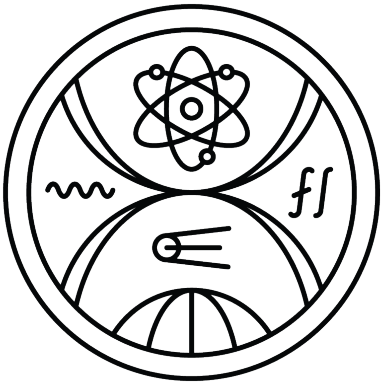
Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

# Semi-supervised learning in Deep Neural Networks

Diploma Thesis

2023
Bc. Sabína Samporová

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics



# Semi-supervised learning in Deep Neural Networks

Diploma Thesis

| | |
|---|---|
| Study Programme: | Applied Computer Science TODO |
| Field of Study: | Computer Science TODO |
| Department: | Department of Computer Science TODO |
| Supervisor: | RNDr. Kristína Malinovská, PhD. |

Bratislava, 2023
Bc. Sabína Samporová


Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

# ZADANIE ZÁVEREČNEJ PRÁCE

**Meno a priezvisko študenta:** Bc. Sabína Samporová
**Študijný program:** aplikovaná informatika (Jednoodborové štúdium, magisterský II. st., denná forma)
**Študijný odbor:** informatika
**Typ záverečnej práce:** diplomová
**Jazyk záverečnej práce:** slovenský
**Sekundárny jazyk:** anglický

**Názov:** Čiastočne riadené učenie hlbokých neurónových sietí
*Semi-supervised learning in Deep Neural Networks*

**Anotácia:** Hlboké neurónové siete sú v súčasnosti pravdepodobne najpoužívanejšími a najskúmanejšími modelmi v strojovom učení s aplikáciami v mnohých rôznych oblastiach. Trénovanie takýchto modelov si však vyžaduje množstvo adekvátne označených trénovacích dát, ale zvyčajne je dobre označených dát z reálneho sveta málo. Paradigma semi-supervised learning (čiastočne riadené učenie) rieši tento problém prostredníctvom rôznych techník, ktoré sú zvyčajne založené na vyjadrení a vyhodnotení vzdialenosti medzi príznakovými vektormi (embeddings) označených a neoznačených trénovacích dát a učenie je založené na miere ich podobnosti. Príkladom tohto prístupu je trieda modelov hlbokých neurónových sietí založených na takzvanom Mean Teacher model.

**Cieľ:** Preskúmajte existujúce modely v rámci čiastočne riadeného učenia na kategorizáciu so zameraním na model Mean Teacher (MT). Urobte prehľad súčasného stavu problematiky a navrhnite a implementujte nový model ako vylepšenie MT modelu. Implementujte a vyhodnoťte experimenty s vybraným benchmark datasetom a porovnajte nový model s existujúcimi.

**Literatúra:**
[1] Goodfellow, I., Bengio, Y. and Courville, A., 2016. Deep learning. MIT press.
[2] Tarvainen, A. and Valpola, H., 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. Advances in neural information processing systems, 30.
[3] Tuna, M., Malinovská, K., Farkaš, I., Kraus, S. and Krsek, P., 2021, October. Semi-supervised Learning in Camera Surveillance Image Classification. In 2021 IEEE 17th International Conference on Intelligent Computer Communication and Processing (ICCP) (pp. 155-162). IEEE.

**Vedúci:** RNDr. Kristína Malinovská, PhD.
**Katedra:** FMFI.KAI - Katedra aplikovanej informatiky
**Vedúci katedry:** doc. RNDr. Tatiana Jajcayová, PhD.

**Dátum zadania:** 30.11.2022

**Dátum schválenia:** 01.12.2022

prof. RNDr. Roman Ďurikovič, PhD.
garant študijného programu

Comenius University Bratislava
Faculty of Mathematics, Physics and Informatics

# THESIS ASSIGNMENT

**Name and Surname:**
**Study programme:**

**Field of Study:**
**Type of Thesis:**
**Language of Thesis:**
**Secondary language:**

**Title:**

**Annotation:**

**Supervisor:**
**Department:**
**Head of department:**

**Assigned:**

**Approved:**

Guarantor of Study Programme

.....
Student

.....
Supervisor

**Acknowledgments:**

# Abstrakt

Hlboké neurónové siete sú v súčasnosti pravdepodobne najpoužívanejšími a najskúmanejšími modelmi v strojovom učení s aplikáciami v mnohých rôznych oblastiach. Trénovanie takýchto modelov si však vyžaduje množstvo adekvátne označených trénovacích dát, ale zvyčajne je dobre označených dát z reálneho sveta málo. Paradigma semi-supervised learning (čiastočne riadené učenie) rieši tento problém prostredníctvom rôznych techník, ktoré sú zvyčajne založené na vyjadrení a vyhodnotení vzdialenosti medzi príznakovými vektormi (embeddings) označených a neoznačených trénovacích dát a učenie je založené na miere ich podobnosti. Príkladom tohto prístupu je trieda modelov hlbokých neurónových sietí založených na takzvanom Mean Teacher model. V tejto práci skúmame spomínaný model a možnosti jeho vylepšenia s využitím samoorganizujúceho princípu.

**Kľúčové slová:**

# Abstract

These days, Deep neural networks are the most widely used and researched models in machine learning, with application in many different domains. However, training of such models requires an abundance of adequately labeled data and labels for the real world data are scarce. The semi-supervised learning paradigm aims at leveraging this problem via various different techniques that would typically involve capturing and evaluating the distance between the feature vectors of the learned labeled and unlabeled data and learning is based on similarity. This approach is used for example in the popular Mean Teacher model (MT). In this thesis, we investigate this model and look for posibilities of improvement using principle of self-organization.

**Keywords:** neural networks, semisupervised learning, mean-teacher model, unsupervised learning, self-organizing map

# Contents

# List of Figures

# List of Tables

# Introduction

# Chapter 1

# Overview and methods

In this chapter, we describe basic concept of artificial models called neural networks. We will breafly explain different aproaches of training and models that are designed to each type of training approach.

