In this experiment, we evaluate the accuracy of PNs. We consider three networks, i.e., PN-4, PN-10 and PN-Conv, and train them under varying projection bounds using Algorithm 1. Each model is evaluated on the test set of (a) Fashion-MNIST and (b) E-MNIST.

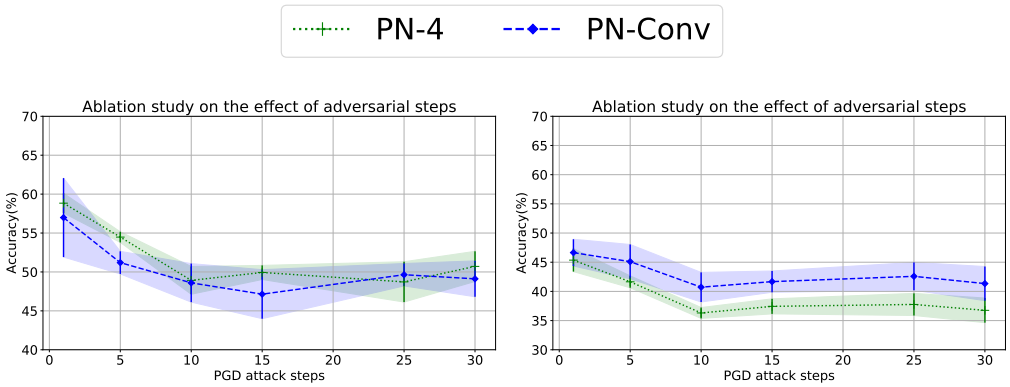
The accuracy of each method is reported in Fig. 8, where the x-axis is plotted in log-scale (natural logarithm). The accuracy is better for bounds larger than 2 (in the log-axis) when compared to tighter bounds (i.e., values less than 0). Very tight bounds stifle the ability of the network to learn, which explains the decreased accuracy. Interestingly, PN-4 reaches similar accuracy to PN-10 and PN-Conv in Fashion-MNIST as the bound increases, while in E-MNIST it cannot reach the same performance as the bound increases. The best bounds for all three models are observed in the intermediate values, i.e., in the region of 1 in the log-axis for PN-4 and PN-10.

We scrutinize further the projection bounds by training the same models only with cross-entropy loss (i.e., no bound regularization). In Table 10, we include the accuracy of the three networks with and without projection. Note that projection consistently improves the accuracy, particularly in the case of larger networks, i.e., PN-10.

G.5 EXPERIMENTAL RESULTS ON ADDITIONAL DATASETS

To validate even further we experiment with additional datasets. We describe the datasets below and then present the robust accuracy in each case. The experimental setup remains the same as in Section 4.2 in the main paper. As a reminder, we are evaluating the robustness of the different models under adversarial noise.

Dataset details: There are six datasets used in this work:



(a) Fashion-MNIST (b) K-MNIST

Figure 7: Ablation study on the effect of adversarial steps in Fashion-MNIST and K-MNIST. All methods are run by considering a projection bound of 1.

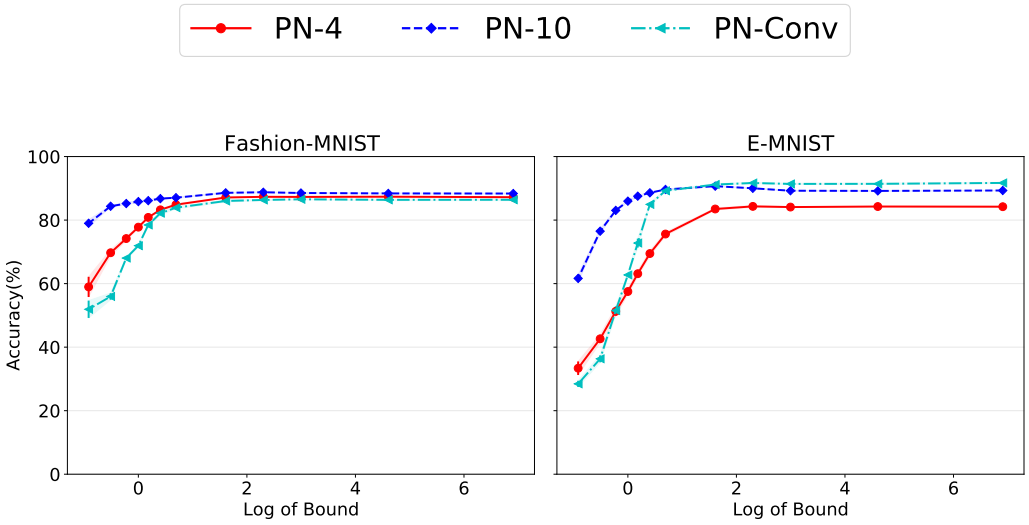


Figure 8: Accuracy of PN-4, PN-10 and PN-Conv under varying projection bounds (x-axis in log-scale) learned on (a) Fashion-MNIST, (b) E-MNIST. Notice that the performance increases for intermediate values, while it deteriorates when the bound is very tight.

Method	PN-4	PN-10	PN-Conv
		Fashion-MNIST	
No projection	$87.28 \pm 0.18\%$	$88.48 \pm 0.17\%$	$86.36 \pm 0.21\%$
Projection	$87.32 \pm 0.14\%$	$88.72 \pm 0.12\%$	$86.38 \pm 0.26\%$
		E-MNIST	
No projection	$84.27 \pm 0.26\%$	$89.31 \pm 0.09\%$	$91.49 \pm 0.29\%$
Projection	$84.34 \pm 0.31\%$	$90.56 \pm 0.10\%$	$91.57 \pm 0.19\%$

Table 10: The accuracy of different PN models on Fashion-MNIST (top) and E-MNIST (bottom) when trained only with SGD (first row) and when trained with projection (last row).

