

# Improving the Performance of Shallow Neural Networks Using Norm-Preserving Regularization

Daniel Stoopendahl



**LUND**  
UNIVERSITY

Department of Automatic Control

MSc Thesis  
TFRT-6317  
ISSN 0280-5316

Department of Automatic Control  
Lund University  
Box 118  
SE-221 00 LUND  
Sweden

© 2026 Daniel Stoopendahl. All rights reserved.  
Printed in Sweden by Tryckeriet i E-huset  
Lund 2026

# Abstract

The Shallow Neural Network (SNN) is one of the most basic forms of machine learning models. It has a high theoretical representational capacity, but this is often not realized in practice, in part due to overfitting. This thesis proposes a new regularization method and compares it against existing techniques across a range of tasks using SNNs. The proposed method, norm-preserving regularization (NP-reg), penalizes  $\ell_2$ -norm difference between the input vector and hidden representation. Adding this penalty term to the objective function was found to affect the generalization performance of SNNs, particularly in heavily overparametrized settings.

Performance improvements were observed for larger models, both in standalone SNNs and when used as a classifier head in transfer learning settings. For smaller models, NP-reg generally had negligible or negative effects on performance. Among existing regularization methods, Batch Norm and Layer Norm showed both positive and negative effects depending on the task, while dropout and weight decay had limited impact.

The mechanisms underlying the performance gains remain unclear. NP-reg produced a rougher loss landscape while still improving performance. This was also observed for Batch Norm, suggesting that smoother optimization alone does not explain the performance gains in the shallow settings. In some tasks, NP-reg reduced rank degradation of feature representations similarly to what Batch Norm was found to do, though neither this effect was consistently associated with improved performance.

Despite the lack of a complete theoretical explanation, NP-reg showed practical value and was shown to be robust across different settings, functioning with multiple activation functions while remaining more computationally feasible than similar approaches.



# Acknowledgments

I want to thank Robert Bridges for supervising the project, helping me understand concepts and providing useful research directions. I am particularly thankful for having had the freedom of exploring different topics.

I would also like to thank Brian Brown for providing feedback and answering any questions I had, especially regarding mathematical intuition.

I want to thank AI Sweden for providing access to computing resources and for welcoming me to the office in Gothenburg.

I thank my supervisor Pontus Giselsson for providing feedback on my writing. This elevated the thesis to a more professional level.

Finally I would like to thank my friends and family. Writing the thesis alone was stressful at times but you helped me keep my self-scrutiny at a reasonable level.

Thank you.



# Contents

|                                                            |           |
|------------------------------------------------------------|-----------|
| <b>1. Introduction</b>                                     | <b>9</b>  |
| 1.1 Contribution . . . . .                                 | 10        |
| <b>2. Background</b>                                       | <b>11</b> |
| 2.1 Shallow Neural Network . . . . .                       | 11        |
| 2.2 Gradient Descent and Regularization . . . . .          | 13        |
| 2.3 Generalized Singular Value Decomposition . . . . .     | 15        |
| 2.4 Related Work . . . . .                                 | 15        |
| <b>3. Methodology: Norm-Preserving Regularization</b>      | <b>17</b> |
| 3.1 Conceptual Motivation . . . . .                        | 17        |
| 3.2 Definition . . . . .                                   | 17        |
| 3.3 Relation to GSVD and Hard Norm-Preservation . . . . .  | 18        |
| 3.4 Relation to Orthogonal Regularization . . . . .        | 18        |
| 3.5 Implementation Details and Design Choices . . . . .    | 19        |
| <b>4. Evaluation: Performance on Simple Classifiers</b>    | <b>20</b> |
| 4.1 General Setup . . . . .                                | 21        |
| 4.2 CIFAR-10 . . . . .                                     | 22        |
| 4.3 IMDb . . . . .                                         | 23        |
| 4.4 UCI HAR . . . . .                                      | 24        |
| <b>5. Analysis: Learning Behavior and Model Properties</b> | <b>25</b> |
| 5.1 Computation Time . . . . .                             | 25        |
| 5.2 Activation Functions . . . . .                         | 26        |
| 5.3 Norm-Preservation in Practice . . . . .                | 26        |
| 5.4 Fixed Norm Ablation . . . . .                          | 27        |
| 5.5 Loss Landscape and Gradients . . . . .                 | 28        |
| 5.6 Singular Values of Hidden Representations . . . . .    | 29        |
| <b>6. Application: Efficient Transfer Learning</b>         | <b>31</b> |
| 6.1 Scenario . . . . .                                     | 31        |
| 6.2 General Setup . . . . .                                | 32        |
| 6.3 Food-101 . . . . .                                     | 32        |
| 6.4 Yelp Open Dataset . . . . .                            | 33        |

|                                            |           |
|--------------------------------------------|-----------|
| <b>7. Discussion</b>                       | <b>35</b> |
| 7.1 Performance . . . . .                  | 35        |
| 7.2 Working Mechanisms of NP-reg . . . . . | 37        |
| 7.3 Limitations . . . . .                  | 38        |
| 7.4 Future Work . . . . .                  | 39        |
| <b>8. Conclusion</b>                       | <b>41</b> |
| <b>Bibliography</b>                        | <b>42</b> |
| <b>9. Appendix</b>                         | <b>46</b> |
| 9.1 Code . . . . .                         | 46        |
| 9.2 Model Sizes . . . . .                  | 46        |
| 9.3 Hyperparameter Grid Search . . . . .   | 46        |
| 9.4 Raw Test Results . . . . .             | 48        |

# 1

## Introduction

*The year is 2034, international tourism is at an all-time high while multilingualism is seeing a steady decline. The reason for this? All cross-language conversation is translated live in people's ears by the software BabelFish. The software is lightweight enough to be embedded in any headphones on the market and the translation is instant, enabling casual conversation without any awkward pauses. Similar software has been in the works for many years but they all struggle with long delays or reliance on external computers. But thanks to recent improvements of Shallow Neural Networks this constrained machine learning application was made feasible.*

The scenario does not seem implausible given the recent advancements in machine learning. In the last few years machine learning has enabled language models that can communicate like humans and image classifiers that are able to detect cancer. These models learn complex relations from large amounts of data through training trillions of parameters. Their accomplishments are undeniably impressive but they require colossal computing resources and their complexity is essentially excessive for most machine learning tasks. So in most cases, a simpler architecture is sufficient.

It has been proven theoretically that any continuous function can be estimated by an artificial neural net with a single hidden layer [Cybenko, 1989]. In the field of machine learning the implementation of these models is usually referred to as Shallow Neural Networks (SNN) which stand in contrast to the complex pipelines of modern models consisting of hundreds of layers from input to output. But despite their powerful theoretical capabilities, they have many practical drawbacks which often make them infeasible for a given problem. Instead of being an architecture handcrafted for a domain, like a convolutional neural net for image classification or transformer for language modeling, the base SNN is a general model which requires thorough training to adapt to a given task. Perhaps the biggest difficulty of SNNs is that this training process rarely converges to a solution that generalizes well. Increasing the size of the model does not necessarily make it better if it cannot fully utilize all of its capacity. This is where regularization comes in. Regularization aims to improve the generalization performance of models. For an image classifier

to perform well, it needs to be able to classify new images just as well as it can classify the images it saw during training. Many different techniques have been developed in machine learning to improve generalization performance. The aim of this thesis is to propose a new method which will then be analyzed and compared to already existing techniques.

The new method of regularization is constructed with inspiration from a new mathematical framework of deconstructing non-linear functions [Brown et al., 2025]. The decomposition enforces an initial norm-preserving non-linear mapping up until the final layer. It is a softer variant of this norm-preserving property that is used as the new regularization technique for the SNN classifiers tested in this work.

## **1.1 Contribution**

The contribution of this thesis is to introduce norm-preserving regularization (Chapter 3) and then answer the following empirical research questions:

1. How do different regularization methods affect the performance of SNN classifiers? (Chapter 4)
2. How does norm-preserving regularization affect the learning dynamics and model behavior of SNN classifiers? (Chapter 5)
3. Can norm-preserving regularization provide accuracy improvement when used in combination with other regularization techniques on a transfer learning task? (Chapter 6)

# 2

## Background

Here the Shallow Neural Network will be defined along with general learning principles and regularization methods. Then the Generalized Singular Value Decomposition will be introduced and finally relevant prior work will be presented.

### 2.1 Shallow Neural Network

A Shallow Neural Network is one of the most basic forms of machine learning models. It refers to a neural network with only a few layers of hidden nodes (usually one or two). This is in contrast to the more common Deep Neural Networks (DNN) which can contain up to hundreds of layers of computation between the input and output. In this thesis SNN classifiers with only one hidden layer will be explored and improved. Furthermore, only linear layers and activation layers will be used even though SNNs in general are not restricted to only these.

The architecture used can be seen in Figure 2.1. It is also defined more formally by the mathematical expression:

$$\mathbf{z} = \mathbf{W}_2 a(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2 \quad (2.1)$$

Where  $\mathbf{z}$  is the output logit vector,  $\mathbf{W}_1, \mathbf{W}_2$  are the weight matrices for the linear layers,  $a(x)$  is the element wise activation function and  $\mathbf{b}_1, \mathbf{b}_2$  are the bias terms. To make a prediction, the model applies a linear transformation to the input, followed by a non-linear activation function, followed by another linear transformation.

Most modern Neural Networks usually utilize many more than just one hidden layer. For example the biggest version of the large language model Llama 3 contains 126 layers [Grattafiori et al., 2024]. Adding depth is undeniably a good way of learning higher level hierarchical relationships and is in practice, the only reasonable way of building these complex machine learning models. That being said, the study of shallower architectures still remains a relevant field. SNNs have many practical applications where a deeper model would not be suitable and in theory, is able to represent any continuous function just like a DNN. Furthermore the simplicity of the architecture comes with the benefit of being easier to analyze and

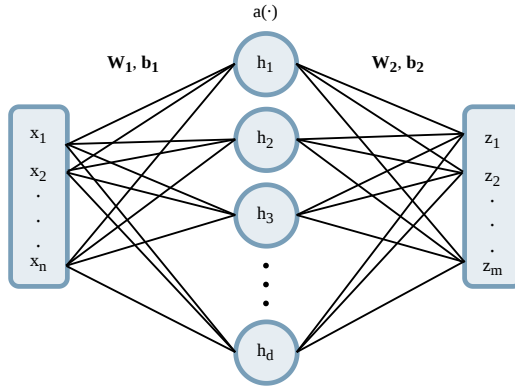


**Figure 2.1** Model of an SNN with input:  $\mathbf{x}$ , parameters:  $\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}_1, \mathbf{b}_2$ , hidden representation:  $\mathbf{h}$  and output:  $\mathbf{z}$ .

understand using tools such as the Generalized SVD (Section 2.3), making them less of a black-box than deeper networks.

## Practical Use

While SNNs to a great extent have been replaced by deeper more task-specific architectures there are still places where they see practical use, including:

- Environments where high inference speed is required, for example computer networks [Lopez-Martin et al., 2019].
- For devices with limited computing/energy resources such as in IoT [Reis, 2025].
- Scenarios with limited data to avoid overfitting [Pasupa and Sunhem, 2016].
- In order to transfer a big pretrained ”foundation model” to a specific task [Imran and Ishmam, 2024].
- In the academic world as a baseline and to justify the need for deeper architectures [Ba and Caruana, 2014], [Urban et al., 2017].

## Universal Approximation Theorem

The Universal Approximation Theorem (UAT) was developed in 1989 simultaneously by [Cybenko, 1989] and [Hornik et al., 1989]. It proves that any continuous function can be approximated by a neural net with a single hidden layer up to arbitrary accuracy. These works assumed a sigmoidal or squashing activation function. This was later generalized to any non-polynomial activation function [Leshno et al.,

1993] such as ReLU [Nair and Hinton, 2010]. Given enough width, the SNN can thus in theory give an accurate solution to almost any machine learning problem. But even with very wide models, this is rarely realized in practice. A big limiting factor is the difficulty of finding good model parameters, commonly done through gradient descent.

