Feature Extraction for Deep Neural Networks: A Case Study on the COVID-19 Retweet Prediction Challenge

Daichi Takehara
Aidemy Inc.
takehara-d@aidemy.co.jp

ABSTRACT

This paper presents our solution for the COVID-19 Retweet Prediction Challenge, which is part of the CIKM 2020 AnalytiCup. The challenge was to predict the number of times it will be retweeted of tweets related to COVID-19. We tackled this challenge using a deep neural network-based retweet prediction method. In this method, we introduced useful feature extraction techniques for retweet prediction. Experiments have confirmed the effectiveness of the techniques, especially for the primary processes: numerical feature transformation and user modeling. Finally, the solution used a stacking-based ensemble method to provide the final predictive result for the competition. The code for this solution is available at https://github.com/haradai1262/CIKM2020-AnalytiCup.

CCS CONCEPTS

• **Information systems** → **Data mining**; *Social recommendation.*

KEYWORDS

Information diffusion, Retweeet prediction, Feature extraction, Deep learning, COVID-19,

1 INTRODUCTION

To understand the mechanisms of information diffusion is an active area of research that has many practical applications. In a crisis like COVID-19, information diffusion directly influences people's behavior and becomes especially valuable [6]. Retweeting, sharing tweets directly to followers on Twitter, can be viewed as amplifying the diffusion of original content. Thus retweet prediction is beneficial for understanding the mechanisms of information diffusion.

Retweet prediction has been widely studied. In recent years, there has been growing interest in methods based on deep neural networks (DNNs), which have reported high performance [10, 15, 19]. DNNs have made it possible to skip many feature engineering, especially in image processing and natural language processing. However, in DNNs for tabular data including retweet predictions, data pre-processing and feature engineering are still often necessary and significantly impact performance [9, 14].

In retweet prediction, the processing of numerical features related to tweets, such as the number of followers, strongly affects performance. To train DNNs effectively, it may be useful to transform the numerical features to different distributions [20]. Furthermore, it is crucial to learn the expression of the user that publish tweets. Although the embedding-based method using the user id is often used in DNN-based methods it may not be that easy to sufficiently learn the representation of the infrequent users included in the training data [5]. As mentioned above, it is necessary to design the input features to the DNN according to the data and tasks, which can be difficult.

As a case study to tackle these difficulties, in this paper, we present our solution to the COVID-19 Retweet Prediction Challenge as part of the CIKM 2020 AnalytiCup. This challenge's task was to predict the number retweets for a given COVID-19-related tweet. We propose a DNN-based retweet prediction method. In the proposed method, we introduce a useful feature extraction method as an input to a DNN for retweet prediction. In the feature extraction, we transform numerical features into multiple different distributions to effectively utilize the metrics related to tweets. Besides, we cluster users based on multiple types of features to enable infrequent users to represent user attributes. Using the obtained features, we train a DNN model. In the experiments, we verify the effectiveness of the transformation of numerical features and user data handling, which are essential issues in retweet prediction. In addition, as a solution for the competition, we introduce a stacking-based ensemble method to improve the prediction results' performance and robustness.

2 CHALLENGE

2.1 Dataset

In the COVID-19 Retweet Prediction Challenge, TweetsCOV19 dataset was provided. This dataset consists of 8151524 tweets concerning COVID-19 on Twitter published by 3664518 users from October 2019 until April 2020. For each tweet, the dataset provides metadata and some precalculated features. The contents of the dataset and the process of their generation are detailed in this paper [3].

2.2 Task Description

Given a tweet from the TweetsCOV19 dataset, the task was to predict the number retweets (#retweets). The test data for the evaluation are tweets published during May 2020, which the month subsequent to the tweets included in the TweetsCOV19 dataset. The mean squared log error (MSLE) is used as the evaluation metric for the task.

3 METHOD

3.1 Overview

An overview of the proposed method is shown in Figure 1. First, we extract the features to be inputed to the DNN. The features input to the DNN are divided into numerical, categorical, and multi-hot categorical features, and categorical and multi-hot categorical features are converted into low-dimensional vectors through the embedding layer. Using the extracted features, we train a multilayer perceptron (MLP) for retweet prediction.

