POst WEngeR
(Predicting weekly wage of football player)

Student Yonghoon Dong, Hyunwoo Park, Jiwoo Jung, Seungyeop Cha,

1 Purpose

The purpose of this project is to gather indicators of soccer players' goals, assist, and runs, and then use these data to predict players' future salaries and evaluate their growth potential. We used data from professional players playing in soccer leagues in various countries around the world, including the Premier League, for this project. Specifically, we used data such as UEFA coefficients by year (changing the result of European competitions such as the Champions League), the weekly salary, number of games played, number of goals scored, number of clean sheets, playing time, number of yellow/red cards, and winning percentage.

2 Datasets

1. UEFA coefficient : www.uefa.com
2. Players information : https://www.footballdatabase.eu.en/Approach
3. Players salaries : www.capology.com

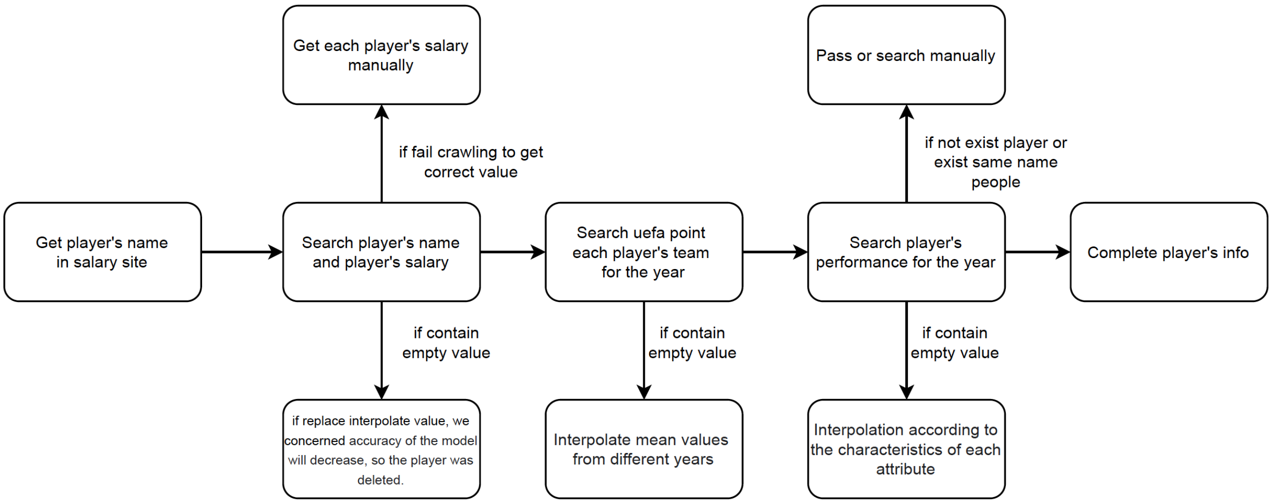

Get each player's salary manually
Pass or search manually
if fail crawling to get correct value
if not exist player or exist same name people
Get player's name in salary site
Search player's name and player's salary
Search uefa point each player's team for the year
Search player's performance for the year
Complete player's info
if contain empty value
if contain empty value
if contain empty value
if replace interpolate value, we concerned accuracy of the model will decrease, so the player was deleted.
Interpolate mean values from different years
Interpolation according to the characteristics of each attribute

Figure 1: Pipeline for crawling

3 Data Preprocessing

3.1 Handle incorrect values

We build a dataset consisting of football player's abilities (e.g. number of games, number of goals, number of assists and so on) and salary by year. There are some missing values and incorrectly collected data in our dataset, since the site prevented crawling. So, in the case of weekly wages, the player was manually filled(Since it is critical value for training), and in the case of the remaining indicators, if strange data were contained, the player was excluded for the stability of learning. Finally we have 521 number of players dataset.

