KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in Mathematics

Second cycle, 30 credits

Enhancing Neural Network Accuracy on Long-Tailed Datasets through Curriculum Learning and Data Sorting

DANIEL BARREIRA

Stockholm, Sweden, 2023

Enhancing Neural Network Accuracy on Long-Tailed Datasets through Curriculum Learning and Data Sorting

DANIEL BARREIRA

Master's Programme, Applied and Computational Mathematics, 120 credits
Date: October 9, 2023

Supervisors: Mattias Sandberg, Therese Persson
Examiner: Mattias Sandberg
School of Engineering Sciences
Host company: Sellpy
Swedish title: Förbättring av noggrannheten för neurala nätverk på Long-Tailed-dataset genom Curriculum Learning och Data Sortering

Abstract

In this paper, a study is conducted to investigate the use of Curriculum Learning as an approach to address accuracy issues in a neural network caused by training on a Long-Tailed dataset. The thesis problem is presented by a Swedish e-commerce company. Currently, they are using a neural network that has been modified by them using a CORAL framework. This adaptation means that instead of having a classic binary regression model, it is an ordinal regression model. The data used for training the model has a Long-Tail distribution, which leads to inaccuracies when predicting a price distribution for items that are part of the tail-end of the data. The current method applied to remedy this problem is Re-balancing in the form of down-sampling and up-sampling. A linear training scheme is introduced, increasing in increments of 10% while applying Curriculum Learning. As a method for sorting the data in an appropriate way, inspiration is drawn from Knowledge Distillation, specifically the Teacher-Student model approach. The teacher models are trained as specialists on three different subsets, and furthermore, those models are used as a basis for sorting the data before training the student model. During the training of the student model, the Curriculum Learning approach is used. The results show that for Imbalance Ratio, Kullback-Liebler divergence, Class Balance, and the Gini Coefficient, the data is clearly less Long-Tailed after dividing the data into subsets. With the correct settings before training, there is also an improvement in the training speed of the student model compared to the base model. The accuracy for both the student model and the base model is comparable. There is a slight advantage for the base model when predicting items in the head part of the data, while the student model shows improvements for items that are between the head and the tail.

Keywords

Machine Learning, Neural Network, CORAL-framework, Long-Tailed Data, Imbalance Metrics, Teacher-Student models, Curriculum Learning, Training Scheme

Sammanfattning

I denna uppsats genomförs en studie för att undersöka användningen av Curriculum Learning som en metod för att hantera noggrannhetsproblem i ett neuralt nätverk som är en konsekvens av träning på data som har en Long-Tail fördelning. Problemställnningen som behandlas i uppsatsen är tillhandagiven av ett svensk e-handelsföretag. För närvarande använder de ett neuralt nätverk som har modifierats med hjälp av ett CORAL-ramverk. Denna anpassning innebär att det istället för att ha en klassisk binär regressionsmodell har en ordinal regressionsmodell. Datan som används för att träna modellen har en Long-Tail fördelning, vilket leder till problem vid prediktering av prisfördelning för diverse föremål som tillhör datans svans. Den nuvarande metod som används för att åtgärda detta problem är en Re-balancing i form av down-sampling och up-sampling. Ett linjärt träningschema introduceras, som ökar i steg om 10% medan Curriculum Learning tillämpas. Metoden för att sortera datan på ett lämpligt sätt inspires av Knowledge-Distillation, mer specifikt lärar-elevmodell delen. Lärarmodellerna tränas som specialister på tre olika delmängder, och därefter används dessa modeller som grund för att sortera datan innan tränandet av elevmodellen. Under träningen av elevmodellen tillämpas Curriculum Learning. Resultaten visar att för Imbalance Ratio, Kullback-Libler-divergens, Class Balance och Gini-koefficienten är datat tydligt mindre Long-Tailed efter att datat delats in i delmängder. Med rätt inställningar innan tränandet finns även en förbättring i träningshastighet för elevmodellen jämfört med basmodellen. Noggrannheten för både elevmodellen och basmodellen är jämförbar. Det finns en liten fördel för basmodellen vid prediktering av föremål i huvuddelen av datan, medan elevmodellen visar förbättringar för föremål som ligger mellan huvuddelen och svansen.

Nyckelord

Maskininlärning, Neuralt Nätverk, CORAL-ramverk, Long-Tailed Data, Imbalance Metrics, Teacher-Student modeler, Curriculum Learning, Träningsscheman

Acknowledgments

A special thanks goes out to my two supervisors. To Therese for endlessly helping me and always being there when I needed it, and to Mattias for being understanding and calm in times of distress. Thank you both very much.

As this also marks the end of my studies at KTH, I would like to take a moment to thank the people I have met on the way. Especially Anmol, Jonathan and Maja without whom I would not have enjoyed the last 5 years nearly as much as I have.

Stockholm, October 2023
Daniel Barreira

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Thesis Background

In a world with growing amounts of information, and companies being able to track and utilize it in different ways, new challenges arise. One of those challenges is that things around us is not generally uniformly distributed, but everything is more or less skewed in one way or another. This together with the fact that it is important to not manipulate data to keep it as representative to reality as possible, leads to problems in all sectors that is somewhat based on data. Furthermore, with the current rise of neural networks in areas such as Large Language Models (for example ChatGPT), and other advances in the field of artificial intelligence of all sorts, it is increasingly important to be able to handle a non-uniform reality. This has lead to that in the world of research it is an increasingly hot subject how to keep down computational time as well as making sure to not neglect the part of the data that is underrepresented.

1.2 Host Company

Sellpy is a company founded in 2014 in Stockholm, Sweden. Being able to bring society towards a more circular life style is one of their main goals, which is reflected in their slogan 'Everyone should be able to live circular". It is an e-commerce company specialized in second hand items. The main idea is that if something can fit in a 75 liter bag, it is processed by the company. If the product is deemed as a suitable product it goes on to their website, if it's not deemed as a suitable object to be sold, it is donated to different organizations such as Myrorna, Stockholm stadsmission and Unicef. [1]

1.3 Problem

The application at hand for this paper is the use of an alternative method for handling the tail-data for a pre-existing model. The existing model aims to produce a price distribution for a second-hand e-commerce platform. This distribution as created with the help of a neural network that is trained on historical sales data. The sales data includes fields such as original sales price and brands. More information about the data can be found in Chapter 3.2. The output from the existing model can be seen in Figure 1.1.

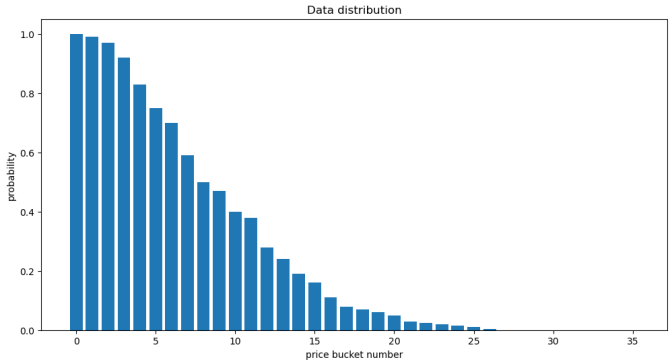


Figure 1.1: Example of an items output from the pricer model

Here each bar represents a price interval also known as a pricebucket. These differ in size, the first ones has the span 25 SEK, and later on they have the span 50 SEK. The x-axis is price in SEK. and the y-axis is the predicted probability for the item to be sold in that given price-bucket. It is based on a CORAL-framework [2] and it is trained on data accessible from Sellpy's own database. The data distribution is skewed towards the lower end of the price range, making it more difficult to achieve a high accuracy for price predictions in the premium segment. This thesis investigates methods for improving the accuracy of price predictions in the high-price tail of the data. There are multiple methods for handling this sort of behaviour and the current method is working to a desired level, but there is some improvement to be done at the tail.