## 1.1 Neural networks

Neural networks are artificial inteligence models with applications in many domains, such as forecasting, computer vision or natural language processing. Since these models are successful, but still has some limitations in performance, it is useful to study them and try new approaches, that could possibly achieve better results.

Neural network is composed of processing units called neurons, which are connected to layers. Layers are then connected to network. Each neuron has input features, and internal weights, which are used for computation of weighted sum. Result of weighted sum is scalar and it is transformed by function, called activation function. Most used activation functions are logistic function, hyperbolic tangent or rectified linear unit - mostly in deep networks, with many layers.

## 1.2 Convolutional neural networks

Convolutional neural networks (CNNs) are a specialized kind of neural network for processing data that has a known grid-like topology, for example image data (2-D grid of pixels). CNNs have special type of layers - convolutional layers. These layers do not consist of neurons, but matrices of weights called kernels or filters. Each layer has it‘s own trainable weights and provides specialized kind of linear operation - convolution. We can see this process described in figure 1.1. Convolution leverages important ideas, such as sparse interactions or parameter sharing that can help improve a machine learning system. Another operation, called pooling, is also often emplyed in CNNs.

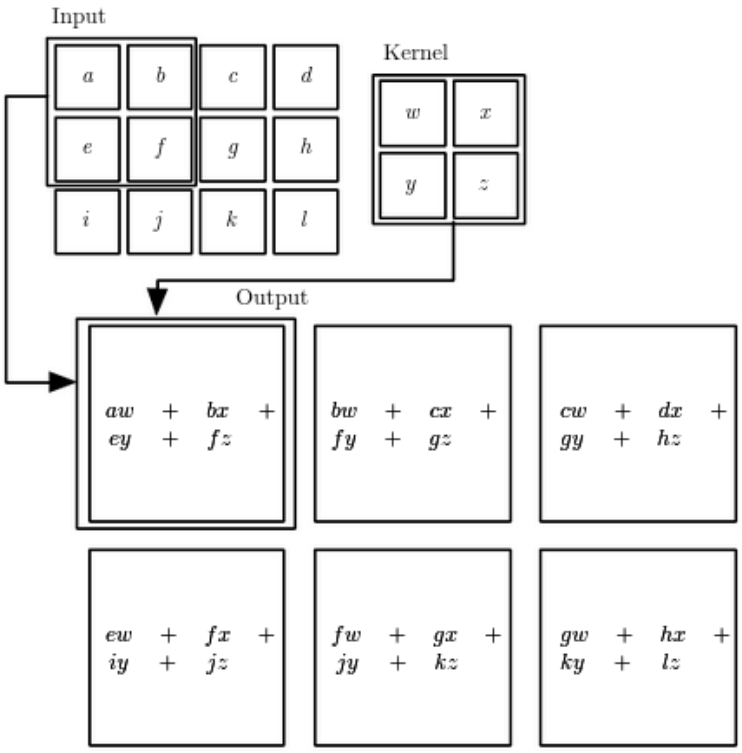



Figure 1.1: Convolution in CNNs [3]

When pooling layers are employed, network can be invariant to augmentations, such as translation, rotation or scaling. [3]

## 1.3 Supervised models

Recently, one of most used method for achieving useful real world results with neural network models is using supervised learning. It is simply mapping input described by set of features to output labels. In the beggining, mapping is not very accurate, so the aim of training is to adjust models parameters, called weights, such that the inference is more correct then before adjustment.

### 1.3.1 Multi layer perceptron

We already introduced neural networks in section 1.1. Multi layer perceptron (MLP) is neural network with input layer, output layer and at least one hidden layer. The goal of a MLP is to approximate some function $f^*$ by samples from this function. Samples are pairs of input and output $(x, y)$. MLP defines mapping $y = f^*(x, \theta)$, where $\theta$ denotes parameters of the model - weights. At the beginning, weight are randomized. Then using supervised learning from training examples, weights are adjusted to minimize loss function, which is expression of prediction error. For training of MLP, error

backpropagation algorithm is used. [3]

## 1.4 Unsupervised models

Unsupervised models work with data, that consist of only input samples, which means they does not have assigned desired output. This is common case, when we have some new task on real world data, that we do not know the desired output. Unsupervised mode They are able to capture the similarities of data or unseen data structures. Examples of such neural model is the Self-organizing map.

### 1.4.1 Self-organizing map

The self-organizing map (SOM), originally proposed by Kohonen in 1990 [4], is unsupervised neural model, based on competitive and cooperative principles. It is mostly used for clustering and visualization of high-dimensional data onto a low-dimensional grid. The grid is composed of units - neurons, each one associated with a prototype vector from the original data space. The learning algorithm enforces a topology constraint, so that neighboring map units correspond to prototypes that are close in the original space, according to euclidean distance. [2]

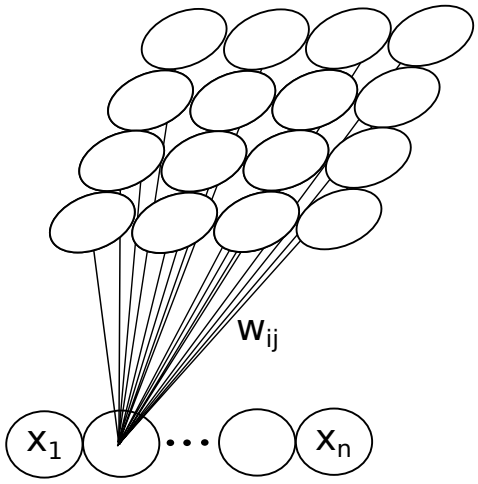



Figure 1.2: Self-organizing map

In process of training, competitive principle is used first. Among all neurons in network, one denoted $i^*$ (winner neuron, prototype) is chosen by equation 1.1. It is the one closest to the input $x$. As a distance measure, Euclidian norm is used.