## 2.2 Gradient Descent and Regularization

The fundamental backbone of most of modern machine learning is training through gradient descent with backpropagation [Rumelhart et al., 1986]. If the model is defined as (2.1), "training" or "learning" is the process of finding good parameters  $\mathbf{W}_1$ ,  $\mathbf{W}_2$ ,  $\mathbf{b}_1$  &  $\mathbf{b}_2$  given some training data. This is implemented by defining a loss function to be minimized usually representing some kind of prediction error with the given data. A model with low loss should thus correspond to a good performing model. Reaching low loss is achieved through iteratively updating model parameters in the direction of the negative gradient of the loss function, so-called gradient descent.

Gradients for deep layers are calculated using the Backpropagation algorithm. Utilizing the chain rule and storing intermediate results, gradients of each layer of parameters can efficiently be calculated. With all gradients calculated, a step is taken scaled by a learning rate, updating all parameters. In reality, the entire dataset is rarely used all at once. Instead, at each step a random batch of the data is sampled and used in the gradient computation. This is what is known as Stochastic Gradient Descent (SGD).

A common difficulty in machine learning is the problem of overfitting. The problem arises when the model learns the training data too well to the point of memorizing individual data points including their noise instead of learning the underlying general function. Thus when tested on new data points the model performs poorly. A solution to this is regularization.

### Regularization

There are many more or less explicit methods of regularization. Weight decay, dropout, Batch Normalization, early stopping, double descent, label smoothing and SGD are all techniques that can have a regularizing effect.

A fair comparison between all regularization techniques would not be feasible since they are implemented in fundamentally different ways. Furthermore in most scenarios, many regularization techniques are used in combination and they often synergize. This work will therefore focus on comparing the new method norm-preserving regularization (Chapter 3) with the techniques: Weight decay, dropout, Batch Norm and Layer Norm.

**Weight Decay** or L2 Regularization is implemented by adding a regularization

term to the loss function in the form of  $\lambda_{wd}||\mathbf{W}||_2^2$ . This penalizes parameters from becoming too large. The original explanation for why weight decay works was that smaller weights are less prone to overfit the sharp noise in the data, making the model "smoother" and thus generalize better [Krogh and Hertz, 1991]. Recent work have looked into how it works together with modern deep architectures trained with SGD [D’Angelo et al., 2024] showing that it improves generalization more implicitly by making the learning dynamics more desirable.

**Dropout** [Srivastava et al., 2014] is a technique applied during training to promote diversification in neural activations. Dropout layers are added with probability  $p$  (usually  $p = 0.1-0.5$ ) and in training, each neuron has probability  $p$  of being replaced by 0. This encourages the model not to rely too much on individual neurons. After training is finished, activations are no longer dropped and instead have to be rescaled to keep the outputs within the same range.

**Batch Normalization** [Ioffe and Szegedy, 2015] (Batch Norm) is a building block for stabilizing and speeding up the training of deeper architectures. Batch Norm layers are usually inserted before the activation functions of hidden layers. For each training batch these calculate the mean and variance to then normalize the signal. Batch Norm also introduces two learnable parameters per feature which can rescale and move the signal, essentially undoing the normalization if that is what is regarded as optimal. Importantly these are initialized to being inactive meaning that in the early stages of training, signals between layers closely follow a normalized distribution. There is no consensus on why Batch Norm works but many favorable properties it gives rise to have been identified. For some problems it has been shown to make the loss landscape smoother and gradients more predictable [Santurkar et al., 2018]. This leads to a more stable training process and better generalization performance. It also prevents rank collapse which is a problem in deeper architectures where different inputs cannot be kept separate [Daneshmand et al., 2020].

**Layer Normalization** [Ba et al., 2016] (Layer Norm) is another technique that utilizes normalization to improve training. Instead of normalizing over a batch, it normalizes the output across a layer in the network. This removes the dependency on the batch size and makes it possible to implement it in scenarios where the batch size is very small or in recurrent models which might not implement batched training. Like Batch Norm it also introduces learned parameters to rescale and shift after the normalization and due to its similarity to Batch Norm it most probably works for the same reasons, e.g. that it smoothens the loss landscape, stabilizes training, and prevents rank collapse.

## 2.3 Generalized Singular Value Decomposition

The regular Singular Value Decomposition (SVD) is a powerful tool in numerical analysis that has enabled applications such as compression and approximation of matrices. SVD makes it possible to identify the most important directions in the input and output space and their corresponding scaling factors.

A similar decomposition was recently proposed, the Generalized Singular Value Decomposition (GSVD) [Brown et al., 2025]. This enables decomposition of non-linear functions such as machine learning models. In the construction, non-linearities are separated from linear mappings as follows:

A function  $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$  that satisfies  $f(0) = 0$  can be decomposed into:

$$f(x) = U\Sigma v(x), \quad \text{for all } x \in \mathbb{R}^n \quad (2.2)$$

Where  $U \in \mathbb{R}^{m \times m}$  is a unitary matrix,  $\Sigma \in \mathbb{R}^{m \times l}$  a real, non-negative, rectangular-diagonal matrix with  $l = m + n$  and  $v : \mathbb{R}^n \rightarrow \mathbb{R}^l$  is an injective, norm-preserving ( $\|v(x)\|_2 = \|x\|_2$  for all  $x \in \mathbb{R}^n$ ) mapping.

Importantly the non-linear function is norm-preserving similar to the orthogonal  $V^T$  matrix of the SVD. This ensures that all scaling is concentrated to the  $\Sigma$  matrix, just like the SVD. A later paper [Brown et al., 2026] explored applications of this theorem to machine learning tasks using two approaches. The first approach decomposes a black-box model using only access to its input and output. The constructed decomposition can then be used in perturbation attacks. The second approach constructs a machine learning model so that it enforces norm-preservation up until the final linear layer during training (SVDNet). A GSVD can then be computed by applying the linear SVD to the final layer enabling useful applications. Here a hard constraint is put on the model to ensure norm-preservation (by rescaling the signal after the activation function). But this could have negative impact on training performance as the model becomes too restricted. With this in mind, a new method is suggested in this thesis which instead uses a softer constraint (Chapter 3).

## 2.4 Related Work

Many modifications have already been made to SNNs. Perhaps the most researched is the Extreme Learning Machine. In terms of regularization techniques for neural networks, the most similar to norm-preserving regularization is orthogonal regularization.

### Extreme Learning Machine

The Extreme Learning Machine [Huang et al., 2006] (ELM) aims to provide a drastic improvement of learning efficiency. The first layer of parameters are set randomly at initialization and not learned. This has the benefit of making the training process extremely efficient. Instead of using gradient descent to stumble across the

loss landscape, potentially getting stuck in local minima, the optimal solution can be found analytically by computing a matrix inverse. So along with the extremely fast inference time, ELMs also have an extremely fast learning time, justifying the name. In the realm of very limited computing resources, ELMs have shown great performance and on top of that, multiple extensions of it have been developed [Huang et al., 2011]. However, despite good empirical performance and the authors' original claim, the ELM is in fact not a universal approximator [Perfilieva et al., 2025]. Furthermore, ELM is developed for scenarios where there is a very low computing budget for training. In the case where a larger computation budget is available for training, but not for inference, significantly better models can be developed using other training algorithms.

### Orthogonal Regularization

Orthogonal regularization was introduced as a remedy to the problem of exploding/vanishing gradients [Vorontsov et al., 2017]. It enforces or encourages the linear weight matrices to be orthogonal. This is implemented either as a soft constraint in the form of an added loss term or as a hard constraint with a parametrization of the weight matrix. In most cases the softer constraint performs better [Vorontsov et al., 2017]. There are many ways the regularization term can be computed but it usually consists of calculating the Gram matrix of the weights which can be computationally expensive for bigger layers. Orthogonal regularization has been applied to many types of networks including CNNs [Bansal et al., 2018] and RNNs [Vorontsov et al., 2017] but has not gained traction as a popular regularization method.

# 3

## Methodology: Norm-Preserving Regularization

The new method norm-preserving regularization (NP-reg) is introduced here. It is motivated, formally defined and conceptually compared to existing methods.

### 3.1 Conceptual Motivation

In general, a machine learning model has a big arsenal of transformations to apply to an input. The linear transformations of rotating and scaling, non-linear folding and bending, as well as lifting and collapsing dimensions. Limiting scaling across the first layers can be seen as a separation of concerns for the model. The first part of the network gets the job of providing unbiased features and leaves the selection confidence through scaling to the second layer. Or to draw a human analogue: You see a coiled object lying in the grass and jump out of fear thinking it is a snake. But upon a slower investigation of its features you can correctly classify it as just a garden hose. Your first response is fast and useful but the slower methodological response of first examining the features is more accurate.

### 3.2 Definition

The proposed norm-preserving regularization is introduced as an extra loss term similar to that of weight decay. For one sample this loss is calculated as:

$$l_{np}(x) = (\|h(x)\|_2 - \|x\|_2)^2 \quad (3.1)$$

where  $x$  is the input vector and  $h(x)$  is the feature vector obtained by passing the input through a linear layer and activation function:  $h(x) = a(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1)$ . Averaged over a batch of  $N$  samples and scaled by  $\lambda_{np} > 0$ , this gives:

$$L_{np} = \lambda_{np} \cdot \frac{1}{N} \sum_{i=1}^N (\|h(x_i)\|_2 - \|x_i\|_2)^2 \quad (3.2)$$

This loss term is added to the objective function of the problem so that they are minimized together during training. Pseudocode of the basic training pipeline can be seen in Algorithm 1.

---

**Algorithm 1** Training loop with Norm-preserving regularization

---

```

1: Initialize weights
2: for epoch = 1 to N do
3:   for (x, y) in dataset do
4:     prediction, features  $\leftarrow$  model(x, weights)           ▷ Save feature vector
5:     pred_loss  $\leftarrow$  loss_function(prediction, y)
6:     reg_loss  $\leftarrow$  np_lambda · mse(norm(x), norm(features))
7:     loss  $\leftarrow$  pred_loss + reg_loss                       ▷ Add NP-reg loss
8:     grad  $\leftarrow$  compute_gradients(loss)
9:     weights  $\leftarrow$  weights − learning_rate · grad
10:   end for
11: end for

```

---

### 3.3 Relation to GSVD and Hard Norm-Preservation

Unlike SVDNet or the strict requirements of GSVD, NP-reg does not enforce injectivity nor is it strictly norm-preserving. Instead it should be seen as a soft constraint with inspiration from the GSVD. It punishes norm scaling on seen data but does not enforce any structural requirements on the architecture such as requiring  $f(0) = 0$ . The relaxation means that it does not break UAT and can potentially achieve better learning results.

### 3.4 Relation to Orthogonal Regularization

NP-reg has many practical similarities to orthogonal regularization. Orthogonality of a linear mapping implies norm-preservation of input but three fundamental differences exist between orthogonal regularization and NP-reg.

First, if the linear layer is an expansion or reduction of dimensions, the weight matrix is rectangular and orthogonal regularization enforces either orthonormal rows or columns. For an expansion the implication of norm-preservation still holds but in the case of a reduction the norm-preservation only holds if the input exists in the row-space of the matrix, otherwise the map is norm-decreasing.

Second, NP-reg is data driven while orthogonal regularization is data-independent. The benefit of being data dependent is that noise in the data can provide an extra implicit regularizing effect. But it comes with the drawback of risking overfitting the norm-preserving mechanism to the training data.

Third and perhaps most important, the implementation of the loss as the mean squared error of the norm is a faster and more feasible computation for big weight matrices as it does not involve matrix multiplications like current implementations of orthogonal regularization. Furthermore the implementation constructs a different loss landscape providing different gradients.

It is also worth noting that by changing  $h(x)$  in (3.2) to the output of the linear layer, a variant of orthogonal regularization can be implemented which is more computationally efficient than the existing methods. Using this method would probably make the loss term smoother due to the exclusion of non-linearities and since ReLU works as an identity mapping in half of the input-space, the performance might not be too dissimilar from NP-reg. However, this would abandon the benefits of analysis through the GSVD.