1.3.1 Initial Problem Formulation

Curriculum learning is explored as a way to remedy long-tail problems in a second-hand e-commerce pricer model which currently is using Re-balancing. The two methods are then compared using different metrics.

1.3.2 Scientific and engineering issues

One of the big scientific issues with Long-tail data is that it naturally becomes very skewed towards certain directions. There is a responsibility from the side of academia to further advance the field of research toward methods that are more including and ethically sustainable. With ethical sustainability the connection is not clear cut with the application of this paper, but seeing that Long-tail data is a problem in many different fields the connection might be more clear. For example, imagine a model training on facial expressions and recreating these in some manner, here it immediately becomes apparent that underrepresented groups might fall between the cracks and not be fairly represented by the model and thus making it ethically unsustainable. The same responsibility does not exist with respect to the data and utilization of the data in this thesis, but the methods used could potentially be the same in both scenarios.

1.4 Purpose

One of the main purposes of this thesis is to further advance current research. Also, applying current state of the art methods from academia for real word use. Being able to get a fair representation of the entire spectrum of objects in the data set, is essential both out of an ethical standpoint, not neglecting minorities, but also practically. From a practical standpoint the company will be able to set better prices for a wider range of goods resulting in a better product and happier costumers in general.

1.5 Goals

Achieving a result for how Curriculum Learning work for pricing models in comparison to the current method used. Further more observing if there might be computational advantages for different methods. Lastly, compare the method at different price intervals, to not only compare the methods on the entire training data, and thus achieving a metric for how it performs for underrepresented data.

1.6 Research Methodology

The literature study was mainly done using google scholar and following which new papers were introduced to the website arXiv. An emphazis was put on reading material released by the research teams from FAANG and open-ai. At the moment the research field of AI seems to be developing at such a rapid speed that it can be hard to be sure that the methods you are implementing are state of the art and not proved inadequate. This paper is mainly based on papers and theory that is a bit older (in the world of AI) but in the section 2.3 some newer papers are presented and how they can be related to the thesis project.

All the data used for comparison and development of the models is directly taken from the host company and is real world data based on items they have received over the last 6 years.

The host company contributed with a computer with a Linux operating system which is the main computer used for everything that is related to model development. The models used by the host company was copied locally from private git-repositories and modified into scripts that could be used for the project. For algorithm development and coding the program Visual Studio Code was used. The models were built using Tensorflow and Keras, and the data was mainly handled using Pandas in combination with NumPy in addition to some other popular python-libraries for machine learning. The smaller models was trained locally, but for the more comprehensive models with higher computational requirements the host company provided access to Amazon Web Service (AWS) as a way to schedule model-training and being able to run parallel training sessions.

1.7 Delimitations

Because of time constraints the models are not trained on the complete data set, which the original model is trained on. Instead, both the baseline model and the alternative model developed in the thesis are trained on a subset of the data to get a fair comparison between them. This means that the results presented for the baseline model in this thesis are not the same as they would be for a full-scale version of the model. Furthermore focus was put on making the model work as intended, and not on computational optimization leaving improvement to be done in that area for future work.

1.8 Structure of the thesis

Chapter 2 is dedicated to the technical background which is mainly focusing on Long-tail problems, possible methods to handle it, and surrounding research on the subject. In the third chapter a focus is put on the model given for the thesis, the data set, and curriculum learning as a tool for handling the Long-tail distribution. This is followed by Chapter 4, where the implementation is explained. The focus In Chapter 5, the results are presented along with a comparison with the current method. Finally in Chapter 6 and 7, a discussion is done about the results and a conclusion is drawn together with prospects for future work to be done.

Chapter 2

Technical Background

2.1 Long-Tail data

It is seldom that it is the noise in the data that creates an inaccurate model. Often in modern datasets it is rather the lack of data when it comes to the more atypical and rare objects that is the root of these inaccurate models. When the data distribution is distributed in such a way that you have loads of data for some of the objects, and a few samples for the rest you get what in practice is called a long-tail distribution. In recent datasets it has been seen that they follow a general power law distribution such as a Zipf Distribution. [3]

2.2 Pricer model

Following is an explanation model which is used as a baseline for this thesis. The model is built using the python libraries Tensorflow and Keras. In order to achieve a desired output from the model a CORAL-framework has been adopted and modified [2]. The desired output in this case is being able to set a percentage chance for an item being sold at a certain price given a set of predetermined set of price buckets. One of the characteristics of this method is that the percentage per bucket is decreasing in order of size (where size is the height of the staple, see Figure 1.1). That is that the second bucket will never have a higher probability than the first bucket, and so on. The model is not pretrained on an existing data set, but fully built upon in-house collected data.

2.2.1 Ordinal Regression

The CORAL-framework that is adapted is based on what is known as Ordinal Classification, also known as Ordinal Regression. This is the method for predicting labels on an ordinal scale. A classifier, also known as a ranking rule, h maps the objects, $\bar{x_i} \in \chi$ into an ordered set $h : \chi \rightarrow y$, where $y = \{r_1 < .. < r_K\}$. An attribute to ordinal classification is that the labels is enough information to be able to order the objects. With that in mind, the following is proposed as a CORAL framework [2].

2.2.2 CORAL-framework

The training data set is denoted $D = \{\bar{x}_i, y_i\}_i^N$, where N is the number of training examples and $\bar{x_i} \in \chi$ is the i:th training example and y_i is the rank, where $y_i \in y$. The purpose of the ordinal regression is to minimize a loss function $L(h)$ given a certain ranking rule. The loss function used for this specific case is defined in Equation(2.4).

Given the aforementioned data set D, the rank index q_i is given by

$$q_i = 1 + \sum_{k=1}^{K-1} f_k(\bar{x_i}), \tag{2.1}$$

where $f_k(\bar{x_i}) \in \{0, 1\}$ is a prediction of the binary classifier in the output layer. Also K is the number of rankings. A requirement is that f_k needs to reflect the existing ordinal information which means that $f_1(\bar{x_i}) \geq f_2(\bar{x_i}) \geq ... \geq f_{K-1}(\bar{x_i})$, this in turn will guarantee consistent predictions. Further more to impose and guarantee binary class consistency, the binary tasks share the same weight parameters but have independent bias units.

2.2.2.1 Loss Function

$\boldsymbol{W}$ is the weight parameters excluding the bias units of the final layer. In the penultimate layer there is an output, $g(\bar{x_i}, \bar{W})$, which share a single weight with the nodes in the final layer. To make sure that this in turn represents the binary classifiers in the final layer, bias units are added to $g(\bar{x_i}, \bar{W})$ such that $\{g(\bar{x_i}, \bar{W}) + b_k\}_{k)1}^{K-1}$ becomes the input to the final layer.
The logistic sigmoid function is defined as

$$\sigma(z) = \frac{1}{(1 + exp(-z))}, \tag{2.2}$$

and the predicted empirical probability for task k is given by:

$$\hat{P}(y_i^k = 1) = \sigma(g(\bar{x_i}, \bar{W}) + b_k). \tag{2.3}$$

The loss function that is minimized in the model training phase is defined as

$$L(\bar{W}, \bar{b}) = -\sum_{i=1}^{N}\sum_{k=1}^{K-1} \lambda^{(k)}[log(\sigma(g(\bar{x_i}, \bar{W})+b_k))y_i^{(k)}+log(1-\sigma(g(\bar{x_i}, \bar{W})+b_k)(1-y_i^{(k)})], \tag{2.4}$$

this is the weighted cross-entropy. The rank prediction is then calculated via

$$f_k(\bar{x_i}) = 1\{(\hat{P}(y_i^k = 1) > 0.5\}. \tag{2.5}$$

λ^k is the weight corresponding to the k-th classifier. It is also referred to as a importance parameter for a task k. For simplicity both the implementation for the pricer model, as well as the authors of the papers has defined it in such a way that

$$\forall k : \lambda^k = 1. \tag{2.6}$$

There is the possibility to introduce a non uniform weighting scheme if that suits the tasks at hand better. The theoretical proof of classifier consistency is presented in [2].