$$i^* = \arg\min_i \|x(t) - w_i\| \tag{1.1}$$

Then all weights are adjusted based on cooperative principle. The change of each weight is computed by equation 1.2. The neurons closest (in low dimensional grid) to

winner are adjusted more, than those farther from it. That means all neurons are moved to represent input $x$ better. The magnitude of change is defined by learning rate $\alpha(t)$ decreasing in time. Distance of neurons in grid is defined by neighbourhood function denoted as $h$. Commonly used neighbourhood functions are Manhattan distance (eq. 1.4) or Gausian distance (eq. 1.3).

$$\Delta w_i = \alpha(t)h(i, i^*)(x(t) - w_i), \tag{1.2}$$

$$h(i, i^*) = e^{-d(i,i^*)^2/\sigma^2(t)} \tag{1.3}$$

$$h(i, i^*) = \begin{cases} 1 & \text{if } d_M(i, i^*) \leq \lambda(t) \\ 0 & \text{if otherwise} \end{cases} \tag{1.4}$$

### 1.4.2 SOM evaluation methods

For evaluation of SOM during the training, three metrics are typically used - quantization error, winner discrimination and entropy.

#### Quantization error

Quantization error (QE) calculates the average error at the unit as a result of quantization process [8]. It is calculated as mean of mean squared errors (MSE) of difference of $k$-th input $x^k(t)$ and its winner $i^*$ (eq. 1.5), where $M$ denotes number of inputs and $N$ denotes their dimensionality. Well trained SOM should have small quantization error.

$$QE = \frac{1}{M}\sum_{k=1}^{M} MSE(x^k(t) - i^*) \tag{1.5}$$

#### Winner discrimination

Winner discrimination (WD) is proportion of neurons chosen as winners during training. We can denote set of chosen neurons (winners) as $U$ and set of all neurons as $A$. Then WD is computed as in equation 1.6. Well trained SOM should choose all or almost all neurons as winners.

$$WD = \frac{|U|}{|A|} \tag{1.6}$$

#### Entropy

Entropy is a measure of order in the system. Entropy (E) for SOM evaluates how often various units become winners, so the highest entropy means most balanced unit

participation in the competition process [8]. Computation of entropy is shown in equation 1.7, where $p$ denotes vector, which contains probability of each unit to be chosen. Probability is estimated from training as a number of usages of a unit devided by all unit usages.

$$E = -\sum_{i}(p_i \log_2 p_i) \tag{1.7}$$

## 1.5 Semisupervised models

Semisupervised learning is method that use unlabeled data to improve model's learning possibilities. Usually, creating huge labeled dataset is difficult, but collecting huge amount of unlabeled and smaller portion of labeled data is feasible. This is, why semisupervised models can be really usefull and worth studying. Most semisupervised models are based on two ideas - consistency regularization and pseudo labels.

### 1.5.1 Consistency regularization models

Consistency regularization

### 1.5.2 Models using pseudolabels

Pseudolabels are iteratively achieved from neural based model, which is trained on labeled data. Then unlabeled data are put as the input of this model and prediction of the model is used as assigned value for this unlabeled datapoint. It is spproximation of real class in which data sample belongs. Then unlabeled set of data then get synthetic labveled, so it is possible to use them in the same way as labeled data, for training model. This idea has some problems such as lack of consideration of any prior knowladge about the visual similarity of classes. To resolve this problem, Nassar et al. created improved model called SemCo. [7]

### 1.5.3 Mean teacher model

The Mean teacher model (MT) proposed by Tarvainen et al.[9] is also example of consistency regularization model. Consistency regularization is provided by using consistency cost 1.10. MT consists of two deep convolutional neural networks with the same layer architecture. Each network has its own trainable weights. First network is called student and its weights are denoted $\theta$, second is called teacher, with weights $\theta'$. Mean teacher is intended for classification problem. During training, model use both, labeled and unlabeled data. Labeled data have assigned desired output (label), while unlabeled data are from the same domain of classes, but they has no label.

Important part of model training is using augmentations, for example translation, rotation, addition of small noise etc. Each datapoint, usually image, is augmented in two different ways. One augmented image is then used as input for student network, the other for teacher network. Augmentation technique is commonly used as it improves results mostly in supervised learning.

Training method of student model differs from teachers training rule. Student's weights are updated using backpropagation, teacher uses Exponential moving average (EMA) of weights of student model. EMA rule can be written as equation 1.8, where $\theta'_t$ is teachers weight matrix in time $t$ and $\theta_t$ is students weight matrix in time $t$ and $\alpha$ is hyperparameter - learning rate. This strategy is called Temporal ensembling [6] and it represents composition of weights in different times, which is type of regularization.

$$\theta'_t = \alpha\theta'_{t-1} + (1-\alpha)\theta_t \tag{1.8}$$

Student's loss function consists of consistency cost and supervised cost of the model. Supervised cost $S(\theta)$ is calculated for student model as cross entropy of predictions for input $x_j$, augmented by augmentation $\eta$ and desired label $y_j$ only for labeled samples(equation 1.9). Consistency cost $J(\theta)$ is the expected distance between the prediction of the student model and the prediction of the teacher model (equation 1.10) It is computed as Mean squared error (MSE) of difference in teacher and student predictions on the same input $x_j$ and different augmentations $\eta$ and $\eta'$. This loss function does not use any labels for input data, therefore it is example of unsupervised learning and we can aply it on unlabeled data as well as on labeled data.

$$S(\theta) = \frac{1}{m}\sum_{j}^{m}[-\log\ P_f(y_j|x_j;\theta,\eta)] \tag{1.9}$$

$$J(\theta) = \frac{1}{n}\sum_{i}^{n}\|f(x_i,\theta',\eta') - f(x_i,\theta,\eta)\|^2 \tag{1.10}$$

Total loss is then computed as weighted sum of $S(\theta)$ and $J(\theta)$, as shown in equation 1.11. Parameter $w_t$ is weight of consistency loss in time $t$. It's value slowly increases in time. Based on this loss function, student model is trained using error back-propagation algorithm and gradient descent. The Mean teacher model and training process is described in figure 1.3.