## 3.5 Implementation Details and Design Choices

This is by no means the only way to implement norm-preservation as a soft constraint but it is an implementation that keeps the loss reasonable across the whole learning process and stable for both good and bad model parameters.

One drawback with the implementation is that it punishes absolute and not relative scaling. A gain where  $\|x\|_2 = 23$  and  $\|h(x)\|_2 = 25$  would be penalized just as much as a gain where  $\|x\|_2 = 2$  and  $\|h(x)\|_2 = 4$  even though their relative scaling is very different. But basing the loss on pure scaling introduces the danger of exploding losses and gradients for badly initialized weights. Still, a more relative measurement could be considered in future work.

In this thesis a single  $L_{np}$  term will be added to the cross entropy loss of the objective function. Here the norm-preservation is encouraged between the input to the SNN and the output of the activation function. But it is worth mentioning that a more generalized method is possible. Norm-preservation could be endorsed between any representations, not just between the input and first hidden layer. A bigger network can thus promote norm-preservation up until the final layer either by applying losses over multiple intermittent representations or as one loss from input to the penultimate layer.

# 4

## Evaluation: Performance on Simple Classifiers

To verify how well different regularization techniques perform on SNN classifiers and to see how well they generalize, experiments had to be done across different domains. Three experiments were constructed based on common machine learning tasks. The experiments were:

1. Image classification using the CIFAR-10 dataset [Krizhevsky and Hinton, 2009]
2. Sentiment analysis using the IMDb dataset [Maas et al., 2011]
3. Human activity recognition using the UCI HAR dataset [Anguita et al., 2013]

The three experiments provide a nice breadth of classification tasks. They represent three domains with different formats of input and also provide different size training datasets (50 000, 25 000, 7352). Furthermore the numbers of output labels (10, 2, 6) vary across the datasets. Examples of datapoints and labels can be seen in Table 4.1.

While vision tasks rarely use pure DNN/SNN architectures without spatial information, it provides a research insight to more correctly classify performance differences between them and the more commonly used convolutional neural networks and vision transformers. For basic language tasks SNN-like models see more real world use along with basic tokenizers and embedders. One example is Google’s Compact Language Detector v3 (CLD3) [Google, 2026] used in the Google Chrome browser. But perhaps the best natural fit for an SNN is on tabular data with no implicit structure. A particularly useful application is in low-compute environments such as IoT devices.

**Table 4.1** Examples of input data and labels for the CIFAR-10, IMDb and UCI HAR datasets.

| Dataset  | Input                                                                                                                                                                                              | Label      |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|
| CIFAR-10 |                                                                                                                   | Automobile |
| IMDb     | “all i can say about this film is to read the back of the video case and then put it back on he shelf and pick anything else, i mean anything, a blank video, would be better than watching this.” | Negative   |
| UCI HAR  | [0.26211587, -0.0031463972, -0.13276921, -0.46545973, -0.28466028, ... , 0.30104294]                                                                                                               | Walking    |

## 4.1 General Setup

The main setup of the architecture as well as the methodology for hyperparameter search is kept consistent across all three experiments.

### Model

For each of the tasks an SNN was used which consisted of a fully connected linear layer followed by a ReLU activation function followed by a second fully connected linear layer, essentially implementing Figure 2.1. Six different model sizes were tested (Section 9.2) for each experiment.

Weights were initialized using a Kaiming uniform distribution and the AdamW optimizer [Loshchilov and Hutter, 2018] was used for training. For all tasks, cross entropy was used as loss function.

### Hyperparameter Search

To determine the best hyperparameters for each experiment and model size, a grid search was conducted using accuracy on the validation set as measurement of performance. The range of the hyperparameters in the grid search were selected from preliminary testing on the biggest “vanilla” model (model without any regularization method applied). For NP-reg, weight decay and dropout, three different hyperparameters were tested ( $[1e-3, 1e-2, 1e-1]$ ,  $[1e-5, 1e-4, 1e-3]$ ,  $[0.1, 0.3, 0.5]$ ). And for all methods three batch sizes and three initial learning rates were tested.

To compare the performance of the best models against each other, the best hyperparameters from each technique were selected and models were trained multiple times using fixed seeds to estimate the mean and standard deviation of the prediction accuracy. Raw data from the hyperparameter search and evaluation can be seen in Section 9.3 and 9.4 in the appendix.

## 4.2 CIFAR-10

### Dataset

CIFAR-10 consists of 60 000 labeled images each belonging to one of 10 classes (e.g. dog, automobile, frog). Each data point is a 32 x 32 pixel RGB image with one label. Out of the 60 000 samples, 10 000 are used as test set. From the remaining samples, 45 000 were used for training and 5000 for validation. The input pixel values are normalized and data augmentation is conducted through random horizontal flipping and cropping as in AlexNet [Krizhevsky et al., 2012]. Then the images are flattened into one-dimensional vectors, removing all structural information and forwarded to the SNN classifier.

### Model

To improve convergence a learning rate scheduler was used which reduced the learning rate by half if no improvement was made to the validation loss over 16 epochs. After training, the best performing model was loaded for testing. The grid search considered the learning rates [1e-5, 3e-5, 1e-4] and the batch sizes [64, 128, 256]. Size of the hidden dimension ranged from 8 to 8192.

### Results

The performance of the different techniques can be seen in Figure 4.1. Weight decay and dropout did not provide any clear improvements over the vanilla model while Batch Norm consistently outperformed across all model sizes. NP-reg and Layer Norm provide clear performance gain when reaching the heavily overparametrized domain. For NP-reg, the bigger models benefited from a large  $\lambda_{np}$  (1e-1) while even the lowest  $\lambda_{np}$  led to negative results for the small models.

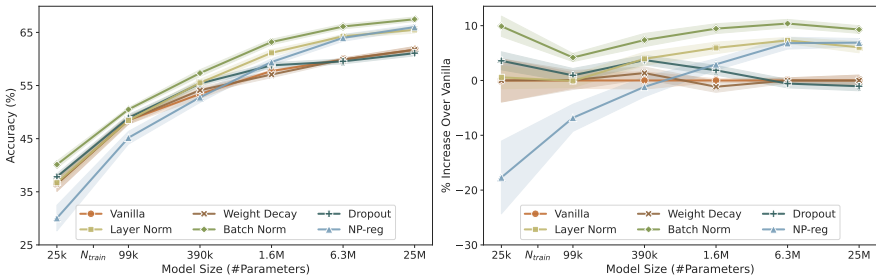


**Figure 4.1** Performance comparison of regularization methods applied to an SNN on the CIFAR-10 dataset. Left plot displays absolute performance, right plot shows relative performance over vanilla. Sample mean and standard deviation over 10 seeds are reported.

## 4.3 IMDb

### Dataset

The IMDb dataset consists of movie reviews which are labeled as either positive or negative. The dataset consists of 50 000 reviews split evenly between positive and negative. The training set is of size 25 000 and the test set is of size 25 000. From the training set a fixed validation set of 5000 samples is extracted making the final split 20 000/5000/25 000 for train/validation/test.

### Model

To convert the text into a format that can be trained on, a Bag-of-Words embedder was used. This stores the counts of individual words in a sparse vector of the size of the vocabulary while disregarding word ordering. This is used as the input to the SNN. The output of the SNN is two logits representing the two labels. For this task early stopping was conducted when validation loss had not improved over eight epochs and for test evaluation the best performing model based on validation accuracy was loaded.

Both the hidden dimension and the vocabulary size (input dimension) were varied which both change the total number of parameters. Vocabulary-size/hidden-dimension range from 1562/8 to 50 000/256. The grid search considered the learning rates  $[3e-6, 1e-5, 3e-5]$  and the batch sizes  $[32, 64, 128]$ .

### Results

The results of this experiment are presented in Figure 4.2. Once again weight decay and dropout provided no clear performance gain over the base model. NP-reg performed better than vanilla especially in the bigger models while Layer Norm and Batch Norm consistently underperformed.

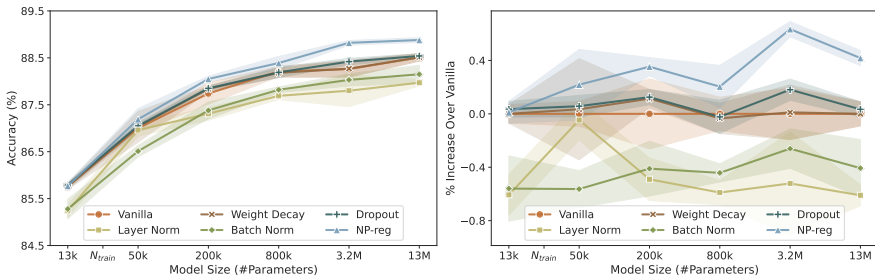


**Figure 4.2** Performance comparison of regularization methods applied to an SNN on the IMDb dataset. Left plot displays absolute performance, right plot shows relative performance over vanilla. Sample mean and standard deviation over 10 seeds are reported.

## 4.4 UCI HAR

### Dataset

The UCI HAR dataset consists of information gathered during six human activities: walking, walking upstairs, walking downstairs, sitting, standing and lying. The data was gathered through the accelerometer and gyroscope of a smartphone. From the raw signals, 561 features were extracted representing movements and forces. The dataset is on the smaller side with 10 299 samples in total. Out of these 7352 were used as training data and 2947 samples were in the test set. Due to the small size of the dataset, 5-fold cross validation was used with validation sets of size 1470.

### Model

Due to the smaller size of this dataset, certain precautions had to be made in order to mitigate noise in the results. First of all a fixed number of epochs (50) with no learning rate scheduler were used. Second, K-fold cross-validation was used with 5 folds for the grid search. Lastly 40 seeds were used for the final test evaluation of the models to make sure the mean and standard deviation were representative. The grid search considered the learning rates  $[3e-5, 1e-4, 3e-4]$  and the batch sizes  $[32, 64, 128]$ . Hidden dimensions of sizes ranging from 8 to 8192 were evaluated.

### Results

The results of this experiment were the noisiest due to the smaller dataset as can be seen in Figure 4.3. The best performing models were of the smaller under-parametrized size using Batch Norm and Layer Norm but the models of this setup had a large variance in performance. As the model sizes increases the models more consistently converged to similar performance. For the bigger sizes, Batch Norm, Layer Norm and NP-reg provided the best performance.

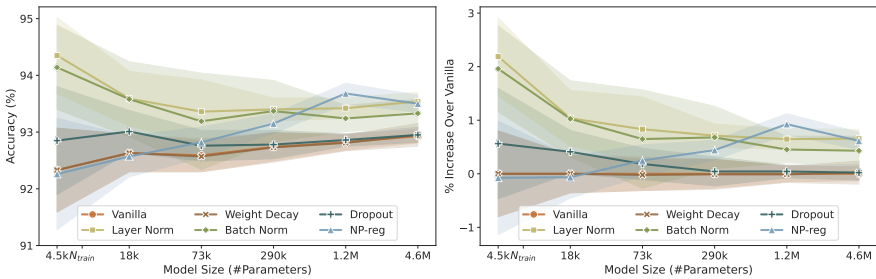


**Figure 4.3** Performance comparison of regularization methods applied to an SNN on the UCI HAR dataset. Left plot displays absolute performance, right plot shows relative performance over vanilla. Sample mean and standard deviation over 40 seeds are reported.

# 5

## Analysis: Learning Behavior and Model Properties

In order to understand the behavior of NP-reg, why it works and its limitations, further experiments had to be conducted. This chapter first handles general behavior of NP-reg analyzing: The impact on computation time (Section 5.1), sensitivity to activation functions (Section 5.2) and how well norm is preserved (Section 5.3). This is followed by an investigation of three hypotheses of why the regularization works: Limiting scaling in the first layer (Section 5.4), improvement of learning dynamics (Section 5.5) and producing more expressive features (Section 5.6).

### 5.1 Computation Time

Runs were made measuring the average wall clock time for epochs to investigate how much overhead comes with using the technique. A model using NP-reg along with a vanilla model were trained with the same batch size and learning rate for analysis. The convergence behavior can be visualized in the learning curves in Figure 5.1.

