University of Stavanger

FACULTY OF SCIENCE AND TECHNOLOGY

MASTER'S THESIS

Study programme/specialisation: Data Science	Spring/ Autumn semester, 2020 Open / ~~Confidential~~
Author: Markus Pettersen	
Programme coordinator: Thomasz Wiktorski Supervisor(s): Antorweep Chakravorty, Aida Mehdipour Pirbazari, Trond Sørås	
Title of master's thesis: Deep Learning Over 20 Years of Cinema Ticket Sales	
Credits: 30	
Keywords: TabularModel, Deep Learning, Fastai, PyTorch, Cinema Ticket Sales	Number of pages: ……65…………… + supplemental material/other: - Link to repository in appendix Stavanger, 14.06.2020 date/year

University
of Stavanger

Faculty of Science and Technology
Department of Electrical Engineering and Computer Science

Deep Learning over 20 Years of Cinema Tickets Sales

Master's Thesis in Data Science
by

Markus Pettersen

Internal Supervisors

Antorweep Chakravorty

Aida Mehdipour Pirbazari

External Supervisors

Trond Sørås

June 15, 2020

"Look to yourself that you may not lose all that we and you have labored for, but that you may persevere until you win and receive back a perfect reward in full."

2. John 8

Abstract

For many years, movie theaters across Norway have been manually setting up its weekly movie schedule according to the prior knowledge of the operators. This is a slow and laborious method that in this day and age is outdated and should be automated.

This thesis explores how this process can be automated using deep learning. By exploring 20 years of sales data we gain insight into what can cause screenings to have a high or low coverage. We use this insight to assist with feature selection for the neural network, which in this thesis will be the TabularModel from fastai. With this model we are able to predict the coverage of a future screening.

Our objective is to use these predictions to create a suggested schedule that will optimize ticket sales for movie theaters.

Acknowledgements

I would like to thank my supervisors at the University for their feedback throughout this project.

I would also like to thank DX and Trond Søràs for the oppurtunity to work on this project and for the help and insight they have provided.

Contents

Abbreviations

DX	**D**ialogue e**x**e
TMDb	**T**he **M**ovie **D**ata**b**ase
MLP	**M**ulit-**L**ayered **P**erceptron
RAM	**R**andom **A**ccess **M**emory
MAE	**M**ean **A**bsolute **E**rror
MSE	**M**ean **S**quared **E**rror
LR	**L**earning **R**ate
ED	**E**mbedding **D**ropout
WD	**W**eight **D**ecay

Chapter 1

Introduction

1.1 Motivation

The motivation for this thesis is to optimize and automate the scheduling process for a weekly cinema program. So far this has been a manual job that relies on the schedulers knowledge to set up the best schedule. Our goal is to provide a system that can make a suggested schedule based on the performance of previous data in hope that this will increase the coverage of the future screenings.

1.2 Problem Definition

As mentioned, our goal is to create a system that can suggest a weekly schedule for a movie theater. In addition to this we want to give some insights as to what separates good from bad screenings.

This thesis consists of two main parts, an analysis of the data to find patterns and gain insights to the data, and a scheduler that can predict the outcome of a given screening and based on this suggest an optimized schedule. The goal of the analysis is to find patterns that can show us why some screenings have a higher coverage than others. The scheduler's job is to predict the coverage of future screenings and to generate a suggested schedule to optimize the sales.

1.2.1 Q&A

When proposing this thesis, Dialogue exe (DX) had a few questions they wanted answers to. Our analysis, in chapter 3, will seek to answer these question.

The main questions are as follows:

- Which time(s) of day has the highest coverage
- Which day(s) of the week has the highest coverage
- Is there a difference in time of day e.g. for Friday and Monday
- Which genre performs the best
- Should screenings start earlier during work days?
- Is there a noticeable difference in ticket sales if the movie starts earlier or later?

1.3 Challenges

Throughout the work of this thesis we faced several challenges, gathering additional information and aggregating it with the original, identifying useful patterns in the data, the unpredictable nature of movies and their screenings, and finding a model that is able to take advantage of the patterns in our data. This section will cover each of these challenges in turn.

Finding a good source for additional metadata for the movies and combining this with our original dataset from DX. Being able to properly combine this extra information with the original dataset proved somewhat problematic as the original data provided often lacked the original title of a movie and often included the version (3D, atmos, HFR, etc.). The cleaning and aggregation process lead to some mismatches which may cause some inaccuracies in our results, this can be solved with access to the right databases, such as Filmweb.

Being able to properly identify interesting patterns in the data proved a to be a challenge. When looking at individual features we try to see if we can find some patterns that can be discerned by a human observer.

All movies have the potential to have both good and bad screenings and isolating when a specific movie will have its best performance is not a simple task. Movies also have a tendency to break with expectations, in other words a movie can sometimes flop even if the expectations for it were high and vice versa a movie no one expected to be good can become a sensational hit. Movies also loose traction over time, the longer since the release the lower the coverage, so our model needs to take this into consideration when making predictions about the performance of any given screening. Just because the coverage was close to 1 on the opening weekend does not mean the same will be true the following weekend, or even the following day.

The biggest challenge we faced have been being able to find a model that is capable of properly learning from the data and tuning the hyperparameters of the model we chose.

1.4 Contributions

In this thesis we suggest a way to create an automated weekly movie scheduler for movie theaters as well as investigate the data and give insight regarding the conditions responsible for both good and bad performing screenings.

1.5 Outline

In chapter 2 we provide the reader the necessary background information for subjects they should have an understanding of when reading this thesis.

Chapter 3 goes through the analysis of the data, the different areas investigated and gives some comments on the results along the way.

Chapter 4 gives an explanation and an overview of our proposed solution. In this chapter we explain how the individual components work together.

In chapter 5 we go through the implementation of our solution and the testing done to achieve our results. We also go through the preprocessing of the data as well as the aggregation of data from different sources.

Chapter 6 discusses the results and the impact of this thesis and provides a conclusion.

Chapter 7 proposes some future directions for further development of this project.

Chapter 2

Background

2.1 DX

This thesis was proposed and was completed in cooperation with DX. Their website states the following:

> DX is a software and services provider to the entertainment industry. DX was founded in Bodø, Norway in 1997. We work with 200+ cinemas, venues and event organizers. [1]

2.2 TMDb

To add metadata for the movies we have used data from TMDb. Their website gives the following description of the service they provide:

> The Movie Database (TMDb) is a community built movie and TV database. Every piece of data has been added by our amazing community dating back to 2008. TMDb's strong international focus and breadth of data is largely unmatched and something we're incredibly proud of. Put simply, we live and breathe community and that's precisely what makes us different. [2]

2.3 Feature embeddings

A given dataset will generally contain a number of features, sometimes they are all of the same type and in the same numeric range and can then be used directly as input to a neural network. However, this is rarely the case in real life observations and we often have to deal with different type of features both categorical and continuous values not often normalized. No matter what features we have, the common denominator is that we want to use them to find patterns we can use for predictions.

The reason we expect these machine learning algorithms to work is that we expect that instances with similar feature values will lead to similar predicted output. And hence the representations of these input features directly effects the nature and the quality of the learned patterns.[3]

We can directly use the continuous features without making any changes to them beforehand, though it is often a good idea to normalize the data to avoid some features having more to say just because of the scale of the feature. E.g. number of seats in a room and the budget of the movie shown, one is in the hundreds range while the other in the millions range, see table 2.1, this could cause the budget to have an exaggerated influence that is not representative of the its real impact, we therefore normalize the values to account for the relative scale.

id	showtime	room	number of seats	movie budget	movie genres	coverage
0	2019-12-31 21:50	3	50	10 000 000	[27, 53]	0.32
1	2019-12-31 22:00	2	66	40 000 000	[35, 80, 18, 9648, 53]	0.39
2	2019-12-31 19:00	1	70	40 000 000	[35, 80, 18, 9648, 53]	0.38
3	2019-12-31 22:00	4	211	250 000 000	[28, 12, 878]	0.07
4	2019-12-31 22:00	1	66	10 000 000	[27, 53]	0.20

Table 2.1: Example dataset

When it comes to categorical features, such as genre or room these cannot be directly used as input for a neural network and needs to be transformed to one or more continuous features, this is done by using feature embedding. There are multiple ways to implement feature embedding, and for this thesis we will be using two implementations; one-hot encoding and Cat2Vec. How these work will be discussed in the following subsections.

But what exactly is an embedding, google gives a good description in of its tutorials:

An embedding is a translation of high-dimensional vector into a low-dimensional space. Ideally, an embedding captures some of the semantics of the input by placing semantically similar inputs close together in the embedding space. [3]

2.3.1 One-Hot encoding

One of the embedding methods used in this thesis is one-hot encoding. One-hot encoding works so that if there are 10 categories in a categorical feature then there will be 10 resulting features where each feature represents the presence or absence of one the corresponding category. An example of this is shown in table 2.2 and 2.3.

id	movie genres
0	[27, 53]
1	[35, 80, 18, 9648, 53]
2	[35, 80, 18, 9648, 53]
3	[28, 12, 878]
4	[27, 53]

Table 2.2: Example dataset, categorical feature

id	27	53	35	80	18	9648	28	12	878
0	1	1	0	0	0	0	0	0	0
1	0	1	1	1	1	1	0	0	0
2	0	1	1	1	1	1	0	0	0
3	0	0	0	0	0	0	1	1	1
4	1	1	0	0	0	0	0	0	0

Table 2.3: Example dataset, one-hot encoded

There are both advantages and disadvantages with using one-hot encoding (lists combined from [4] and [5]):

Advantages:

- Determining the presence of a category has a low and constant cost of accessing one feature
- Changing the category has the constant cost of accessing two features
- Easy to design and modify

Disadvantages

- Requires more features than other encodings
- For features with many unique categories - the dimensionality of the transformed vector becomes unmanageable
- The mapping is completely uninformed: "similar" categories are not placed closer to each other in the embedding space

2.3.2 Cat2Vec

In addition to one-hot encoding we will also be using Cat2Vec [6] for embedding categorical features. This method is used in fastai's TabularModel by default. This is used to try and capture the relationships between the categories of the features [7].

And unlike one-hot encoding, Cat2Vec should be able to capture more of the semantic relationship between the categories.

2.4 PyTorch

PyTorch is an open source deep learning library, primarily developed by Facebook's AI Research lab [8]. Utilizing the kernel modules from Cuda enables efficient use of the GPU (if from NVIDIA) which for deep learning is much faster than the CPU.

2.5 fastai

The fastai library is built on top of PyTorch. This library is designed to make it easier to quickly build models and to experiment.

> The fastai library simplifies training fast and accurate neural nets using modern best practices. It's based on research into deep learning best practices undertaken at fast.ai, including "out of the box" support for vision, text, tabular, and collaborative filtering models. [9]

When using fastai, much of the manual setup steps that is required by PyTorch is done automatically. Their databunch class allows for easier treatment of training, validation and test sets, making sure that the same processes are done to all the sets in the right order and divides the sets into mini-batches.

Their code-base is well documented and structured in an intuitive manner such that changing the source code to make customizations is still easier than building it all from scratch using PyTorch or TensorFlow.

2.5.1 TabularModel

This is a model provided by fastai designed for tabular data such as pandas dataframes. The model combines embeddings of the categorical features with the continuous features. By default each layer of this model consist of batchnorm, a linear transformation and a ReLU activation function, resulting in a specialized multi-layered perceptron (MLP).

The TabularModel uses Cat2Vec as the embedding method for the categorical features, hence managing to reduce dimensionality of each categorical feature.

fastai describes this model as a bridge between the pandas library and PyTorch, and provides the following description:

> The pandas library already provides excellent support for processing tabular data sets, and fastai does not attempt to replace it. Instead, it adds additional functionality to pandas DataFrames through various pre-processing functions, such as automatically adding features that are useful for modelling with date data. fastai also provides features for automatically creating appropriate DataLoaders with separated validation and training sets, using a variety of mechanisms, such as randomly splitting rows, or selecting rows based on some column.
>
> fastai also integrates with NVIDIA's cuDF library, providing end-to-end GPU optimized data processing and model training. fastai is the first deep learning framework to integrate with cuDF in this way. [10]

2.6 Random Forest

We are using a Random Forest model as our baseline in this thesis.

> Random Forest is an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes or mean prediction of the individual trees. [11]

Chapter 3

Initial Analysis

This chapter will cover the different analysis of the data done and their importance going forward. In this chapter there is a subsection for each question that we wanted an answer to.

The sections of this chapter will go into the details regarding the result from our analysis. To give an overview of the different stats we will go through, they are as follows:

- Coverage by Hour of Day
- Coverage by Day of Week
- Coverage by Hour of Day per Day of Week
- Coverage by Genre
- Coverage by TMDb score
- Coverage by TMDb popularity
- Coverage relative to Release Date

3.1 Coverage by Hour of Day

At first we looked to see if there were any specific time periods that sold better than others. The results of this can be seen in figure 3.1. As we can observe there is a spike during the early hours while from 12:00-23:59 it remains somewhat stable. The numbers on top of each column in the figure represents the number of instances observed at each timestep. This shows us that most of the screenings happens in the time-period 16:00-21:59 and that screenings outside this period can be considered to be special occasions or outliers.