3.2 Time series data

Because the purpose of this project is to predict weekly salary of a football player using history data about player's abilities, we construct the input value of one observation as a player's ability for given window size of years and the output value as an weekly salary for the next year. We compared the experimental results by changing the window size. However, increasing the window size reduces the amount of data available. Related information will be mentioned in the experiment section

3.3 Normalization

There are many player's features in our data, therefore for this reason embedding have different scale and unit for different coordinate. Leaning could be hindered by strongly reflecting certain features and ignoring other features. To solve this problem, we used normalization technique. However, using normalization can weaken the expressiveness of the model, as seen in batch normalization. Therefore, as presented a method to supplement this in our batch regularization paper, we have applied a similar method. Related information also will be mentioned in the experiment section.

4 Model Architecture

Basically we are using Transformer model which is good at inferring relationship between different embeddings. This is because in order to predict the weekly salary, the model must fully understand the relationship between the previous players' performance and the resulting weekly salary. We inferred that the relationship between player's performance and weekly salary will be well inferred through multi-head attention in the transformer. In this respect, the basic model for our architecture follows the transformer.

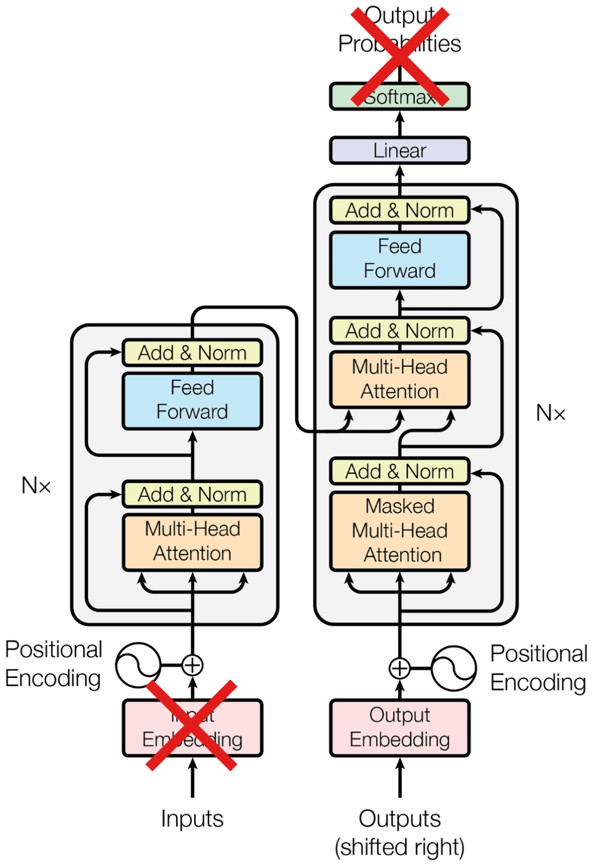


Figure 2: Our modified Transformer architecture

4.1 Input for Encoder

Encoder maps an input sequence of performance representation $(x_1, \cdots, x_n)$ to a sequence of representation $z = (z_1, \cdots, z_n)$ (n : window size). Each coordinate for the input means a specific performance of the competitor.

4.2 Input for Decoder

Given z, decoder generates an output sequence $y = (y_1, y_n, y_{n+1})$ of predicted salaries one element at a time.

4.3 Expected result

If the window size is 6, the weekly salary for the next year is predicted if the performance of the player for 6 years is provided to the encoder. In other words, it becomes possible to predict future predictions only with the current player's performance.

5 Training

5.1 Optimizer

We used the Adam optimizer with StepLR (Learning Rate Scheduler).

5.2 Regularization

We used the EarlyStopping to halt the training our neural networks at the right time.