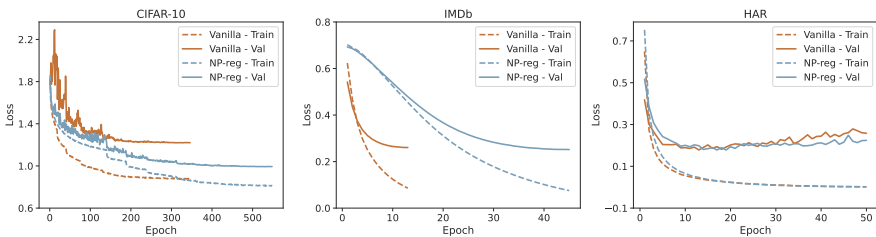


**Figure 5.1** Training and validation loss during training of a vanilla model versus an NP-reg model for the three datasets. The added loss of the regularization term is not included.

For CIFAR-10, the average time per epoch using NP-reg was 3.024 seconds on an NVIDIA 3060 Ti GPU. Without NP-reg, an epoch took an average of 2.969 seconds making it an overhead of 1.87%. The more notable difference in time was seen in the experiments where the models trained until convergence. Here, models with NP-reg trained for more epochs in general as can be seen in Figure 5.1.

## 5.2 Activation Functions

Since the regularization is constraining the magnitude of the features, one might assume that the model relies on many "dead" neurons where the ReLU outputs 0 and that it might break down when using other activation functions without the large range of 0 output. To test this, ReLU was compared against Leaky ReLU with slope 0.01 and hyperbolic tangent (Tanh). The results are presented in Table 5.1.

**Table 5.1** Accuracy using different activation functions across datasets. (+) indicates improvement over vanilla and (−) indicates decrease over vanilla.

| Dataset  | Model   | ReLU       | Leaky ReLU | Tanh       |
|----------|---------|------------|------------|------------|
| CIFAR-10 | NP-reg  | 66.21% (+) | 65.89% (+) | 47.27% (+) |
|          | Vanilla | 58.17%     | 58.78%     | 45.02%     |
| IMDb     | NP-reg  | 88.94% (+) | 88.93% (+) | 88.45% (−) |
|          | Vanilla | 88.42%     | 88.45%     | 88.57%     |
| UCI HAR  | NP-reg  | 93.45% (+) | 93.38% (+) | 92.74% (+) |
|          | Vanilla | 92.84%     | 92.87%     | 92.12%     |

For all tests except one NP-reg provided performance gain, but it was most noticeable when using the ReLU activation function.

## 5.3 Norm-Preservation in Practice

Whether the regularization actually makes the layer preserve the norm of unseen data is analyzed here. The three  $\lambda_{np}$  values used in the grid search were tested. How well it preserves norm is measured the same way the loss term is calculated excluding the scaling factor:

$$L_{np}^* = \frac{1}{N} \sum_{i=1}^N (\|h(x_i)\|_2 - \|x_i\|_2)^2 \quad (5.1)$$

This was calculated over both the training data and test data. The results are presented together with the accuracy in Table 5.2. For comparison, a vanilla baseline and a network implementing a hard norm-preserving constraint are included.

**Table 5.2** Norm-preservation on train and test data for different  $\lambda_{np}$  values and a model with hard constraint.  $L_{np}^* = 0$  indicates perfect norm-preservation across the entire dataset.

|                   | $\lambda_{np} = 0$ | $\lambda_{np} = 0.001$ | $\lambda_{np} = 0.01$ | $\lambda_{np} = 0.1$ | Hard   |
|-------------------|--------------------|------------------------|-----------------------|----------------------|--------|
| Accuracy          | 58.17%             | 62.63%                 | 66.40%                | 66.21%               | 55.65% |
| Train- $L_{np}^*$ | 3992               | 19.19                  | 40.42                 | 3.946                | 0      |
| Test- $L_{np}^*$  | 28160              | 408.5                  | 22.43                 | 2.529                | 0      |

Higher  $\lambda_{np}$  leads to higher preservation of norm on the training data as expected as well as higher preservation on the unseen test data. The only exception is the Train- $L_{np}^*$  with  $\lambda_{np} = 0.01$ . Using a hard constraint on norm scaling had negative effect on performance making it worse than the vanilla model.

## 5.4 Fixed Norm Ablation

It could be hypothesized that keeping the feature scale limited is the cause of the performance improvement, and that it does not necessarily have to remain close to the input norm but instead just to some fixed value. This hypothesis was analyzed by replacing  $\|x\|_2$  in (3.2) with the mean norm of all inputs:  $E[\|x\|_2]$ . So instead of encouraging norm-preservation across the first layer, it encourages all norms of features to be a fixed value regardless of input. The performance of this fixed-norm loss (Norm-constraining) is compared against NP-reg and the vanilla baseline in Table 5.3.

**Table 5.3** Comparison of accuracy between penalizing norm deviance from input (NP-reg), penalizing norm deviance from fixed value (Norm-Constraining) and a baseline model on the three datasets.

| Model             | CIFAR-10 | IMDb   | UCI HAR |
|-------------------|----------|--------|---------|
| NP-reg            | 66.21%   | 88.94% | 93.45%  |
| Norm-Constraining | 61.59%   | 88.03% | 93.59%  |
| Vanilla           | 58.17%   | 88.42% | 92.84%  |

The performance of Norm-Constraining varies widely across tasks, this could be explained by the variance of the input norm. For UCI HAR the mean input norm was tighter  $17.16 \pm 2.958$  than for CIFAR-10  $66.89 \pm 17.78$  and IMDb  $26.72 \pm 17.01$ . This explains the good performance on UCI HAR where the two methods are not very different. On the other tasks where the norm carries more information, utilizing it makes a contribution to performance.

## 5.5 Loss Landscape and Gradients

One of the main explanations for the success of Batch Norm and Layer Norm is that they make the loss landscape smoother when combined with modern optimizers. [Santurkar et al., 2018] showed that it not only made the variation of loss smaller, but also the variation of gradients. The same experiment is conducted here on the biggest CIFAR-10 model and the smallest UCI HAR model where Batch Norm provided clear performance improvements. The same measurements are also done on the vanilla model and the NP-reg model for comparison. A fixed number of epochs was used along with no learning rate scheduler.

The measurement is done as follows: At fixed intervals in training a measurement is done. The gradient is calculated and three evaluations are done along this direction from  $0.1x$  to  $3x$  step-size. First the range of loss values are measured, then the range of the  $\ell_2$  change of the gradient and third, the effective  $\beta$ -smoothness measured as the highest  $\ell_2$  change in gradient over distance moved. The results can be visualized in Figure 5.2.

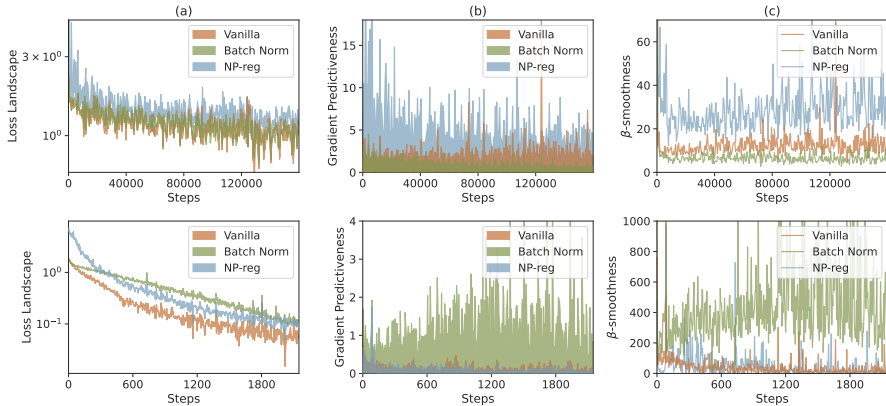


**Figure 5.2** Loss landscape and gradient smoothness of Vanilla, Batch Norm and NP-reg for two models. At fixed intervals in training, three calculations are made by traversing the gradient direction from  $0.1x$  to  $3x$  step-size. (a) indicates the range of losses, (b) indicates the range of gradient  $\ell_2$  change, (c) indicates the maximum gradient change over distance. For the top row the biggest model on the CIFAR-10 dataset was used, for the bottom row the smallest model for the UCI HAR dataset was used.

For the big model, Batch Norm provided a clear improvement of both the gradient predictiveness and effective  $\beta$ -smoothness. NP-reg on the other hand made the loss landscape rougher in every regard even though the resulting model performed better. Likewise, for the small model Batch Norm made the landscape more rugged while still providing better accuracy.

In comparison to [Santurkar et al., 2018] which also used CIFAR-10, the ranges of loss and gradient predictiveness here were tighter for all methods as the shallow model used does not have the same loss spikes that come with deeper, more non-linear networks.

## 5.6 Singular Values of Hidden Representations

As an input propagates through a neural network the rank of the representation tends to decrease as noise is removed from the signal. But with very deep architectures this can have a detrimental effect on representational power. Batch Norm layers have been shown to reduce this effect and preserve rank in deep networks [Daneshmand et al., 2020]. Due to the similarities between NP-reg and orthogonal regularization it might have a similar rank-preserving effect.

To look into this, the rank distribution and effective rank is computed for the hidden representation of the data. This is done by sending a big batch ( $N=8192$ ) through the network after training. From the matrix of resulting feature vectors  $H$ , singular values are computed and sorted. In order to get a metric on the diversity of the features the effective rank is also calculated as defined by [Roy and Vetterli, 2007].

The computation was done using the largest and smallest of each model for the methods NP-reg, Batch Norm and vanilla. The singular values are reported in Figure 5.3 and the effective ranks in Table 5.4.

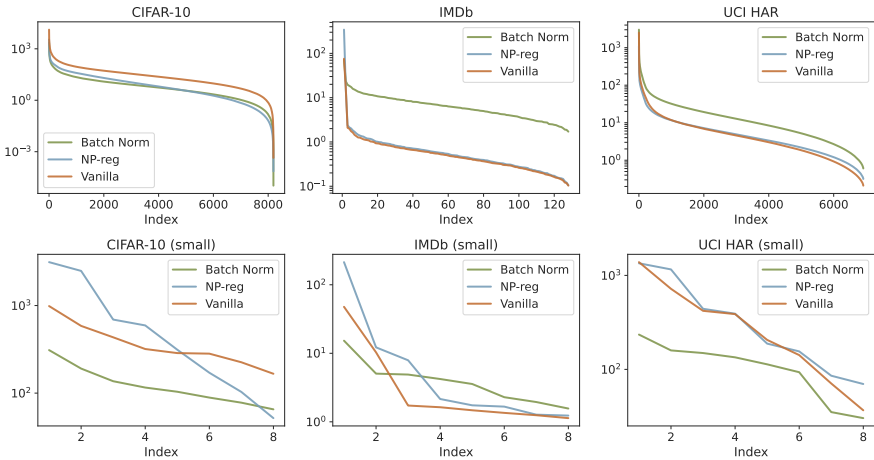


**Figure 5.3** Distribution of singular values for the hidden representations  $H$ . The singular values are plotted from largest to smallest. Top row displays the largest model for each task and the bottom row displays the smallest model for each task.

**Table 5.4** Effective rank of the hidden representations  $H$  comparing Batch Norm, NP-reg and Vanilla on the 3 datasets.

|       | <b>Model</b> | <b>CIFAR-10</b> | <b>IMDb</b> | <b>UCI HAR</b> |
|-------|--------------|-----------------|-------------|----------------|
| Large | Batch Norm   | 2694            | 99.80       | 2840           |
|       | NP-reg       | 2509            | 4.159       | 2997           |
|       | Vanilla      | 2501            | 19.14       | 1855           |
| Small | Batch Norm   | 7.010           | 6.051       | 6.862          |
|       | NP-reg       | 4.309           | 1.712       | 5.188          |
|       | Vanilla      | 6.826           | 2.776       | 5.148          |

Both NP-reg and Batch Norm have an effect on the spectrum of singular values and the effective rank. Batch Norm increased the effective rank in all experiments providing more diverse features. NP-reg increased the effective rank for UCI HAR but had the opposite effect for IMDb where it also outperformed all other models. Furthermore it significantly reduced the effective rank of the small CIFAR-10 model where it had sub-par accuracy.