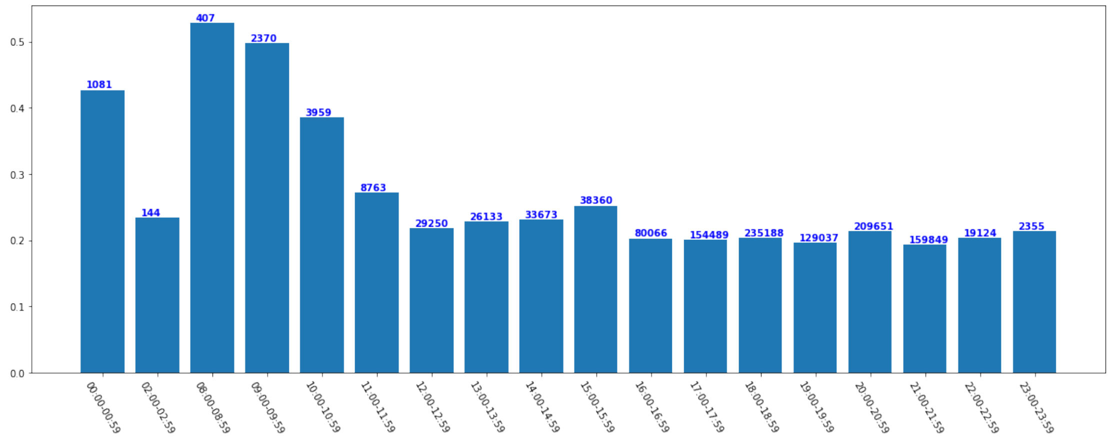

Figure 3.1: Coverage by Hour. (Number on top of each column represents the number of instances observed at each time period)

We select a subset of the data, that would represent normal opening hours, 12:00-22:59. The subset can be observed in figure 3.2. For the regular opening we see an almost uniform distribution with a marginal difference here and there. This distribution closely approximates the average of the data, here the average is 0.213, while the average of the entire dataset is 0.208.

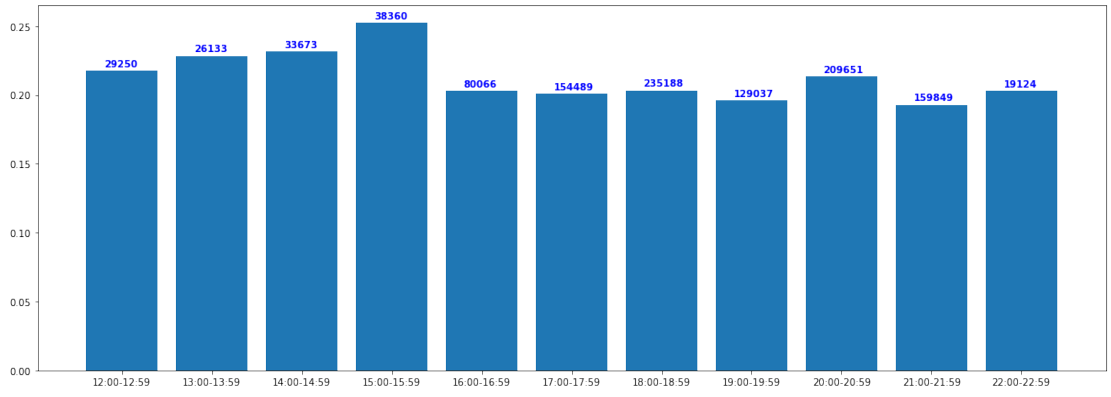

Figure 3.2: Coverage by Hour. Regular opening hours.

With this we conclude that within regular opening hours the time a movie is shown has a very little if any effect on the coverage of the screening. Outside the regular opening hours we see some differences, but this might be simply due to the few amounts of screenings those period, if the number of screenings outside regular hours increase this difference might decrease and stabilize around the mean, or it may very well turn out to be a worse time than regular hours if done too often.

3.2 Coverage by Day of Week

Next we looked into the effect a given weekday will have on the coverage of a screening. Before we began our assumption was that weekends will most likely have significantly higher coverage than normal workdays.

As can be seen in figure 3.3 there is a noticeable spike during the weekend (Thursday-Saturday) as suspected. From this we confirm that more people go to see movies during the weekend, and we can also observe that these days, especially Saturday, has a significant higher number of screenings than the other days. So during this period there is not only a higher average coverage but this is also the period with the highest number of screenings.

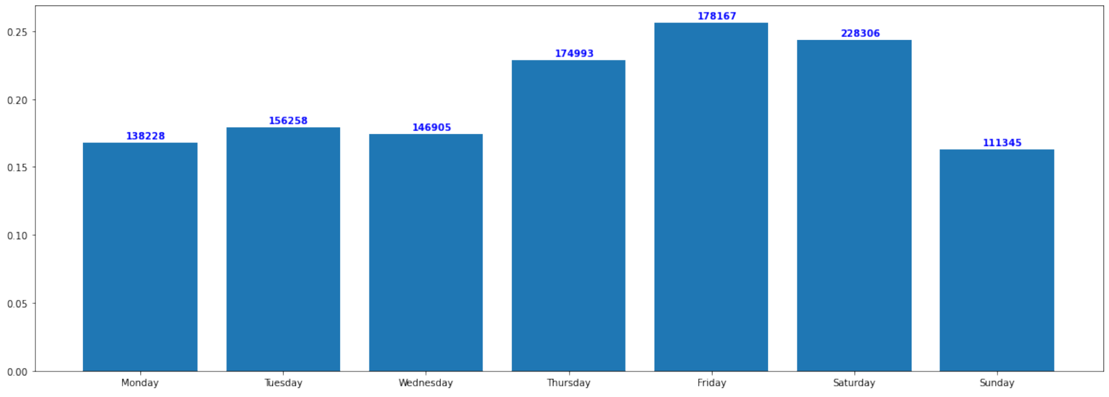

Figure 3.3: Coverage by day of week

Somewhat surprisingly, Sundays were the worst performing day of the week, by a small margin. Even though Sundays are technically a part of the weekend, the Norwegian culture is probably a reason for its low performance. Sundays are often considered as a church or family day in Norway, and its only in recent years that Sundays are not as much regarded as a holiday as it used to be.

The stats from figure 3.3 contains all the data and if we looked at the same stats for only the opening hours then we would see the same results, with Sunday even lower, but if we were to look into the times outside normal opening hours then we would get the results we see in figure 3.4.

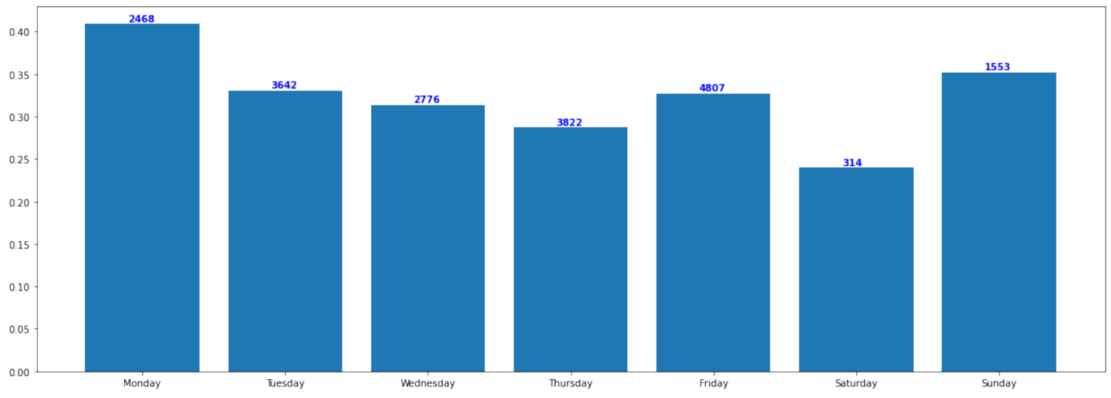

Figure 3.4: Coverage by day of week, outside regular opening hours

From this we conclude that under normal opening hours the best days to show a movie is on Friday, Saturday, and Thursday respectively. Outside the regular opening hours we something different, here Mondays is the best performing and the coverage is higher across the board, while on Saturdays there are rarely screenings. One thing to remark is that even though there tends to be a higher coverage for times outside the norm, this does not mean that increasing these number would be a good idea. The higher coverage might very well be due to the lower amount of screenings.

3.3 Coverage by Hour of Day per Day of Week

Here we take a deeper look into the data by combining hour a day and day of week. We again separate regular opening hours from outside opening hours. We also look into the coverage of the different times for Fridays and Tuesdays.

When looking at the data one day at a time, we noticed that Friday, Thursday and Saturday were the ones that stuck out, while the rest were somewhat similar. The same was observed when looking at each day hour by hour, so for the weekend we use Friday as its representative, and as for the representative of the workdays we will be using Tuesday. The data from Friday's regular opening hours can be seen in figure 3.5 and for Friday outside the regular opening hours in figure 3.6.

On Friday during regular opening hours there is a slight increase at 20:00, but only marginally. While during the off hours we see a bit more of a pattern where 10:00 seems to be significantly higher with an average of 0.45 where the rest lie around 0.27.

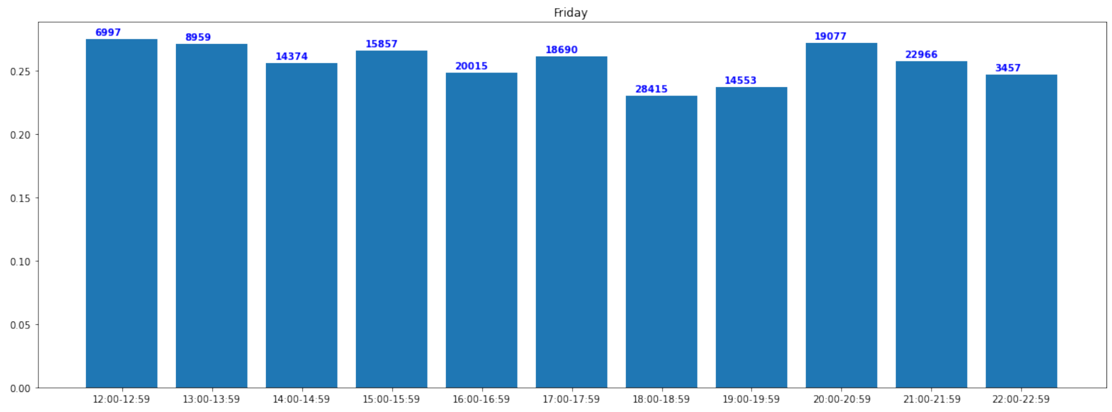

Figure 3.5: Coverage hour by hour for Friday, regular opening hours

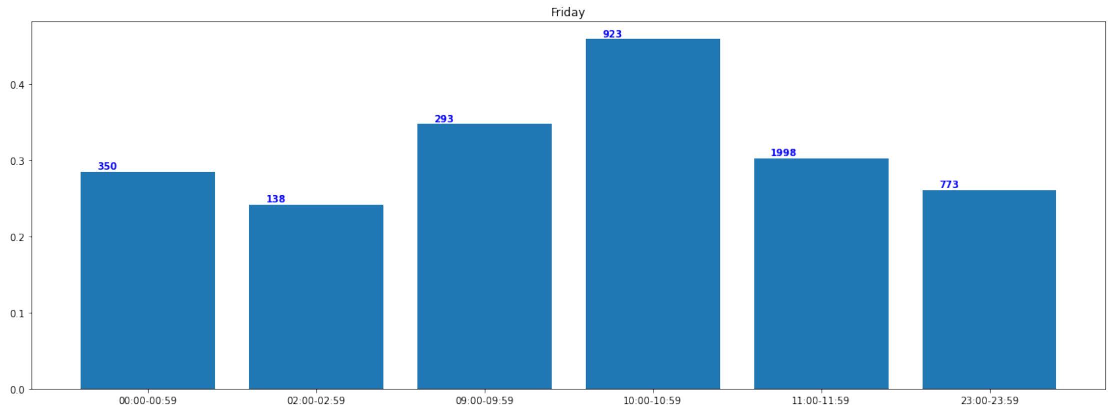

Figure 3.6: Coverage hour by hour for Friday, outside regular opening hours

For Tuesday we see a bit more of a pattern, a peak at 15:00, and it increases from 16:00-20:00 before it start decreasing again, this is during the regular opening hours as seen in figure 3.7. Outside the regular opening hours, figure 3.8, we can see a clear pattern with midnight screenings at the top and it only decreasing from then on.