2.2.3 Model evaluation

Both the mean absolute error(MAE) and root mean squared error (RMSE) is defined for evaluation the model.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - h(\boldsymbol{x}_i)| \tag{2.7}$$

is the definition of MAE, where y_i in this case is the ground truth rank for the sample i, $h(\boldsymbol{x_i i}$ is the predicted rank. RMSE is defined as

$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - h(\boldsymbol{x}_i))^2}. \tag{2.8}$$

2.3 Related Work

Longtail problems in Machine Learning has been around for a long time, and different sorts of methods are proposed as a solution. This paper is focusing on Curriculum Learning, but other more classical approaches could also be used. The model that is used as reference throughout the paper is currently using re-sampling as the method. [4]

2.3.1 Re-sampling

This method seeks to take a longtail problem, and make it more uniform by either down-sampling the head, up-sampling the tail or by doing both at the same time. This is done in a random manner, more known as random over-sampling (ROS) and random under-sampling (RUS). One problem that is associated with these approaches is that either the tail gets overfit by using ROS or the head gets a worse performance. To circumvent this problem more recent studies are looking at special kinds of re-sampling methods. Some of those are Classifier Re-training and Nearest Class Mean Classifier [5]. Classifier Re-training is a simple method that involves retraining the classifier through class-balanced sampling. In other words, while maintaining the representations constant randomly reinitialize and fine-tune the weights W and b (bias units) of the classifier for a limited number of epochs, ensuring a balance among the different classes. Nearest Class Mean Classifier is a bit more complex. This approach involves calculating the average feature representation for every class within the training set, followed by conducting a nearest neighbor search using either cosine similarity or the Euclidean distance, computed on L2-normalized mean features. Despite its straightforward nature, this method serves as a strong baseline. The use of cosine similarity effectively avoids the weight imbalance issue due to its built-in normalization properties.

2.3.2 Cost-sensitive Learning

By adjusting different loss values depending on which class it is during training, you can in a way punish the model for making a wrong guess. One of the standard methods is called weighted softmax loss, which uses label frequencies of the training sample for re-weightning [6]. A way of improving this method is by tuning the influence of label frequencies on loss weights, based on sample influence. Further more balanced softmax proposed to use label frequency to improve model predictions during training. By doing this

one lessens the burden of having to ”get to know old knowledge again”. There are a couple of different takes of how to do Cost-sensitive Learning in good ways. One take is that instead of using label frequencies one would instead use a concept called effective number to approximate the expected sample number of different classes. The concept effective number is an exponential function of the training sample number. Lastly the new loss function introduces a weighting term which is inversely proportional to the effective number of classes.
Another take is to use weights that are learned from Data. In a special case done by Meta-Weight-Net, the validation set is balanced whereby the weighting function is approximated by a one-layer MLP which in turn is fitted for the long-tailed distribution [4].

2.3.3 Logit Adjustment

The idea is based on post-hoc shifting model logits based upon label frequencies. Some alternative ideas within the space of logit adjustment is to change the logits based on the test data instead of the training data, which would make the model function for more arbitrary data. [7] A causal classifier was introduced by De-confound. By computing the exponential moving average of features during training this data is then used as bias information. The bias information is furthermore then used for removing bad casual effects by subtracting the bias [8].

2.3.4 Data augmentation

By using augmentation techniques there is a possibility of enhancing the size and quality of data sets which is then used for model training. A solution that is built for long-tailed problems is Non-transfer augmentation. MiSLAS proposed in their paper [9] that (1) data mix-up helps to remedy model over-confidence; (2) mix-up has a positive effect on representation learning in the decoupled training scheme. That is, they proposed that data mix-up can enhance representation learning.

2.3.5 Representation Learning

The current spectra of representation Learning is focused around a couple of different fields. Metric learning establishes similarity or dissimilarity between objects. When it comes to long-tailed learning the methods are based on seeking after distance-based losses and thus achieving a more discriminating

feature space. [4]
There is a couple of different Sequential Training models such as Hierarchical feature learning and Unequal-training. The first mentioned seeks to cluster the data into visually similar groups which then forms a cluster tree. The original node is pre-trained on a known set and the children inherits model parameters. Then the new child is trained by using the samples in the cluster and gets a bit better. Unequal training on the other hand aims to treat the head and the tail differently while training. First using the head to train a resilient model, and afterwards using the tail-class samples to strengthen and enhance the models interpretation of hard identities [10].

2.3.6 Classifier Design

It is common practice for deep learning to use a linear classifier such that

$$p = \phi(w^T f + b) \tag{2.9}$$

where ϕ is a softmax function and b is a biasterm, which usually can be ignored. Class imbalance does so that the head classes has a larger classifier weight. A proposed method to circumvent this is to use scaleinvariant cosine classifier such that

$$p = \phi((\frac{w^T f}{||w||||f||})/\tau + b) \tag{2.10}$$

where the temperature τ is set to a reasonable level. By using a τ-normalized procedure there is a possibility to address imbalance of decision boundaries. Let $\tilde{w} = \frac{w}{||w||_2^\tau}$. When the temperature τ is 1 you have the usual L_2-normalization. If the temperature is 0, then there is no scaling. This is a method that can also be trained with class-balanced sampling. This resulting classifier is named the learnable weight scaling classifier. Another property that comes along is that the nearest class mean classifier computes the L_2-normalized mean features for each class on the training set. This is followed up by using an nearest neighbor algorithm that can be either based on Euclidean distance or cosine similarity [4].

Chapter 3

Method

3.1 Research Method

For starters, a thorough introduction to the existing model was conducted by the hosting company. The problem was presented, and a literature study was made to see different ways of being able to tackle the problem at hand. A method was chosen in consultation with supervisors, and then further investigation was made into the subject to get a comprehension of how to apply it to the problem at hand. From this point a modification to the main model was made, which was where the majority of the time allocated for this thesis was put into. Finally an evaluation of the project was made with a focus on data distribution, speed of the training process and prediction accuracy.

3.2 Data

The data is collected in-house, and has been for many years, and the data is saved to a database. Not all data available from the server is used for the task at hand, but only the fields that are used for model training are extracted. First of all the data is filtered between certain date intervals. The date is set by when the item was paid for. After that the information extracted is which category the item falls under, if it is clothing or an electronic product etc, and which type of item it is, that is if it is a clothing item, is it a shirt or something else. On a more item specific level the condition, brand, patterns and measurement is also fetched. As for condition it is also relevant to know if the item has any defects, which is also extracted. Most of these stats are manually collected by the people working at the warehouse with the sorting process of new items. Finally also the original price of which the item was sold for is extracted which

is used for setting a ground truth for the models as well as the evaluation of the models. The original price is translated into a price bucket which also is the representation of a class. For example, if an item is sold for 57 SEK it belongs to the class 50. For a full table of all classes see Appendix **??**.

The data itself is long-tail distributed, with a vast majority of the data being of more common brands, for example H&M and sub-brands to them. In Figure 3.1 it is visible how there is a clear long-tailed distribution. The x-axis is numbered between 1-59 which corresponds to the different classes used. They differ in sizes between 25 SEK and 50 SEK.