# 6

## Application: Efficient Transfer Learning

As a final test to validate the utility of the method in a real world application, two tasks of transfer learning were conducted. This involves taking a larger pretrained model and adapting it to a specific task. Transfer learning can be done by retraining the whole model but this is often infeasible especially for small companies with limited computing resources. Instead some form of Parameter-Efficient Fine-Tuning (PEFT) is often used. A very basic form of PEFT is to only train and update the weights of the final classifier layer, often implemented as an SNN. This also comes with the benefit of not needing full access to the underlying model, instead only the embeddings of the training data are required.

### 6.1 Scenario

A local ramen restaurant has been facing challenges with employee expectations such as requests for higher wages and set working hours. The head chef is thus considering the possibility of using AI models as an alternative workforce to potentially mitigate these problems. Unfortunately the ramen joint does not have their own GPU cluster where they can train models from scratch so instead they are considering taking a large pretrained model and adapting it to their in-house data.

Specifically they want to be able to classify an image of a dish and classify the sentiment of a review. To evaluate if NP-reg can improve performance here, two transfer learning tasks were tested:

1. Dish image classification using the Food-101 dataset [Bossard et al., 2014]
2. Review sentiment analysis using the Yelp Open Dataset [Yelp Inc., 2026]

Examples of data points and labels can be seen in Table 6.1.

**Table 6.1** Examples of input data and labels for Food-101 and Yelp Open Dataset.

| Dataset           | Input                                                                                                                          | Label  |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------|--------|
| Food-101          |                                               | Ramen  |
| Yelp Open Dataset | “The a/c really wasn’t working, they sold out on the ramen burger during Sunday lunch, and the ramen wasn’t all that great...” |        |
|                   |                                                                                                                                | 1 star |

## 6.2 General Setup

In most scenarios, multiple regularization methods are used together. Therefore these experiments also look into how well NP-reg works together with other regularization techniques. For comparison, a model without any regularization (Vanilla) was compared against a model using NP-reg (NP-reg), and two models using combination of regularization methods. One with one of [Batch Norm, Layer-Norm, no normalization] in combination with dropout and weight decay (BN/LN/None + D + WD) and a second one with NP-reg in combination with dropout and weight decay (NP-reg + D + WD). The hyperparameters were selected with a grid search using the same ranges as in Chapter 4. To see how well the techniques perform in the low data regime, subsets using (100%, 30%, 10%, 3%, 1%) of the training data was used. Batch size was fixed in both experiments while three learning rates were tested. Raw data from the hyperparameter search and evaluation can be seen in Section 9.3 and 9.4 in the appendix.

## 6.3 Food-101

### Dataset

Food-101 consists of RGB images of 101 different kinds of dishes. In total there are 101 000 images out of which 25 250 are in the test set. From the training data a validation set of size 7575 was extracted.

### Model

The backbone of the model is a ResNet-50 [He et al., 2016] CNN pretrained on ImageNet [Deng et al., 2009]. This is used to precompute the feature vector for each image in the dataset. The classifier head that is trained is an SNN with hidden dimension 2048. A fixed batch size of 128 was used, learning rates [1e-5, 3e-5, 1e-4] were tested and training was stopped when the validation loss did not improve over three epochs. Thereafter, the best performing model was loaded for testing.

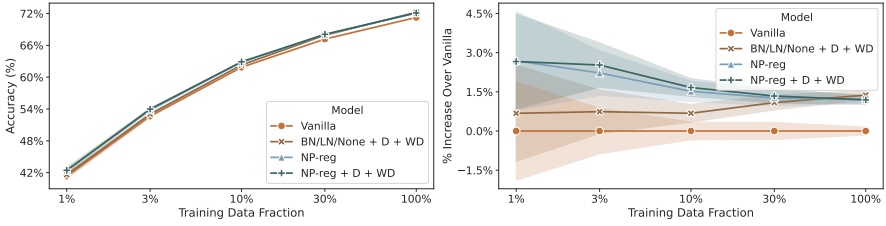


**Figure 6.1** Comparison of methods of regularization on a transfer learning task using subsets of the Food-101 dataset. Right plot shows relative increase in accuracy over the baseline. Sample mean and standard deviation over 40 seeds are reported.

## Results

The results can be visualized in Figure 6.1 showing both the absolute and relative increase in accuracy over the baseline model. The biggest but also most noisy difference was in the low data regime where (NP-reg) and (NP-reg + D + WD) showed the biggest performance increase. The benefit of adding dropout and weight decay to NP-reg was not big. With the full dataset, all three methods showed similar performance gain.

## 6.4 Yelp Open Dataset

### Dataset

The Yelp Open Dataset consists of customer reviews of services and restaurants on the website Yelp. The reviews contain a text along with a 1-5 star rating. A subset of 200 000 restaurants reviews was used, out of which 20 000 were selected for testing. From the remaining training data a validation set of size 19 000 was extracted.

### Model

As backbone a BERT base model [Devlin et al., 2019] was used. The embeddings were calculated in advance for computational efficiency using the maximum BERT context window of 512 tokens. Instead of using the classification [CLS] token for prediction which was pretrained for next sentence prediction, all of the output tokens are averaged. The hidden dimension was set to width 768, batch size to 256 and the learning rates [3e-5, 1e-4, 3e-4] were tested. Training was stopped when the validation loss did not improve over three epochs. Thereafter, the best performing model was loaded for testing.

## Results

The resulting performance is presented in Figure 6.2. Here the results are not as conclusive, the relative performance gain was not very big. The variance in accuracy

was comparable to that of Food-101 but none of the methods provided consistent improvement in any of the dataset sizes.

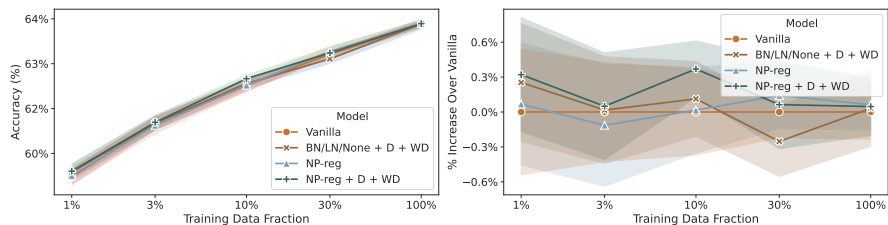


**Figure 6.2** Comparison of methods of regularization on a transfer learning task using subsets of the Yelp Open Dataset. Right plot shows relative increase in accuracy over the baseline. Sample mean and standard deviation over 40 seeds are reported.

# 7

## Discussion

Here an attempt will be made to paint a cohesive picture that connects the empirical results in the previous chapters with the theory. The aim is to provide some rationale to when and why NP-reg works and when other alternatives are better. Finally limitations and future work will be discussed.

### 7.1 Performance

There is no one solution that consistently performed the best which is not surprising given the broad variety of tasks. But out of the methods tested it can be said that Layer Norm and Batch Norm along with NP-reg were the methods that had the most noticeable impact on the models' performance. Both for better and for worse.

#### Batch Norm and Layer Norm

Batch Norm and Layer Norm, methods that are usually motivated to stabilize training and improve the performance of deeper models can also have a noticeable positive effect on these shallower models. Even though shallower and wider models in general have smoother loss landscape [Nguyen and Hein, 2018], [Li et al., 2018] than deeper ones, Batch Norm was still able to contribute to a smoothing effect of the gradients along the loss landscape for the big model. But they both also showed a clear performance boost in the underparametrized model for UCI HAR where there was no smoothing, showing that the positive impact of the methods reaches far wider than just deep overparametrized models. This suggests that the smoothing effect is not the sole contributor to the performance boost.

#### NP-reg

NP-reg had a positive impact when the model reached into the heavily overparametrized domain. Different optimal  $\lambda_{np}$  were needed for different model sizes and experiments. This is perhaps not as intuitive to select as hyperparameters for weight decay and dropout as the effect of the regularization depends not only on  $\lambda_{np}$ , but also input dimension, hidden dimension and the magnitude of the input.

That being said, for the bigger models all three  $\lambda_{np}$ , representing a wide range, showed a performance increase over the baseline. For smaller model sizes though, NP-reg showed either no effect or a detrimental effect. This could imply that solutions with low norm scaling tend to generalize well but that the regularization can have a harmful effect if the model does not have the capacity to satisfy both the constraint and the objective function. Section 7.2 further investigates the impact of NP-reg.

## Dropout and Weight Decay

Dropout and weight decay on the other hand, perhaps the two most widely used regularization methods, showed no noticeable performance increase for the SNN classifiers in these tasks even when expanding to the heavily overparametrized models. This might be due to the fact that the used optimizer (AdamW) with mini-batch learning already provided the regularization effect by itself. Optimizers that use stochastic gradient descent and momentum already provide regularization through their tendency to find flatter minima. As [D’Angelo et al., 2024] states, the usefulness of weight decay is no longer its explicit regularizing effect but instead its helpful effects on optimizers. Perhaps the SNNs used were too shallow to benefit from these stabilizing effects.

## Transfer Learning

The usefulness of regularization differed in the two transfer learning tasks. For the image classification there was a consistent performance boost when using regularization methods, especially for the NP-reg methods in the low-data domain. This low-data domain is of particular interest because it is in these scenarios where fine-tuning just the head might be preferred over fine-tuning the whole model as that could lead to catastrophic forgetting.

The benefits of adding dropout and weight decay to these models are questionable. Adding it to the NP-reg model did not provide a clear benefit and since it did not provide performance in the previous experiments it seems probable that the same was true for these experiments.

For Yelp Open Dataset the effects of regularization were not as distinct. The limited effect of the methods might be due to the fact that the BERT backbone already provided sufficient regularization and that the downstream task thus becomes easily solvable, perhaps even linearly so. Noise might also not be as apparent in the language setting compared to the visual domain where the compressed images almost certainly have imperfections. Adding to this, the number of labels in this task (5) were significantly lower than in the other (101), meaning that the task for the classification head were not as nuanced and that a lower rank might be fine. Finally the models were not as significantly overparametrized due to the larger dataset and due to the smaller hidden dimension and therefore less likely to overfit.

## 7.2 Working Mechanisms of NP-reg

NP-reg was shown to be a robust and feasible regularization option both in terms of computational overhead, sensitivity to hyperparameters and change in activation function. Using it does implement a network that has the property of preserving norm on unseen data across the first layer to a much greater extent than if it is not used. This in combination with its empirical performance makes it an attractive regularization choice. But why does it work and when might it be preferred over existing methods?

There are essentially two lenses one can look through to explain why a regularization method works. The more classical is that the final model ends up with parameters that generalizes better by introducing a bias in minima. This is how weight decay originally was explained (models with smaller magnitude weights do not overfit to the noise). The more modern explanation is that it changes the learning dynamics of optimizers in a beneficial way. In reality, both lenses provide useful insight, a good regularization technique needs to provide minima that generalize well while also providing favorable conditions for gradient descent to reach the minima.

### Change in Learning Dynamics

It is questionable how much can be improved upon the learning dynamics of the shallow networks since they do not define as complex of a loss landscape as deeper models do. The results from the experiments (Figure 5.2) showed that NP-reg made the loss landscape more rugged while still improving performance. Gradient smoothing is neither a complete explanation for why Batch Norm works in the shallow models. The landscape was improved for the overparametrized implementation but worsened for the smaller model while still improving performance. There must therefore be some other explanation.

### Reducing Representational Collapse

As could be seen in Section 5.6, NP-reg seemed to have a noticeable effect on the singular value distribution of the features, the same could also be seen for Batch Norm. For UCI HAR the feature representations did not collapse as much as they did with the base model. If the resulting directions are representative of the output labels they can then more easily be separated by the final linear layer. The effect of rank-diminishing comes from the learning process where the signal is separated from the noise. But limiting this and instead forcing the first layer to produce more unbiased features seem to improve generalization. In this task this seems like a reasonable explanation for the success of NP-reg. This explanation also satisfactorily explains Batch Norm, better than the explanation of smoother gradient landscape as that did not apply to the small models. Furthermore it might explain why Extreme Learning Machines can sometimes perform better than models trained with gradient descent as the random first layer preserves a diverse representation but with the drawback of having to accept noisy features.