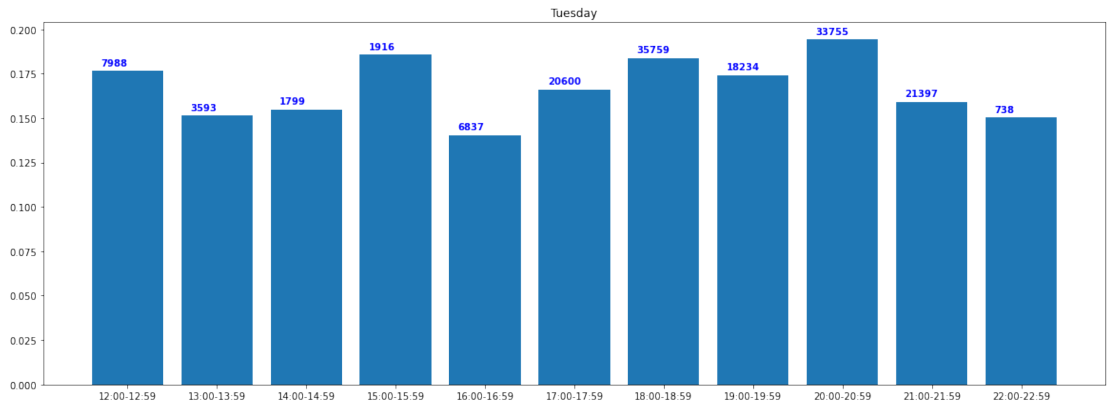

Figure 3.7: Coverage hour by hour for Tuesday, regular opening hours

What we can observe from the screenings outside regular opening hours is the low amount

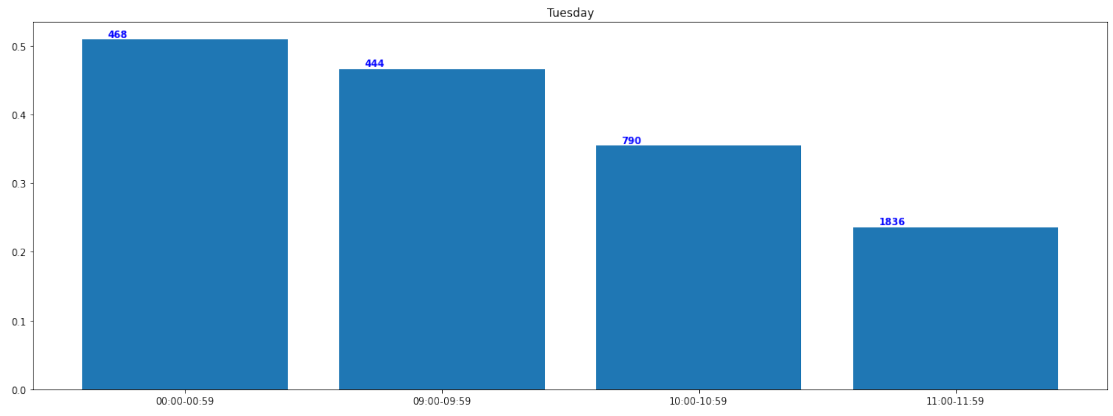

Figure 3.8: Coverage hour by hour for Tuesday, outside regular opening hours

of screenings. In fact there is only 17 000 screenings outside the regular hours, with 1 million screening during. Giving how few screenings there are outside the regular hours we will onward only focus on the regular opening hours.

3.4 Coverage by Genre

When looking at what impact the different genres have on the coverage the only one truly stands out are movies in the War genre. The screenings with movies in the War genre have an average coverage of 0.26 while the closest competitors (Family, Adventure, Fantasy, Animation, History, Music) are at approximately 0.22.

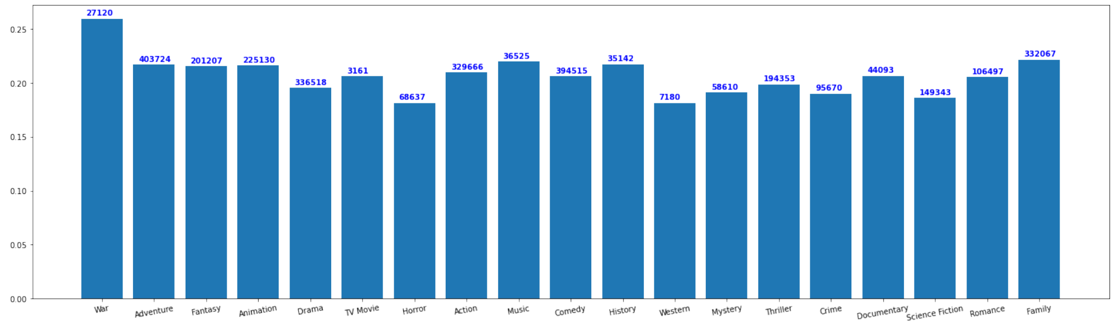

Figure 3.9: Coverage by genre (any genre with less than 5000 screenings have been excluded from the plot)

When looking at the genres during regular opening hours we see the same pattern, with only a slight decrease in the coverage for movies in the Family genre.

To get a better understanding of these differences we look at how the Family genre is at different times of day compared to War which is the overall best performing genre.

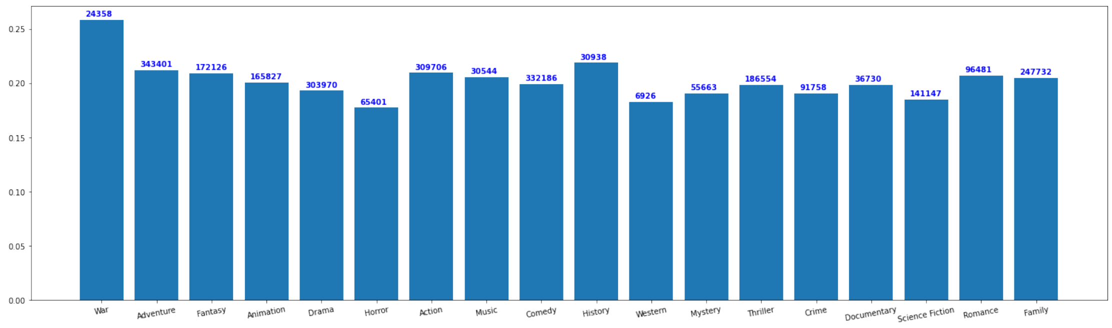

Figure 3.10: Coverage by genre, regular opening hours (any genre with less than 5000 screenings have been excluded from the plot)

There are a few things to note about this comparison. Firstly, we observe that during the period 12:00-15 the family genre outperforms the war genre and in addition the we can clearly see that there are a lot more screenings of family movies than there are of war movies. Secondly, we observe that from 20:00 the number of screenings of war movies are higher than the number of screenings of family movies. From this we conclude that Family movies perform better during the early hours, while the opposite is true for war movies.

If we were to look at the days and not just the time of day we would get the same result as we see here with only some slight variation from day to day, but the pattern remains.

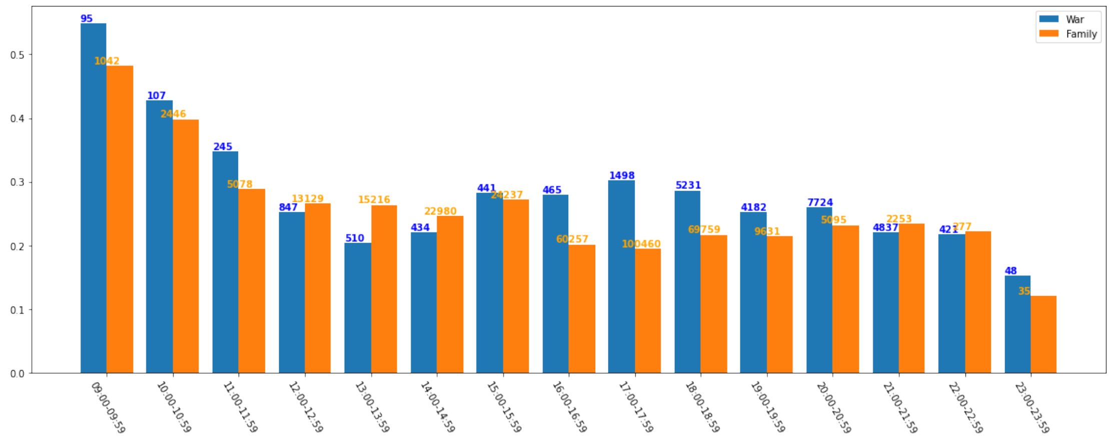

Figure 3.11: War and Family genre comparison

3.5 Coverage by TMDb Score

Next we take a look at the coverage in comparison to the TMDb score, figure 3.12. As we can see from the plot, most of the data is in the region from 5 to 8, anything outside this region are rare occurrences and is considered outliers.

What we can tell from this plot is that there is a slightly positive correlation between the score and the coverage within the region. Given that that this is only true within this region it might be a good idea to use this feature as a categorical feature rather than a continuous feature.

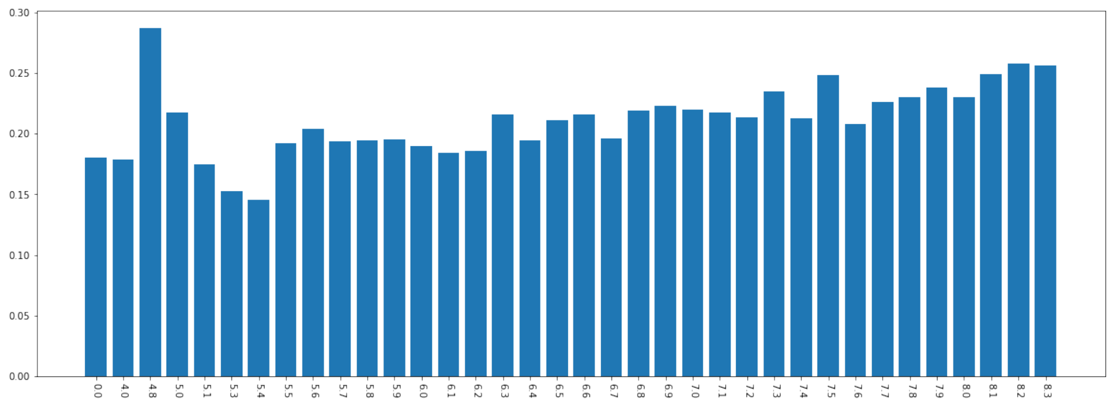

Figure 3.12: Coverage by TMDb score (any score with less than 5000 screenings have been excluded from the plot)

3.6 Coverage by TMDb Popularity

From figure 3.13 we see the effects that popularity has on the coverage. And with the exception of a few outliers we see a positive correlation between the TMDb popularity and the coverage of the screenings.

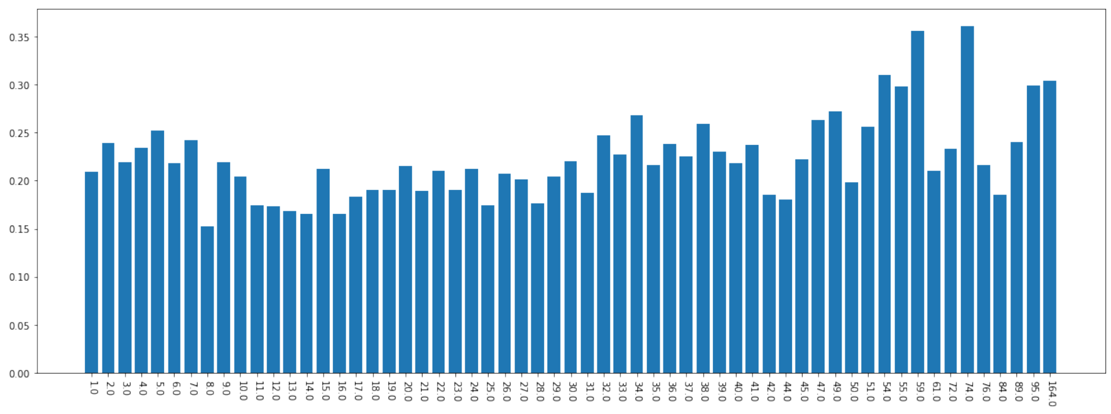

Figure 3.13: Coverage by TMDb popularity (any score with less than 2000 screenings have been excluded from the plot)

3.7 Coverage relative to the Release Date

Our assumption is that the longer a movie has been shown in the theater the lower the coverage will be. We also go by the assumption that the coverage will be significantly higher during the first week. In this section we will explore these aspects and discuss the discoveries made from our observations.

3.7.1 Coverage by Days Since Release

From figure 3.14, we can see a clear pattern, there are mainly two things to note about this plot. First is that there are regular four day spikes, these occur with a seven day interval and start with 0, 1, 2, and 3 days since release, these days correspond most often to Thursday, Friday, Saturday, and Saturday respectively (not a typo). The same holds true for every four day spike we see in the figure. This supports our prior discoveries regarding coverage per day of week with Friday, Saturday, and Thursday being the most popular. Secondly we can see that there is a steady decrease over time, and the spikes tend to flatten over time.

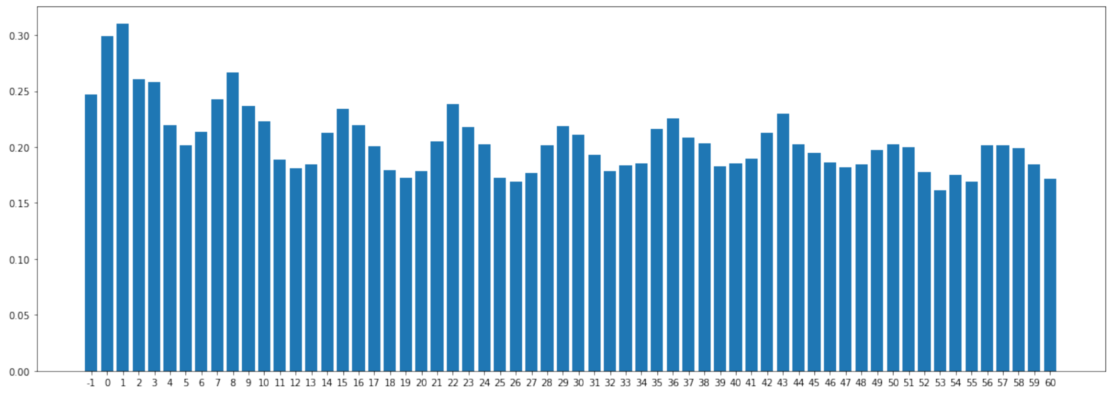

Figure 3.14: Coverage by day days since release (-1 corresponds to any screenings prior to official release)

What this tells us it that screenings will generally have a higher coverage on the opening weekend and that it will decrease with time, but that succeeding weekends will still have a higher coverage than workdays. To note is that the first two weekends are significantly higher than the rest.