5.3 Hyperparameter Tuning

epoch	100	100	100	100	100	100	100	100	100	200
batch size	128	128	128	128	128	128	128	128	128	256
initial lr	0.001	0.0008	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001
window size	3	3	3	3	3	3	3	3	3	3
d ff	2048	2048	1024	2048	4096	4096	8192	4096	4096	4096
dr rate	0.3	0.3	0.4	0.2	0.3	0.2	0.1	0.2	0.2	0.2
valid loss	0.199767	0.207508	0.214862	0.192663	0.197211	0.189034	0.189564	0.193505	0.193384	0.190752

Table 1: Valid error with respect hyperparameter

epoch	1000	100	100	100	100	100	300	300
batch size	128	128	64	128	128	128	128	128
initial lr	0.001	0.0011	0.001	0.0011	0.0011	0.005	0.005	0.005
window size	6	6	6	6	6	6	6	6
n layer	4	3	3	3	3	3	3	3
d model	512	512	512	1024	1024	2048	2048	2048
h	8	8	8	8	16	8	8	8
dr rate	0.3	0.1	0.1	0.1	0.1	0.1	0.1	0.1
lr scheduler	step LR	step LR	step LR	step LR	step LR	step LR	step LR	CosineAnnealingLR
valid loss	0.135456	0.132385	0.139951	0.126350	0.128236	0.126164	0.124723	0.126673

Table 2: Valid error with respect hyperparameter

epoch	batch size	initial lr	gamma	window size	n layer	d_model	h	d ff	dr rate	norm eps	loss function	optimizer	learning rate scheduler
300	128	0.0005	0.99	6	3	2048	8	2048	0.1	10^{-6}	MSE	Adam	stepLR

Table 3: Hyperparameter list for minimum Valid error

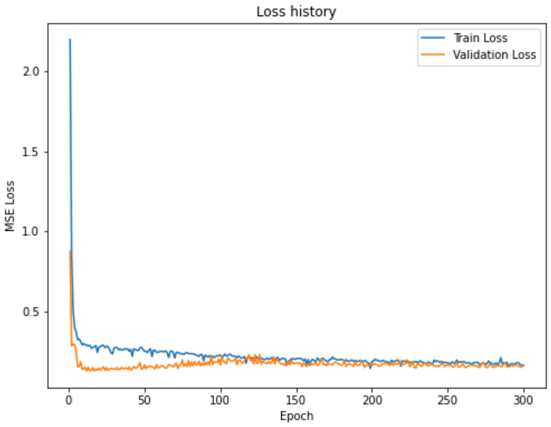
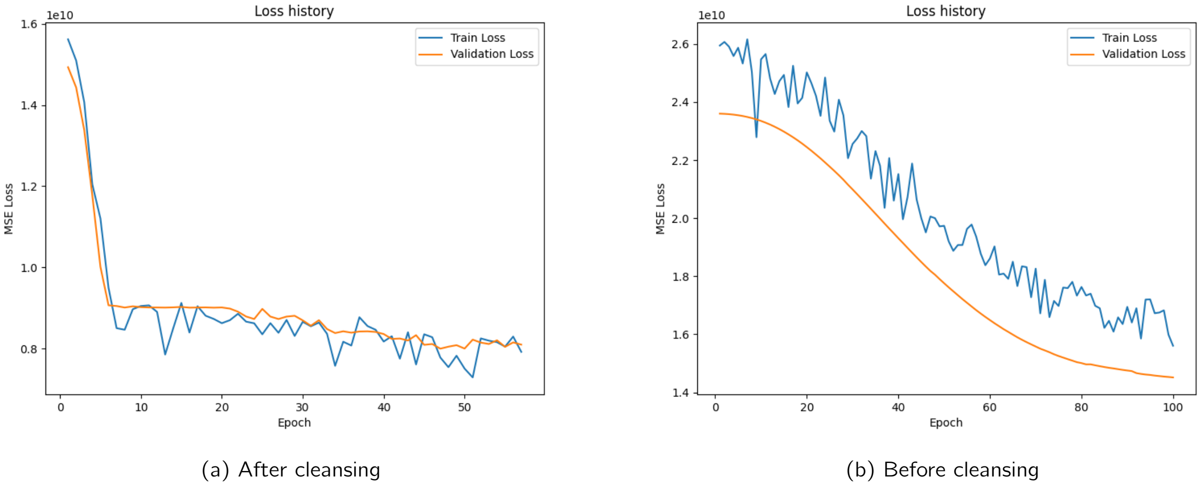


Figure 3: Best result of validation loss

6 Experiment

6.1 Data Cleansing

When data was first collected through crawling, there were quite a lot of incorrect data. An experiment was conducted to compare the results of running the model at that point in time and the results of running the model after manually correcting the incorrect data.