1. *Fashion-MNIST* (Xiao et al., 2017) includes grayscale images of clothing. The training set consists of $60,000$ examples, and the test set of $10,000$ examples. The resolution of each image is 28×28, with each image belonging to one of the 10 classes.
2. *E-MNIST* (Cohen et al., 2017) includes handwritten character and digit images with a training set of $124,800$ examples, and a test set of $20,800$ examples. The resolution of each image is 28×28. E-MNIST includes 26 classes. We also use the variant *EMNIST-BY* that includes 62 classes with $697,932$ examples for training and $116,323$ examples for testing.
3. *K-MNIST* (Clanuwat et al., 2018) depicts grayscale images of Hiragana characters with a training set of $60,000 examples$, and a test set of $10,000$ examples. The resolution of each image is 28×28. K-MNIST has 10 classes.
4. *MNIST* (Lecun et al., 1998) includes handwritten digits images. MNIST has a training set of $60,000$ examples, and a test set of $10,000$ examples. The resolution of each image is 28×28.
5. *CIFAR10* (Krizhevsky et al., 2014) depicts images of natural scenes. CIFAR10 has a training set of $50,000$ examples, and a test set of $10,000$ examples. The resolution of each RGB image is 32×32.
6. *NSYNTH* (Engel et al., 2017) is an audio dataset containing $305,979$ musical notes, each with a unique pitch, timbre, and envelope.

We provide a visualization[3] of indicative samples from all four datasets in Fig. 9.

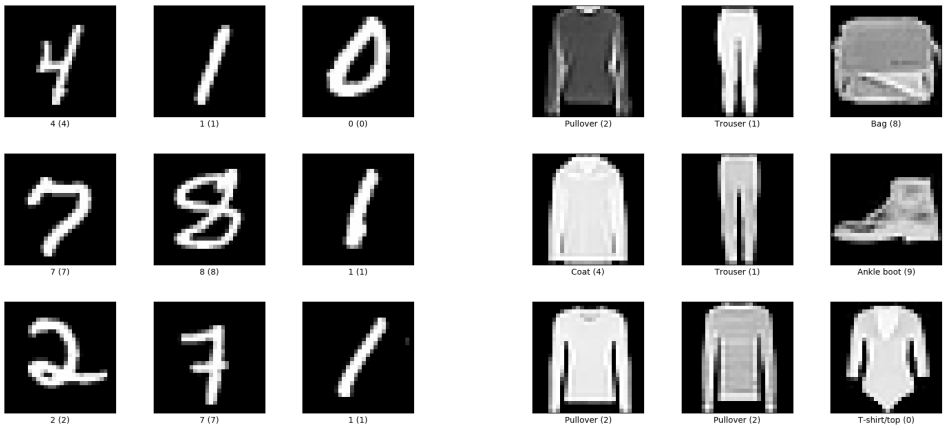
We originally train PN-4, PN-10 and PN-Conv without projection bounds. The results are reported in Table 11 (columns titled 'No proj') for MNIST and E-MNIST, Table 13 (columns titled 'No proj') for E-MNIST-BY and Table 14 (columns titled 'No proj') for NSYNTH. Next, we consider the performance under varying projection bounds; the accuracy in each case is depicted in Fig. 10 for K-MNIST, MNIST and E-MNIST-BY and Fig. 11 for NSYNTH. The figures (and the tables) depict the same patterns that emerged in the two main experiments, i.e., the performance can be vastly improved for intermediate values of the projection bound. Similarly, we validate the performance when using adversarial training. The results in Table 12 demonstrate the benefits of using projection bounds even in the case of adversarial training.

G.6 Experimental results of more types of attacks

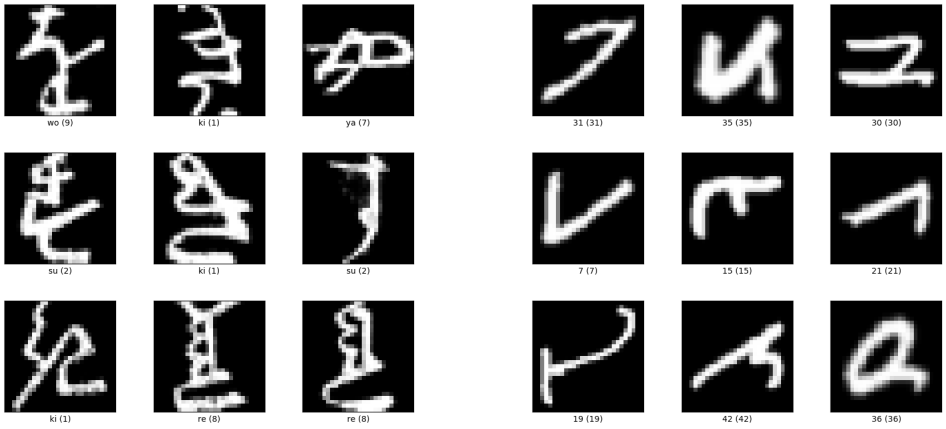
To further verify the results of the main paper, we conduct experiments with three additional adversarial attacks: a) FGSM with $\epsilon = 0.01$, b) Projected Gradient Descent in Trades (TPGD) (Zhang et al., 2019), c) Targeted Auto-Projected Gradient Descent (APGDT) (Croce and Hein, 2020). In TPGD and APGDT, we use the default parameters for a one-step attack.