3.7.2 First week after release

Here we will reevaluate the stats from the previous sections, but will only look at the subset of the data corresponding to the first ten days after the release. We use the first ten days so as to include the first two weekends which contains the highest performing screenings.

Hour of Day

What we clearly see here, figure 3.15, is that the first week following the release is consistently higher than the other period during regular opening hours. This goes a way to confirm our assumption of the correlation between the days since the release date and the coverage of the screening.

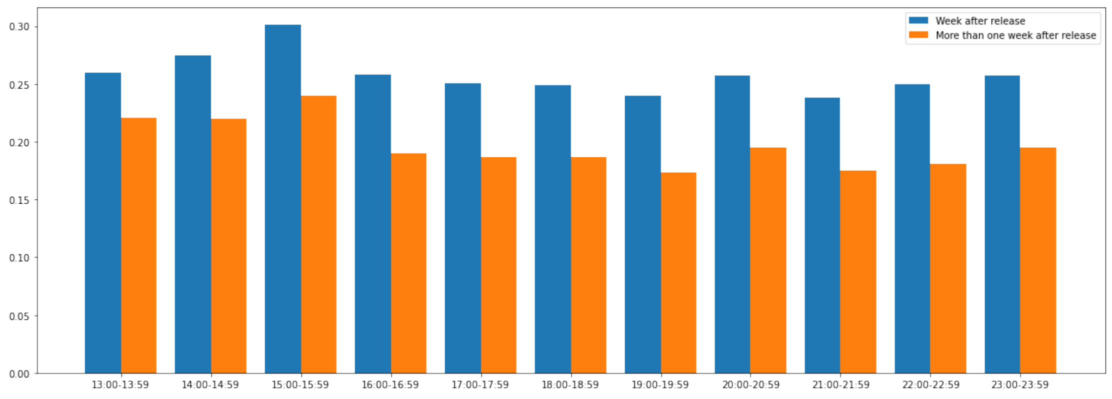

Figure 3.15: Coverage by hour of day, comparing week of release with entire period

Day of Week

Just like with hour by hour, day by day shows the same result that the first week after release performs better on any given day of the week than a screening with a movie that is more than one week after its release. Figure 3.16.

This relationship also holds true for the other features we have gone through in earlier sections. To conclude, the release date has a significant impact on the coverage of a screening.

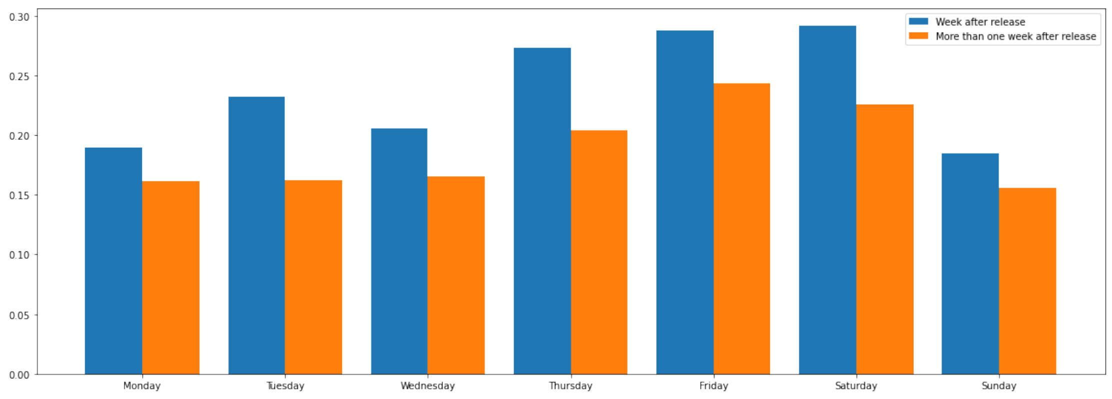

Figure 3.16: Coverage by day of week, comparing week of release with period after first week

Chapter 4

Solution Approach

As a reminder to what our task with this thesis was, we set out to automate the scheduling of screenings for movie theaters. In practical terms this means we want to create a system that can based on prior knowledge automatically suggest a weekly schedule given a list of movies and information regarding the movie theater. To solve this we break down the process into four main parts; preprocessor, analysis, neural network, and scheduler. The preprocessor is there to handle the raw input data and to get additional information needed and make sure it is the correct format accepted by our neural network. The analysis is what we went into detail in chapter 3, and this insight was used for the feature selection for the neural network. The neural network is an MLP model which we use to predict the coverage of new screenings. The scheduler takes the output from the neural network and uses the predictions to generate a suggested schedule for the coming week.

4.1 Baseline

We intend to use a neural network for the scheduler, and to assess how good the neural network performs we will be using a Random Forest model as a baseline to compare our results to. The input for the Random Forest model will be the output of the embedding layer from the neural network to account for the categorical features present in our data and to ensure the same input for both models.

4.2 Proposed Solution

As a solution we propose a pipeline consisting of three components. A preprocessor, a neural network, and a scheduler. A simplified version of our proposed system architecture is shown in figure 4.1.

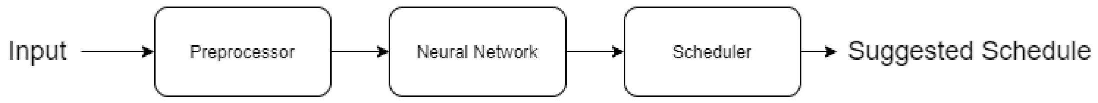

Figure 4.1: Simplified System Architecture

We propose to use a neural network to predict the performance of a movie given the showtime and where it will be shown. We will be using the TabularModel from the fastai library. The scheduler will consist of a series of procedures to ensure the most optimized schedule.

4.2.1 System Architecture

Figure 4.2 shows the complete system architecture first mentioned at the beginning of this chapter. For each movie theater there is some static information, e.g. information regarding the rooms, this can be changed but that would be outside the norm. The user input is the movies that are to be currently shown in that location.

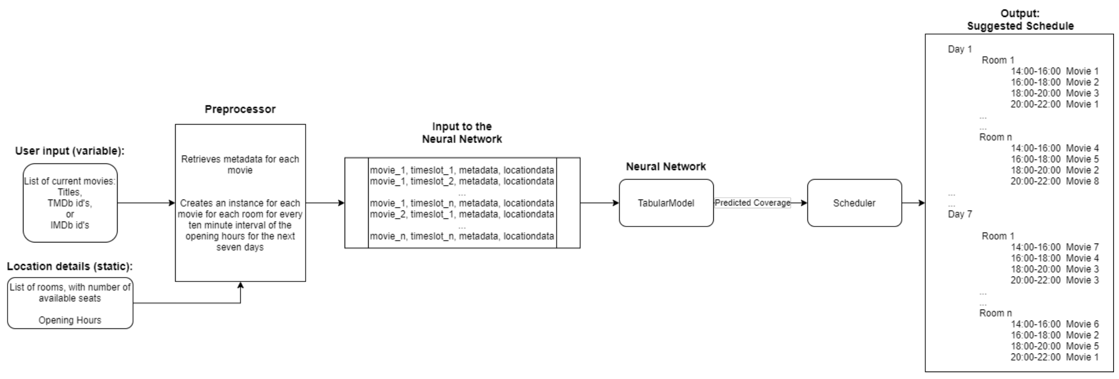
User input (variable):
List of current movies: Titles, TMDb id's, or IMDb id's
Location details (static):
List of rooms, with number of available seats
Opening Hours
Preprocessor
Retrieves metadata for each movie
Creates an instance for each movie for each room for every ten minute interval of the opening hours for the next seven days
Input to the Neural Network
movie_1, timeslot_1, metadata, locationdata
movie_1, timeslot_2, metadata, locationdata
...
movie_1, timeslot_n, metadata, locationdata
movie_2, timeslot_1, metadata, locationdata
...
movie_n, timeslot_n, metadata, locationdata
Neural Network
TabularModel
Predicted Coverage
Scheduler
Output: Suggested Schedule
Day 1
Room 1
14:00-16:00 Movie 1
16:00-18:00 Movie 2
18:00-20:00 Movie 3
20:00-22:00 Movie 1
...
Room n
14:00-16:00 Movie 4
16:00-18:00 Movie 5
18:00-20:00 Movie 2
20:00-22:00 Movie 8
...
Day 7
Room 1
14:00-16:00 Movie 7
16:00-18:00 Movie 4
18:00-20:00 Movie 3
20:00-22:00 Movie 3
...
Room n
14:00-16:00 Movie 6
16:00-18:00 Movie 2
18:00-20:00 Movie 5
20:00-22:00 Movie 1

Figure 4.2: System Architecture

Preprocessor

The preprocessor takes two main inputs; a list of movies that are to be shown, and information regarding the movie theater the movies are to be shown in. The preprocessor retrieves additional information about the movies from TMDb's API and appends it to the list of movies.

It then creates an instance of each movie in each room for each ten minute interval of the movie theater's opening hours. E.g. given a list of 10 movies for a location with 10 rooms with opening hours from 14:00 to 23:59 gives a total of 42 000 instances.

Neural network

As mentioned we will be using the TabularModel from fastai. This model will for each instance sent from the preprocessor predict the coverage.

This information is then sent to the scheduler that does the rest.

Figure 4.3 shows the architecture of the TabularModel we ended up using for this thesis.

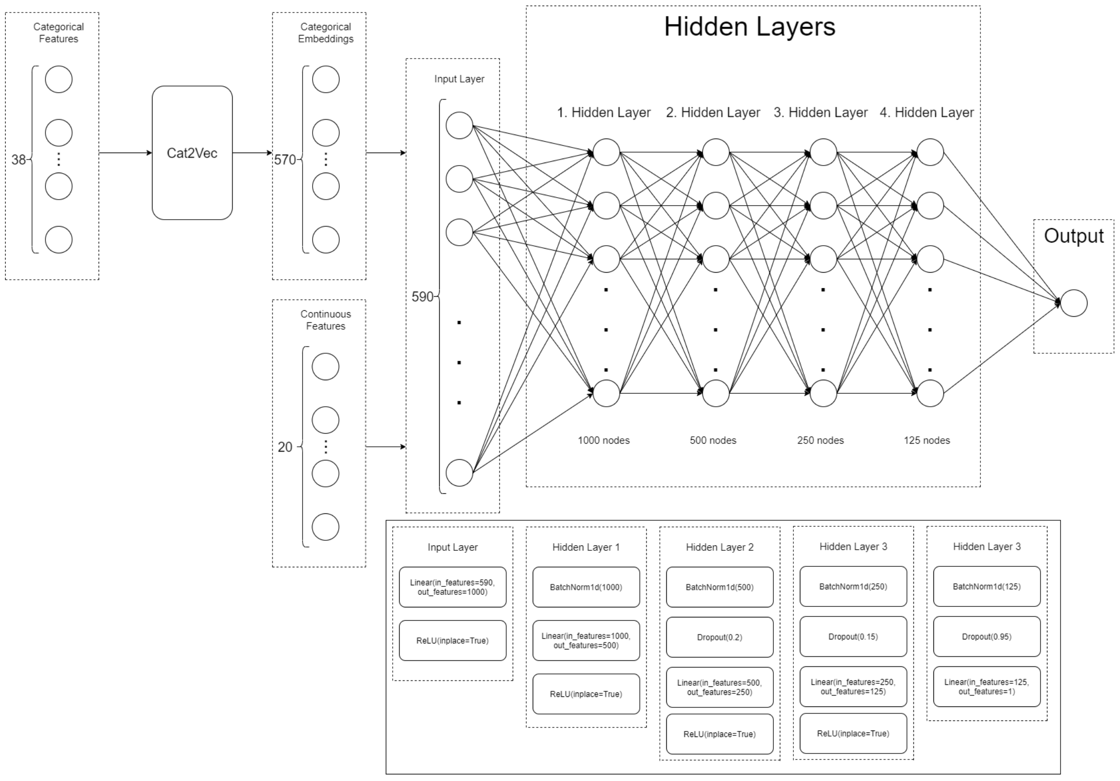

Figure 4.3: TabularModel Architecture

Scheduler

The scheduler will receive predictions from the neural network and based on this information generate a suggested schedule. To do this the scheduler sorts the predictions by coverage and for each day it creates a schedule for each room starting with the screening with the highest coverage and adds the runtime with an additional buffer to allow for commercials/maintenance/cleaning. It goes through each subsequent suggested screening for that room that day and if it does not collide with the screenings already in the schedule then this new screening will be added to the schedule. This is done until the schedule for each room for each day is filled. Finally the suggested schedule is returned to the user and they can then choose to use it or not.

Chapter 5

Experimental Evaluation

This chapter will cover how we implemented the proposed solution and what experimental procedures were done to achieve our results. The chapter will conclude with a summary of the results from our approach and an example of the final output of our scheduling system.

5.1 Experimental Setup and Data Set

For this thesis we use a combination of different datasets. Our main dataset, provided by DX, containing the ticket sales for each cinema screening from a hundred and forty one locations for the past twenty years. This dataset contains the following information for each screening; location, room, seats available, title of the movie, see table 5.1. This is our base dataset, other datasets are used to supplement this with additional information.

id	showtime	title	movie_id	room	seats	sold	coverage
56979	2019-10-14 20:30	Gemini Man (2D)	UIP20180847	3	180	8	0.0444
56981	2019-10-15 17:30	Brillebjørn på feire	SEM20190038	4	70	4	0.0571
56982	2019-10-15 18:00	Den avskylige snømannen (Norsk tale)	UIP20180853	3	180	14	0.0778
56988	2019-10-15 19:30	Wild Rose	NFD20190198	4	70	6	0.0857
56987	2019-10-15 20:30	Joker	SFN20190978	3	180	59	0.3278

Table 5.1: Subset from DX's dataset

In addition to the data from DX we also use data from TMDb, an open source movie database that provides us with additional information about each movie. This information gives us a better insight into the type of the movies. The features extracted form TMDb is explained in table 5.2.