$$Loss(\theta) = S(\theta) + w_t J(\theta), \tag{1.11}$$

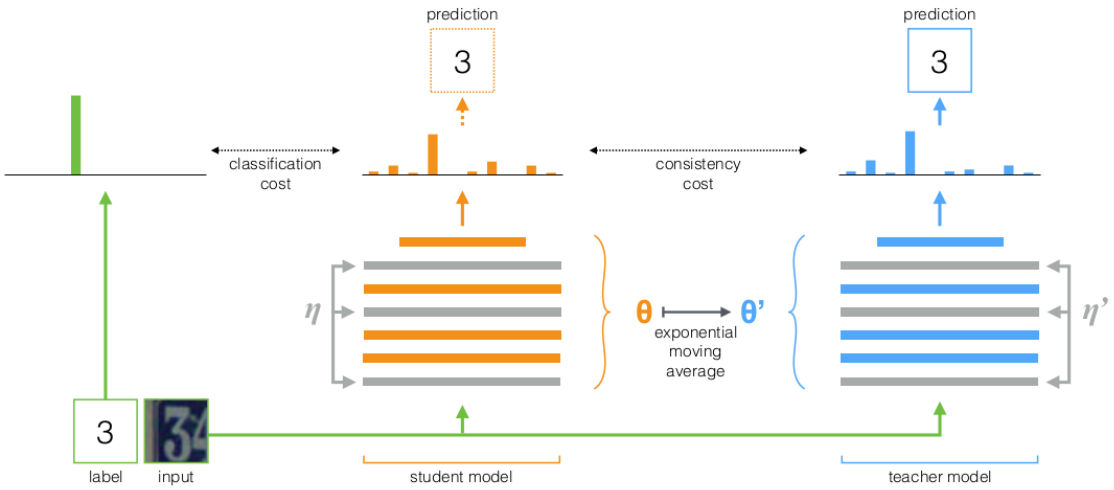



Figure 1.3: Mean teacher model

### 1.5.4 Binary mean teacher model

Binary mean teacher model (BMT) is derived from previously described Mean teacher model. Binary Mean Teacher was introduced by Tuna et al.[11] and worked well for binary clasification whether the image contains containing wearable object (backpack) or not. BMT is described in figure 1.4.

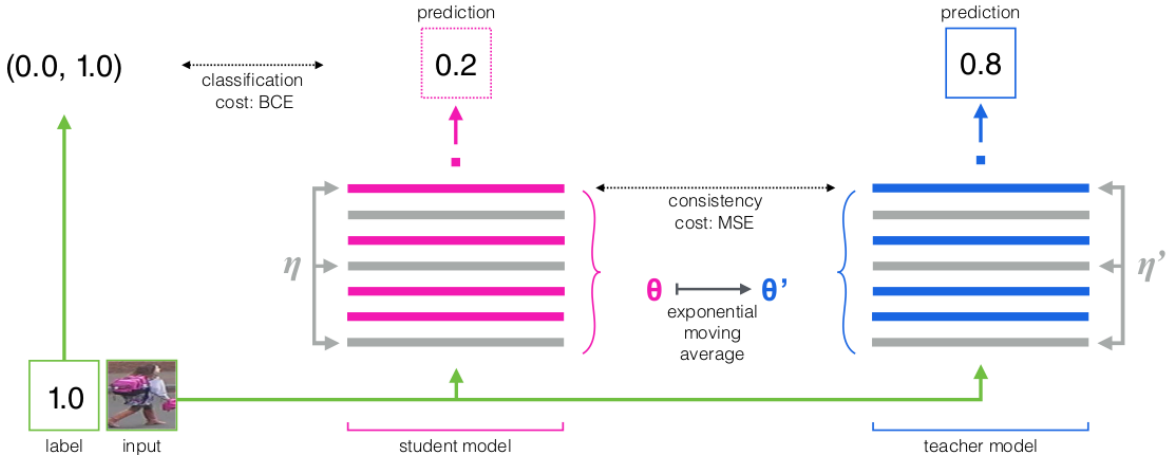



Figure 1.4: Binary mean teacher model

There are several changes in comparison to Mean teacher model. Firstly, inferences of student's and teacher's network are only scalars, so they are not suitable for consistency loss. They are replaced by feature vectors produced by last convolutional layer. Another difference is, since the output is scalar, supervised cost which in MT is computed as Cross entropy is in BMT computed as Binary cross entropy (BCE). Supervised loss is shown in equation 1.12, consistency loss is shown in equation 1.13. In consistency loss we denote weights from convolutional layers $\tau$. We can not use original notation $\theta$, because it denotes all weights (also those in fully connected layers).

We can say $\tau \subset \theta$ and $\tau' \subset \theta'$. Inference in convolutional part is denoted by function $g$. Last change is in activation function of the last fully connected layer. In MT, softmax function was used. In BMT authors proposed sigmoid activation function. Total loss (eq. 1.14) is computed similarly to MT model.

$$S(\theta) = -\sum_{i}^{n} [y_j \log \hat{y_j} + (1 - y_j) \log (1 - \hat{y_j})] \tag{1.12}$$

$$J(\tau) = \frac{1}{n} \sum_{i}^{n} \left\| g(x_i, \tau', \eta') - g(x_i, \tau, \eta) \right\|^2 \tag{1.13}$$

$$Loss(\theta) = S(\theta) + w_t J(\tau), \tag{1.14}$$

# Chapter 2

# Our research

We aimed to create new semi-supervised model which include concept of self-organization into uninformed part of semisupervised model. We devided our research into several experiments which investigate and fine tune components of this new model.

## 2.1 Self-organization in supervised models

We decided to use Self-organizning map (SOM) described in section 1.4.1 as unsupervised component of loss function. Unlike other unsupervised techniques, SOM can provide nonlinear transformation of the input and is neural network based, so its complexity is scalable.

We studied related work in literaturen and found article by Forest et al. [2] from 2019, where they introduced model called DESOM. Authors say it is first ever usage of SOM in autoencoder model.