The quantitative results are reported in Table 15 for four datasets and the curves of Fashion-MNIST and E-MNIST are visualized in Fig. 12 and the curves of K-MNIST and MNIST are visualized in Fig. 13. The results in both cases remain similar to the attacks in the main paper, i.e., the proposed projection improves the performance *consistently* across attacks, types of networks and adversarial attacks.

[3]The samples were found in https://www.tensorflow.org/datasets/catalog.

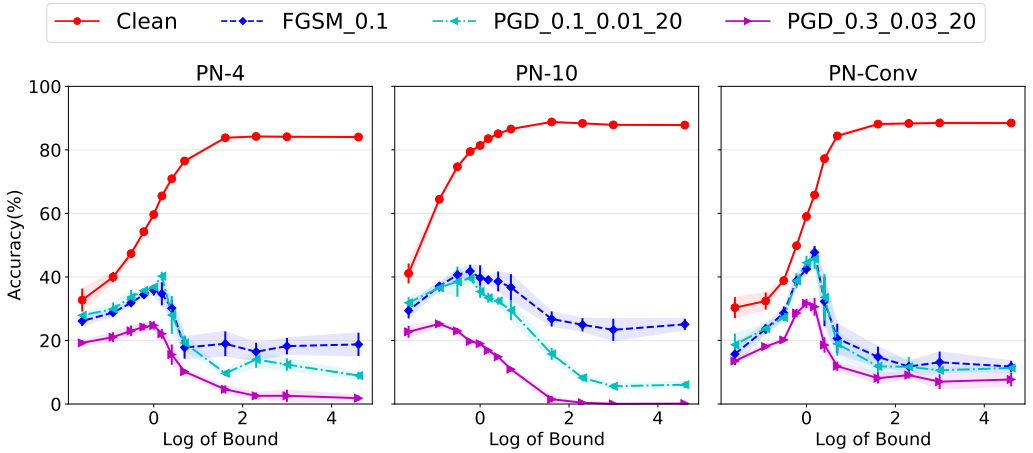


(a) MNIST (b) Fashion-MNIST

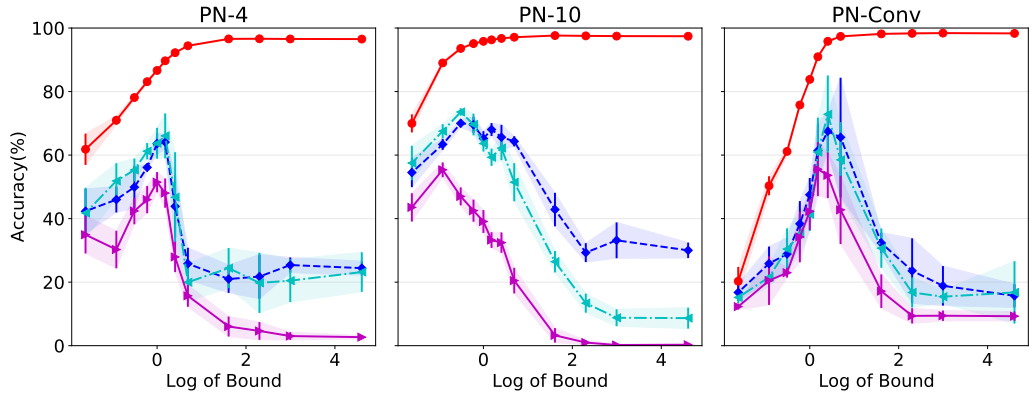


(c) K-MNIST (d) E-MNIST

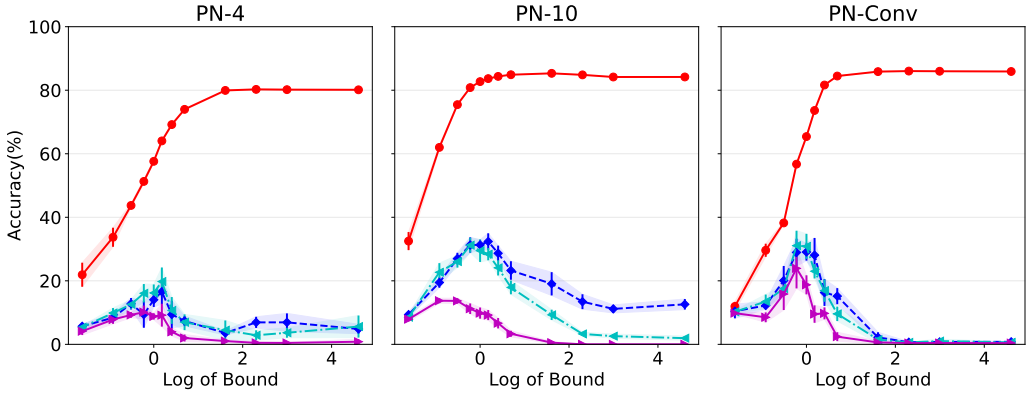
Figure 9: Samples from the datasets used for the numerical evidence. Below each image, the class name and the class number are denoted.