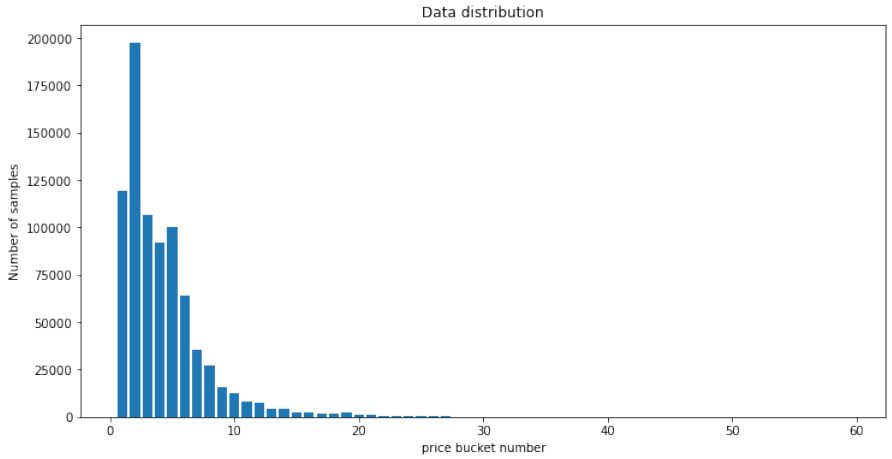


Figure 3.1: How the data is distributed into price buckets

The data in the tail might be everything from more expensive items that naturally has less data, or just lesser known brands.

3.3 Data Metrics

Some of the basis for this thesis is based upon the observation that training a model on a uniform distribution with fewer samples might be easier than training it on a large sample size that has a long-tailed distribution[11]. Thus it is also important to be able to determine if a dataset is uniform, or how far away from a uniform distribution it is. With this aim in mind, below is different metrics all with the goal of somehow measuring how uniform a set is.

3.3.1 Imbalance Ratio

Imbalance Ratio is a basic measurement of looking at the class with most samples, and the class with the fewest samples and thus getting a sense of how imbalanced the data is. In the case of this thesis a class is represented by a pricebucket. It is bounded by 1 from below, which also represents a pure uniform data [11]. It is defined as:

$$I_{Ratio} = \frac{N_{imax}}{N_{imin}} \tag{3.1}$$

where N_{imax} is the maximum number of samples in the data set and N_{imin} is the minimum number of samples.

3.3.2 KL divergence

The Kullback-Leibler divergence (KL divergence) is a measurement of how different two distributions are compared to the data. In this case we compare the existing distribution with a uniform one. A low value would indicate a uniformly distributed set, and a high value the opposite[12]. It is defined as:

$$I_{KL} = D(P||Q) = \sum_i p_i log(\frac{p_i}{q_i}) \tag{3.2}$$

where p_i is the class probability given by

$$p_i = \frac{N_i}{N} \tag{3.3}$$

and q_i is the uniform distribution probability given by

$$q_i = 1/C \tag{3.4}$$

where C is the number of different classes.

3.3.3 Class Balance

The class balance metric was proposed by Collins et al. [13]. It the paper the theoretical proof is shown for why it is bounded between $[0, 2]$. In this case, if there exists a class with all the data the value is 2, and if there is a uniform amount of samples in each class, it will have the value 0. It is defined as:

$$I_{abs} = \sum |\frac{1}{C} - \frac{N_i}{N}| \tag{3.5}$$

this can be seen as the sum of the distance between the uniform distribution probability and class probability for each class.

3.3.4 Gini Coefficient

The last metric is a metric more known as an inequality metric used in finance or to measure countries wealth[14]. But the Gini Coefficient can also be used as a metric for inequality in general for data. It is defined as:

$$I_{Gini} = \frac{\sum_{i=1}^{C}(2i - C - 1)N_i}{C\sum_{i=1}^{C} N_i} \tag{3.6}$$

It is defined between $[-1, 1]$, where 0 is the value for a true equal dataset, and a negative value is interpreted as the data being very spread out. If the value is closer to 1, it means that there is 1 class with all the samples in it, or a few classes.

3.4 Curriculum learning

Curriculum learning was first introduced by Yoshua Bengio in a paper published in 2009 [15]. The method can be tracked back to 1993, where the main idea is to start with a small task, then gradually increase the difficulty of training. This can much be compared to how the school system for us humans work. It is proposed that a well written curriculum can act as a method to find a better local minima of a non-convex training criteria [16]. This is also known as a Continuation Method which is a well known method in non-convex optimization [17]. In this method, you start from a simple smooth objective function, gradually going towards the original non-convex function. From the lens of Machine Learning the objective function is mostly also the performance measure. Applying this would mean that it is expected that the model performs better when using easier data samples, and as more complex data is added the complexity of the objective function should also increase. A proposed algorithm exists for how a general algorithm for curriculum learning should look like, seen in Algorithm 1. For this to work, a model M is trained on a training data set D. Further more there is a a curriculum criterion, that is a criterion for for how the ordering of the data is composed. There is also a level l which is a sort of definition of where in the curriculum the data belongs to. An

example of this is if data belong to the easy part of the training data, or medium or hard, given that there are 3 levels. Traditionally the criterion is based on a easy-to-hard ranking system where the definition of easy and hard is task-specific. An important part of the methodology for curriculum learning is the scheduling function, which is when to update the training process. One way is that the scheduler is bound by the performance level p, that is when a certain performance is achieved even more data is introduced to the training. Another approach for the scheduler is that it is solely based on iterations/epochs before moving up in level.

Algorithm 1 General curriculum learning algorithm

Data: M – a machine learning model;
D – a training data set;
P – performance measure;
n – number of iterations / epochs;
C – curriculum criterion / difficulty measure;
l – curriculum level;
S – curriculum scheduler;

```
1 for t = 1, 2, ..., n do
2     p ← P(M)
      if S(t, p) = true then
3         (M, D, P) ← C(l, M, D, P)
4     D* ← select(D)
      M ← train(M, D*, P)
```

In the first step of the algorithm the number of training instances is defined, going up to the number of epochs. In the second row, the current performance level is calculated. If this fulfills a predetermined curriculum scheduler, the learning part starts as described on row 3. Given a level , a model, a dataset and a performance measure, the model, data and performance is updated. Examples of this could be to increase model capacity, sorting the data in a new way and unsmoothing the performance measure. How the modifications are made is based on which level the algorithm is currently at. At row 4 a new subset is chosen, and then the model is trained on this new subset.

3.4.1 Theoretical Analysis of Curriculum learning

As mentioned before, this can be seen as a form of Continuation Method. Using a easier version is the key for understanding the global picture. Starting

with a very smooth function where it easy to identify a global minima, and then slowly introducing complexity. By doing this it is easier to track the minima and thus getting a better approximation in the end. The training guides the training towards better results. Recent research has also focused on curriculum learning in the aspect of data distribution. Just as in the spirit with this thesis, deep-learning in general uses large-scale data sources. On a global scale, data is some times collected from many different sources and the noise in the data that is added by this is noticeable. In the case of Sellpy, all the data is in-house, but yet it is very probable that there is also noise in the data. The advantage of curriclum learning in this aspect is that noisy data should generally be associated with harder examples, and the clean data is more so associated with easy data. From this it is quite intuitively understood that the model is spending more time training on data without noise than with noise which leads to faster training and denoising.

The denoising is discussed by Gong et al. [18] and is built upon the notion that there inherently exists a difference between the training data and the testing data. This difference is a product of noise in the data. Looking at the total distribution of both the training data and the testing data, there exists a region that has more high-confidence data. This data should then in theory correspond to the easier data, and thus by starting the training on easier samples you minimize the negative effect of noise.