Feature	Details
TMDb ID	TMDb's unique identifier for the movie
IMDb ID	IMDb's unique identifier for the movie
TMDb Title	The title of the movie used by TMDb
Original Title	The original title of the movie in the movies original language
Release Date	The movie's release date
Runtime	The runtime of the movie, in minutes
Genre Ids	A list of the ids of the genres the movie belongs to
Popularity	A numeric representation of the movie's popularity
Vote Average	The average vote score of the movie
Vote Count	The number of votes a movie has received
Overview	A summary of the movie
Tagline	The tagline of the movie, one sentence
Keywords	The keywords for the movie
Original Language	The original language of the movie
Budget	The budget of the movie, in USD
Revenue	The movie's reported revenue
Production Companies	Information regarding the production companies that made the movie
Cast	The cast featured in the movie
Crew	The crew that worked on the movie

Table 5.2: Features extracted from TMDb

The data from TMDb was extracted using their open API [12]. Gathering this information proved to be a challenge given the nature of our dataset from DX. In table 5.1 we see some examples showcasing this challenge, some of the movie titles were in Norwegian and some includes a version such as (2D) and (norsk tale). There is no standard for how these versions are appended to the titles and the removal process of these were generalized which introduced some degree of uncertainty to the accuracy of the titles after removing the version.

Looking back just a few years at the Norwegian standard for movie titles, we see that most foreign titles used to be translated to Norwegian. With the translations of the titles we faced a challenge, some of the Norwegian titles directly corresponds to the English title of a different movie, so when searching through the TMDb API not all titles were correctly identified as demonstrated in table 5.3. Table 5.3 also shows us that if there is uncertainty as to the search string then a list of possible matches is returned, finding the correct one from this list is not a simple task without additional information that can identify the correct movie or manually selecting the right movie. Given that we are searching for tens of thousands of movies, manually checking is not practical. Our method for selecting the movie most probable of being correct is a title match and the difference between the release date of the search result and the showtime of the movie, the movie from the list that we deem most likely will be selected. This process is not flawless as demonstrated by the movie "Frost 2" where none of the search results is the correct one, the movie "Frozen" is in the same franchise, but is years apart and the title is not as similar to the search string as some of the other results. Without access to other databases that may provide us with more accurate data, this inaccuracy is something we have to accept.

Provided title	Correct translation	movies retrieved from TMDb
Nasse Nøff	Piglet's Big Movie	Piglet's Big Movie
Frost 2	Frozen 2	Jack Frost 2 Frost/Nixon Frosty the Snowman Frosty Returns Jack Frost Frostbitten Mister Frost Before the Frost Father Frost The Legend of Frosty the Snowman Frozen Frost
Død Snø	Dead Snow	Dead Snow Dead Snow 2 Red vs. Dead

Table 5.3: Wrong translations

Our dataset contains 1 134 202 screenings over the past twenty years spread across 141 locations. In this time there has been shown a total of 6 546 different movies and a total of 35 750 551 tickets have been sold. 10 962 827 is the number of tickets sold within the first week of release which account for 30.7% of all tickets sold. In the second week 5 552 877 tickets were sold, 4 040 907 in the third, and 2 785 177 in the fourth. By the end of the second week after release 16 515 704 were sold, 20 556 611 by the end of the third, and 23 341 788 by the end of the fourth week following the release. 65.3% of all tickets sold were sold within the first month following the release.

We have split our dataset into three parts, a training, a validation and a test set. The training data contains screenings from 2001 through 2016, the validation data contains screenings from 2017 and 2018, and the test data contains screenings from 2019. The final model will be fitted with the training and validation sets.

In table 5.4 we see how many screenings there has been for each year. Some of these numbers look somewhat conspicuous, 1970, 1990 and 2200 as they are all quite a few years away from the rest and especially the year 2200 which is 180 years in the future and is clearly nothing more than someone making clerical error. In 1970 the movies that were apparently shown were "Star Wars: The Rise of Skywalker" and "Ralph Breaks the Internet" which originally premiered in 2019 and 2018 respectively. And in 1990 they apparently had pre-premiere of the 2018 movie "Jurassic World: Fallen Kingdom". Since these screenings account for a total of eleven screenings out of one million, we made the decision to simply remove these outliers from the dataset as this would not affect the overall dataset in any meaningful manner.

Year	Number of screenings
1970	7
1990	2
2001	35
2002	121
2003	2201
2004	4064
2005	5181
2006	6507
2007	29585
2008	39804
2009	48902
2010	60905
2011	72285
2012	79776
2013	87530
2014	89165
2015	88764
2016	121255
2017	125162
2018	132778
2019	132328
2020	7843
2200	2

Table 5.4: Number of screenings by year

Ignoring the noise in table 5.4 we see that the number of screenings in 2001 and 2002 is significantly lower than the other years. The reason for this is that DX started collecting data at one location in 2001 and for most of 2002 the screenings are from the two rooms in that location, in December of 2002 they expanded to a second location. Unlike the screenings from 1970 and 1990 which are just noise, the low number of screenings from 2002 to 2006 is due to a startup/establishing period for DX where they acquired more clients growing their business and database.

5.1.1 Preprocessing and data aggregation

As we mentioned in the previous section, the titles comes with the version appended, which makes it challenging to find the corresponding movie. So before we retrieve the additional information from TMDb we need strip the version from the title leaving us with a clean title. At time of writing there is yet to be a standard for how the versions are to be formulated and how it should be added to the titles, table 5.5 shows an example of different representations of the same versions (the dataset is in Norwegian and so are the formulations of the versions). Also the placement of the version within the title is not consistent, "Snoopy og Charlie Brown: Knøttene-filmen (2D, norsk tale)", "Star Wars 2D: The Force Awakens" and "Den Gode Dinosaur 3D (norsk tale)" are examples of different placements of the version, all in brackets, some in brackets, in the middle of the title. This causes the filtering process to become more complex resulting in 185 lines of "replace" statements to account for every different representation of each version, the code for that function is shown in code listing A.1.

As previously mentioned, retrieving a movie from TMDb when only having the Norwegian title is not all that reliable. This combined with a filter that is not perfect causes some mismatched movies and in some cases we were not able to find any matching movies, the latter was rare and when not finding any matches the screening was removed from the dataset. This resulted in a few thousand screenings being removed which should not affect the dataset too much.

Finally we combine the information retrieved from TMDb with our dataset from DX, this does cause some redundancy in the dataset but nowadays storage is not really of any concern and it does make the computations quicker.

What it means	Different representations
Original language with Norwegian subtitles	Original tale
	Orig. versjon
	Org. versjon
	Originalversjon
	Tekstet original versjon
	Original tale med norsk tekst
	Tekstet org.versjon
	Tysk tale
	Engelsk tale, norsk tekst
	Tekstet original
	Original tale, norsk tekst
	Eng. tale
	Originalversjon med norske underteksten
Original language without subtitles	Utekstet org. versjon
	Utekstet original tale
	Original tale u/tekst
	Original versjon u/tekst
	Org. tale, utekstet
	Original tale utekstet
Norwegian dub	No. versjon
	Norsk versjon
	Norsk tale
	Norske stemmer
	Norsk tale utekstet
	Utekstet, norsk tale
	Norsk dub
	Dubbet
	Dubbet versjon

Table 5.5: Examples of different representations of the same versions

5.1.2 Feature Selection

In chapter 3 we went through some statistics of the features from our dataset. These stats is some of what we are basing our choice of features moving forward into the neural network.

As we saw earlier on, we have three features that seems to have most of the deciding factor; day of week, hour of day, and number of days since the release of the movie. However, these features are only representative of where and when the movies were shown and not what type of movie were shown, so we need to include features that can represent the differences in the movies that are shown. This is why we combined the data from DX with the additional metadata from TMDb, this provides information enabling us to differentiate between the different movies and combined with the sales data we expect this information to be able to predict the coverage of future screenings.

The feature that help us differentiate when movies were shown is the showtime. However, the format of this feature is a timestamp which does not directly translate well into either categorical or continuous features. To be able to properly utilize this feature we extract the relevant information from the timestamp before using it in our model. From the timestamp we extract the following features:

- Day of month
- Day of year
- Day of week
- Month of year
- Year
- Hour
- Minute
- Whether it is at the end or the beginning of the month
- Whether it is at the end or the beginning of the quarter
- Whether it is at the end or the beginning of the year
- Whether or not it is a holiday
- Sine and Cosine features of:
  - Day of week
  - Day of month
  - Day of year
  - Hour
  - Minute

These features should help us with the time data, and the Sine and Cosine features should help with the periodic nature of the dates and times.

For information regarding where movies are shown the information we have is the number of seats and the room number, we also have a location ID though this is combined with the room number. When it comes to the room number they are generally number from one through the number of rooms the movie theater has, meaning that nearly all movie theaters has a room with room number one, and there is no correlation between room number n for location x and room number n for location y. Because of this we append the ID of the locations to the room number to make each room/location combination unique.

Our movie related features are the genres, the budget, runtime, popularity and rating of the movies that are being screened.

From our combined dataset we also calculate the difference between the screening and the date of release, as we saw in chapter 3.7.1 that this features is correlated with the coverage and we saw that especially the opening weekend were higher than the rest, and that every subsequent weekend had a lower coverage that the opening weekend, but still a higher coverage than the workdays.

Table 5.6 gives an overview over the features we chose to work with and whether we treat it as a continuous or categorical feature.

Embedding of categorical features

So far, deep learning models does not natively support categorical features and so for us to able to include these features in our model we need to use a form of feature embedding to convert each categorical feature into a continuous representation that the model can work with. There are several different types of embeddings that can be used, and each has its own pro's and con's, these were discussed in chapter 2. We will be using the feature embedding technique of Cat2Vec, this embedding method is built-in to fastai's TabularModel and will automatically be applied to each feature defined as categorical.

Cat2Vec will convert the categorical feature to a series of continuous vectors. Unless explicitly specified, the number of vectors is defined as the lower value: 600 or $1.6 *$

Categorical Features	room
	location
	genres
	days since release
	year
	month
	week
	day
	hour
	minute
	day of week
	day of year
	is month end
	is month start
	is quarter end
	is quarter start
	is year end
	is year start
	is holiday
Continuous Features	seats
	tmdb popularity
	tmdb vote average
	runtime
	budget
	weekday cosine
	weekday sine
	day of month cosine
	day of month sine
	day of year cosine
	day of year sine
	hour cosine
	hour sine
	clock cosine
	clock sine
	minute cosine
	minute sine

Table 5.6: Features chosen

$num\_categories^{0.56}$. E.g. one of our categorical features, $dayofweek$, has 7 unique values, this results in 5 embedding vectors, another categorical feature, *room*, has 333 unique categories resulting in 41 embedding vectors.

For the genre feature we faced a bit of a problem, where the other categorical features only belong to one category at a time, a movie can belong to multiple categories (genres) it is in fact abnormal for a movie to be contained within a single category. This data is then given to us a list of categories each movie belongs to, so far Cat2Vec does not support such input and therefore we will be using one-hot encoding for this feature. Unlike when we use Cat2Vec, with one-hot we need to preprocess the data before we send it to the model, the on-hot encoding results in a set of boolean vectors each declared as a categorical feature for the input of the TabularModel.

5.1.3 Random Forest

We used a Random Forest model to get a baseline performance to compare our model to. Given the difference in complexity seeing how a Random Forest model compares to neural network was quite interesting. If the neural net is not capable of getting a better result than the Random Forest, then it might be preferable to go with the less complex model.

To ensure comparability we use the same embeddings for the Random Forest model as for the TabularModel. To achieve this we started with the embedding code from fastai and made some modifications to allow for it to be used as the input of the Random Forest model.

We are using a Random Forest model with 10 trees, each with a max-depth of two, $max\_features = \sqrt{num\_features}$ and mean absolute error (MAE) as the criterion. Our reasoning for these hyperparameters were mostly due to time/hardware limitations. With the hyperparameters we chose the model trained in 13 hours, we tried a few times to train a model with 100 and 1 000 trees, and in both cases our computer got the all too familiar *blue screen* error after running for several days, it also killed one of the RAM chips on our computer. We tried a couple of different hyperparameter values for the depth, but given the time to train each model and that this is a baseline model and not the final model we chose the best performing hyperparameters our limited search yielded.

5.1.4 TabularModel

For this project we chose the TabularModel from fastai as the model to be used for our solution. This model is especially designed for dataframe like data and has built-in embeddings for categorical features. This model uses mean squared error (MSE) as the loss function and the ADAM optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.99$ as default values.

This model is what we call our default model. To improve these results we go through a tuning process were we try to find better values for the hyperparameters used by the TabularModel's optimizer.

Hyperparameter Tuning

Our default model uses the default values and it gets decent result, but the test MAE is much higher than the train and validation MAE which tells us our model has room for improvement and that it is possibly overfitted to the training data. We have then gone through a tuning process to find a better combination of values other than the default out-of-the-box values.