(a) K-MNIST



(b) MNIST



(c) E-MNIST-BY

Figure 10: Adversarial attacks during testing on (a) K-MNIST (top), (b) MNIST (middle), (c) E-MNIST-BY (bottom) with the x-axis is plotted in log-scale. Note that intermediate values of projection bounds yield the highest accuracy. The patterns are consistent in all datasets and across adversarial attacks.

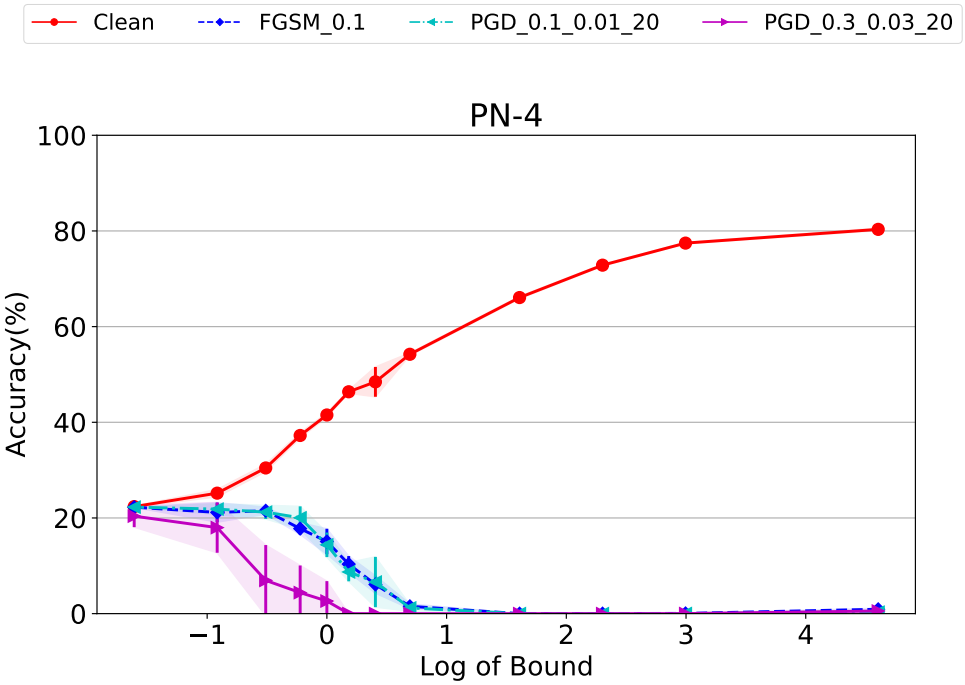
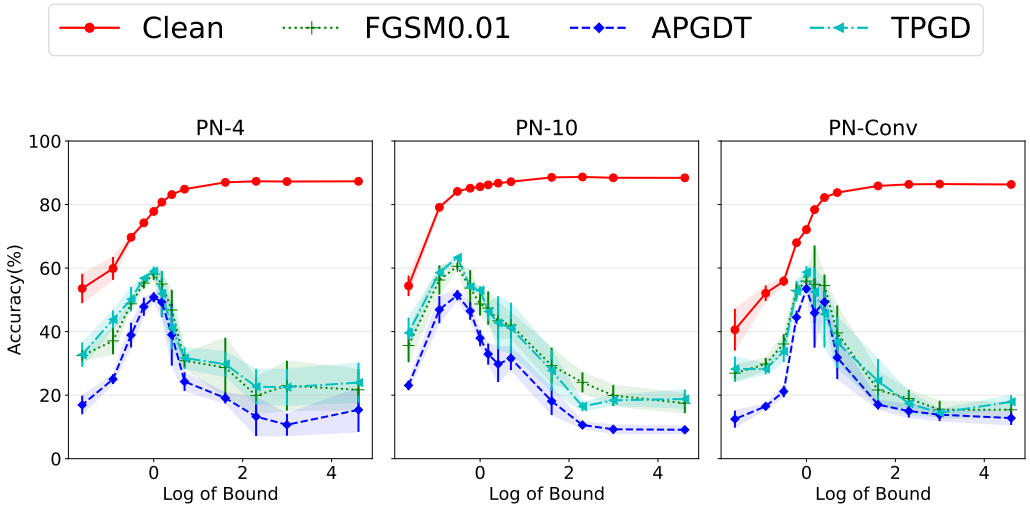
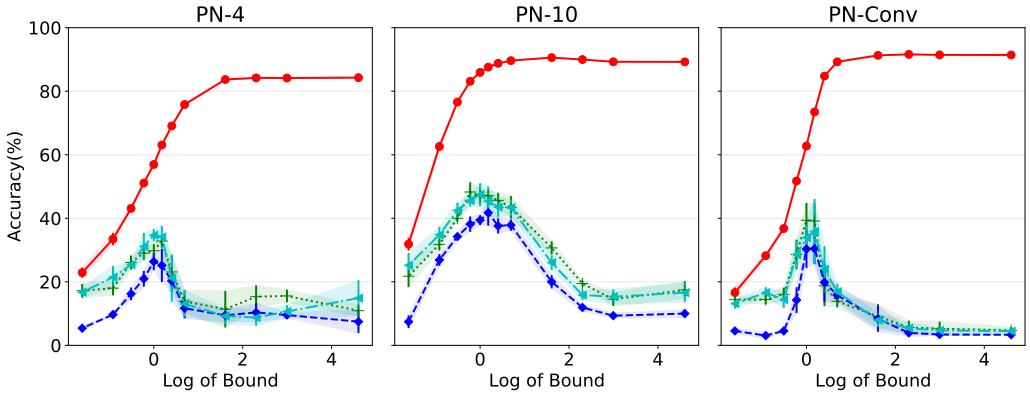


Figure 11: Adversarial attacks during testing on NSYNTH.