3.4.2 Application

There are two general motivations for using curriculum learning as a method [17]. Either, the objective is to guide the model. By doing this the model will achieve a more regularized training in the optimal space, which gives steeper gradients i.e faster training. The other motivation is to denoise the data putting more of the model-attunement on high-confident data and thus creating a more fair and realistic model. When the motivation is to guide, this is often coupled with the fact that some of the tasks are difficult to predict with direct training and yielding in poor performance or slow convergance. Whereas when the motivation is to denoise, this is rather linked with the desire to increasing training speed or making the model more robust and generalizable.

3.5 Teacher-Student model

One of the more popular methods in recent research is Knowledge Distillation. It is primarily used as a method to keep down computational costs, since the

teacher models can be trained in parallel. Another advantage is that since the teachers do not need to be retrained every time the model is updated you are able to only retrain the student. It has been shown that one could use smaller models to mimic the behaviour and the complexity of larger models without any significant loss in accuracy [19]. The point of many models is to find ways of reproducing an accurate answer to what an objective function might represent. In the case of a pricer model, this would mean to give an accurate representation of consumer pattern, and especially functioning for new data. For the case of this paper only the teacher-student relationship part is used and not the complete method of knowledge distillation.

3.6 Evaluation

The evaluation of the method is divided into three different parts. For starters the data distribution is evaluated with the help of 4 metrics, before and after introducing Curriculum learning. Afterwards the models themselves are compared in terms of raw numeric values from the training instance. Because the model is an adaption of the CORAL-framework the loss function and the mean absolute error are also custom made, but these are presented in the results as well as the speed of the training. Finally the accuracy of the models are evaluated. This is done by creating three different test sets, and measuring how far in terms of pricebuckets they are from the ground truth.

Chapter 4

Implementation

4.1 Pre-processing

Before the models could be trained the adequate data is needed. The data is collected from the database using SQL-scripts, and the data is within the date interval 2022-08-16 to 2022-10-03. This particular interval was chosen to train the models on enough data to produce relevant results, but still at a level that it is possible to iterate different settings for the models without it taking too much time. The steps taken before training the models were fetching the data as described above, after that the data is cleaned, removing irrelevant rows together with outliers where some essential information might be missing. During the cleaning stage the data in the interval of 0SEK to 125 SEK is also down-sampled by 40%. Lastly a transformation is made, that is creating numerical data out of string metadata with the help of tokenizers. A tokenizer is a common tool in Machine learning for converting text to numbers, and making sure that text is always converted to the same number. From this point the transformed data is saved and loaded into the teacher models for training.

4.2 Teacher models

The teacher models were made following the original model as presented by the host company, with some modifications to both make it easier to follow and because the original model included bits for analysis that was not needed for the thesis. Three teachers were created were the first teacher had data in the intervall $0 - 125$ SEK, the second teacher had data between the interval $125 - 500$ SEK and lastly the third teacher had data from 500 SEK and all the way up to the most expensive. Each model was trained with 10 epochs, and

without early stopping. Early stopping is a Tensorflow setting for the models, making it possible to automatically stop the training in case the validation loss has not increased in a certain amount of epochs given a certain tolerance.

4.3 Ranking

The ranking made is a way to sort the data after it has been trained on the teacher models. The first step in the sequencing is to load the teacher models. After that all the data which the teacher models are trained on are loaded. The training set beloging to their corresponding models are then used as the input for prediction. The prediction is then translated into which pricebucket the guess belongs to, and the next step is creating the ranking. The ranking is done as a measure of distance between a prediction and the ground truth, that is which price bucket the original selling price belongs to. This leads to a distance of 0 being the lowest and the highest is in theory the number of price buckets.

4.4 Student model

The ranked data is loaded into the student script. The student model is trained using a training scheme as a part of applying curriculum learning. The important part with this training scheme is that it is monotonically increasing, meaning that the function is training on more and more data after each epoch. The training scheme used is a simple one increases with 10% more of the data every fourth epoch. With this as a training scheme the model is training on half the data after 16 epochs, and all the data after 36 epochs. It is built on the same basis as the teacher models and the base model, the only difference being the curriculum learning scheme and not introducing early stopping until epoch 36. Here the metric for early stopping is set to 3 and a tolerance of 0.01. This means that if there goes 3 epochs without the 0.01 improvement in validation loss the training should be stopped. The maximum amount of epochs is set to 100, so that in theory 136 epochs could be trained.

4.5 Base model, the existing model

The base model itself is trained in the same manner as the teacher models, with some modifications to values. Furthermore it is trained on the same data as the student model, with addition of up-sampling on the rare pricebuckets in the

training set in the interval of 500 SEK all the way up to the most expensive. The up-sampling variable is set to 15. This means that it looks at buckets with a few samples in it, and up-samples it by multiplying it by the variable, creating duplicates in the data. It uses early stopping with a tolerance of 0, and an epoch interval of 3. That is, if there is 3 epochs without the validation loss improving the training is stopped.

4.6 Evaluation

A separate script is made for the evaluation loading all relevant models and data. Tensorboard is an extension made by Tensorflow, which can be used to evaluate models trained with the help of Tensorflow. This tool is used for extracting training loss, validation loss, as well as the mean absolute error. Lastly three different test sets are created. The test sets are referred to as ”cheap”, ”medium” and ”expensive”.They follow the same logic as the split of the teacher models, so the cheap set is data within the interval of $0-125$ SEK, medium is within $125-500$ SEK and the expensive is from 500SEK and up. Each set consists of 100 random items within the date interval of $2022-06-16$ and $2022-07-03$. These sets are used for evaluating accuracy.

Chapter 5

Results

5.1 The data

In Table 5.1 the results after calculating the four different metrics presented in Chapter 3.4 can be found. The first metric to notice is Imbalance Ratio, denoted **IR** in the table. The full data set and the third data set both have a high value. This is a result of classes at the end of the spectrum only consisting of 1 sample. For the medium set and the cheaper set the number is greatly reduced from the original number. From the Imbalance Divergence column, denoted as **ImDIV**, it is apparent that the cheap and medium data set has a lower value, indicating a more uniform distribution. The complete data set is the highest number and the expensive set shows an improvement trending towards a bit more of an uniform distribution. The class imbalance metric, denoted as **CI** shows that there is nearly no difference between the total data set and the expensive data set. There is some improvements going from expensive to medium, and numerically almost the same improvement from the medium set to the cheap set. Lastly is the inequality measure, the Gini Coefficient denoted as **GC**. For the complete data, it is evident that it is very inequal with a slight improvement for the expensive version. As with the other metrics there is a greater improvement for the cheaper set, and somewhere in between for the medium set.

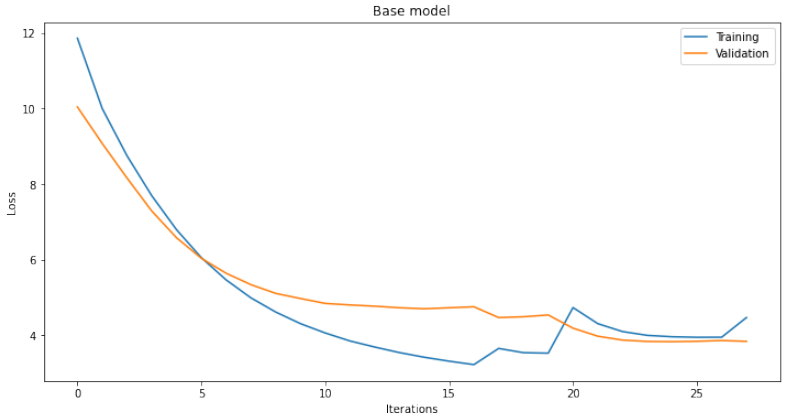
Dataset	Sample Size	Classes	ImDIV	CI	GC	IR
Cheap	554317	5	0.1099	0.6515	-0.5926	5.1074
Medium	230271	8	0.2768	0.7979	-0.6558	10.9945
Expensive	27393	45	1.2270	0.9657	-0.8103	4504
All	811981	58	1.7921	0.9986	-0.9055	197362