## 2.2 Our model

As base of our model, we chose semi-supervised model Mean teacher described in section 1.5.3. We suggested to improve unsupervised loss of the model by using distances of feature vectors in Self-organizing map. Our model is described in details in figure 2.1.

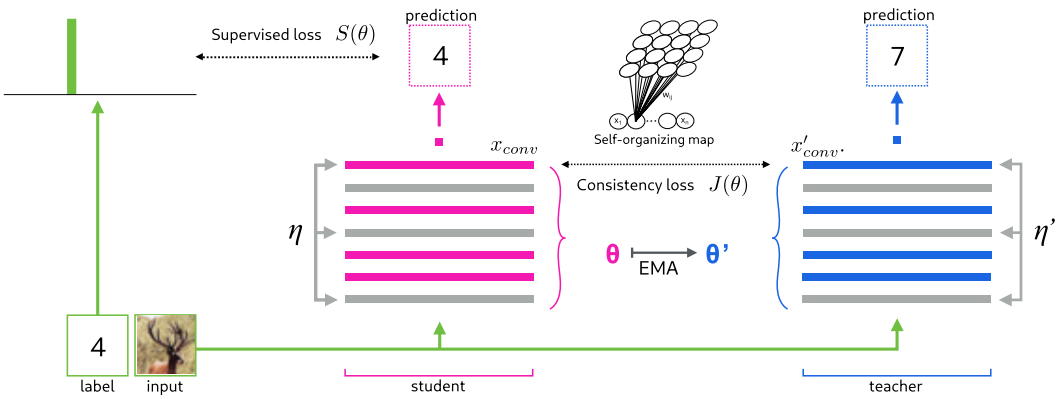



Figure 2.1: Description of our model

# Chapter 3

# Classification of animate and inanimate objects

In this chapter, we focused on existing model Binary mean teacher, that is derived from semisupervised model Mean Teacher, and is used for binary classification. Our task was to classify images of animate and inanimate objects. We introduced our custom made dataset, that contains images from standard dataset CIFAR10 which we relabeled into only two classes. The main goal of this experiment was to test performance of the Binary mean teacher model. We have done several experiments with this new dataset and compared semisupervised model performance with supervised baseline.

## 3.1 Task description

Motivation for our task was to find out, whether model is able to discover features that are specific for appearence of living creatures and others that define lifeless objects and differenciate these two classes of objects. Simply, our task was classification of images into two classes - animate objects and inanimate objects.

## 3.2 Dataset

As far as we know, there is no dataset for binary classification of images, with suitable labels. We chose to transform and use standard CIFAR10 dataset [5] which consists of 60000 colour images with resolution $32 \times 32$ pixels in 10 classes (bird, cat, deer, dog, frog, horse, airplane, automobile, ship, truck), with 6000 images per class.

In this experiment, we transformed dataset from original ten classes into two - animate and inanimate. Animate class was formed of images from original classes bird, cat, deer, dog, frog and horse. Inanimate class consisted of images that was in originally in classes airplane, automobile, ship and truck.

## 3.3 Models

We wanted to compare semisupervised model performance with supervised model, to find out how unsupervised part of model support learning. As far as the task was designed for binary classification, we decided to use semisupervised model Binary mean teacher described in section 1.5.4. As a baseline we used supervied feedforward neural network with the same layer architecture as semisupervised model. Our dataset was completely new, so we needed to compute accuracy for both models with various of different hyperparameters.

### 3.3.1 Network description

Architecture of hidden layers was taken from Muhammad Sarmad's github repository [1], from his implementation of Mean teacher model. Neural net had 25 layers described in table 3.1 with ReLU as activation functions for all layers except last two. In last two layers, he used sigmoid activation function.

| hidden layer with parameters |
|---|
| 3 × BatchNorm2d(128) |
| Conv2d(3, 128, 3,padding=1) |
| 2 × Conv2d(128, 128, 3,padding=1) |
| MaxPool2d(2, 2) |
| Dropout(0.5) |
| 3 × BatchNorm2d(256) |
| Conv2d(128, 256, 3, padding=1) |
| 2 × Conv2d(256, 256, 3, padding=1) |
| MaxPool2d(2, 2) |
| Dropout(0.5) |
| BatchNorm2d(512) |
| BatchNorm2d(256) |
| BatchNorm2d(128) |
| Conv2d(256, 512, 3) |
| Conv2d(512, 256, 1) |
| Conv2d(256, 128, 1) |
| AvgPool2d(6,6) |
| nn.Linear(128, 1) |
| BatchNorm1d(1) |

Table 3.1: Table of layers in network

[1] `https://github.com/iSarmad/MeanTeacher-SNTG-HybridNet`

### 3.3.2 Implementation

We developed this experiment in this GitHub repository [2]. Repository contains code for dataset transformation and the experiments. Core of the model implementation was taken from Sarmad's Mean teaches. We used his hidden architecture, but changed the last layer dimension to one, which is necessary for binary classification. We also changed the unsupervised loss calculation to use representation from last convolution layer instead of network output. These changes were necessary to transform Mean teacher into Binary mean teacher model. During developement, we validated, that it is necessary to use feature vectors from last convolutional layer when computing MSE. It was esential for network convergence.

### 3.3.3 Baseline

Our baseline was deep network trained on same portion of labeled data. We will run fully supervised network and compouted accuracy on our custom made dataset. As we mentioned, this network had the same architecture as student or teacher network, so difference in accuracy should show the effect of using additional information from unsupervised loss in learning of semisupervised model.

We trained network on 4000 labeled data. Training consisted of 30 epochs, learning rate was set to 0.5, optimizer was SGD. Best accuracy net achieved was **93%**, as evaluation on 10000 labeled samples.

## 3.4 Training

We trained all networks for 30 epochs and used SGD optimizer. Ratio of labeled and unlabeled samples in training set was constant and was 4000 : 46000. Network was evaluated on 10000 labeled samples. As image augmentations, we used flipping and rotation.

## 3.5 Experiments

We tried 3 different subtasks. First one focused on looking for learning rate that gives good results for semisupervised model, second discovered how different type of augmentation influence model accuracy and last one was about trying different amount of labeled data in training set and looking whether unlabeled data which semisuperised model use helped its performance in comparison to supervised model with only labeled data.