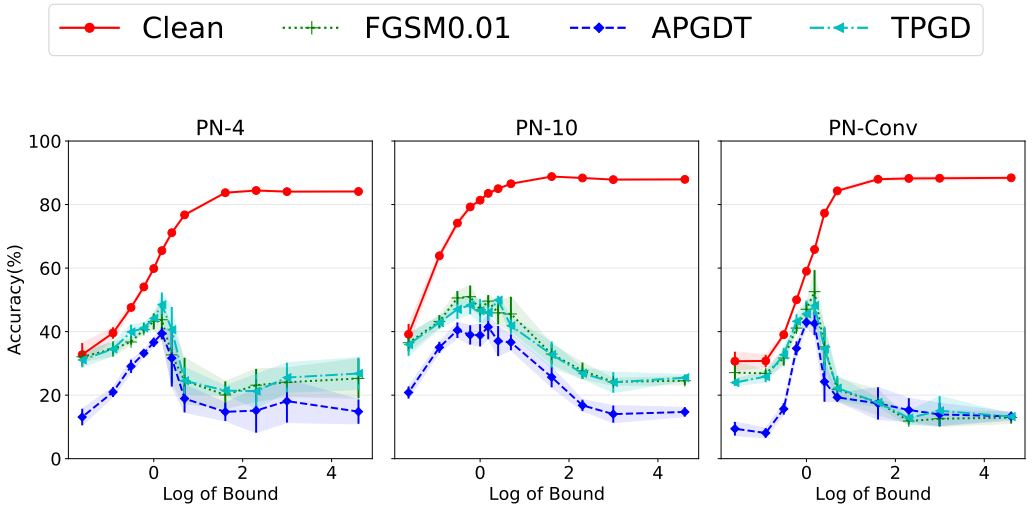


(a) Fashion-MNIST

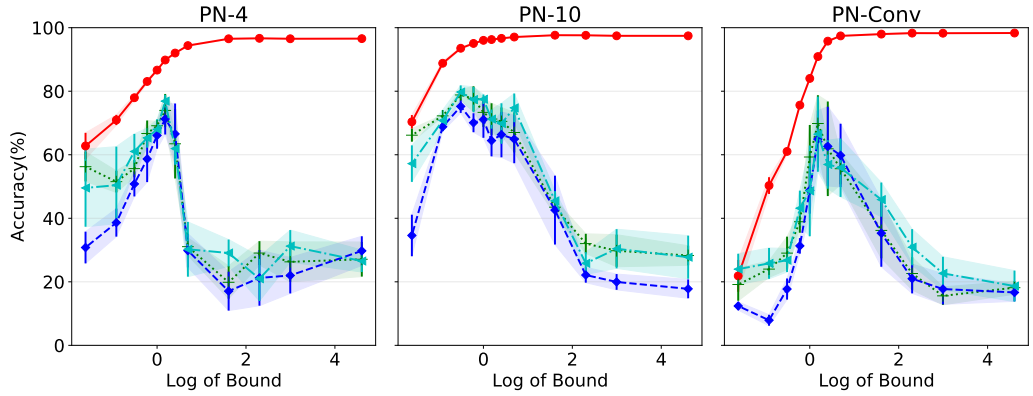


(b) E-MNIST

Figure 12: Three new adversarial attacks during testing on (a) Fashion-MNIST (top), (b) E-MNIST (bottom) with the x-axis is plotted in log-scale. Note that intermediate values of projection bounds yield the highest accuracy. The patterns are consistent in both datasets and across adversarial attacks.



(a) K-MNIST



(b) MNIST

Figure 13: Three new adversarial attacks during testing on (a) K-MNIST (top), (b) MNIST (bottom) with the x-axis is plotted in log-scale. Note that intermediate values of projection bounds yield the highest accuracy. The patterns are consistent in both datasets and across adversarial attacks.