(a) After cleansing

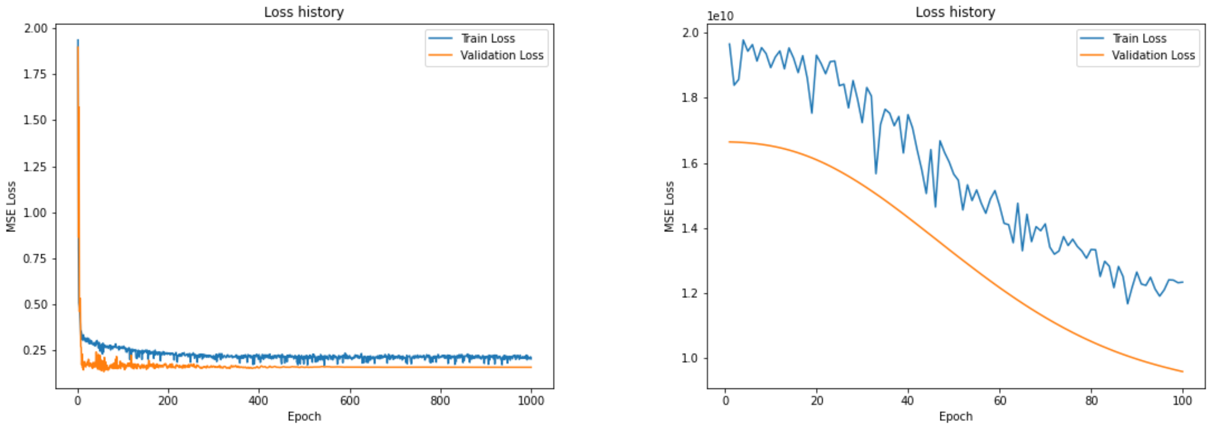
(b) Before cleansing

Figure 4: Compare train and validation error with same hyperparameter

Figure 4 shows that data cleansing plays an important role in improving the performance of the model.

6.2 Normalize

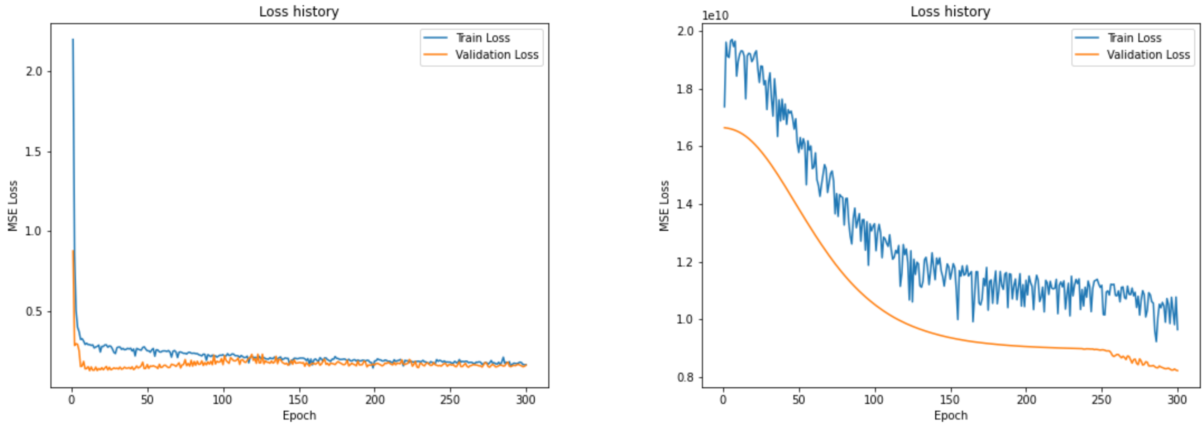
An experiment was conducted to find out how much Normalization affects learning stably. To this end, the test results when normalized and when not done were compared.