[2] https://github.com/Sabka/DT-mean-teacher

### 3.5.1 Learning rate

We tried several values of learning rate hyperparameter and trained only semisupervised model with other parameters set as described in section 3.4. Our goal was to observe, which values of learning rate were good for the task. We state table 3.2 which contains best learning rate with accuracies.

| Learning rate | Student validation acc |
|---|---|
| 0.05 | 87% |
| 0.1 | 89% |
| 0.5 | 91% |

Table 3.2: Learning rate accuracies

### 3.5.2 Different augmentation

In next experiment, we tried to use color augmentation for input data. We chose Pytorch ColorJitter, which change colors according to parameters brightness, contrast, saturation and hue. The result of transformation can look like in image 3.1. Our best model with data augmented by ColorJitter, on 4000 labeled datapoints and 46000 unlabed datapoints and learning rate 0.5 achieved accuracy **91%**, which is not better than with same hyperparameters and only flip and rotate augmentation.

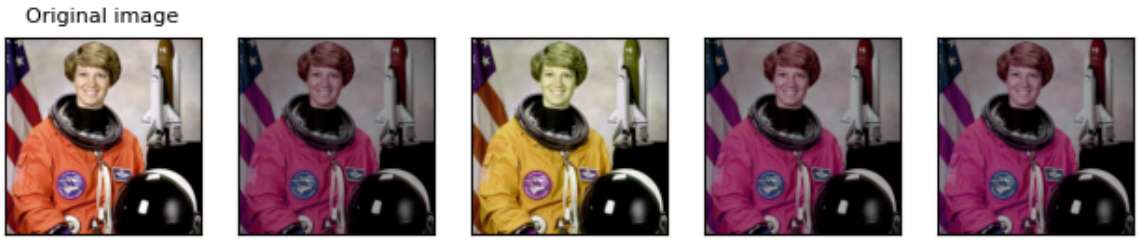



Figure 3.1: Color Jitter augmentation example [10]

### 3.5.3 Size of portion of labeled data

In this final experiment, we tended was to show, how unlabeled examples helped model to learn better when there were smaller portion of labeled examples, which is the main goal of semisupervised learning. We introduced new hyperparameter - portion of labeled data and monitor accuracy of net and computed new baselines. For each portions of labeled data, we trained supervised model. The results are shown in table 3.3. Highlighted results show cases where Binary Mean Teacher model achived better accuracy than baseline model.

| portion | LR | Student validation acc | Baseline (with same portion) |
|---|---|---|---|
| 100 | 0.1 | 78% | 60% |
| 100 | 0.5 | 80% | 60% |
| 500 | 0.5 | 88% | 85% |
| 500 | 0.1 | 88% | 85% |
| 1000 | 0.1 | 89% | 83% |
| 1000 | 0.5 | 90% | 83% |
| 4000 | 0.1 | 89% | 93% |
| 4000 | 0.5 | 91% | 93% |

Table 3.3: Comparison of models

## 3.6 Discussion of results

First experiment showed, that 4000 labeled samples are enough for supervised model to train well. Semisupervised model didn't achieved this high accuracy even when trying different values of learning rate hyperparameter.

In second experiment, we tried to use different augmentation - change of colors in the image. In this case, model did not work any better than using only flip and rotation. Accuracy stayed the same.

Last experiment focussed on how different size of portion of labeled data influenced model's results. This showed that semisupervised model worked much better for smaller portions of data, when supervised model wasn't able to acomplish that high accuracy. This experiment showed, that huge amount of unlabeled data used for trainig of semisupervised model helped its performance.

There are several things about Binary Mean Teacher model, that could be studied in future. We recommend future research to focus on more complex architectures, which was not able for us, because of our hardware limitations. Study of different hyperparameters, such as EMA decay or consistency cost weight in weighted sum of costs could also help to reach better performance.

# Chapter 4

# Developement of SOM-loss

In this chapter, we mainly focus on developing method which involve structure represented by Self-organising map (SOM) into learning of Multi layer perceptron (MLP). We designed experiment and compared the improvement in performence of MLP when structural information of data from SOM representation are involved. This experiment was designed on table dataset containing atributes acquired from samples of several wine species. We chose the table dataset rather than image dataset to accelerate our research.

## 4.1 Consistency in semi-supervised learning

Unsupervised loss in semisupervised learning is mostly based on prediction consistency. In case of Mean teacher model, this consistency is between predictions of teacher and student model. Other models, for example Siamese neural network, holds consistency between predictions of one model on sample augmented in two different ways. In our research, we decided to use self-organizing map as model of consistency of data points from same class.

## 4.2 SOM-loss idea

Self-organizing maps are able to naturally cluster data and project them into low dimensional representation. This is determined by neurons also called prototypes. Each prototype has its weighs, which are adjusted during training to represent some of the input data samples. Weights of prototype has the same dimension $n$ as input data, therefore we can consider prototype weights as prototype position in $n$-dimensional space.

We developed another idea, to use these prototypes during learning of supervised model as supportive element. So we used distance of SOM prototypes of two samples

as approximation of difference of samples. By that, we helped network to learn from information about difference of two samples.

## 4.3 SOM-loss propositions

We were considering several forms of SOM loss. In each we use distance of some two 13-dimensional vectors, which is computed based on equation 4.1.

$$dist(x^1, x^2) = \sqrt{\sum_{i=1}^{n} (x_i^1 - x_i^2)^2} \tag{4.1}$$

### 4.3.1 Prototype distance SOM-loss

First SOM-loss we considered was based on pair of datapoints $x^1$ and $x^2$. They could be from same or different class. We found their prototypes $p^1$ and $p^2$ as predictions of datapoints. Then we computed their distance $dist(p^1, p^2)$ using equation 4.1. If datapoints were from the same class, we wanted the distance close to zero. If second datapoint was from different class, distance should have been high. Our proposed SOM-loss is then equal to distance in 4.1.