Table 5.1: Summary of Different Datasets

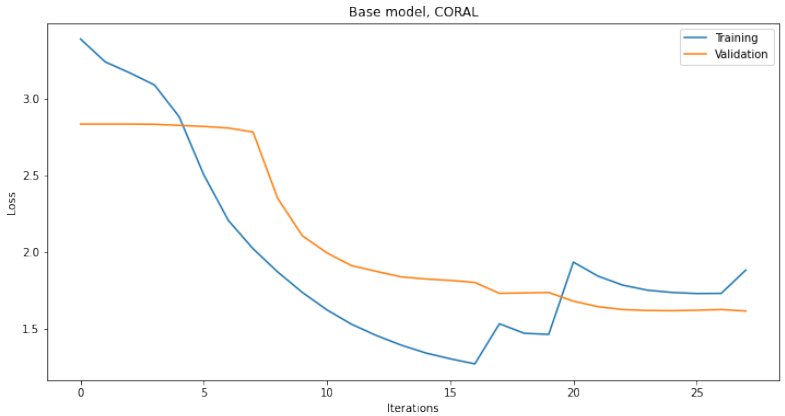
5.2 Model training

5.2.1 Base model

The base model had an average epoch time of $318s$. After epoch 25 there was no improvement in validation loss which resulted in early stopping and terminating the training session. In Figure 5.1(a) the graph is shown for how the loss function behaves for both the training set and the validation set. Furthermore in Figure 5.1(b) the CORAL-adaptation of Mean Absolute Error is shown. Very similar trends are seen in both graphs, where it starts converging towards an optimal validation loss in both cases around epoch 7, finding an improvement at epoch 20 and then stagnating. At the end it finished with a validation loss of 3.8658 and a validation CORAL-MAE of 1.6136. The model took **2 hours 27 minutes** to finish training.



(a) Training and Validation loss for the base model



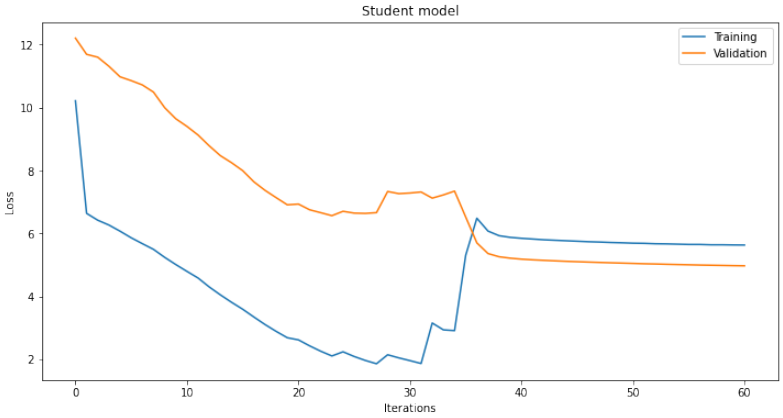
(b) Training and Validation Mean Absolute Error for the base model

Figure 5.1: Overall loss for the base model

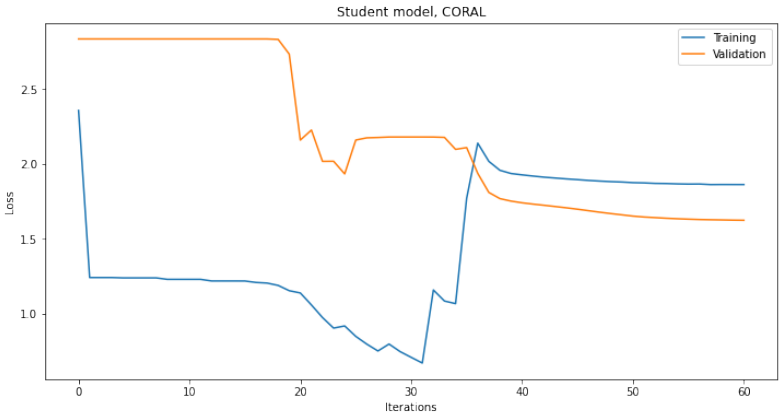
5.2.2 Student model

The student model starts out with the first four epochs with a speed of 39 seconds per, and slowly increasing in 4 epoch intervals as more and more data is introduced to the model. Upon reaching epoch 37, when the complete dataset is introduced to the model, each epoch is training at the same speed as the base model, which is approximately 318 seconds per epoch. When introducing the last parts of the data, around epoch 30, it is visible that the training loss increases drastically. The validation loss is decreasing slowly, and as soon as the entire data is introduced there is only marginal improvements per epoch. It stopped at epoch 60, but with almost no improvement compared to epoch 40. The model reached epoch 36 after

1hours40min and finished at epoch 60 after **3hours47min**. At the end it finished with a validation loss of 4.9705 and a validation CORAL-MAE of 1.6228



(a) Training and Validation loss for the student model



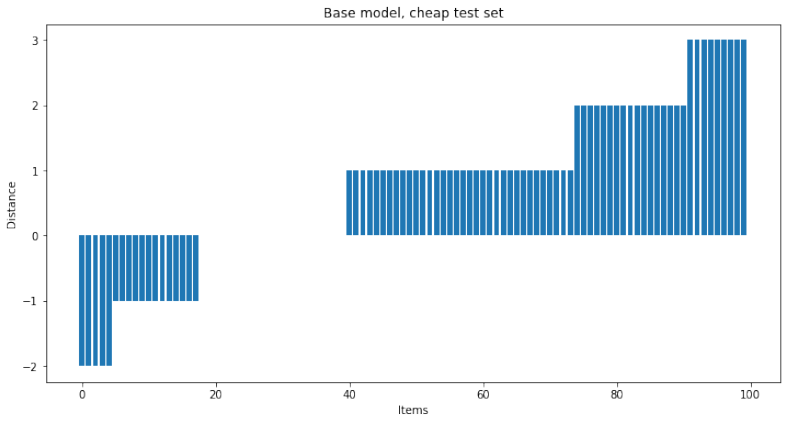
(b) Training and Validation Mean Absolute Error for the student model

Figure 5.2: Overall loss for the student model

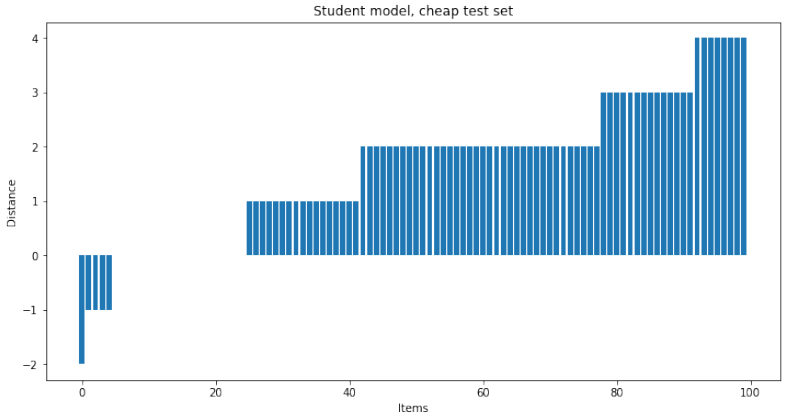
5.3 Prediction

The results of how well the two different models performed in forms of prediction is divided into three different parts. As a measure for accuracy, distance is used. The distance refers to how many prickebuckets of the ground truth the prediction is. In the case of the cheap set the base model has more than 60 predictions within the range ± 1, with a max deviation of 3. The student model has 40 prediction within the same range, and almost 40 predictions with

a distance of 2 compared to the base models 20. Finally the student model also has a higher max deviation of 4.