The explanation does not on the other hand, explain the results from the IMDB dataset. For this task NP-reg outperformed the baseline while Batch Norm underperformed. Here NP-reg had the opposite effect on singular value distribution, making it more skewed and essentially reducing it to fewer dimensions. Batch Norm on the other hand increased the effective rank but performed worse. These counterintuitive results might be explained by the fact that IMDB was a binary classification task, so even when collapsing the input into fewer dimensions, if the representation is correct the classes are still linearly separable. Furthermore, the IMDB task was the one with the highest variance in input norm and it is reasonable to think that the input norm (length of the review) carries an important signal that NP-reg was able to utilize. The normalization of Batch Norm and Layer Norm might also have been disturbed by the sparsity of the input format. Using raw counts with the Bag-of-Words embedder meant that the vast majority of inputs to the model were zero and this might lead to unstable calculations of mean and variance in the Batch Norm and Layer Norm layers.

## Other Effects of the Regularization

Applying too high penalty did harm the performance of the models just like applying a hard constraint did. Therefore the minima with no norm scaling are not inherently better generalizers but instead have a neighborhood of points that can generalize well. Similarly, encouraging a fixed feature norm was shown to perform worse than NP-reg which slightly points towards the orthogonal regularization effect being important.

An interesting property is that it could have a harmful effect when applied to smaller models as seen in Figure 4.1, seemingly it only provides benefit in the heavily overparametrized domain with many solutions with low training loss. Here the bias of the regularization is allowed to come forward without negative impact on the objective function.

NP-reg also outperformed Batch Norm and Layer Norm on the small subsets of Food-101. This, which might be the most useful result, has no clear explanation. But the low amount of data might make the bias of the regularization more apparent, that the signal of the feature norm is useful and that this could be leveraged. But further analysis would have to be conducted to explore this.

## 7.3 Limitations

### Model Selection

Because of the many variables and places with randomness it was difficult to get reliable results. Especially with the smaller dataset (UCI HAR). This was mitigated by running the experiments with many different seeds (10 for CIFAR-10 and IMDB and 40 for UCI HAR) which helped in getting a more reliable mean. But the practice of repeated evaluation was not conducted for the grid search in the CIFAR-10

and IMDb experiments so the "best" hyperparameters for each model might not necessarily be the best parameters in general.

Another limitation is in the range and granularity of hyperparameters tested. Only three hyperparameter values were tested for weight decay, dropout and NP-reg and they were kept the same for all three experiments. Having more fine tuned values that are separate for each experiment would undoubtedly increase performance of all methods but since raw performance was not the goal of the experiments, wider and coarser ranges were used.

Conducting a search over more values was considered using Bayesian optimization tools such as Optuna [Akiba et al., 2019]. While this would automate the process of hyperparameter tuning, possibly enabling a bigger search space, it also comes with drawbacks which made it less suitable than grid search. Specifically, the computational expense of a thorough search and the lack of transparency in the search process.

## Evaluation

Because of its similarities to orthogonal regularization, a direct comparison to this would be of interest in the performance analysis. Preliminary experiments showed it to have similar performance to Layer Norm on the different tasks but the major limitation was the big overhead in computation time. Due to this computational overhead and that the similarity was found late in the project, a full grid search of hyperparameters was not possible within the time frame.

A final ablation study could have been done looking at normalization of the input and how omitting it might impact performance. Since the effect of NP-reg is directly dependent on the scale of the input it is reasonable to think that this has an impact. But none of the used datasets would provide a useful analysis. An unnormalized CIFAR-10 would just change the pixel values to be within  $[0,1]$  not providing much of a difference in magnitude from the normalized version. The IMDb input was raw counts and thus not normalized and UCI HAR is normalized by default so this could not be undone.

Other evaluation metrics than accuracy could have been used to test performance. F1-score and similar metrics might have been more suitable, especially on unbalanced datasets. Nonetheless, it was decided against in order to keep the results of all experiments consistent and easily interpretable.

## 7.4 Future Work

Two directions of future work can be seen with this as basis. More improvements upon SNN and further analysis of NP-reg.

## SNN Improvements

For real world scenarios where SNNs are used, further performance improvements would be useful. One example is to make use of distillation in order to provide more potent data and thus better models. Furthermore SNNs could easily be implemented in distributed learning environments. Any small IoT device could essentially learn a basic task and the results could be ensembled or kept personal.

Another direction is to use the GSVD to gain useful insights into the model. Even though NP-reg does not strictly follow norm-preservation across the first layer, the decomposed  $\Sigma$  and  $U$  matrix might still carry useful information. Furthermore, as the method is directly linked to the  $\ell_2$ -norm of the input, tasks where this is a relevant signal might be of particular interest. For example in energy applications.

## NP-reg Work

Applying NP-reg to other architectures is perhaps the most useful and interesting research direction. As mentioned, NP-reg is not limited to only being applied across the first layer but could just as well be used in intermediate layers or across multiple layers. A natural starting point would be to try to mimic the results of [Daneshmand et al., 2020] and see if NP-reg can avoid rank-diminishing in deeper networks. Given the rank-preserving similarities to Batch Norm shown here this seems quite probable.

Moving on to more complex architectures, one difficulty is the use of residual layers which is ubiquitous in deeper architectures like transformers and deep CNNs. If NP-reg were to be implemented in its current form with skip connections it would push the residual towards zero since the entire norm is already passed through the skip connection. This would make the layer nothing more than an identity function. Nonetheless this is by no means an unsolvable problem. [De and Smith, 2020] showed that the deeper the architecture of ResNet blocks, the more closely the earlier layers approximate the identity function. So perhaps a gradually increasing  $\lambda_{np}$  could be used for deeper networks to mimic this behavior.

Another research direction would be to take inspiration from Batch Norm and implement NP-reg as a hard constraint at initialization, but with a learned hyperparameter that can lessen the effect of the constraint possibly improving upon the results of the hard-constraint model with the added benefit of not requiring hand-picked hyperparameters.

# 8

## Conclusion

Although Shallow Neural Networks are less common than deep architectures, they remain relevant in settings where simplicity, low inference cost, or limited compute are important. While research has mostly been directed to modifications of the architecture, this thesis has provided results in improving the base model by analyzing different methods of regularization.

The results in regards to the empirical research questions were that: (1) Batch Norm, Layer Norm and NP-reg have an effect on the performance of SNNs in the evaluated classification tasks. The effects depend on the format of the input and output but they were most noticeable in the heavily overparametrized models. For all methods there were also cases where they produced subpar accuracy indicating that no single method is suitable for all problems. (2) NP-reg remained computationally feasible and robust across different activation functions. It can have a rank increasing effect on features for some datasets but there is not a single cohesive explanation for when and why it works across the board. (3) NP-reg can be useful as a utility when doing transfer learning of classifier heads given that the task at hand gives room for improvement via regularization, this was especially apparent with access to limited data.

NP-reg showed promising results across several tasks and model sizes. But in order to be truly useful it would have to be applied to deeper architectures. Further investigation could also be conducted on improvements to SNNs. Evidently their learning behavior differ from DNNs so standard machine learning practice might not necessarily apply to SNNs. With more progress, SNNs may open doors to new applications in the world of efficiency-driven AI.

# Bibliography

- Akiba, T., S. Sano, T. Yanase, T. Ohta, and M. Koyama (2019). “Optuna: A Next-generation Hyperparameter Optimization Framework”. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. Association for Computing Machinery. DOI: 10 . 1145 / 3292500 . 3330701.
- Anguita, D., A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz (2013). “A Public Domain Dataset for Human Activity Recognition Using Smartphones”. *Computational Intelligence*.
- Ba, J. L., J. R. Kiros, and G. E. Hinton (2016). *Layer Normalization*. arXiv: 1607 . 06450v1.
- Ba, L. J. and R. Caruana (2014). “Do Deep Nets Really Need to be Deep?” In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc.
- Bansal, N., X. Chen, and Z. Wang (2018). “Can We Gain More from Orthogonality Regularizations in Training Deep Networks?” In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc.
- Bossard, L., M. Guillaumin, and L. Van Gool (2014). “Food-101 – Mining Discriminative Components with Random Forests”. In: Fleet, D. et al. (Eds.). *Computer Vision – ECCV 2014*. Springer International Publishing. DOI: 10 . 1007 / 978 - 3 - 319 - 10599 - 4 \_ 29.
- Brown, B. C., R. Bridges, D. Grimsman, M. Munoz, and S. Warnick (2026). *A Generalized Singular Value Theory for Neural Networks*. arXiv: 2605 . 06938v1.
- Brown, B. C., M. King, S. Warnick, E. Yeung, and D. Grimsman (2025). “An SVD-like Decomposition of Functions with Finite 2-induced Norm”. In: *2025 American Control Conference (ACC)*. DOI: 10 . 23919 / ACC63710 . 2025 . 11108047.
- Cybenko, G. (1989). “Approximation by superpositions of a sigmoidal function”. *Mathematics of Control, Signals, and Systems* 2:4. DOI: 10 . 1007 / BF02551274.

- D'Angelo, F., M. Andriushchenko, A. Varre, and N. Flammarion (2024). "Why Do We Need Weight Decay in Modern Deep Learning?" *Advances in Neural Information Processing Systems* **37**. DOI: 10.52202/079017-0730.
- Daneshmand, H., J. Kohler, F. Bach, T. Hofmann, and A. Lucchi (2020). "Batch normalization provably avoids ranks collapse for randomly initialised deep networks". In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc.
- De, S. and S. Smith (2020). "Batch Normalization Biases Residual Blocks Towards the Identity Function in Deep Networks". In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc.
- Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei (2009). "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. DOI: 10.1109/CVPR.2009.5206848.
- Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: Burstein, J. et al. (Eds.). *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics. DOI: 10.18653/v1/N19-1423.
- Google. *Compact Language Detector v3*. URL: <https://github.com/google/cld3>. (Visited on 2026-05-15).
- Grattafiori, A. et al. (2024). *The Llama 3 Herd of Models*. arXiv: 2407.21783v3.
- He, K., X. Zhang, S. Ren, and J. Sun (2016). "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR.2016.90.
- Hornik, K., M. Stinchcombe, and H. White (1989). "Multilayer feedforward networks are universal approximators". *Neural Networks* **2**:5. DOI: 10.1016/0893-6080(89)90020-8.
- Huang, G.-B., D. H. Wang, and Y. Lan (2011). "Extreme learning machines: a survey". *International Journal of Machine Learning and Cybernetics* **2**:2. DOI: 10.1007/s13042-011-0019-y.
- Huang, G.-B., Q.-Y. Zhu, and C.-K. Siew (2006). "Extreme learning machine: Theory and applications". *Neurocomputing*. *Neural Networks* **70**:1. DOI: 10.1016/j.neucom.2005.12.126.
- Imran, A. A. and M. F. Ishmam (2024). "FourierKAN outperforms MLP on Text Classification Head Fine-tuning". In: *NeurIPS 2024 Workshop on Fine-Tuning in Modern Machine Learning: Principles and Scalability*.
- Ioffe, S. and C. Szegedy (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *Proceedings of the 32nd International Conference on Machine Learning*. PMLR.