Drawback of this deffinition was, that we would need to somehow discriminate which distances are small enough to represent the same class and which are big enough to represent datapoints from different class.

### 4.3.2 Prototype distance and datapoint distance SOM-loss

Another problem with loss described in previous section 4.3.1 was, that it only uses distances of prototypes, but did not reflect distance from original datapoint to its prototype. We decided to add this sample-prototype distance for both samples in the loss as shown in equation 4.2.

$$dist2(x^1, x^2) = dist(p^1, p^2) + dist(x^1, p^1) + dist(x^2, p^2) \tag{4.2}$$

### 4.3.3 Congruent and incongruent distance SOM-loss

First problem, mentioned in 4.3.1 still persisted. We decided to resolve it using normalization, for which we needed to compute two distances. The distance of two samples from the same class $dist_c$, and the distance of two samples from different classes $dist_i$. We use first sample $x^1$, second sample $x^2$ from congruent class to $x^1$ and third $x^3$ from incongruent class. Distances are computed based on equations 4.3, 4.4. Then, we compute normalised loss $SOMloss$ as described in equation 4.5.

$$dist_c(x^1, x^2, x^3) = dist2(x^1, x^2) \tag{4.3}$$

$$dist_i(x^1, x^2, x^3) = dist2(x^1, x^3) \tag{4.4}$$

$$SOMloss = \frac{dist_i - dist_c}{dist_i + dist_c} \tag{4.5}$$

In the following table 4.1, different walues of distances were investigated. As a result, we can see that proposed $SOMloss$ resolve all cases when distances of congruent and incongruent samples are high or low. We can explain these cases. First and last cases are, when distances are really close and the loss is 0, so it has no information value, because distances give no infomation about sample is closer to original sample, whether congruent or incongruent. In case 2, result is positive and it is a good case when incongruent sample is far from original and congruent is close to original. In third case, loss is negative and this is the case, when distances are opposite as they should be. We decided to rescale this loss one more time, based on 4.6. After this operation, negative values are not possible, loss is from interval (0, 1) and it is lower when distance of congruent samples is small and of incongruent is high. Loss is high if mistake in distance is high.

| $dist_i$ | $dist_c$ | $SOMloss$ | $SOMlossRescaled$ |
|---|---|---|---|
| 1000 (high) | 1000 (high) | 0 (middle) | 0.5 (middle) |
| 1000 (high) | 1 (low) | 0.998 (high) | 0.0009 (low) |
| 1 (low) | 1000 (high) | -0.998 (low) | 0.99 (high) |
| 1 (low) | 1 (low) | 0 (middle) | 0.5 (middle) |

Table 4.1: SOMloss with different distances

$$SOMlossRescaled = 0.5 - 0.5 \cdot SOMloss \tag{4.6}$$

## 4.4 Experiment

We designed the experiment to investigate how proposed SOM loss influence the performace of supervised MPL. We used pretrained SOM for determining datapoint distances and add this unsupervised loss to supervised loss of model.

### 4.4.1 Wine dataset

We chose table dataset called Wine dataset. These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different

cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. Hence the data is 13-dimensional with three classes defined by the three cultivars. The training sets were large with 59, 71 and 48 training samples per class. [1]

We split dataset into training and testing part such that test dataset contained approximately 25% of dataset, and each class had the same ratio of testing dataset. That meant 15 samples from each class. Other 75% of data created training dataset. In this training dataset, data were reorganized in the way that each sample was paired with another sample from the same class and another sample form different class. Final training datasret was then composed of 6650 triplets. We will later call the first sample of triplet as original sample, second sample as congruent sample and third sample as incongruent sample.

### 4.4.2 Supervised model baseline

As our supervised model we chose basic Multi layer perceptron (MLP). We started with architecture with 4 layers with few tenths of neurons in each layer and sigmoid activation functions. We found out, that such model was too complex and after selection of appropriate learning rate parameter, it was able to train on dataset and absolutely discriminate classes. Accuracy was 100%. Since we wanted the baseline that we can compete with, we decrease the complexity of model. At last we chose architecture with only one hidden layer, with 15 neurons and sigmoid activation function. Architecture is shown in table 4.2. This architecture was not able to achieve 100% testing accuracy, so it was suitable for our experiment.

| MLP layer with parameters |
|---|
| nn.Linear(13, 15) |
| nn.Sigmoid() |
| nn.Linear(15, 3) |
| nn.Softmax(dim=1) |

Table 4.2: Table of layers of MLP

### 4.4.3 Semi-supervised model

We were considering several ways how to include SOM information into MLP. We tried to cotrain SOM with MLP, but since SOM inputs did not change in time, it was not necessary and what is more important, it led to overfit in SOM. Then we decided to pretrain SOM on data samples and during training of MLP, use information from SOM predictions.

We proposed combined loss *SomSupLoss* which combine supervised loss of MLP and *SOMlossRescaled* multiplied by hyperparameter $\kappa$, as shown in equation 4.7.

$$SomSupLoss = MSE(x, \hat{y}) + \kappa \cdot SOMlossRescaled \tag{4.7}$$

### 4.4.4 Setup

The setup of the experiment consisted of MLP architecture and parameters, SOM parameters and type of SOM loss used. MLP architecture - dimensions of layers and activation functions - are described in section 4.4.2, learning rate was set to 0.0002.

SOM with topology of $5 \times 5$ neurons was trained and final representation is shown in figure 4.1. SOM metrics at the end of training were quantization error = 616.809, winner discrimination = 1.0, entropy = 4.523.