There is no standardized way to tune the hyperparameters of a neural network and this is generally done one of two ways, using extensive prior knowledge of the data and the workings of the optimizer and loss function and then making an educated guess as to what would probably work best, or an extensive search of each combination of hyperparameter values. The former, as mentioned, requires an extensive prior knowledge of the data and which optimizer and loss function would work best and what the parameter values should be, in addition this method would only give a starting point and some clues as to the hyperparameter search space. We have not found any other project working on a similar problem with a similar dataset and since neural networks are not one-model-fits-all we will go for the second method of searching the hyperparameter space as extensively as time allows. However, even for a single hyperparameter this is infeasible if the search space is not limited, e.g. learning rate (LR) can have any positive value but checking every value results in an infinite amount of possible values, but using best practises developed for years we in this case will limit the search space to values from 0 to 1. Even limiting this search space to values between 0 and 1 would still give us an infinite amount of possible values so we need to further limit the search space, we do this by setting regular increments so that we only have a finite number of values to test. Table 5.7 shows which hyperparameters we tuned and the search space and increment for each of them.

Given the search space and increments for the hyperparameters above we would need to run 5.33 quadrillion tests ($5.33 * 10^{15}$). We had one machine running three test concurrently, each test taking roughly twenty minutes. With the limited processing power we would need $6.76 * 10^{10}$ years to run every test, and given the time left before

Parameter	Default values	Search-space	Increments
Learning Rate (LR)	1e-3	1e-10 $\rightarrow$ 1e-1	by times 10 (1e-7, 1e-6, 1e-5, etc.)
β_1	0.90	$0 \rightarrow 1$	by 0.05
β_2	0.99	$0 \rightarrow 1$	by 0.05
ϵ	1e-8	1e-10 $\rightarrow$ 1e+5	by times 10
Weight Decay (WD)	0.01	$0 \rightarrow 1$	by 0.05
Embedding Dropout (ED)	0	$0 \rightarrow 1$	by 0.05
Layer Dropout (4 layers)	0 (for each layer)	$0 \rightarrow 1$	by 0.05
Momentum	(0.95, 0.85)	$0 \rightarrow 1$ (for each)	by 0.05
AMSGrad	False	True/False	

Table 5.7: Search space for the TabularModel's hyperparameters

delivery of this thesis (or even our lifetime) that would not be feasible. To get around this limitation we have to test only a subset of the hyperparameter combinations, choosing this subset is an important choice which can greatly impact the final results.

We start our tuning process by testing all the values in the search space for the LR, we choose the value that got us the best results and this will the LR we will be using when tuning the other hyperparameters. We use this logic for the rest of the hyperparameters, testing all values of a given hyperparameter choosing the one that got the best result and moving on to the next. After testing different values of all the hyperparameters we select the top ten results.

We ran a total of 221 tests, the results of all these can be found in appendix B, our overall top ten results can be seen in table 5.8. Each of the top ten tests were ran ten times, and the MAE shown in the table is the average of those ten runs, the same applies to the default model. Probably the most interesting part of the table is that for the Random Forest model the training error is the highest while the test error is lowest, this is not results you would often see, and is most probably due to the limitations mentioned regarding the training process of the Random Forest baseline model.

	epochs	LR	(β_1, β_2)	ϵ	WD	ED	Layer Dropout	momentum	Train MAE	Valid MAE	Test MAE
Random Forest									0.1526	0.1400	0.1340
Default	5	1e-3	(0.9, 0.99)	1e-8	0.01	0	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0718	0.1272	0.1820
Test 180	8	1e-3	(0.99, 0.99)	1e-3	0.15	0.15	(0.00, 0.20, 0.15, 0.95)	(0.95, 0.85)	0.0806	0.1183	0.1448
Test 159	8	1e-3	(0.99, 0.99)	1e-3	0.15	0.15	(0.00, 0.20, 0.90, 0.00)	(0.95, 0.85)	0.0822	0.1192	0.1451
Test 125	8	1e-3	(0.99, 0.99)	1e-3	0.15	0.15	(0.00, 0.20, 0.00, 0.00)	(0.95, 0.85)	0.0778	0.1195	0.1454
Test 195	8	1e-3	(0.99, 0.99)	1e-3	0.15	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.85)	0.0787	0.1196	0.1452
Test 185	8	1e-3	(0.99, 0.99)	1e-3	0.15	0.15	(0.00, 0.20, 0.15, 0.00)	(0.15, 0.85)	0.0788	0.1196	0.1465
Test 105	8	1e-3	(0.99, 0.99)	1e-3	0.15	0.15	(0.20, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0806	0.1200	0.1448
Test 144	8	1e-3	(0.99, 0.99)	1e-3	0.15	0.15	(0.00, 0.20, 0.15, 0.00)	(0.95, 0.85)	0.0786	0.1200	0.1465
Test 202	8	1e-3	(0.99, 0.99)	1e-3	0.15	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.00)	0.0787	0.1200	0.1457
Test 84	8	1e-3	(0.99, 0.99)	1e-3	0.15	0.15	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0765	0.1203	0.1446
Test 219	8	1e-3	(0.99, 0.99)	1e-3	0.15	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.85)	0.	0.	0.

Table 5.8: Top ten after Hyperparameter Tuning

The top ten results in table 5.8 are sorted by validation MAE and test MAE if the validation MAE between two are a tie. From the table we can see that our top ten as well as the default model has all beaten the baseline model with the exception of the baseline test MAE. When comparing our top ten with the default model we see that our tuning process has managed to find several hyperparameter combinations that outperforms the default settings. As we can see tuning the hyperparameters resulted favorably and we at least found a local optima, given how we limited the hyperparameter search space were there is no guarantee that the best combination we found is the global optima, but it at the very least gives us results that are a significant improvement over the default values.

The Final Model

For our final model we chose the model from the top ten that had the best validation/test MAE combination, test 180. This model had a test MAE higher than the baseline model, but significantly lower than the default model and our chosen model got a better validation MAE than both the default and the baseline. We conclude that test 180 is the best performing model of our tests, improving on the default model showing the importance of tuning the hyperparameters.

5.2 Experimental Results

With our final model selected, we now combine this with our preprocessor and our scheduler to make a suggestion for an optimized schedule. For comparison we use a schedule from one of the locations for a week in September of 2019 (16.-22. of September) and use this as a baseline against our optimized schedule. The input to our scheduler will be the movies that were shown that week in that location. We will consider 14:00-22:50 as the regular opening hours and our suggested schedule will only suggest screening in this period, so we will only compare our suggested schedule with the original schedule from that same period.

For this final test we train the model on both training and validation sets, screenings from the inclusive interval 2001-2018. The resulting schedule can be seen in table 5.9.

There were 19 movies that had been screened in that location that week, our suggested schedule only included 5 of these. The original schedule had 145 screenings that week, while our suggestion had 176, 115 of these were the same movie. From this we can see that even though our TabularModel is trained to predict the coverage, basing the schedule on only this aspect may lead to an undesirable solutions. Some additional heuristics that helps the scheduler not pick only one movie to screen all the time would probably be beneficial.

We tested several weeks for several locations and all tests revealed the same, that one movie ended up being suggested for most of the screenings. Even if this would be the movie that would get the highest coverage, screening it all the time might very well have the opposite effect.

Monday, 16.09.2019						
	Room 1		Room 2		Room 3	
Time	Actual	Suggested	Actual	Suggested	Actual	Suggested
14:00		Downton Abbey	Downton Abbey		Downton Abbey	Downton Abbey
15:00				The Ash Lad		
16:00	Hobbs & Shaw					
17:00		The Ash Lad (Askeladden)	Downton Abbey	The Ash Lad	Toy Story 4	The Ash Lad
18:00						
19:00	Toy Story 4	The Ash Lad (Askeladden)			It Chapter Two	The Ash Lad
20:00			Downton Abbey	Orps: the movie		
21:00	Scary Stories to tell in the dark					
22:00		Orps: The Movie		Orps: the movie		Orps: The Movie
Tuesday, 17.09.2019						
	Room 1		Room 2		Room 3	
Time	Actual	Suggested	Actual	Suggested	Actual	Suggested
14:00		The Ash Lad		The Ash Lad		The Ash Lad
15:00						
16:00				The Ash Lad		The Ash Lad
17:00		The Ash Lad	Downton Abbey		Toy Story 4	
18:00				The Ash Lad		The Ash Lad
19:00	Toy Story 4	The Ash Lad			It Chapter Two	
20:00			Downton Abbey			Orps: The Movie
21:00	Scary Stories to Tell in the Dark					
22:00		Orps: The Movie		Orps: the movie		Orps: The Movie
Friday, 20.09.2019						
	Room 1		Room 2		Room 3	
Time	Actual	Suggested	Actual	Suggested	Actual	Suggested
14:00	Hobbs & Shaw		Toy Story 4		The Ash Lad	
15:00		The Ash Lad		The Ash Lad		It Chapter Two
16:00	Good Boys		Toy Story 4		The Ash Lad	
17:00						
18:00	Good Boys	The Ash Lad	The Ash Lad	The Ash Lad	It Chapter Two	The Ash Lad
19:00						
20:00	Toy Story 4		Ad Astra			Orps: The Movie
21:00		Orps: The Movie		Orps: the movie	It Chapter Two	
22:00	Angel Has Fallen					Tpy Story 4
23:00			Rambo: Last Blood			
Saturday, 21.09.2019						
	Room 1		Room 2		Room 3	
Time	Actual	Suggested	Actual	Suggested	Actual	Suggested
14:00		The Ash Lad	Toy Story 4			The Ash Lad
15:00	Beware of Children			The Ash Lad	The Ash Lad	
16:00	Hobbs & Shaw	The Ash Lad	Toy Story 4			The Ash Lad
17:00				The Ash Lad	The Ash Lad	
18:00		The Ash Lad	Downton Abbey			The Ash Lad
19:00	Dora and the Lost City of Gold			The Ash Lad	Rambo: Last Blood	
20:00		The Ash Lad	Ad Astra			The Ash Lad
21:00	Angel Has Fallen			Orps: The Movie	Rambo: Last Blood	
22:00		Orps: The Movie				Toy Story 4

Table 5.9: Actual schedule vs Suggested schedule

Chapter 6

Discussion & Conclusion

From our findings during this period we see great potential for this to be further researched and developed into a working product for the movie theater industry. There is still work to be done to improve the accuracy of the model and the scheduler needs a few extra procedures, and you will find some suggestions to this in chapter 7.

We might get better results with a better embedding for genre as one-hot is somewhat inefficient, and that would also allow us to incorporate the data for the cast and crew as well which is of the same format, but with a much higher cardinality.

As we saw in chapter 5 our scheduler predicted a very different schedule than the already existing one, whether our optimized version would be better there is no certain way of knowing without actually testing how it would turn out. Testing whether our scheduler is actually better than the manual is not scientifically possible as we would need to test both scenarios under the same conditions and since datetime plays an important part and unless someone invents time-travel, we can not recreate the same conditions and a true comparison is not possible. The goal of this scheduler is to try an get even a slight overall improvement, given the irregularities we have seen in the data even if this scheduler is implemented and tested for a year and we see an increase or decrease in overall coverage there is no way of knowing if this scheduler is the cause or if there is some other underlying cause of this, maybe the movies that year were simply better and more people wanted to see them, or the opposite could be just as likely, maybe the general interest in going to see a movie at a movie theater simply have gone. We might eventually find a correlation, but there is no way to prove the causation from the data alone, it would be somewhat like predicting the stock market, there are too many variables to account for that are not easily measurable.

Overall we are happy with the results we have made and the final product delivered.

Chapter 7

Future Directions

An idea for further development would be to treat the text features, such as overview, with a natural language model separately before adding that to the final model. And to investigate using a combined model, a model consisting of several neural networks. One could use a separate neural network for the location only data, one for the movie only, where the movie only data could be a combination of a network that has the categorical and the continuous feature with another that treats the text fields. Separating the models in this manner and combine the output from each of these might prove better than our model, it would at the very least be interesting to see the results from such an approach.

Exploring other forms of embedding would likely greatly alter the results and more custom embedding methods might reveal a clearer pattern in the data which could be more optimal for a neural network.

As we have mentioned, the scheduler has a tendency of suggesting a single movie most of the time. This is undesirable behaviour and for further work it would be a good idea to implement some extra procedures to improve the how the schedule handles the predictions from the neural network.