(a) Normalized

(b) Not Normalized

Figure 5: Compare train and validation error with same hyperparameter



(a) Normalized

(b) Not Normalized

Figure 6: Compare train and validation error with same hyperparameter

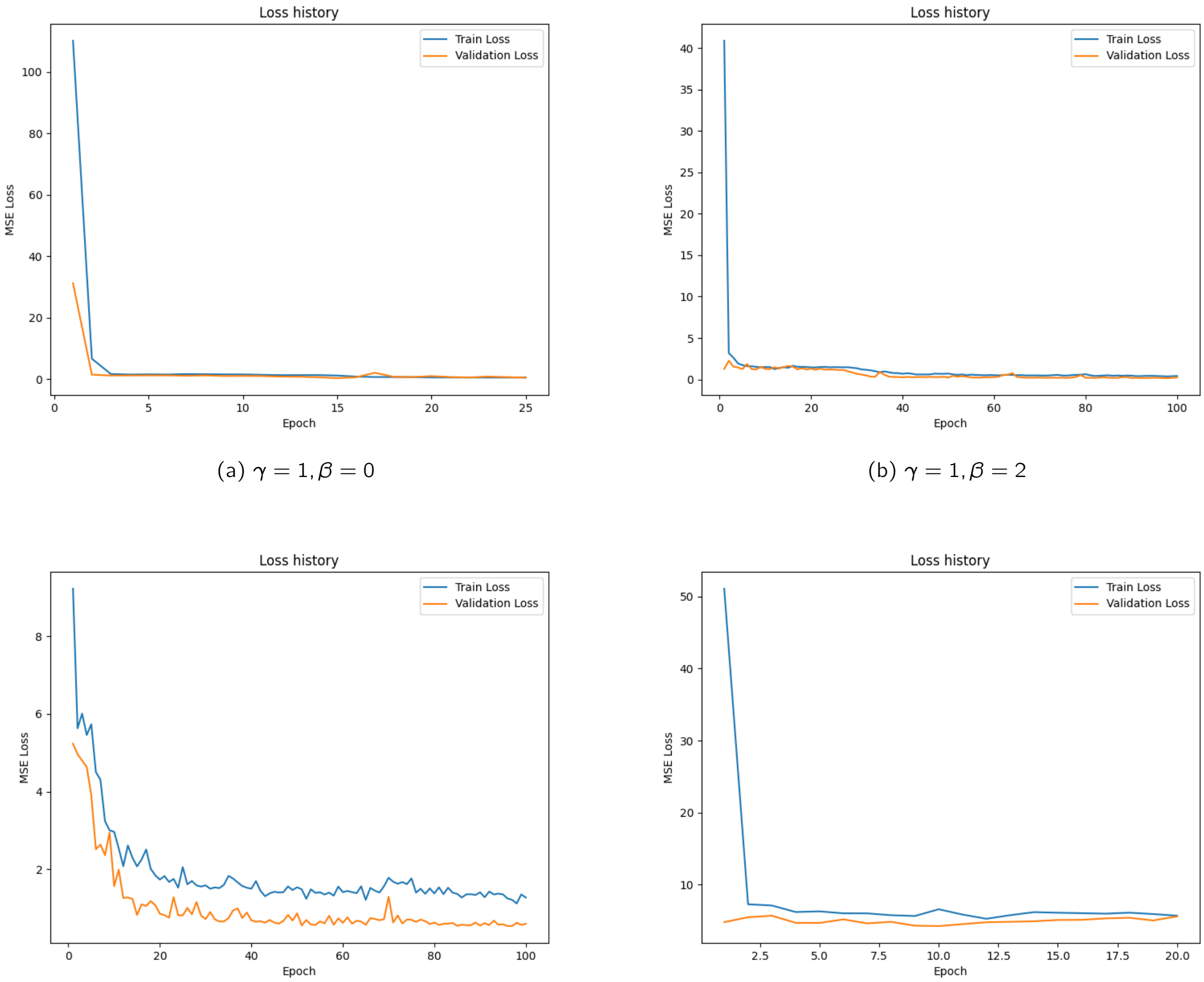
Figure 5 and 6 shows that normalization plays an important role in improving the performance of the model. When normalization is performed, it can be confirmed that the model is trained stably.