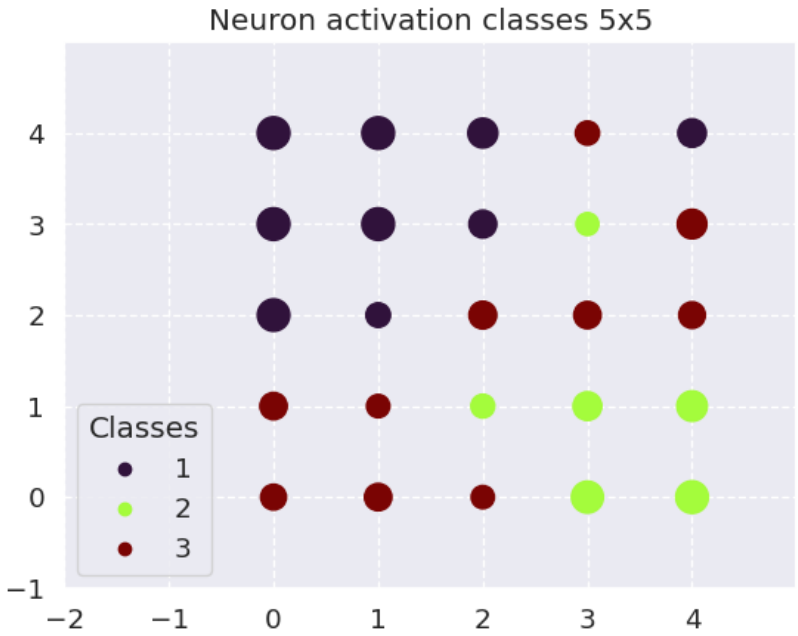



Figure 4.1: SOM used for experiment

For semi-supervised model, we used SOM loss introduced in previous section and the same architecture and parameters of MLP as for supervised baseline.

### 4.4.5 Results

Each model was trained for 30 epochs. We trained both supervised and semi-supervised model 20 times, so we can compare their perforamnce. All accuracies (in %) sorted from best are shown in table 4.3. Confusion matrices of best performing supervised and semi-supervised model are shown in figure 4.4. We computed mean of these accuracies.

For MLP, mean accuracy is 74.89% and for semi-supervised model it is 82.89%. We can see, that SOM supported supervised model to produce better predictions.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| MLP | 2 × 93.33 | 7 × 91.11 | 88.89 | 86.67 | 82.22 | 3 × 64.44 | 62.22 | 60.0 | 3 × 33.33 |
| com | 2 × 95.56 | 4 × 93.33 | 7 × 91.11 | 80.0 | 2 × 64.44 | 3 × 62.22 | 60.0 | - | - |

Table 4.3: Semi-supervised and supervised test accuracier

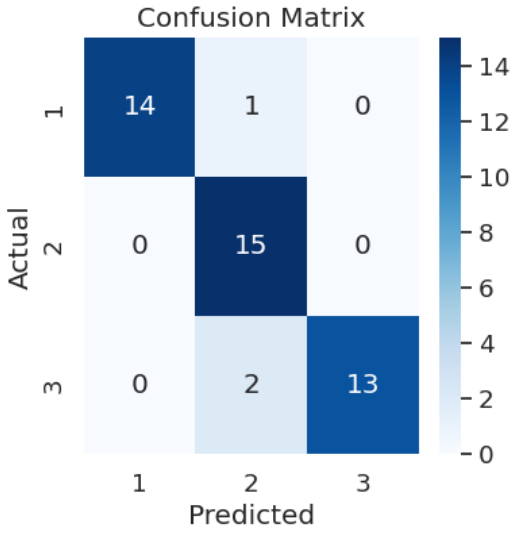



Figure 4.2: Supervised model

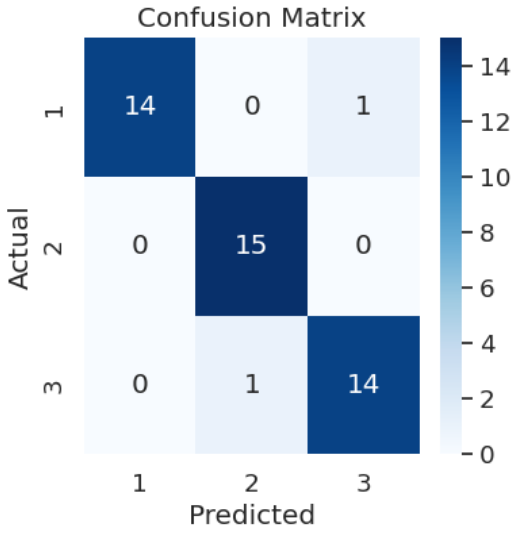



Figure 4.3: Semi-supervised model

Figure 4.4: Confusion matrices of best performing models

### 4.4.6 Discussion

As we can see, information from SOM prototype distances can significantly improve model performance. We can also see, that MLP model has in 3 cases accuracy 33.33% which in classification into 3 classes means model consistently choose the same class, so model did not converge. For semi-supervised model, this never happened during our experiment.

# Chapter 5

# SOM and evolving feature vectors

# Chapter 6

# Winner selection methods

# Conclusion

# Bibliography


[1] Stefan Aeberhard and M. Forina. Wine. UCI Machine Learning Repository, 1991. DOI: https://doi.org/10.24432/C5PC7J.

[2] Florent Forest, Mustapha Lebbah, Hanene Azzag, and Jérôme Lacaille. Deep embedded som: joint representation learning and self-organization. *reconstruction*, 500:500, 2019.

[3] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[4] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[5] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, Toronto, 2009.

[6] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016.

[7] Islam Nassar, Samitha Herath, Ehsan Abbasnejad, Wray Buntine, and Gholamreza Haffari. All labels are not created equal: Enhancing semi-supervision via label grouping and co-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7241–7250, June 2021.

[8] RNDr. Kristína Rebrová. *Grounding the meaning in sensorimotor cognition: a connectionist approach*. Phd thesis, Comenius University in Bratislava, Bratislava, 2013.

[9] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[10] Pytorch color jitter documentation. https://pytorch.org/vision/main/auto_examples/transforms/plot_transforms_illustrations.html#sphx-glr-auto-examples-transforms-plot-transforms-illustrations-py. Accessed: 2023-12-07.

[11] Matúš Tuna, Kristína Malinovská, Igor Farkaš, Svatopluk Kraus, and Pavel Krsek. Semi-supervised learning in camera surveillance image classification. In *2021 IEEE 17th International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 155–162, 2021.