- Krizhevsky, A. and G. E. Hinton (2009). *Learning Multiple Layers of Features from Tiny Images*. MA thesis. University of Toronto.
- Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc.
- Krogh, A. and J. Hertz (1991). “A Simple Weight Decay Can Improve Generalization”. In: *Advances in Neural Information Processing Systems*. Morgan-Kaufmann.
- Leshno, M., V. Y. Lin, A. Pinkus, and S. Schocken (1993). “Multilayer feedforward networks with a nonpolynomial activation function can approximate any function”. *Neural Networks* **6**:6. DOI: 10.1016/S0893-6080(05)80131-5.
- Li, H., Z. Xu, G. Taylor, C. Studer, and T. Goldstein (2018). “Visualizing the Loss Landscape of Neural Nets”. In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc.
- Lopez-Martin, M., B. Carro, A. Sanchez-Esguevillas, and J. Lloret (2019). “Shallow neural network with kernel approximation for prediction problems in highly demanding data networks”. *Expert Systems with Applications* **124**. DOI: 10.1016/j.eswa.2019.01.063.
- Loshchilov, I. and F. Hutter (2018). “Decoupled Weight Decay Regularization”. In: *International Conference on Learning Representations*.
- Maas, A. L., R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts (2011). “Learning Word Vectors for Sentiment Analysis”. In: Lin, D. et al. (Eds.). *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.
- Nair, V. and G. E. Hinton (2010). “Rectified linear units improve restricted boltzmann machines”. In: *Proceedings of the 27th International Conference on Machine Learning (ICML)*. Omnipress.
- Nguyen, Q. and M. Hein (2018). “Optimization Landscape and Expressivity of Deep CNNs”. In: *Proceedings of the 35th International Conference on Machine Learning*. PMLR.
- Pasupa, K. and W. Sunhem (2016). “A comparison between shallow and deep architecture classifiers on small dataset”. In: *2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE)*. DOI: 10.1109/ICITEED.2016.7863293.
- Perfilieva, I., P. Artiemjew, and A. Niemczynowicz (2025). “Randomness Versus Backpropagation in Machine Learning”. In: Rutkowski, L. et al. (Eds.). *Artificial Intelligence and Soft Computing*. Springer Nature Switzerland. DOI: 10.1007/978-3-031-84353-2\_19.

- Reis, M. J. C. S. (2025). “Lightweight Signal Processing and Edge AI for Real-Time Anomaly Detection in IoT Sensor Networks”. *Sensors (Basel, Switzerland)* **25**:21. DOI: 10.3390/s25216629.
- Roy, O. and M. Vetterli (2007). “The effective rank: A measure of effective dimensionality”. In: *2007 15th European Signal Processing Conference*.
- Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). “Learning representations by back-propagating errors”. *Nature* **323**:6088. DOI: 10.1038/323533a0.
- Santurkar, S., D. Tsipras, A. Ilyas, and A. Madry (2018). “How Does Batch Normalization Help Optimization?” In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc.
- Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov (2014). “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. *Journal of Machine Learning Research* **15**:56.
- Urban, G., K. J. Geras, S. E. Kahou, O. Aslan, S. Wang, A. Mohamed, M. Philipose, M. Richardson, and R. Caruana (2017). “Do Deep Convolutional Nets Really Need to be Deep and Convolutional?” In: *International Conference on Learning Representations*.
- Vorontsov, E., C. Trabelsi, S. Kadoury, and C. Pal (2017). “On orthogonality and learning recurrent networks with long term dependencies”. In: *Proceedings of the 34th International Conference on Machine Learning*. PMLR.
- Yelp Inc. *Yelp Open Dataset*. URL: <https://business.yelp.com/data/resources/open-dataset>. (Visited on 2026-05-15).

# 9

## Appendix

### 9.1 Code

The experiments were implemented in Python 3.13 using PyTorch 2.10. A repository containing implementation of all experiments and results can be found at: <https://github.com/danielstoopendahl/np-reg>

### 9.2 Model Sizes

Model sizes for the SNNs in Chapter 4 are presented in Table 9.1.

**Table 9.1** Model sizes used in the CIFAR-10, IMDB and UCI HAR experiments. For CIFAR-10 and UCI HAR the hidden dimension is varied while the input dimension is fixed (3072 and 561). For IMDB both the vocabulary size and hidden dimension are varied. Parameter counts include all weights and biases of both linear layers.

| Size | CIFAR-10 |            | IMDb           |            | UCI HAR |           |
|------|----------|------------|----------------|------------|---------|-----------|
|      | Hidden   | # Params   | Vocab / Hidden | # Params   | Hidden  | # Params  |
| 1    | 8        | 24 674     | 1562 / 8       | 12 522     | 8       | 4550      |
| 2    | 32       | 98 666     | 3125 / 16      | 50 050     | 32      | 18 182    |
| 3    | 128      | 394 634    | 6250 / 32      | 200 098    | 128     | 72 710    |
| 4    | 512      | 1 578 506  | 12 500 / 64    | 800 194    | 512     | 290 822   |
| 5    | 2048     | 6 313 994  | 25 000 / 128   | 3 200 386  | 2048    | 1 163 270 |
| 6    | 8192     | 25 255 946 | 50 000 / 256   | 12 800 770 | 8192    | 4 653 062 |

### 9.3 Hyperparameter Grid Search

The best found configuration of hyperparameters for each experiment are presented in Tables 9.2 and 9.3.

**Table 9.2** Best hyperparameters for the CIFAR-10, IMDB and UCI HAR experiments. Cells report  $lr, bs$  for Vanilla/Batch Norm/Layer Norm, and  $lr, bs, extra$  for Weight Decay ( $\lambda_{wd}$ ), Dropout ( $p$ ), and NP-reg ( $\lambda_{np}$ ).

| Model Size | Method       | CIFAR-10        | IMDb            | UCI HAR        |
|------------|--------------|-----------------|-----------------|----------------|
| Size 1     | Vanilla      | 1e-4, 128       | 3e-5, 32        | 3e-4, 32       |
|            | Weight Decay | 1e-4, 128, 1e-4 | 3e-5, 32, 1e-5  | 3e-4, 32, 1e-3 |
|            | Dropout      | 1e-4, 256, 0.1  | 3e-6, 32, 0.1   | 3e-4, 32, 0.3  |
|            | Batch Norm   | 1e-4, 64        | 1e-5, 128       | 3e-4, 128      |
|            | Layer Norm   | 3e-5, 256       | 3e-5, 128       | 3e-4, 64       |
|            | NP-reg       | 1e-4, 256, 1e-3 | 3e-5, 64, 1e-3  | 3e-4, 32, 1e-3 |
| Size 2     | Vanilla      | 1e-4, 64        | 1e-5, 32        | 3e-4, 64       |
|            | Weight Decay | 1e-4, 256, 1e-3 | 1e-5, 32, 1e-5  | 3e-4, 64, 1e-5 |
|            | Dropout      | 1e-4, 128, 0.1  | 1e-5, 64, 0.1   | 3e-4, 32, 0.5  |
|            | Batch Norm   | 1e-4, 256       | 3e-5, 128       | 3e-4, 32       |
|            | Layer Norm   | 1e-4, 128       | 3e-5, 64        | 3e-4, 32       |
|            | NP-reg       | 1e-4, 128, 1e-3 | 3e-5, 32, 1e-1  | 3e-4, 32, 1e-3 |
| Size 3     | Vanilla      | 1e-4, 256       | 3e-6, 32        | 3e-4, 32       |
|            | Weight Decay | 1e-4, 64, 1e-3  | 3e-6, 32, 1e-5  | 3e-4, 32, 1e-4 |
|            | Dropout      | 1e-4, 128, 0.1  | 3e-6, 64, 0.5   | 3e-4, 64, 0.5  |
|            | Batch Norm   | 3e-5, 256       | 3e-6, 128       | 3e-4, 32       |
|            | Layer Norm   | 3e-5, 128       | 3e-6, 64        | 3e-4, 32       |
|            | NP-reg       | 1e-4, 128, 1e-3 | 3e-5, 128, 1e-2 | 3e-4, 32, 1e-3 |
| Size 4     | Vanilla      | 1e-4, 256       | 3e-6, 32        | 3e-4, 32       |
|            | Weight Decay | 3e-5, 128, 1e-3 | 3e-6, 32, 1e-5  | 1e-4, 32, 1e-3 |
|            | Dropout      | 1e-4, 128, 0.3  | 3e-6, 64, 0.1   | 1e-4, 64, 0.3  |
|            | Batch Norm   | 1e-4, 128       | 3e-6, 128       | 1e-4, 32       |
|            | Layer Norm   | 1e-4, 256       | 3e-6, 32        | 3e-4, 32       |
|            | NP-reg       | 1e-4, 128, 1e-1 | 3e-5, 128, 1e-1 | 3e-4, 64, 1e-2 |
| Size 5     | Vanilla      | 3e-5, 64        | 1e-5, 32        | 1e-4, 32       |
|            | Weight Decay | 3e-5, 64, 1e-4  | 1e-5, 32, 1e-4  | 1e-4, 32, 1e-3 |
|            | Dropout      | 3e-5, 128, 0.3  | 3e-6, 32, 0.5   | 1e-4, 32, 0.1  |
|            | Batch Norm   | 1e-4, 256       | 3e-6, 32        | 1e-4, 128      |
|            | Layer Norm   | 1e-4, 256       | 3e-5, 32        | 3e-4, 128      |
|            | NP-reg       | 1e-4, 128, 1e-1 | 3e-5, 128, 1e-1 | 3e-4, 64, 1e-3 |
| Size 6     | Vanilla      | 1e-5, 64        | 3e-6, 32        | 3e-4, 128      |
|            | Weight Decay | 1e-5, 64, 1e-4  | 3e-6, 32, 1e-4  | 3e-5, 32, 1e-4 |
|            | Dropout      | 3e-5, 256, 0.1  | 3e-6, 32, 0.3   | 3e-4, 128, 0.3 |
|            | Batch Norm   | 1e-4, 256       | 3e-6, 128       | 1e-4, 128      |
|            | Layer Norm   | 3e-5, 256       | 3e-5, 64        | 1e-4, 32       |
|            | NP-reg       | 1e-4, 64, 1e-1  | 3e-5, 128, 1e-1 | 3e-5, 32, 1e-2 |

**Table 9.3** Best hyperparameters for the Food-101 and Yelp Open Dataset experiments. Vanilla cells report  $lr$ ; NP reports  $lr, \lambda_{np}$ ; OPT reports  $lr$ , norm,  $\lambda_{wd}$ ,  $p$ ; NP+OPT reports  $lr, \lambda_{np}, \lambda_{wd}, p$ .

| Data Fraction | Method              | Food-101              | Yelp Open Dataset     |
|---------------|---------------------|-----------------------|-----------------------|
| 1%            | Vanilla             | 1e-4                  | 3e-5                  |
|               | NP-reg              | 1e-4, 1e-2            | 1e-4, 0.1             |
|               | BN/LN/None + D + WD | 1e-4, None, 1e-4, 0.5 | 1e-4, None, 1e-3, 0.5 |
|               | NP-reg + D + WD     | 1e-4, 0.1, 1e-5, 0.5  | 1e-4, 1e-2, 1e-3, 0.5 |
| 3%            | Vanilla             | 1e-4                  | 3e-5                  |
|               | NP-reg              | 1e-4, 1e-2            | 1e-4, 1e-2            |
|               | BN/LN/None + D + WD | 1e-4, LN, 1e-3, 0.5   | 3e-5, None, 1e-3, 0.1 |
|               | NP-reg + D + WD     | 3e-5, 1e-2, 1e-3, 0.5 | 3e-5, 1e-3, 1e-5, 0.1 |
| 10%           | Vanilla             | 1e-5                  | 1e-4                  |
|               | NP-reg              | 1e-4, 1e-2            | 1e-4, 1e-2            |
|               | BN/LN/None + D + WD | 3e-5, BN, 1e-3, 0.3   | 3e-5, None, 1e-3, 0.5 |
|               | NP-reg + D + WD     | 3e-5, 1e-2, 1e-4, 0.5 | 3e-5, 1e-1, 1e-5, 0.5 |
| 30%           | Vanilla             | 3e-5                  | 3e-5                  |
|               | NP-reg              | 1e-4, 1e-2            | 1e-4, 0.1             |
|               | BN/LN/None + D + WD | 1e-4, BN, 1e-3, 0.5   | 3e-5, LN, 1e-3, 0.5   |
|               | NP-reg + D + WD     | 1e-4, 1e-3, 1e-5, 0.5 | 1e-4, 1e-2, 1e-3, 0.5 |
| 100%          | Vanilla             | 3e-5                  | 3e-5                  |
|               | NP-reg              | 1e-4, 1e-2            | 1e-4, 1e-3            |
|               | BN/LN/None + D + WD | 3e-5, LN, 1e-4, 0.5   | 3e-5, LN, 1e-5, 0.5   |
|               | NP-reg + D + WD     | 1e-4, 0.1, 1e-3, 0.1  | 1e-4, 1e-1, 1e-3, 0.5 |

## 9.4 Raw Test Results

The performance mean and standard deviation for all dataset and methods are presented in Table 9.4 and 9.5.