List of Figures

List of Tables

Appendix A

Code

GitHub repository with source code for this thesis:

https://github.com/MPettersen/Deep-Learning-over-20-Years-of-Cinema-Ticket-Sales

```python
def remove_version(txt):
  return (
    txt.strip()
    .replace('(arabisk/bosnisk/engelsk/tysk/kurdisk/svensk tale, norsk tekst)', '')
    .replace('25th anniversary show from royal albert hall - direkte ove', '')
    .replace('ekstraforestilling fred. 09.sept. kl. 22.00.', '')
    .replace('25th anniversary show from royal albert hall', '')
    .replace('ekstraforestilling sund. 11.sept. kl.20.00.', '')
    .replace('daybreakers(utekstet originalversjon)', '')
    .replace('norsk actionthriller. alder 15 r.', '')
    .replace('50th anniversary celebration atmos', '')
    .replace('(nb! filmen er tatt over av fox)', '')
    .replace('(nb! gratis! - kun kjp i dra)', '')
    .replace('premire!  norsk.  alder: 9 r.', '')
    .replace('(3d - briller kjpes i kiosken)', '')
    .replace('gratis - ingen forhndsbest.', '')
    .replace('briller (25,kjpes i kiosk)', '')
    .replace('(2\x1f\x1f\x1fd,norsk tale)', '')
    .replace('(nyrestaurert/digitalisert)', '')
    .replace('2011 jubileumsforestilling', '')
    .replace('i3d(orginal) full rulle', '')
    .replace('(mnedens joker nr. 2!)', '')
    .replace('(festivalpass gjelder)', '')
    .replace('(nb! eneste visning!)', '')
    .replace('med regissrbesk', '')
    .replace('stord filmklubb', '')
    .replace(':knttenefilmen', '')
    .replace('(b filmklubb)', '')
    .replace('10-rsjubileum', '')
    .replace('(gammel reg.)', '')
    .replace('stor sthai)', '')
    .replace('halvmaraton', '')
    .replace('direkte ove', '')
    .replace('(1978)', '')
    .replace('83d)', '')
    .replace('(n)', '')
    .replace('org.versj stor sthai', '')
    .replace('ekstraforestilling', '')
    .replace('og valgfritt tale', '')
    .replace('ikke billettsalg', '')
    .replace('30-rs jubileum', '')
    .replace('direkteoverfrt', '')
    .replace('forest. kl.1400', '')
    .replace('originalversjon', '')
    .replace('verdenspremiere', '')
    .replace('forf. innleder', '')
    .replace('i atmos 3d lyd', '')
    .replace('ikke book tale', '')
    .replace('mnedens joker', '')
    .replace('norgespremiere', '')
    .replace('orginalversjon', '')
    .replace('usikker lengde', '')
    .replace('barnehagekino', '')
    .replace('gratisvisning', '')
    .replace('norgespremier', '')
    .replace('originalsprk', '')
    .replace('forestilling', '')
    .replace('dolby atmos', '')
    .replace('frpremiere', '')
    .replace('horrornight', '')
    .replace('ingen tekst', '')
    .replace('nederlandsk', '')
    .replace('originalutg', '')
    .replace('portugisisk', '')
    .replace('sing- along', '')
    .replace('sing-a-long', '')
    .replace('strikkekino', '')
    .replace('.vises i 3d', '')
    .replace('gratiskino', '')
    .replace('indonesisk', '')
    .replace('norgesprem', '')
    .replace('uten tekst', '')
    .replace('seniorkino', '')
    .replace('sing-along', '')
    .replace('bollywood', '')
    .replace('eng tekst', '')
    .replace('filmdager', '')
    .replace('filmklubb', '')
    .replace('italiensk', '')
    .replace('med norsk', '')
    .replace('med tekst', '')
    .replace('nei tekst', '')
    .replace('subtitles', '')
    .replace('trehundre', '')
    .replace('utektstet', '')
    .replace('babykino', '')
    .replace('extended', '')
    .replace('islandsk', '')
    .replace('litauisk', '')
    .replace('original', '')
    .replace('premiere', '')
    .replace('utekstet', '')
    .replace('17. mai', '')
    .replace('blueray', '')
    .replace('digital', '')
    .replace('engelsk', '')
    .replace('forest.', '')
    .replace('japansk', '')
    .replace('med nor', '')
    .replace('n-prem.', '')
    .replace('preview', '')
    .replace('reprise', '')
    .replace('stemmer', '')
    .replace('tekster', '')
    .replace('tekstet', '')
    .replace('u/tekst', '')
    .replace('versjon', '')
    .replace('version', '')
    .replace('2\x1fd', '')
    .replace('3\x1fd', '')
    .replace('17 mai', '')
    .replace('dubbet', '')
    .replace('fransk', '')
    .replace('gratis', '')
    .replace('m. eng', '')
    .replace('m/ no.', '')
    .replace('n.tale', '')
    .replace('n-prem', '')
    .replace('norske', '')
    .replace('polish', '')
    .replace('polske', '')
    .replace('samisk', '')
    .replace('spansk', '')
    .replace('tale-a', '')
    .replace('teksta', '')
    .replace('35 mm', '')
    .replace('atmos', '')
    .replace('dansk', '')
    .replace('dubba', '')
    .replace('hindi', '')
    .replace('m .no', '')
    .replace('norsk', '')
    .replace('og 2d', '')
    .replace('og 3d', '')
    .replace('polsk', '')
    .replace('tekst', '')
    .replace('versj', '')
    .replace('35mm', '')
    .replace('dig.', '')
    .replace('dubb', '')
    .replace('eng.', '')
    .replace('film', '')
    .replace('i 2d', '')
    .replace('i 3d', '')
    .replace('imax', '')
    .replace('m.no', '')
    .replace('org.', '')
    .replace('orig', '')
    .replace('tale', '')
    .replace('tysk', '')
    .replace('); r', '')
    .replace('2 d', '')
    .replace('3 d', '')
    .replace('4dx', '')
    .replace('5.1', '')
    .replace('dub', '')
    .replace('dvd', '')
    .replace('hfr', '')
    .replace('no.', '')
    .replace('org', '')
    .replace('txt', '')
    .replace('2d, no', '')
    .replace('3d, no', '')
    .replace('2d, n', '')
    .replace('3d, n', '')
    .replace('2d no', '')
    .replace('3d no', '')
    .replace('2d', '')
    .replace('3d', '')
    .replace('4k', '')
    .replace('m/', '')
    .replace('u/', '')
    .replace('*', '')
    .replace(’!’, ’’)
    .replace(’+’, ’’)
    .replace(’.’, ’ ’)
    .replace(’/’, ’ ’)
    .replace(’-’, ’ ’)
    .replace(’"’, ’’)
    .replace(’:’, ’ ’)
    .replace(’;’, ’ ’)
    .replace(’,’, ’ ’)
    .replace(’(’, ’’)
    .replace(’)’, ’’)
    .replace("’", ’’)
    .replace(’  ’, ’ ’)
    .strip())
```

Listing A.1: Code for removing the versions from the titles (a problem with listings and Latex causes ÆØÅ to be removed)

Appendix B

Tuning Results

	LR	(β_1, β_2)	ϵ	WD	AMSgrad	ED	Layer Dropout	momentum	Train MAE	Valid MAE	Test MAE
Random Forest									0.1526	0.1400	0.1340
Default	1e-3	(0.9, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0718	0.1272	0.1820
Test 1	1e-10	(0.90, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.3874	0.3872	0.3393
Test 2	1e-9	(0.90, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.3477	0.3534	0.3537
Test 3	1e-8	(0.90, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.3343	0.3423	0.3406
Test 4	1e-7	(0.90, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.3402	0.3540	0.3513
Test 5	1e-6	(0.90, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.2978	0.3093	0.3107
Test 6	1e-5	(0.90, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.1163	0.1389	0.1717
Test 7	1e-4	(0.90, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0809	0.1289	0.1965
Test 8	1e-3	(0.90, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0718	0.1272	0.1820
Test 9	1e-2	(0.90, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0776	0.1273	0.1993
Test 10	1e-1	(0.90, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0898	0.1346	0.1851
Test 11	1e-3	(0.00, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0738	0.1324	0.2014
Test 12	1e-3	(0.10, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0723	0.1264	0.1830
Test 13	1e-3	(0.20, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0722	0.1268	0.1786
Test 14	1e-3	(0.30, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0724	0.1253	0.2220
Test 15	1e-3	(0.40, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0716	0.1245	0.1812
Test 16	1e-3	(0.50, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0714	0.1277	0.1820
Test 17	1e-3	(0.60, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0716	0.1270	0.1885
Test 18	1e-3	(0.70, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0724	0.1283	0.1984
Test 19	1e-3	(0.80, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0722	0.1264	0.2067
Test 20	1e-3	(0.99, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0715	0.1270	0.1854
Test 21	1e-3	(0.00, 0.00)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.2261	0.2071	0.1967
Test 22	1e-3	(0.00, 0.10)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0825	0.1278	0.2605
Test 23	1e-3	(0.00, 0.20)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0939	0.1351	0.2862
Test 24	1e-3	(0.00, 0.30)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0801	0.1399	0.3047
Test 25	1e-3	(0.00, 0.40)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0791	0.1356	0.3575
Test 26	1e-3	(0.00, 0.50)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0790	0.1391	0.2760
Test 27	1e-3	(0.00, 0.60)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0783	0.1568	0.3376
Test 28	1e-3	(0.00, 0.70)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0763	0.1514	0.3275
Test 29	1e-3	(0.00, 0.80)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0763	0.1514	0.3275
Test 30	1e-3	(0.00, 0.90)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0741	0.1492	0.2445
Test 31	1e-3	(0.99, 0.00)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.7933	0.8125	0.8313
Test 32	1e-3	(0.99, 0.10)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.7934	0.8101	0.8200
Test 33	1e-3	(0.99, 0.20)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0789	0.1344	0.2716
Test 34	1e-3	(0.99, 0.30)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0800	0.1313	0.3432
Test 35	1e-3	(0.99, 0.40)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0783	0.1338	0.2709
Test 36	1e-3	(0.99, 0.50)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0792	0.1514	0.3122
Test 37	1e-3	(0.99, 0.60)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0785	0.1479	0.2691
Test 38	1e-3	(0.99, 0.70)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0758	0.1445	0.2437
Test 39	1e-3	(0.99, 0.80)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0762	0.1610	0.2556
Test 40	1e-3	(0.99, 0.90)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0740	0.1450	0.2291
Test 41	1e-3	(0.99, 0.99)	1e-10	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0723	0.1290	0.1847
Test 42	1e-3	(0.99, 0.99)	1e-9	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0718	0.1261	0.1743
Test 43	1e-3	(0.99, 0.99)	1e-7	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0719	0.1281	0.2084
Test 44	1e-3	(0.99, 0.99)	1e-6	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0720	0.1275	0.1960
Test 45	1e-3	(0.99, 0.99)	1e-5	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0724	0.1265	0.1827
Test 46	1e-3	(0.99, 0.99)	1e-4	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0747	0.1249	0.1739
Test 47	1e-3	(0.99, 0.99)	1e-3	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0805	0.1278	0.1519
Test 48	1e-3	(0.99, 0.99)	1e-2	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0934	0.1325	0.1507
Test 49	1e-3	(0.99, 0.99)	1e-1	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.1131	0.1380	0.1497
Test 50	1e-3	(0.99, 0.99)	1e-0	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.1702	0.1732	0.1694

Table B.1: Hyperparameter Tuning, test 1-50

	LR	(β_1, β_2)	ϵ	WD	AMSgrad	ED	Layer Dropout	momentum	Train MAE	Valid MAE	Test MAE
Random Forest									0.1526	0.1400	0.1340
Default	1e-3	(0.9, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0718	0.1272	0.1820
Test 51	1e-3	(0.99, 0.99)	1e+1	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.3001	0.3118	0.3083
Test 52	1e-3	(0.99, 0.99)	1e+2	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.3341	0.3447	0.3438
Test 53	1e-3	(0.99, 0.99)	1e+3	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.3402	0.3446	0.3504
Test 54	1e-3	(0.99, 0.99)	1e+4	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.3368	0.3420	0.3452
Test 55	1e-3	(0.99, 0.99)	1e+5	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.3576	0.3604	0.3698
Test 56	1e-3	(0.99, 0.99)	1e-3	0.00	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0817	0.1286	0.1607
Test 57	1e-3	(0.99, 0.99)	1e-3	0.05	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0784	0.1272	0.1545
Test 58	1e-3	(0.99, 0.99)	1e-3	0.10	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0764	0.1232	0.1536
Test 59	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0771	0.1217	0.1480
Test 60	1e-3	(0.99, 0.99)	1e-3	0.20	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0785	0.1216	0.1493
Test 61	1e-3	(0.99, 0.99)	1e-3	0.25	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0813	0.1247	0.1574
Test 62	1e-3	(0.99, 0.99)	1e-3	0.30	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0831	0.1234	0.1590
Test 63	1e-3	(0.99, 0.99)	1e-3	0.35	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0841	0.1247	0.1543
Test 64	1e-3	(0.99, 0.99)	1e-3	0.40	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0860	0.1267	0.1526
Test 65	1e-3	(0.99, 0.99)	1e-3	0.45	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0876	0.1259	0.1575
Test 66	1e-3	(0.99, 0.99)	1e-3	0.55	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0907	0.1281	0.1625
Test 67	1e-3	(0.99, 0.99)	1e-3	0.50	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0882	0.1259	0.1589
Test 68	1e-3	(0.99, 0.99)	1e-3	0.60	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0910	0.1254	0.1554
Test 69	1e-3	(0.99, 0.99)	1e-3	0.65	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0928	0.1269	0.1606
Test 70	1e-3	(0.99, 0.99)	1e-3	0.70	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0924	0.1271	0.1605
Test 71	1e-3	(0.99, 0.99)	1e-3	0.75	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0935	0.1263	0.2698
Test 72	1e-3	(0.99, 0.99)	1e-3	0.80	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0944	0.1270	0.1612
Test 73	1e-3	(0.99, 0.99)	1e-3	0.85	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0954	0.1264	0.1627
Test 74	1e-3	(0.99, 0.99)	1e-3	0.90	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0970	0.1284	0.1656
Test 75	1e-3	(0.99, 0.99)	1e-3	0.95	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0990	0.1390	0.1831
Test 76	1e-3	(0.99, 0.99)	1e-3	0.99	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0991	0.1317	0.1713
Test 77	1e-3	(0.99, 0.99)	1e-3	1.00	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0985	0.1319	0.1705
Test 78	1e-3	(0.99, 0.99)	1e-3	2.00	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.1098	0.1329	0.1776
Test 79	1e-3	(0.99, 0.99)	1e-3	5.00	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.1445	0.1522	0.2017
Test 80	1e-3	(0.99, 0.99)	1e-3	10.0	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.1846	0.1928	0.2320
Test 81	1e-3	(0.99, 0.99)	1e-3	0.15	True	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0786	0.1261	0.1537
Test 82	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.05	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0784	0.1210	0.1461
Test 83	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.10	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0792	0.1214	0.1451
Test 84	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0801	0.1210	0.1436
Test 85	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.20	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0802	0.1211	0.1462
Test 86	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.25	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0814	0.1219	0.1439
Test 87	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.30	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0816	0.1220	0.1425
Test 88	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.35	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0835	0.1229	0.1448
Test 89	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.40	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0834	0.1231	0.1463
Test 90	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.45	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0840	0.1231	0.1418
Test 91	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.50	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0840	0.1220	0.1405
Test 92	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.55	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0856	0.1233	0.1430
Test 93	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.60	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0861	0.1241	0.1428
Test 94	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.65	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0879	0.1256	0.1459
Test 95	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.70	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0892	0.1259	0.1471
Test 96	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.75	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0901	0.1260	0.1482
Test 97	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.80	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0909	0.1257	0.1476
Test 98	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.85	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0923	0.1267	0.1454
Test 99	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.90	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0952	0.1278	0.1425
Test 100	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.95	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0997	0.1307	0.1386