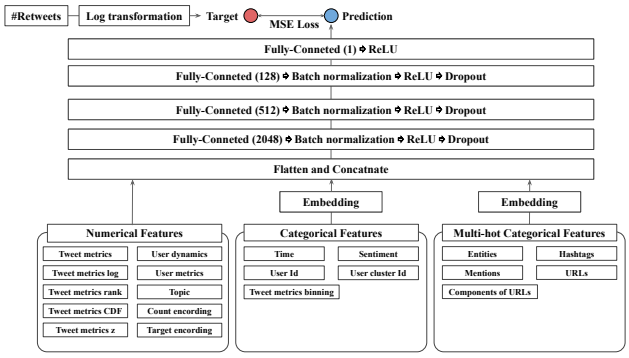

Figure 1: Overview of our proposed method. The notation of features corresponds to the name columns in Table 1.

3.2 Features

The features used in the proposed method are shown in Table 1. Numerical feature transformation and user modeling, the critical issues of retweet prediction, are discussed in the following subsections. Please refer to the published code[1] for strict processing.

3.2.1 Numerical Feature Transformation. #retweets is strongly related to metrics of a tweet, such as the number of followers (#followers) and favorites (#followers), which are expected to have a significant impact on the performance of our prediction model. In the proposed method, we attempt to represent various distributions of these metrics and improve the performance by combining them as input for the DNN. Specifically, we introduce the following five numerical feature transformations.

Z transformation. We transform each value $x_i \in X$ by the following function using the mean $\overline{x}$ and standard deviation μ of the dataset X:

$$F_z(x_i) = \frac{x_i - \overline{x}}{\mu} \tag{1}$$

CDF transformation. We derived a normal distribution from the mean and standard deviation observed from the dataset. Using the distribution, we transformed the original values by the cumulative distribution function (CDF). To implement this function, we used the Python library SciPy[2].

Rank transformation. We transform each value $x_i \in X$ by the following function:

$$F_{rank}(x_i) = \sum_{x_j \in X} \mathrm{I}_{x_j < x}, \ \mathrm{I}_{x_j < x} = \begin{cases} 1 & x_j < x \text{ is true} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Log transformation. We transform each value $x_i \in X$ by the following function:

$$F_{log}(x_i) = \log_e(x_i + 1) \tag{3}$$

Here we add one to x_i to avoid the output being infinity when x_i is zero.

Binning transformation. We separate each value into buckets of the same size based on the quantiles of the sample. In the proposed method, the values are divided into ten quantiles. Unlike other transformations, we use the transformed values as categorical features.

We apply these transformations to tweet metrics (Table 1) and input the obtained values into a MLP.

3.2.2 User Modeling. Appropriately representing the user who published the tweet is essential for predicting #retweets. For DNN-based prediction models, a common and effective method is to learn by inputting the user id into embedding layers. However, the infrequent users included in the training data are not sufficiently trained [5]. By clustering users from various points of view and embedding based on their cluster Ids, we can even learn the user attributes for infrequent users. Specifically, we introduced the following three types of user clustering.

User topic clustering. We clustered users using topics contained in tweets. Specifically, we combined the entities, hashtags, mentions, and URLs included in the tweet and set them as sequences for each user. Next, user topic features were extracted by applying the term frequency-inverse document frequency (TFIDF) [2] to the sequences and dimensionality reduction using singular value decomposition (SVD) [4]. Using the extracted features, we applied the K-means clustering [11] to the users.

User metric clustering. We clustered users using user-related metrics. User metric features consist of the mean and standard deviation of the user's followers, friends, likes, as well as the unique numbers of entities, hashtags, mentions, and URLs from the tweet log posted by each user. Using the obtained features, we applied the K-means clustering to the users.

User topic and metric clustering. Using the features that combine user topic features and user metric features, we applied the K-means clustering to the users.

Note that the number of clusters is set to 1000 in each clustering.