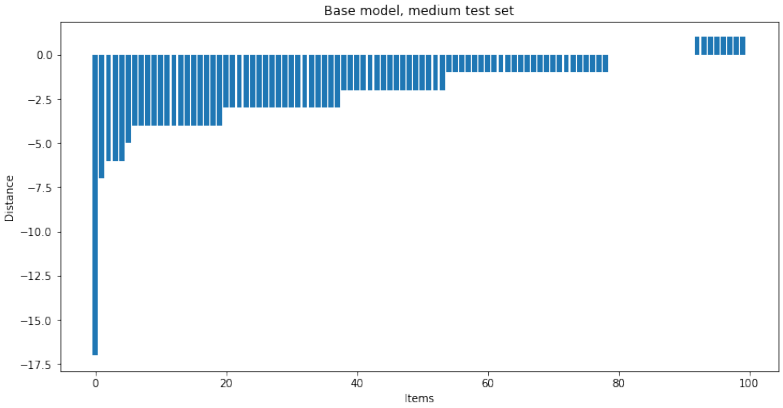
(a) Predictions of the cheap set using the base model



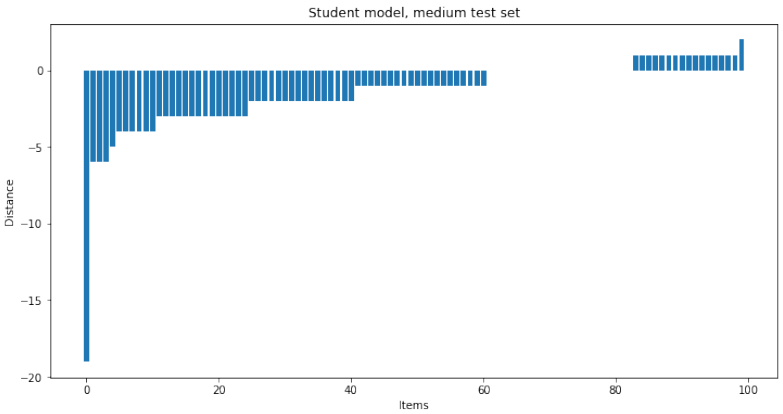
(b) Predictions of the cheap set using the student model

Figure 5.3: Predictions of the cheap test set of the models

Focusing in on the medium test set, seen in Figure 5.4, both models have a high max deviation. The base models at a distance of 16, and the student model 19. Within the range of ± 1 in distance, the student model is at the range of 60 predictions whereas the base model has 45 guesses in that interval. The remaining trends are similar for both the prediction-cases. Worth noticing is that both models are undershooting, meaning that almost every prediction is below the ground truth.



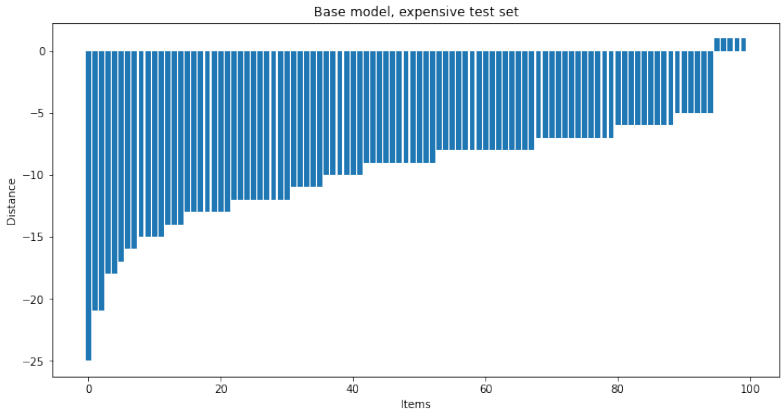
(a) Predictions of the medium set using the base model



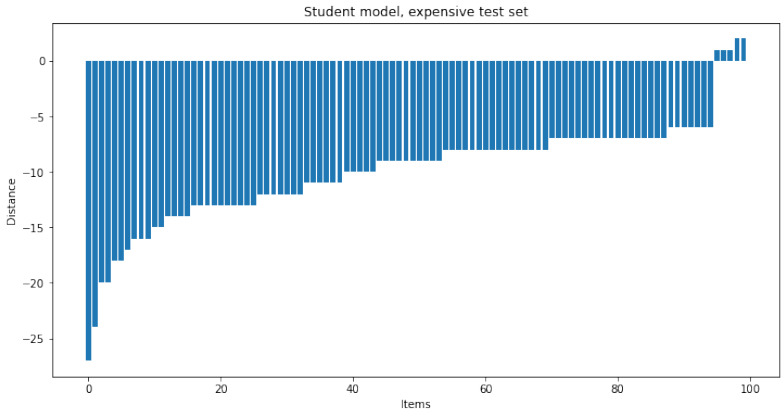
(b) Predictions of the medium set using the student model

Figure 5.4: Predictions of the cheap test set of the models

Finally is the predictions of the expensive test set seen in Figure 5.5. Worth reiterating is that the expensive set consists of 45 different classes. Within the range of ± 5 in distance the student model has 30 predictions and the base model has 20. Other than that the both models has very similar trends with a max deviation of 25 for the base model and 27 for the student model.



(a) Predictions of the expensive set using the base model



(b) Predictions of the expensive set using the student model

Figure 5.5: Predictions of the cheap test set of the models

Chapter 6

Discussion

6.1 Data

The data was clearly very long-tailed from the beginning as indicated from the Imbalance Ratio. This is an effect of classes in the tail consisting of few samples, and classes in the head being very data point heavy. With a high value of KL-divergence it is also apparent that complete set is a long way from being uniformly distributed, which also is an expected result for the same reason as the high value of Imbalance Ratio. The cheap and the medium set consists of more than 95% of the data, which means that almost all data has an original selling price of 500 SEK or less. Even, with dividing the set into three different parts, the expensive set is showing clear signs of being long-tailed albeit there is improvement in every field from the original set. There is a couple of possible solutions that in theory would improve the metrics over the board in regards to the expensive set. Either, even more teacher models can be made. Since the data has a natural decreasing trend of number of samples per class, dividing the expensive set into even more subsets would naturally improve the metrics. The problem with this approach is that it does not addressing the core problem at hand, which is that the expensive part of the data is a small portion of the complete set. In the teacher models only down-sampling has been used for the cheap set, but no up-sampling for the expensive set. By using up-sampling this would achieve better data metrics and by proxy also give the student model more expensive data to train on. The cheap and the medium set has flourished in terms of metrics as an effect of splitting up the data. The cheap sets albeit only consisting of 5 classes, shows tendencies of a more uniformly even set. Both the Imbalance Ratio and the KL-divergence is showing promising numbers. The class imbalance and the Gini coefficient

still shows that there is improvements to be made over the classes. The same analysis can be made of the medium set.

6.2 Model training

From the results it looks like the base model is performing better with respect to training the model. It is almost **1 hour 20 minutes** faster and reaching a better result with regards to validation loss, with almost identical mean absolute error(MAE) loss. For the student model, it took 36 epochs before all the data was introduced to the model, and thus also before the early stopping is introduced into the model and it could stop training. It is also visible that after a while, when introducing the data that has a worse ranking, the training-loss increases quite drastically meaning that it is having a hard time determining the noisy values, which is as predicted and showing that the intention of ranking has some of the desired result. Furthermore, by using another value for early stopping in the Tensorflow model the training would have stopped at around epoch 40. After that the increase on both validation-MAE and validation loss is marginal. The model then would have finished after 2 hours, showing an increase in speed in comparison to the base model.