	Method	No proj.	Our method	Jacobian	L_2
			K-MNIST		
PN-4	Clean	$84.04 \pm 0.30\%$	$\mathbf{84.23 \pm 0.30}\%$	$83.16 \pm 0.31\%$	$84.18 \pm 0.44\%$
	FGSM-0.1	$18.86 \pm 2.61\%$	$\mathbf{35.84 \pm 1.67}\%$	$22.61 \pm 1.30\%$	$22.05 \pm 2.76\%$
	PGD-(0.1, 20, 0.01)	$11.20 \pm 4.27\%$	$\mathbf{40.26 \pm 1.36}\%$	$16.00 \pm 5.63\%$	$10.93 \pm 2.42\%$
	PGD-(0.3, 20, 0.03)	$1.94 \pm 1.11\%$	$\mathbf{24.75 \pm 1.32}\%$	$4.46 \pm 2.59\%$	$2.70 \pm 1.26\%$
PN-10	Clean	$87.90 \pm 0.24\%$	$\mathbf{88.80 \pm 0.19}\%$	$88.73 \pm 0.16\%$	$87.93 \pm 0.18\%$
	FGSM-0.1	$24.52 \pm 1.44\%$	$\mathbf{41.83 \pm 2.00}\%$	$26.90 \pm 1.02\%$	$26.62 \pm 1.59\%$
	PGD-(0.1, 20, 0.01)	$7.54 \pm 0.79\%$	$\mathbf{39.55 \pm 0.64}\%$	$11.50 \pm 1.35\%$	$5.09 \pm 0.68\%$
	PGD-(0.3, 20, 0.03)	$0.05 \pm 0.04\%$	$\mathbf{25.24 \pm 0.93}\%$	$1.24 \pm 0.64\%$	$0.19 \pm 0.12\%$
PN-Conv	Clean	$88.41 \pm 0.37\%$	$88.48 \pm 0.42\%$	$86.57 \pm 0.46\%$	$\mathbf{88.56 \pm 0.62}\%$
	FGSM-0.1	$13.34 \pm 2.01\%$	$\mathbf{47.75 \pm 2.03}\%$	$14.16 \pm 3.05\%$	$12.43 \pm 2.58\%$
	PGD-(0.1, 20, 0.01)	$10.81 \pm 1.25\%$	$\mathbf{45.68 \pm 3.11}\%$	$12.05 \pm 0.82\%$	$11.05 \pm 0.85\%$
	PGD-(0.3, 20, 0.03)	$6.91 \pm 2.04\%$	$\mathbf{31.68 \pm 1.43}\%$	$7.54 \pm 1.39\%$	$6.28 \pm 2.37\%$
			MNIST		
PN-4	Clean	$96.52 \pm 0.13\%$	$\mathbf{96.62 \pm 0.17}\%$	$95.88 \pm 0.16\%$	$96.44 \pm 0.18\%$
	FGSM-0.1	$20.96 \pm 5.16\%$	$\mathbf{64.09 \pm 2.41}\%$	$33.59 \pm 8.46\%$	$26.07 \pm 5.64\%$
	PGD-(0.1, 20, 0.01)	$14.23 \pm 5.39\%$	$\mathbf{66.05 \pm 7.06}\%$	$20.83 \pm 5.64\%$	$16.06 \pm 5.84\%$
	PGD-(0.3, 20, 0.03)	$2.59 \pm 2.01\%$	$\mathbf{51.47 \pm 3.17}\%$	$4.92 \pm 1.18\%$	$4.26 \pm 2.44\%$
PN-10	Clean	$97.46 \pm 0.11\%$	$\mathbf{97.63 \pm 0.06}\%$	$97.36 \pm 0.05\%$	$97.53 \pm 0.10\%$
	FGSM-0.1	$30.12 \pm 4.58\%$	$\mathbf{70.02 \pm 1.28}\%$	$40.22 \pm 2.31\%$	$28.77 \pm 2.41\%$
	PGD-(0.1, 20, 0.01)	$9.70 \pm 2.11\%$	$\mathbf{73.57 \pm 1.17}\%$	$18.74 \pm 5.39\%$	$10.91 \pm 2.32\%$
	PGD-(0.3, 20, 0.03)	$0.47 \pm 0.53\%$	$\mathbf{55.36 \pm 2.32}\%$	$2.49 \pm 1.46\%$	$0.44 \pm 0.29\%$
PN-Conv	Clean	$98.32 \pm 0.12\%$	$\mathbf{98.40 \pm 0.12}\%$	$97.88 \pm 0.12\%$	$98.32 \pm 0.11\%$
	FGSM-0.1	$18.98 \pm 2.99\%$	$\mathbf{67.50 \pm 6.22}\%$	$27.02 \pm 9.88\%$	$23.77 \pm 5.58\%$
	PGD-(0.1, 20, 0.01)	$12.57 \pm 2.81\%$	$\mathbf{72.85 \pm 12.23}\%$	$13.96 \pm 2.57\%$	$13.84 \pm 3.18\%$
	PGD-(0.3, 20, 0.03)	$10.57 \pm 4.08\%$	$\mathbf{55.56 \pm 8.48}\%$	$10.22 \pm 0.52\%$	$9.10 \pm 3.62\%$

Table 11: Comparison of regularization techniques on K-MNIST (top) and MNIST (bottom). In each dataset, the base networks are PN-4, i.e., a 4th degree polynomial, on the top four rows, PN-10, i.e., a 10th degree polynomial, on the middle four rows and PN-Conv, i.e., a 4th degree polynomial with convolutions, on the bottom four rows. Our projection method exhibits the best performance in all three attacks, with the difference on accuracy to stronger attacks being substantial.

Method	AT	Our method + AT	Jacobian + AT	L_2 + AT
		Adversarial training (AT) with PN-10 on *K-MNIST*		
FGSM-0.1	$70.93 \pm 0.46\%$	$\mathbf{71.14 \pm 0.30}\%$	$64.48 \pm 0.51\%$	$70.90 \pm 0.57\%$
PGD-(0.1, 20, 0.01)	$60.94 \pm 0.71\%$	$61.20 \pm 0.39\%$	$57.89 \pm 0.31\%$	$\mathbf{61.47 \pm 0.44}\%$
PGD-(0.3, 20, 0.03)	$30.77 \pm 0.26\%$	$\mathbf{33.07 \pm 0.58}\%$	$29.96 \pm 0.21\%$	$30.35 \pm 0.42\%$
		Adversarial training (AT) with PN-10 on *MNIST*		
FGSM-0.1	$91.89 \pm 0.30\%$	$91.94 \pm 0.17\%$	$87.85 \pm 0.27\%$	$\mathbf{92.22 \pm 0.30}\%$
PGD-(0.1, 20, 0.01)	$87.36 \pm 0.29\%$	$\mathbf{87.38 \pm 0.37}\%$	$84.96 \pm 0.25\%$	$87.26 \pm 0.49\%$
PGD-(0.3, 20, 0.03)	$61.96 \pm 0.92\%$	$\mathbf{63.96 \pm 1.02}\%$	$62.24 \pm 0.24\%$	$62.44 \pm 0.76\%$

Table 12: Comparison of regularization techniques on (a) K-MNIST (top) and (b) MNIST (bottom) along with adversarial training (AT). The base network is a PN-10, i.e., 10th degree polynomial. Our projection method exhibits the best performance in all four attacks.