Table B.2: Hyperparameter Tuning, test 51-100

	LR	(β_1, β_2)	ϵ	WD	AMSgrad	ED	Layer Dropout	momentum	Train MAE	Valid MAE	Test MAE
Random Forest									0.1526	0.1400	0.1340
Default	1e-3	(0.9, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0718	0.1272	0.1820
Test 101	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.99	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.1065	0.1335	0.1359
Test 102	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.05, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0805	0.1207	0.1427
Test 103	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.10, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0821	0.1222	0.1489
Test 104	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.15, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0818	0.1197	0.1453
Test 105	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.20, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0822	0.1206	0.1443
Test 106	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.25, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0840	0.1224	0.1471
Test 107	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.30, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0851	0.1215	0.1457
Test 108	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.35, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0854	0.1215	0.1439
Test 109	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.40, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0869	0.1208	0.1492
Test 110	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.45, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0880	0.1220	0.1402
Test 111	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.50, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0895	0.1237	0.1515
Test 112	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.55, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0906	0.1231	0.1545
Test 113	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.60, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0920	0.1237	0.1563
Test 114	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.65, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0934	0.1221	0.1512
Test 115	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.70, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0954	0.1225	0.1549
Test 116	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.75, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0972	0.1221	0.1533
Test 117	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.80, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.1004	0.1238	0.1617
Test 118	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.85, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.1043	0.1254	0.1659
Test 119	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.90, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.1076	0.1247	0.1650
Test 120	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.95, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.1158	0.1286	0.1736
Test 121	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.99, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.1413	0.1400	0.1535
Test 122	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.05, 0.00, 0.00)	(0.95, 0.85)	0.0801	0.1205	0.1432
Test 123	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.10, 0.00, 0.00)	(0.95, 0.85)	0.0803	0.1210	0.1442
Test 124	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.15, 0.00, 0.00)	(0.95, 0.85)	0.0811	0.1199	0.1441
Test 125	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.00, 0.00)	(0.95, 0.85)	0.0806	0.1184	0.1464
Test 126	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.25, 0.00, 0.00)	(0.95, 0.85)	0.0818	0.1209	0.1488
Test 127	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.30, 0.00, 0.00)	(0.95, 0.85)	0.0822	0.1214	0.1452
Test 128	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.35, 0.00, 0.00)	(0.95, 0.85)	0.0827	0.1224	0.1492
Test 129	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.40, 0.00, 0.00)	(0.95, 0.85)	0.0832	0.1220	0.1472
Test 130	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.45, 0.00, 0.00)	(0.95, 0.85)	0.0836	0.1207	0.1501
Test 131	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.50, 0.00, 0.00)	(0.95, 0.85)	0.0838	0.1194	0.1477
Test 132	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.55, 0.00, 0.00)	(0.95, 0.85)	0.0839	0.1208	0.1437
Test 133	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.60, 0.00, 0.00)	(0.95, 0.85)	0.0846	0.1213	0.1501
Test 134	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.65, 0.00, 0.00)	(0.95, 0.85)	0.0845	0.1208	0.1527
Test 135	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.70, 0.00, 0.00)	(0.95, 0.85)	0.0842	0.1199	0.1461
Test 136	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.75, 0.00, 0.00)	(0.95, 0.85)	0.0861	0.1232	0.1563
Test 137	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.80, 0.00, 0.00)	(0.95, 0.85)	0.0853	0.1212	0.1488
Test 138	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.85, 0.00, 0.00)	(0.95, 0.85)	0.0857	0.1213	0.1473
Test 139	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.90, 0.00, 0.00)	(0.95, 0.85)	0.0859	0.1216	0.1492
Test 140	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.95, 0.00, 0.00)	(0.95, 0.85)	0.0888	0.1225	0.1503
Test 141	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.99, 0.00, 0.00)	(0.95, 0.85)	0.1356	0.1377	0.1499
Test 142	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.05, 0.00)	(0.95, 0.85)	0.0815	0.1198	0.1483
Test 143	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.10, 0.00)	(0.95, 0.85)	0.0817	0.1198	0.1463
Test 144	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.95, 0.85)	0.0813	0.1190	0.1463
Test 145	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.20, 0.00)	(0.95, 0.85)	0.0828	0.1210	0.1492
Test 146	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.25, 0.00)	(0.95, 0.85)	0.0831	0.1226	0.1517
Test 147	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.30, 0.00)	(0.95, 0.85)	0.0818	0.1210	0.1464
Test 148	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.35, 0.00)	(0.95, 0.85)	0.0821	0.1198	0.1457
Test 149	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.40, 0.00)	(0.95, 0.85)	0.0820	0.1212	0.1499
Test 150	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.45, 0.00)	(0.95, 0.85)	0.0819	0.1189	0.1454

Table B.3: Hyperparameter Tuning, test 101-150

	LR	(β_1, β_2)	ϵ	WD	AMSgrad	ED	Layer Dropout	momentum	Train MAE	Valid MAE	Test MAE
Random Forest									0.1526	0.1400	0.1340
Default	1e-3	(0.9, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0718	0.1272	0.1820
Test 151	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.50, 0.00)	(0.95, 0.85)	0.0824	0.1193	0.1489
Test 152	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.55, 0.00)	(0.95, 0.85)	0.0817	0.1186	0.1450
Test 153	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.60, 0.00)	(0.95, 0.85)	0.0829	0.1217	0.1464
Test 154	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.65, 0.00)	(0.95, 0.85)	0.0828	0.1206	0.1467
Test 155	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.70, 0.00)	(0.95, 0.85)	0.0824	0.1194	0.1490
Test 156	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.75, 0.00)	(0.95, 0.85)	0.0831	0.1193	0.1440
Test 157	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.80, 0.00)	(0.95, 0.85)	0.0832	0.1220	0.1490
Test 158	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.85, 0.00)	(0.95, 0.85)	0.0843	0.1216	0.1479
Test 159	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.90, 0.00)	(0.95, 0.85)	0.0879	0.1180	0.1402
Test 160	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.95, 0.00)	(0.95, 0.85)	0.1001	0.1199	0.1396
Test 161	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.99, 0.00)	(0.95, 0.85)	0.1149	0.1218	0.1331
Test 162	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.05)	(0.95, 0.85)	0.0816	0.1202	0.1480
Test 163	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.10)	(0.95, 0.85)	0.0813	0.1200	0.1458
Test 164	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.15)	(0.95, 0.85)	0.0812	0.1184	0.1422
Test 165	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.20)	(0.95, 0.85)	0.0812	0.1184	0.1430
Test 166	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.25)	(0.95, 0.85)	0.0817	0.1205	0.1504
Test 167	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.30)	(0.95, 0.85)	0.0825	0.1204	0.1479
Test 168	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.35)	(0.95, 0.85)	0.0812	0.1203	0.1463
Test 169	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.40)	(0.95, 0.85)	0.0818	0.1185	0.1466
Test 170	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.45)	(0.95, 0.85)	0.0814	0.1184	0.1465
Test 171	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.50)	(0.95, 0.85)	0.0817	0.1184	0.1425
Test 172	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.55)	(0.95, 0.85)	0.0817	0.1203	0.1460
Test 173	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.60)	(0.95, 0.85)	0.0820	0.1200	0.1456
Test 174	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.65)	(0.95, 0.85)	0.0815	0.1196	0.1455
Test 175	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.75)	(0.95, 0.85)	0.0823	0.1195	0.1507
Test 176	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.70)	(0.95, 0.85)	0.0813	0.1188	0.1430
Test 177	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.80)	(0.95, 0.85)	0.0819	0.1191	0.1450
Test 178	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.85)	(0.95, 0.85)	0.0826	0.1201	0.1481
Test 179	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.90)	(0.95, 0.85)	0.0820	0.1193	0.1460
Test 180	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.95)	(0.95, 0.85)	0.0837	0.1172	0.1432
Test 181	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.99)	(0.95, 0.85)	0.0987	0.1230	0.1435
Test 182	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.00, 0.85)	0.0823	0.1194	0.1469
Test 183	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.05, 0.85)	0.0818	0.1217	0.1465
Test 184	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.10, 0.85)	0.0816	0.1209	0.1461
Test 185	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.15, 0.85)	0.0829	0.1200	0.1477
Test 186	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.20, 0.85)	0.0817	0.1206	0.1455
Test 187	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.25, 0.85)	0.0815	0.1195	0.1472
Test 188	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.30, 0.85)	0.0824	0.1202	0.1468
Test 189	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.35, 0.85)	0.0822	0.1220	0.1496
Test 190	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.40, 0.85)	0.0822	0.1227	0.1496
Test 191	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.45, 0.85)	0.0819	0.1211	0.1460
Test 192	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.50, 0.85)	0.0817	0.1179	0.1431
Test 193	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.55, 0.85)	0.0817	0.1222	0.1493
Test 194	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.60, 0.85)	0.0825	0.1231	0.1493
Test 195	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.85)	0.0819	0.1177	0.1429
Test 196	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.70, 0.85)	0.0813	0.1205	0.1459
Test 197	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.75, 0.85)	0.0819	0.1209	0.1471
Test 198	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.80, 0.85)	0.0823	0.1187	0.1475
Test 199	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.85, 0.85)	0.0827	0.1229	0.1497
Test 200	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.90, 0.85)	0.0818	0.1210	0.1513

Table B.4: Hyperparameter Tuning, test 151-200

	LR	(β_1, β_2)	ϵ	WD	AMSgrad	ED	Layer Dropout	momentum	Train MAE	Valid MAE	Test MAE
Random Forest									0.1526	0.1400	0.1340
Default	1e-3	(0.9, 0.99)	1e-8	0.01	False	0.00	(0.00, 0.00, 0.00, 0.00)	(0.95, 0.85)	0.0718	0.1272	0.1820
Test 201	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.99, 0.85)	0.0827	0.1191	0.1443
Test 202	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.00)	0.0808	0.1183	0.1428
Test 203	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.05)	0.0819	0.1210	0.1468
Test 204	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.10)	0.0820	0.1191	0.1469
Test 205	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.15)	0.0823	0.1221	0.1481
Test 206	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.20)	0.0817	0.1206	0.1475
Test 207	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.25)	0.0818	0.1215	0.1476
Test 208	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.30)	0.0820	0.1205	0.1472
Test 209	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.35)	0.0810	0.1190	0.1429
Test 210	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.40)	0.0809	0.1196	0.1413
Test 211	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.45)	0.0812	0.1190	0.1443
Test 212	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.50)	0.0818	0.1215	0.1465
Test 213	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.55)	0.0810	0.1212	0.1443
Test 214	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.60)	0.0829	0.1239	0.1512
Test 215	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.65)	0.0812	0.1208	0.1435
Test 216	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.70)	0.0815	0.1191	0.1451
Test 217	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.75)	0.0809	0.1200	0.1451
Test 218	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.80)	0.0811	0.1189	0.1448
Test 219	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.85)	0.0811	0.1181	0.1433
Test 220	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.90)	0.0821	0.1201	0.1481
Test 221	1e-3	(0.99, 0.99)	1e-3	0.15	False	0.15	(0.00, 0.20, 0.15, 0.00)	(0.65, 0.99)	0.0818	0.1209	0.1499

Table B.5: Hyperparameter Tuning, test 201-230

Bibliography

[1] DX. About DX. https://en.dx.no/.

[2] TMDb. About TMDb. https://www.themoviedb.org/about, .

[3] Google. Extracting and serving feature embeddings for machine learning. https://cloud.google.com/solutions/machine-learning/overview-extracting-and-serving-feature-embeddings-for-machine-learning.

[4] Wikipedia. One-hot encoding. https://en.wikipedia.org/wiki/One-hot, .

[5] Towards Data Science. Neural Network Embeddings Explained. https://towardsdatascience.com/neural-network-embeddings-explained-4d028e6f0526, .

[6] Towards Data Science. Deep embedding's for categorical variables. https://towardsdatascience.com/deep-embeddings-for-categorical-variables-cat2vec-b05c8ab63ac0, .

[7] fastai. An Introduction to Deep Learning for Tabular Data. https://www.fast.ai/2018/04/29/categorical-embeddings/, .

[8] PyTorch. PyTorch Wikipedia. https://en.wikipedia.org/wiki/PyTorch.

[9] fastai. Welcome to fastai. https://docs.fast.ai/, .

[10] fastai. Tabular. https://www.fast.ai/2020/02/13/fastai-A-Layered-API-for-Deep-Learning/tabular, .

[11] Wikipedia. Random forest. https://en.wikipedia.org/wiki/Random$_forest$, .

[12] TMDb. TMDb API. https://www.themoviedb.org/documentation/api, .