3.3 Model

Using the extracted features, we trained the MLP. In the proposed method, the inputs of the MLP can be divided into numerical, categorical, and multi-hot categorical features. Numerical features were applied to min-max scaling and converted to a scale of [0, 1]. In the proposed method, we transformed categorical features into low-dimensional vectors using the embedding layers. Specifically, we represent one-hot vector $\boldsymbol{x_i}$, a categorical feature, with low-dimensional vector $\boldsymbol{e_i}$,

$$\boldsymbol{e_i} = \boldsymbol{E_c}\boldsymbol{x_i}, \tag{4}$$

where $\boldsymbol{E_c}$ is an embedding matrix for categorical feature c. We further modify it and represent multi-hot vector $\boldsymbol{x_i}$, a multi-hot categorical feature, in the following way:

$$\boldsymbol{e_i} = \frac{1}{n_c}\boldsymbol{E_c}\boldsymbol{x_i}, \tag{5}$$

where n_c is the number of items that a sample has for categorical feature c. These processed values are concatenated and flattened before inputting to the MLP.

As shown in Figure 1, the MLP is a structure that uses ReLU [13] as the activation function and includes batch normalization [7] and dropout [17]. In the proposed method, mean squared error (MSE) loss is calculated as a loss function from the ground truth of #retweets using log transformation and the prediction of MLP.

[1] https://github.com/haradai1262/CIKM2020-AnalytiCup/blob/master/src/feature_extraction.py

[2] https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.norm.html

Table 1: Feature table. Numerical, categorical, and multi-hot categorical features are denoted by N, C, and MC in the Type column, respectively. The values in the #Dim column is the number of dimensions of the features

Name	Description	Type	#Dim
Tweet metrics	Metrics related to a tweet. Specifically, we use #followers, #friends, and #favorites, as well as the multiplication of #followers and #favorites, #friends and #favorites, and #followers and #friends and #favorites	N	6
Tweet metrics z	Values obtained by applying z transformation to "tweet metrics"	N	6
Tweet metrics CDF	Values obtained by applying CDF transformation to "tweet metrics"	N	6
Tweet metrics rank	Values obtained by applying rank transformation to "tweet metrics"	N	6
Tweet metrics log	Values obtained by applying log transformation to "tweet metrics"	N	6
Tweet metrics binning	Values obtained by applying binning transformation to "tweet metrics"	N	6
Sentiment	Positive (1 to 5) and negative (-1 to -5) sentiment scores extracted from the text of a tweet by SentiStrength [18]	C	2
Time	Features obtained from the timestamp of a tweet. Specifically, we use "weekday," "hour," "day," and "week of month" as categorical features, and the difference between the timestamp of the tweet and 2020/6/1 as numerical features	N, C	5
Entities	Entities extracted from the text of a tweet by the Fast Entity Linker [1]	MC	1
Hashtags	Hashtags included in a tweet	MC	1
Mentions	Mentions included in a tweet	MC	1
URLs	URLs included in a tweet	MC	1
Components of URLs	Components of URLs included in a tweet. We extract the three components "protcol," "host," and "top level domain" from the URL (*e.g.*, "http," "www.youtube.com," and ".com" are extracted from http://www.youtube.com/)	MC	3
User ID	User identifier	C	1
User cluster ID	Identifier assigned to a user by three clustering methods described in section 3.2.2.	C	3
User metrics	Metrics related to a user. Specifically, we use the mean and standard deviation of the followers, friends, favorites, and unique numbers of entities, hashtags, mentions, and URLs from the user's tweet history.	N	10
User dynamics	Metrics related to the dynamics in the #followers and #friends of a user. We use the increase in follower and friends from the previous day, the previous week, on the same day, and within the same week	N	8
Topic	5-dimensional features extracted by applying TFIDF [2] to sequences consisted of entities, hashtags, mentions, and URLs included in tweets and dimensionality reduction by SVD [4]	N	5
Count encoding	Values obtained by applying count encoding [16] to the categorical features "sentiment" and "time "	N	6
Target encoding	Values obtained by applying target encoding [12] to the categorical features "tweet metrics binning," "sentiment," "time," and "user Id"	N	11

Note that, at the time of inference, the output value is applied to the inverse transformation and returned to the original scale.