6.3 Accuracy

The accuracy for the cheap set is similar for both the models. On average the base model is performing better, both having less maximal deviation and having more predictions in the interval of ± 1. It is also worth noting that both of them are trending to overvalue items indicating that it the models both might be optimized for data points in the low spectrum of the medium set. This conclusion is strengthened by the fact that when analyzing the medium set on both models they are more prone to make predictions that corresponds to an undervaluation. Continuing the analysis with the medium test set, there is some prediction differences between the two models. The student model has a wider span of predictions within the range of ± 1 showing a better accuracy. This shows that the intended purpose of introducing a teacher-student model and applying a curriculum learning approach is somewhat showing promising results. Since the knowledge distillation was not implemented fully, the improvement in the medium set is an effect of down-sampling the cheap set so that the ranking could have an effect. Thus, the model initially trained on both cheap and medium data resulting in better predictions for the medium data. In

regards to the tail end of the data, both model shows clear lackluster results. Even if the case is that the expensive set consists of 45 classes in contrast to 5 and 8 in the case of the cheap and medium set, the predictions is undervalued and the student model is not an improvement to the existing model which was the intended result.

Chapter 7

Conclusions and Future work

7.1 Conclusions

As presented from the host company, the data used for training of their current model clearly follows a long-tailed distribution. Adapting a teacher-student approach meant that the original dataset is split into different subsets. By choosing to split it into three subsets it was evident that from a distribution point of view the cheap and medium set greatly improved going towards a more uniform distribution. The expensive set also improved but only marginally, not reaching the desired level. Possible solutions to this would be to even further subsets of the expensive data. This solution is first relevant when a knowledge-distillation approach is implemented so that the student model has actual communication with the teacher models. As far as it comes to the model training, at the current settings the base model had a lower training time. Here there is room for improvement in terms of pre-training settings for the student model,which can be explored using a somewhat of a trial and error method. Further comments on this can be found in the future works. Without any modifications made on the current iteration, it is still visible that the student model is an improvement in regards to speed, because the majority of time was spent on minimal increment improvement it is now shown in the absolute numbers, but after further analysis it becomes visible. As for the predictions, starting with the conclusion for the lackluster results of the expensive test set. The results of the ground model with the expensive set does not reflect what the host company is used to. The explanation for this is probably because the current model is trained on 7 weeks worth of data while the model used in production is trained on 3 years of data. This might also be the reason to why the student model is behaving oddly at this level. With a correct

implementation of knowledge distillation there is also possible improvement to be seen for the student model. From the predictions of the medium the conclusion is drawn that introducing ranking with the help of teacher models made a positive impact. The initial problem formulation had a focus on how to improve the model with regards to data in the tail of the distribution. Because the data is so head-heavy some of the data in medium set is also part of tail and thus the claimed improvement.

7.2 Limitations

The work got limited mainly because of time constrains. As a consequence of this the intended implementation of knowledge distillation is not made, and the decision was made to focus on curriculum learning and parts of the knowledge distillation method, namely the ranking method. Other limitations because of lack of time involves tweaking with hyperparameters for the model training as well as trying something else than a linear scheme for the curriculum learning. Furthermore the models were trained locally introducing a RAM-memory problem and not being able to train models on more than 7 weeks worth of data, and also putting limitations of not being able to use the computer while models were in training. Not a problem during development but a problem as soon as trying to scale the models.

7.3 Future work

Since the time constrains introduced quite some limitations this also opens up for a large section of future work to be made. The first one is improving the implementation making it compatible with cloud-services and thus being able to scale. Secondly different forms of training schemes can be introduced as an improvement of the current way of applying curriculum learning. Lastly even more focus can be put into implementing knowledge distillation. More precisely the LFME framework can be adapted [11].

References

[1] "Sellpy - how it works," Sellpy, Sellpy. [Online]. Available: https://www.sellpy.se/howItWorks [Page 1.]

[2] W. Cao, V. Mirjalili, and S. Raschka, "Consistent rank logits for ordinal regression with convolutional neural networks," *CoRR*, vol. abs/1901.07884, 2019. [Online]. Available: http://arxiv.org/abs/1901.07884 [Pages 2, 7, 8, and 9.]

[3] V. Feldman, "Does learning require memorization? a short tale about a long tail," in *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, ser. STOC 2020. New York, NY, USA: Association for Computing Machinery, 2020. doi: 10.1145/3357713.3384290. ISBN 9781450369794 p. 954–959. [Online]. Available: https://doi.org/10.1145/3357713.3384290 [Page 7.]

[4] Y. Zhang, B. Kang, B. Hooi, S. Yan, and J. Feng, "Deep long-tailed learning: A survey," 2021. [Online]. Available: https://arxiv.org/abs/2110.04596 [Pages 10, 11, and 12.]

[5] B. Kang, S. Xie, M. Rohrbach, Z. Yan, A. Gordo, J. Feng, and Y. Kalantidis, "Decoupling representation and classifier for long-tailed recognition," 2020. [Page 10.]

[6] W. Verbeke, D. Olaya, J. Berrevoets, S. Verboven, and S. Maldonado, "The foundations of cost-sensitive causal classification," 2021. [Page 10.]

[7] F. Provost, "Machine learning from imbalanced data sets 101," *Invited paper for the AAAI'2000 Workshop on Imbalanced Data Sets*, 11 2008. [Page 11.]

[8] K. Tang, J. Huang, and H. Zhang, "Long-tailed classification by keeping the good and removing the bad momentum causal effect," in

Advances in Neural Information Processing Systems, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 1513–1524. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/1091660f3dff84fd648efe31391c5524-Paper.pdf [Page 11.]

[9] T. Wu, Z. Liu, Q. Huang, Y. Wang, and D. Lin, "Adversarial robustness under long-tailed distribution," 2021. [Page 11.]

[10] W. Ouyang, X. Wang, C. Zhang, and X. Yang, "Factors in finetuning deep model for object detection," 2016. [Page 12.]

[11] L. Xiang, G. Ding, and J. Han, "Learning from multiple experts: Self-paced knowledge distillation for long-tailed classification," 2020. [Online]. Available: https://arxiv.org/abs/2001.01536 [Pages 14, 15, and 38.]

[12] J. Han. (unknown) Kl-divergence lecture notes. 2023-08-13. [Online]. Available: http://hanj.cs.illinois.edu/cs412/bk3/KL-divergence.pdf [Page 15.]

[13] E. Collins, N. Rozanov, and B. Zhang, "Evolutionary data measures: Understanding the difficulty of text classification tasks," 2018. [Page 15.]

[14] StatsDirect. (2023) Gini coefficient. 2023-08-13. [Online]. Available: https://www.statsdirect.com/help/nonparametric_methods/gini_coefficient.htm [Page 16.]

[15] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: Association for Computing Machinery, 2009. doi: 10.1145/1553374.1553380. ISBN 9781605585161 p. 41–48. [Online]. Available: https://doi.org/10.1145/1553374.1553380 [Page 16.]

[16] P. Soviany, R. T. Ionescu, P. Rota, and N. Sebe, "Curriculum learning: A survey," 2022. [Page 16.]

[17] X. Wang, Y. Chen, and W. Zhu, "A survey on curriculum learning," 2021. [Pages 16 and 18.]

[18] T. Gong, Q. Zhao, D. Meng, and Z. Xu, "Why curriculum learning self-paced learning work in big/noisy data: A theoretical perspective,"

Big Data and Information Analytics, vol. 1, no. 1, pp. 111–127, 2016. doi: 10.3934/bdia.2016.1.111. [Online]. Available: https://www.aimspress.com/article/doi/10.3934/bdia.2016.1.111 [Page 18.]

[19] C. Buciluundefined, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '06. New York, NY, USA: Association for Computing Machinery, 2006. doi: 10.1145/1150402.1150464. ISBN 1595933395 p. 535–541. [Online]. Available: https://doi.org/10.1145/1150402.1150464 [Page 19.]