**Table 9.4** Mean test accuracy and standard deviation (%) for CIFAR-10, IMDB and UCI HAR experiments.

| Model Size | Method       | CIFAR-10         | IMDb             | UCI HAR          |
|------------|--------------|------------------|------------------|------------------|
| Size 1     | Vanilla      | 36.51 $\pm$ 1.44 | 85.76 $\pm$ 0.06 | 92.33 $\pm$ 0.74 |
|            | Weight Decay | 36.51 $\pm$ 1.44 | 85.76 $\pm$ 0.06 | 92.33 $\pm$ 0.74 |
|            | Dropout      | 37.82 $\pm$ 0.60 | 85.79 $\pm$ 0.05 | 92.85 $\pm$ 0.95 |
|            | Batch Norm   | 40.12 $\pm$ 0.66 | 85.28 $\pm$ 0.21 | 94.14 $\pm$ 0.74 |
|            | Layer Norm   | 36.72 $\pm$ 0.76 | 85.24 $\pm$ 0.13 | 94.35 $\pm$ 0.67 |
|            | NP-reg       | 30.05 $\pm$ 2.41 | 85.77 $\pm$ 0.07 | 92.26 $\pm$ 0.98 |
| Size 2     | Vanilla      | 48.47 $\pm$ 0.69 | 87.00 $\pm$ 0.08 | 92.63 $\pm$ 0.33 |
|            | Weight Decay | 48.51 $\pm$ 0.84 | 87.03 $\pm$ 0.33 | 92.63 $\pm$ 0.33 |
|            | Dropout      | 48.92 $\pm$ 0.61 | 87.05 $\pm$ 0.07 | 93.01 $\pm$ 0.37 |
|            | Batch Norm   | 50.50 $\pm$ 0.36 | 86.51 $\pm$ 0.12 | 93.58 $\pm$ 0.66 |
|            | Layer Norm   | 48.43 $\pm$ 0.59 | 86.96 $\pm$ 0.13 | 93.59 $\pm$ 0.48 |
|            | NP-reg       | 45.17 $\pm$ 1.20 | 87.19 $\pm$ 0.23 | 92.57 $\pm$ 0.36 |
| Size 3     | Vanilla      | 53.40 $\pm$ 0.84 | 87.74 $\pm$ 0.23 | 92.59 $\pm$ 0.29 |
|            | Weight Decay | 54.12 $\pm$ 0.59 | 87.84 $\pm$ 0.06 | 92.57 $\pm$ 0.27 |
|            | Dropout      | 55.39 $\pm$ 0.46 | 87.85 $\pm$ 0.05 | 92.76 $\pm$ 0.27 |
|            | Batch Norm   | 57.34 $\pm$ 0.64 | 87.38 $\pm$ 0.18 | 93.19 $\pm$ 0.85 |
|            | Layer Norm   | 55.51 $\pm$ 0.65 | 87.31 $\pm$ 0.14 | 93.36 $\pm$ 0.56 |
|            | NP-reg       | 52.78 $\pm$ 0.97 | 88.05 $\pm$ 0.06 | 92.82 $\pm$ 0.27 |
| Size 4     | Vanilla      | 57.73 $\pm$ 0.18 | 88.21 $\pm$ 0.11 | 92.74 $\pm$ 0.25 |
|            | Weight Decay | 57.07 $\pm$ 0.59 | 88.18 $\pm$ 0.10 | 92.73 $\pm$ 0.26 |
|            | Dropout      | 58.80 $\pm$ 0.51 | 88.19 $\pm$ 0.11 | 92.78 $\pm$ 0.25 |
|            | Batch Norm   | 63.19 $\pm$ 0.52 | 87.82 $\pm$ 0.06 | 93.37 $\pm$ 0.54 |
|            | Layer Norm   | 61.16 $\pm$ 0.61 | 87.69 $\pm$ 0.08 | 93.40 $\pm$ 0.20 |
|            | NP-reg       | 59.43 $\pm$ 0.60 | 88.39 $\pm$ 0.14 | 93.15 $\pm$ 0.17 |
| Size 5     | Vanilla      | 59.89 $\pm$ 0.29 | 88.26 $\pm$ 0.17 | 92.82 $\pm$ 0.14 |
|            | Weight Decay | 59.89 $\pm$ 0.29 | 88.27 $\pm$ 0.18 | 92.81 $\pm$ 0.14 |
|            | Dropout      | 59.55 $\pm$ 0.47 | 88.42 $\pm$ 0.07 | 92.86 $\pm$ 0.11 |
|            | Batch Norm   | 66.11 $\pm$ 0.45 | 88.03 $\pm$ 0.13 | 93.24 $\pm$ 0.23 |
|            | Layer Norm   | 64.27 $\pm$ 0.33 | 87.80 $\pm$ 0.34 | 93.42 $\pm$ 0.18 |
|            | NP-reg       | 63.97 $\pm$ 0.65 | 88.82 $\pm$ 0.05 | 93.68 $\pm$ 0.18 |
| Size 6     | Vanilla      | 61.74 $\pm$ 0.61 | 88.51 $\pm$ 0.08 | 92.93 $\pm$ 0.12 |
|            | Weight Decay | 61.74 $\pm$ 0.61 | 88.51 $\pm$ 0.08 | 92.95 $\pm$ 0.20 |
|            | Dropout      | 61.09 $\pm$ 0.51 | 88.54 $\pm$ 0.05 | 92.95 $\pm$ 0.13 |
|            | Batch Norm   | 67.48 $\pm$ 0.43 | 88.15 $\pm$ 0.19 | 93.33 $\pm$ 0.25 |
|            | Layer Norm   | 65.48 $\pm$ 0.62 | 87.97 $\pm$ 0.07 | 93.54 $\pm$ 0.16 |
|            | NP-reg       | 66.00 $\pm$ 0.46 | 88.88 $\pm$ 0.05 | 93.50 $\pm$ 0.15 |

**Table 9.5** Mean test accuracy and standard deviation (%) for Food-101 and Yelp Open Dataset experiments.

| Training Fraction | Method              | Food-101         | Yelp Open Dataset |
|-------------------|---------------------|------------------|-------------------|
| 1%                | Vanilla             | 41.34 $\pm$ 0.78 | 59.22 $\pm$ 0.32  |
|                   | NP-reg              | 42.45 $\pm$ 0.78 | 59.26 $\pm$ 0.31  |
|                   | BN/LN/None + D + WD | 41.62 $\pm$ 0.76 | 59.37 $\pm$ 0.30  |
|                   | NP-reg + D + WD     | 42.44 $\pm$ 0.75 | 59.41 $\pm$ 0.29  |
| 3%                | Vanilla             | 52.65 $\pm$ 0.46 | 61.01 $\pm$ 0.26  |
|                   | NP-reg              | 53.82 $\pm$ 0.45 | 60.94 $\pm$ 0.32  |
|                   | BN/LN/None + D + WD | 53.04 $\pm$ 0.44 | 61.02 $\pm$ 0.28  |
|                   | NP-reg + D + WD     | 53.98 $\pm$ 0.45 | 61.04 $\pm$ 0.28  |
| 10%               | Vanilla             | 61.87 $\pm$ 0.21 | 62.27 $\pm$ 0.23  |
|                   | NP-reg              | 62.81 $\pm$ 0.24 | 62.28 $\pm$ 0.23  |
|                   | BN/LN/None + D + WD | 62.29 $\pm$ 0.21 | 62.34 $\pm$ 0.20  |
|                   | NP-reg + D + WD     | 62.90 $\pm$ 0.21 | 62.50 $\pm$ 0.15  |
| 30%               | Vanilla             | 67.18 $\pm$ 0.22 | 63.32 $\pm$ 0.14  |
|                   | NP-reg              | 68.03 $\pm$ 0.20 | 63.41 $\pm$ 0.18  |
|                   | BN/LN/None + D + WD | 67.91 $\pm$ 0.19 | 63.16 $\pm$ 0.19  |
|                   | NP-reg + D + WD     | 68.08 $\pm$ 0.18 | 63.36 $\pm$ 0.24  |
| 100%              | Vanilla             | 71.26 $\pm$ 0.11 | 64.31 $\pm$ 0.15  |
|                   | NP-reg              | 72.12 $\pm$ 0.11 | 64.35 $\pm$ 0.14  |
|                   | BN/LN/None + D + WD | 72.24 $\pm$ 0.13 | 64.33 $\pm$ 0.21  |
|                   | NP-reg + D + WD     | 72.11 $\pm$ 0.12 | 64.34 $\pm$ 0.16  |

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                                       |                                                                                                                                                                                                                                          |             |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| <b>Lund University</b><br><b>Department of Automatic Control</b><br><b>Box 118</b><br><b>SE-221 00 Lund Sweden</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |                                       | <i>Document name</i><br><b>MASTER'S THESIS</b>                                                                                                                                                                                           |             |
|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                                       | <i>Date of issue</i><br><b>June 2026</b>                                                                                                                                                                                                 |             |
|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                                       | <i>Document Number</i><br><b>TFRT-6317</b>                                                                                                                                                                                               |             |
| <i>Author(s)</i><br><b>Daniel Stoopendahl</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                                       | <i>Supervisor</i><br><b>Robert Bridges, AI Sweden, Sweden</b><br><b>Kristian Soltész, Dept. of Automatic Control, Lund University, Sweden</b><br><b>Pontus Giselsson, Dept. of Automatic Control, Lund University, Sweden (examiner)</b> |             |
| <i>Title and subtitle</i><br><b>Improving the Performance of Shallow Neural Networks Using Norm-Preserving Regularization</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                                       |                                                                                                                                                                                                                                          |             |
| <i>Abstract</i><br><p>The Shallow Neural Network (SNN) is one of the most basic forms of machine learning models. It has a high theoretical representational capacity, but this is often not realized in practice, in part due to overfitting. This thesis proposes a new regularization method and compares it against existing techniques across a range of tasks using SNNs. The proposed method, norm-preserving regularization (NP-reg), penalizes <math>\ell_2</math>-norm difference between the input vector and hidden representation. Adding this penalty term to the objective function was found to affect the generalization performance of SNNs, particularly in heavily overparametrized settings.</p> <p>Performance improvements were observed for larger models, both in standalone SNNs and when used as a classifier head in transfer learning settings. For smaller models, NP-reg generally had negligible or negative effects on performance. Among existing regularization methods, Batch Norm and Layer Norm showed both positive and negative effects depending on the task, while dropout and weight decay had limited impact.</p> <p>The mechanisms underlying the performance gains remain unclear. NP-reg produced a rougher loss landscape while still improving performance. This was also observed for Batch Norm, suggesting that smoother optimization alone does not explain the performance gains in the shallow settings. In some tasks, NP-reg reduced rank degradation of feature representations similarly to what Batch Norm was found to do, though neither this effect was consistently associated with improved performance.</p> <p>Despite the lack of a complete theoretical explanation, NP-reg showed practical value and was shown to be robust across different settings, functioning with multiple activation functions while remaining more computationally feasible than similar approaches.</p> |                                       |                                                                                                                                                                                                                                          |             |
| <i>Keywords</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                                       |                                                                                                                                                                                                                                          |             |
| <i>Classification system and/or index terms (if any)</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                                       |                                                                                                                                                                                                                                          |             |
| <i>Supplementary bibliographical information</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |                                       |                                                                                                                                                                                                                                          |             |
| <i>ISSN and key title</i><br><b>0280-5316</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                                       |                                                                                                                                                                                                                                          | <i>ISBN</i> |
| <i>Language</i><br><b>English</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | <i>Number of pages</i><br><b>1-50</b> | <i>Recipient's notes</i>                                                                                                                                                                                                                 |             |
| <i>Security classification</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                                       |                                                                                                                                                                                                                                          |             |

<http://www.control.lth.se/publications/>