3.4 Validation Strategy

The method for dividing the dataset into training and validation data was as follows. In this competition, the test data for evaluation is May 2020, one month after the data included in the training data. To bring the distribution of the validation data and the test data closer, we need to use the validation data that is as close to the test data as possible in time series. Thus, we used the data from May 2020 as the validation data. We also wanted to utilize the May 2020 data to perform better learning with fresh data close to test data. For this reason, the May 2020 data was divided into five validation data point and five models to be trained. Here, when verifying with one verification data point, the remaining four are used for training data. Finally, the prediction value of the test data was calculated for each model, and the evaluation score was calculated from their average value.

4 EXPERIMENTS

4.1 Settings

The experimental results are not the scores of the test dataset, but the average of the 5-fold validation described in section 3.4. We empirically set the sizes of the three fully-connected layers to 2048, 512, and 128, respectively, dimension of embedding to 32, dropout rate to 0.3, and batch size to 256. We use Adam [8] to optimize all models. Other hyperparameters can be strictly checked in the published code.

Table 2: Comparison of numerical feature transformations.

Method	MSLE
Tweet metrics	0.187028
Tweet metrics + z transformation	0.173821
Tweet metrics + CDF transformation	0.151882
Tweet metrics + log transformation	0.129360
Tweet metrics + rank transformation	0.130994
Tweet metrics + binning transformation	0.174810
Tweet metrics + all transformations	**0.127761**

4.2 Results

First, we verified the effectiveness of the numerical feature transformation introduced in section 3.2.1. In the experiment, we tried using the tweet metrics without the transformation, with the application of each transformation, and with the application of all transformations. The experimental results are shown in Table 2. It was confirmed that log, rank, CDF, z, and binning transformation contributed to improving the performance, in this order. Since the number of followers and favorites in tweet metrics follows the power law, it is reasonable that log transformation is useful. Also, the best MSLE was obtained when applying all transformations. The result shows the effectiveness of transforming tweet metrics

Table 3: Comparison of user modeling features.

Method	MSLE
Both user ID and user cluster ID are unused	0.144809
User ID	0.128004
User cluster ID	0.137432
User ID and user cluster ID	**0.127761**

Table 4: Models used for the ensemble of our solution. MAE in the LOSS column denotes mean absolute error loss.

Embedding dim	Sizes of FC layers	Dropout rate	Loss	MSLE
32	2048, 512, 128	0.1	MSE	0.128448
32	2048, 512, 128	0.3	MSE	0.127761
32	2048, 512, 128	0.5	MSE	0.128413
40	4096, 1024, 128	0.1	MSE	0.127964
40	4096, 1024, 128	0.3	MSE	0.127810
40	4096, 1024, 128	0.5	MSE	0.128520
40	4096, 1024, 128	0.1	MAE	0.132143

Table 5: Final submission results of the top six teams (semi-finalists) in the competition.

Rank	Team	MSLE (Test dataset)
1	vinayaka	0.120551
2	mc-aida	0.121094
3	**myaunraitau (ours)**	**0.136239**
4	parklize	0.149997
5	JimmyChang	0.156876
6	Thomary	0.169047

into different distributions and inputting them into the DNN model for retweet prediction.

Next, we verified the effectiveness of the user modeling introduced in section 3.2.2. In the experiment, in regard to embedding of user ID and user cluster ID, we tried not using either, using either one, and using both. The experimental results are shown in Table 3. It has been found that the performance is improved when the user cluster ID is also used compared to when only using the user ID.

5 SOLUTION

We used ensemble on multiple models with modified hyperparameters (size of embedding dimension, sizes of fully-connected layers, and dropout rate) and loss function. Table 4 shows the seven models used for the ensemble. Stacking ridge regression [2] was used as the ensemble method. Stacking ridge regression is a method of blending each model's prediction results by a linear sum based on the weights learned by ridge regression. The integer value was obtained as the final predicted value by rounding according to the competition's manners. The final leaderboard looked like Table 5. Our solution was located in the 3rd place.