G.7 Experimental results in NCP model

To complement, the results of the CCP model, we conduct an experiment using the NCP model. That is, we use a 4th degree polynomial expansion, called NCP-4, for our experiment. We conduct an experiment in the K-MNIST dataset and present the result with varying bound in Fig. 14. Notice that the patterns remain similar to the CCP model, i.e., intermediate values of the projection bound can increase the performance significantly.

	Method	PN-4, PN-10 and PN-Conv on E-MNIST-BY	
		No proj.	Our method
PN-4	Clean	$80.18 \pm 0.19\%$	$\mathbf{80.26 \pm 0.17}\%$
	FGSM-0.1	$3.65 \pm 0.76\%$	$\mathbf{16.58 \pm 3.87}\%$
	PGD-(0.1, 20, 0.01)	$4.57 \pm 1.98\%$	$\mathbf{19.77 \pm 4.42}\%$
	PGD-(0.3, 20, 0.03)	$0.59 \pm 0.40\%$	$\mathbf{10.13 \pm 2.08}\%$
PN-10	Clean	$84.17 \pm 0.06\%$	$\mathbf{85.32 \pm 0.04}\%$
	FGSM-0.1	$11.67 \pm 1.21\%$	$\mathbf{32.37 \pm 2.58}\%$
	PGD-(0.1, 20, 0.01)	$2.48 \pm 0.66\%$	$\mathbf{31.22 \pm 2.32}\%$
	PGD-(0.3, 20, 0.03)	$0.03 \pm 0.05\%$	$\mathbf{13.74 \pm 0.77}\%$
PN-Conv	Clean	$85.92 \pm 0.08\%$	$\mathbf{86.03 \pm 0.08}\%$
	FGSM-0.1	$0.65 \pm 0.17\%$	$\mathbf{29.07 \pm 2.72}\%$
	PGD-(0.1, 20, 0.01)	$1.57 \pm 1.40\%$	$\mathbf{31.06 \pm 4.70}\%$
	PGD-(0.3, 20, 0.03)	$0.33 \pm 0.06\%$	$\mathbf{23.93 \pm 6.32}\%$

Table 13: Comparison of regularization techniques on E-MNIST-BY. The base network are PN-4, i.e., 4th degree polynomial, on the top four rows, PN-10, i.e., 10th degree polynomial, on the middle four rows and PN-Conv, i.e., a 4th degree polynomial with convolution, on the bottom four rows. Our projection method exhibits the best performance in all three attacks, with the difference on accuracy to stronger attacks being substantial.

Model	PN-4	
Projection	No-proj	Proj
Clean accuracy	$80.25 \pm 0.27\%$	$\mathbf{80.33 \pm 0.26}\%$
FGSM-0.1	$0.91 \pm 0.14\%$	$\mathbf{22.25 \pm 0.04}\%$
PGD-(0.1, 20, 0.01)	$0.31 \pm 0.11\%$	$\mathbf{22.27 \pm 0.00}\%$
PGD-(0.3, 20, 0.03)	$0.46 \pm 0.29\%$	$\mathbf{20.38 \pm 2.30}\%$

Table 14: Evaluation of the robustness of PN models on NSYNTH. Each line refers to a different adversarial attack. The projection offers an improvement in the accuracy in each case; in PGD attacks projection improves the accuracy by a remarkable margin.