6.3 Scale and shift for Normalization

Using normalization could weaken the expressiveness of the model, as seen in batch normalization. Therefore, as presented a method to supplement this in our batch regularization paper, we experimented with how the scale and shift affect the learning of the model.

$\gamma = 1, \beta = 0$	$\gamma = 1, \beta = 2$	$\gamma = 2, \beta = 0$	$\gamma = 2, \beta = 2$	$\gamma = 3, \beta = 0$	$\gamma = 3, \beta = 2$
0.304582	0.311013	1.578759	7.345860	4.302127	2.034763

Table 4: Test error with respect to scale and shift value



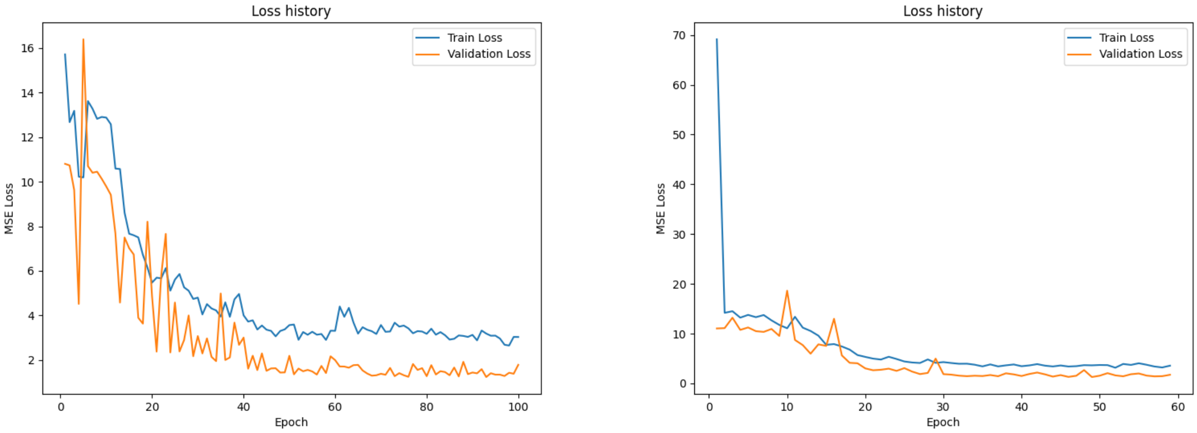
(a) $\gamma = 1, \beta = 0$

(b) $\gamma = 1, \beta = 2$

(c) $\gamma = 2, \beta = 0$

(d) $\gamma = 2, \beta = 2$

Figure 7: Train and validation error with respect to scale and shift value



(a) $\gamma = 3, \beta = 0$

(b) $\gamma = 3, \beta = 2$

Figure 8: Train and validation error with respect to scale and shift value

Through Figures 7 and 8, it can be seen that there is no need to set r and b to preserve expressive power as considered in batch normalization. Looking at the trend of the graph, it can be seen that the result is better when no correction is performed.

7 Discussion

First, the importance of performance is different for each soccer position. So the accuracy can be further improved by reflecting the position information of the players in the embedding.

Second, comparing team between different league is very difficult. We have used UEFA score for this purpose, however this is not enough criteria for comparing different league. Because the UEFA indicator is formed based on qualifying for the Champions League, so the more intense the league, the more difficult it is to advance. So if we can find a better metric to compare different leagues with, the performance could be better.