6 CONCLUSION

This paper presents our solution for the COVID-19 Retweet Prediction Challenge. We proposed a DNN-based retweet prediction method. To improve the performance, we introduced a feature extraction method to be input into the DNN (mainly focusing on numerical feature transformation and user modeling) and confirmed its effectiveness with experiments. As a solution for the competition, we introduced a stacking-based ensemble method for multiple models, which positioned us in the 3rd place.

REFERENCES

[1] Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. 2015. Fast and Space-Efficient Entity Linking for Queries. In *Proceedings of the Eighth ACM Int. Conf. on Web Search and Data Mining*. ACM, 179–188.

[2] Leo Breiman. 1996. Stacked regressions. *Machine learning* 24, 1 (1996), 49–64.

[3] Dimitar Dimitrov, Erdal Baran, Pavlos Fafalios, Ran Yu, Xiaofei Zhu, Matthäus Zloch, and Stefan Dietze. 2020. TweetsCOV19–A Knowledge Base of Semantically Annotated Tweets about the COVID-19 Pandemic. In *Proceedings of the 29th ACM Int. Conf. on Information Knowledge Management, Resource Track*.

[4] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. 2011. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review* 53, 2 (2011), 217–288.

[5] Casper Hansen, Christian Hansen, Jakob Grue Simonsen, Stephen Alstrup, and Christina Lioma. 2020. Content-aware Neural Hashing for Cold-start Recommendation. In *Proceedings of the 43rd Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*. 971–980.

[6] Cindy Hui, Yulia Tyshchuk, William A Wallace, Malik Magdon-Ismail, and Mark Goldberg. 2012. Information cascades in social media in response to a crisis: a preliminary model and a case study. In *Proceedings of the 21st Int. Conf. on World Wide Web*. 653–656.

[7] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd Int. Conf. on Machine Learning*. 448–456.

[8] Diederik P Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*.

[9] Yuanfei Luo, Mengshuo Wang, Hao Zhou, Quanming Yao, Wei-Wei Tu, Yuqiang Chen, Wenyuan Dai, and Qiang Yang. 2019. Autocross: Automatic feature crossing for tabular data in real-world applications. In *Proceedings of the 25th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining*. 1936–1945.

[10] Renfeng Ma, Xiangkun Hu, Qi Zhang, Xuanjing Huang, and Yu-Gang Jiang. 2019. Hot Topic-Aware Retweet Prediction with Masked Self-attentive Model. In *Proceedings of the 42nd Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*. 525–534.

[11] James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1. 281–297.

[12] Daniele Micci-Barreca. 2001. A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems. *ACM SIGKDD Explorations Newsletter* 3, 1 (2001), 27–32.

[13] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th Int. Conf. on Machine Learning*. 807–814.

[14] Jean-François Puget. 2017. Feature Engineering For Deep Learning. https://medium.com/inside-machine-learning/feature-engineering-for-deep-learning-2b1fc7605ace. Accessed: 2020-09-28.

[15] Jiezhong Qiu, Jian Tang, Hao Ma, Yuxiao Dong, Kuansan Wang, and Jie Tang. 2018. Deepinf: Social influence prediction with deep learning. In *Proceedings of the 24th ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining*. 2110–2119.

[16] Shubham Singh. 2020. Categorical Variable Encoding Techniques. https://medium.com/analytics-vidhya/categorical-variable-encoding-techniques-17e607fe42f9. Accessed: 2020-09-28.

[17] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[18] Mike Thelwall, Kevan Buckley, Georgios Paltoglou, Di Cai, and Arvid Kappas. 2010. Sentiment strength detection in short informal text. *Journal of the American Society for Information Science and Technology* 61, 12 (2010), 2544–2558.

[19] Qi Zhang, Yeyun Gong, Jindou Wu, Haoran Huang, and Xuanjing Huang. 2016. Retweet prediction with attention-based deep neural network. In *Proceedings of the 25th ACM Int. on Conf. on Information and Knowledge Management*. 75–84.

[20] Honglei Zhuang, Xuanhui Wang, Michael Bendersky, and Marc Najork. 2020. Feature transformation for neural ranking models. In *Proceedings of the 43rd Int. ACM SIGIR Conf. Research and Development in Information Retrieval*. 1649–1652.