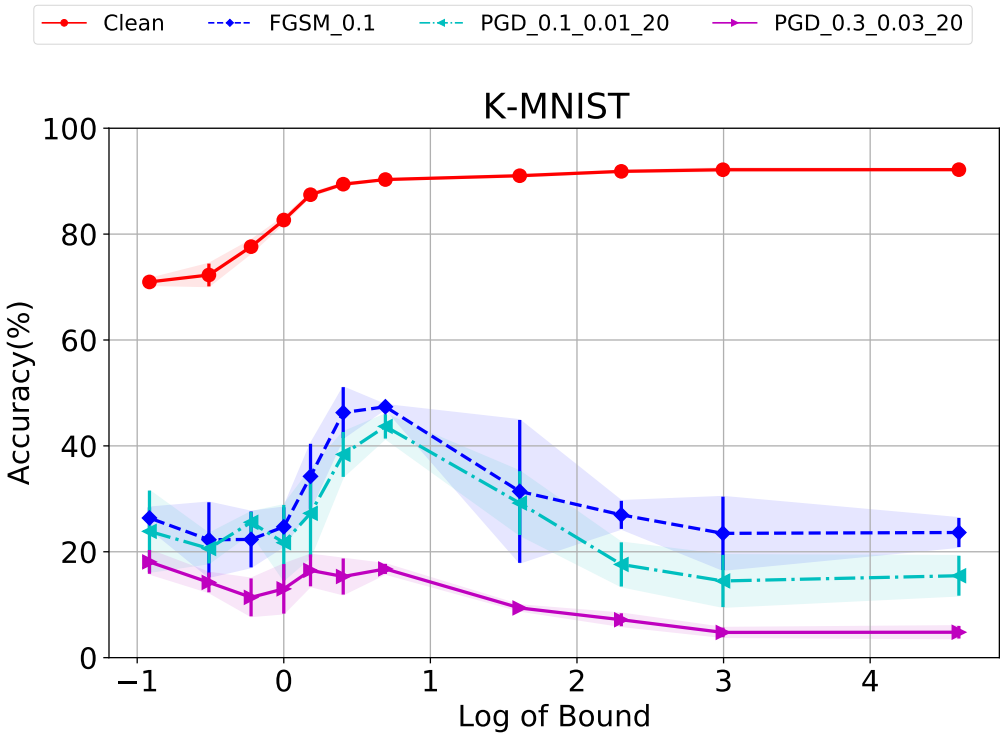


Figure 14: Experimental result of K-MNIST in NCP model.

	Method	No proj.	Our method
		Fashion-MNIST	
PN-4	FGSM-0.01	$26.49 \pm 3.13\%$	$\mathbf{58.09 \pm 1.63}\%$
	APGDT	$16.59 \pm 4.35\%$	$\mathbf{50.83 \pm 1.55}\%$
	TPGD	$26.88 \pm 6.78\%$	$\mathbf{59.03 \pm 1.45}\%$
PN-10	FGSM-0.01	$18.59 \pm 1.82\%$	$\mathbf{60.56 \pm 1.06}\%$
	APGDT	$8.76 \pm 1.14\%$	$\mathbf{51.93 \pm 1.91}\%$
	TPGD	$14.53 \pm 1.49\%$	$\mathbf{63.33 \pm 0.51}\%$
PN-Conv	FGSM-0.01	$15.30 \pm 3.10\%$	$\mathbf{55.90 \pm 2.60}\%$
	APGDT	$11.88 \pm 1.33\%$	$\mathbf{53.49 \pm 0.72}\%$
	TPGD	$14.50 \pm 1.59\%$	$\mathbf{58.72 \pm 1.87}\%$
		E-MNIST	
PN-4	FGSM-0.01	$13.40 \pm 5.16\%$	$\mathbf{32.83 \pm 2.08}\%$
	APGDT	$9.33 \pm 4.00\%$	$\mathbf{26.38 \pm 2.70}\%$
	TPGD	$17.40 \pm 3.11\%$	$\mathbf{34.68 \pm 1.92}\%$
PN-10	FGSM-0.01	$14.47 \pm 1.80\%$	$\mathbf{48.28 \pm 3.06}\%$
	APGDT	$10.13 \pm 0.93\%$	$\mathbf{41.72 \pm 4.05}\%$
	TPGD	$13.97 \pm 0.88\%$	$\mathbf{47.44 \pm 3.62}\%$
PN-Conv	FGSM-0.01	$4.71 \pm 1.10\%$	$\mathbf{39.37 \pm 5.43}\%$
	APGDT	$3.58 \pm 0.66\%$	$\mathbf{30.43 \pm 4.87}\%$
	TPGD	$4.08 \pm 0.33\%$	$\mathbf{35.85 \pm 10.20}\%$
		K-MNIST	
PN-4	FGSM-0.01	$23.31 \pm 5.34\%$	$\mathbf{43.74 \pm 5.97}\%$
	APGDT	$17.02 \pm 6.97\%$	$\mathbf{39.43 \pm 1.89}\%$
	TPGD	$23.45 \pm 7.67\%$	$\mathbf{48.46 \pm 3.84}\%$
PN-10	FGSM-0.01	$26.87 \pm 2.14\%$	$\mathbf{50.99 \pm 3.52}\%$
	APGDT	$16.23 \pm 1.32\%$	$\mathbf{41.46 \pm 3.85}\%$
	TPGD	$22.63 \pm 0.99\%$	$\mathbf{49.91 \pm 1.37}\%$
PN-Conv	FGSM-0.01	$12.31 \pm 2.03\%$	$\mathbf{52.58 \pm 6.80}\%$
	APGDT	$13.47 \pm 2.19\%$	$\mathbf{42.94 \pm 1.68}\%$
	TPGD	$14.25 \pm 2.51\%$	$\mathbf{48.19 \pm 3.02}\%$
		MNIST	
PN-4	FGSM-0.01	$34.14 \pm 7.63\%$	$\mathbf{73.95 \pm 5.18}\%$
	APGDT	$29.88 \pm 9.47\%$	$\mathbf{71.26 \pm 4.88}\%$
	TPGD	$27.01 \pm 9.77\%$	$\mathbf{76.88 \pm 1.98}\%$
PN-10	FGSM-0.01	$32.34 \pm 4.67\%$	$\mathbf{78.83 \pm 1.63}\%$
	APGDT	$19.55 \pm 1.72\%$	$\mathbf{75.22 \pm 2.05}\%$
	TPGD	$28.11 \pm 3.87\%$	$\mathbf{79.74 \pm 2.07}\%$
PN-Conv	FGSM-0.01	$22.73 \pm 3.10\%$	$\mathbf{69.83 \pm 8.91}\%$
	APGDT	$17.95 \pm 3.39\%$	$\mathbf{64.94 \pm 8.96}\%$
	TPGD	$21.82 \pm 3.07\%$	$\mathbf{66.47 \pm 11.83}\%$

Table 15: Evaluation of the robustness of PN models on four datasets with three new types of attacks. Each line refers to a different adversarial attack. The projection offers an improvement in the accuracy in each case.