Categorization of text-based forum posts using neural networks

Joaquín Cardona Ruiz, Agustín Delmonti, Joaquín Sánchez

Departamento de Ingeniería en Sistemas de la Información
Universidad Tecnológica Nacional
Facultad Regional Rosario
joaquincardonaruiz@gmail.com , adelmonti@frro.utn.edu.ar, josanchez@frro.utn.edu.ar

Abstract

Nowadays, online forums represent one of the largest repositories of user generated content on the Internet, thus one of the main sources of information. Categorizing this information within a topic hierarchy in a forum is essential to enable fast access for users to useful information. This task is performed by human users based on their knowledge on the topics. However, sometimes posts are submitted to the wrong category, which requires an amount of time to manually classify these posts in the right category. Current literature shows how to classify content in categories, but using limited amounts of data and applying a clustering based approach. In this paper we propose an automatic model for assessing categories for posts based on their content within an online discussion forum. This task is performed using a deep learning approach. This is a novel approach compared to those used in the current literature. We have found promising results on large real datasets. Therefore, future applications of this method can be studied for automatic post categorization in online forums. Some experimental results are shown to validate the proposed learning algorithm, and to demonstrate its generalization capabilities.

Keywords

Forum classification, Supervised Learning, Deep Learning

Introduction

Online forum boards are spaces of discussion hosted in the Web (such as Reddit or Stack Exchange) where people can hold conversations in the form of posted messages. These forum boards, although different in structure, are primarily divided into a finite set of generic categories of forums. Each category has several relevant threads where members can start discussing a topic related to the corresponding category. A group of users known as moderators have the task of supervising the threads and categories[1].

Forum moderation is an important task because of the great amount of discussions that are created. In these discussions, users can degenerate the main point of a topic, produce content that causes lower ranks on search engines, or discourage users from returning to the forum. These are examples of scenarios that need to be moderated to preserve the quality of the forum. A moderator also contributes with useful and user centered content, publishes, teaches and helps other users to follow the rules. Therefore, moderation is an expensive and time consuming task that can be improved with the support of a text analysis method.

Different categories are inherently targeted to certain groups of topics. This contributes to the organization of the information within the forum hierarchy enhancing the overall user experience with the platform.

However, posts are sometimes submitted to the wrong forum. Therefore, human intervention is required to manually classify these posts in the right category. Predicting whether a post is being published in the correct forum instead of its actual location contributes to the auto-moderation of the forum.

[1]In this work, to avoid confusion, we will use the term "forum board" to refer to the entire online discussion platform, and the terms "categories" and "forums" interchangeably to refer to the different topics that are treated in that platform.

Some authors, such as [1] and [2], already tackle the problem of classifying posts within the hierarchy of a forum by reorganizing the posts in clusters and then assigning the topic labels according to a probability matrix. However, this approach uses vector space metrics —such as the cosine similarity— which are computationally expensive for obtaining predictions.

Using text analysis techniques, a recent study addresses another related discussion forum problem that is the inappropriate content, i.e. messages that do not follow the forum rules. They create a model employing partially labeled comments which execute classification tasks determining whether the content is improper or not [3].

Another approach is the analysis of tools for the automatic generation of spam content —spontaneous messages with the aim of gaining new users to their sites— in order to classify new content as genuine or spam, and block the latter [4].

Forums that are well organized, without content regarded as spam and, most importantly, with posts in the right categories, obtain better scores in search engines. Webmasters should consider this point as modern search engines take into account the underlying structure of web pages in order to show more and better information in their results, giving the site owner the advantage of having a website that is easier to find [5].

In this work, we propose an automatic classification method for assessing the most appropriate category in which posts should be submitted based on their content. This task is a sentence classification problem, similar in nature to that of sentence sentiment classification. The general approach for our method is to use word embeddings as an input for a deep learning model that classifies text content with satisfying success rates.

This work is organized as follows. In the Methodology section we show our approach for preparing the input data, train the model and validate it. In the Experiments section we describe in detail the tools used for applying our method. Finally we draw the conclusions reached in this paper and possible lines of work for future research.

Methodology

In this section we present an automatic method for the classification of posts into appropriate forum categories within a forum board. This method processes the text contained in thousands of online posts in order to create a model that classifies new posts into the appropriate forum board category.

This method consists of four steps as shown in Figure 1:

Step 1: Parsing. The words from each post of every forum are extracted, counted, and posts are stored as a list of words.

Step 2: Filtering. Non-representative words from the forums are discarded.

Step 3: Embedding. Posts are converted into a vector representation.

Step 4: Training. the neural network is training using the processed data.

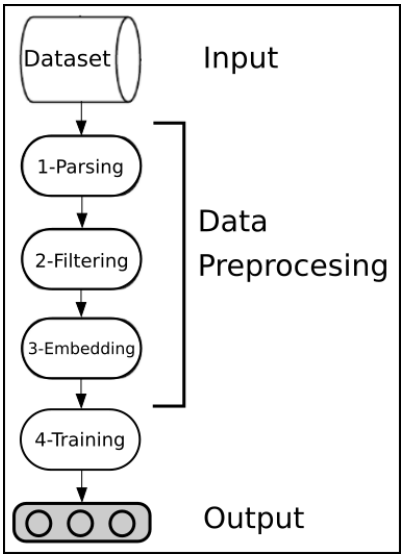

Figure 1: Worflow for the proposed method.

These steps are organized in two stages, data preprocessing and model training. The complete workflow is explained below.

Data preprocessing

In this stage, a simple bag-of-words (BoW)[12] method is used to represent the words of a post in a higher-dimensional, uniform space. This is done in order to transform the posts into a consistent and numeric input for a neural network.The assumption behind this is that two posts are likely to belong in the same category if they have similar content.

The first step in preprocessing the data to make it usable by a neural network consists on parsing the posts contained in each forum, as shown in Step 1 of the method. This is done in order to isolate the words, and eliminate text considered irrelevant like html tags and punctuation marks. During this parsing a *word dictionary* is generated. This is a list which contains every word appearing in the entire dataset, and an integer indicating the number of occurrences of the word in the corpus.

The dataset is processed to determine the vocabulary, and then a filtering process is applied, as shown on Step 2 of the method. This is performed in order to avoid words considered as non-representative of their respectives forums either because they are too frequent in the corpus —words like "the", "a" and "I" which are considered too generic to be used in classifying a post to a forum— or because they are too infrequent, and can be considered either spelling mistakes, errors that were made during processing of the dataset or statistical outliers that do not reliably represent the forum from which they were taken.

After the filtering process, an embedding process is applied as shown on Step 3 of the method. This process involves converting posts from their original representation —a list of words stored as binary strings— to a numeric representation that can be used as an input for a neural network. This is done by representing each post as a multi-hot vector $\mathbf{X}_{D\times 1}$, where D is the size of the word dictionary. The vector consists of values 0 where the corresponding entry in the word dictionary is not present in the post, and values 1 where the entry is present, as shown in the example presented in Figure 2.

Similarly, a one-hot vector $\mathbf{Y}_{C\times 1}$ is generated for each post, where C is the total number of forums. This vector indicates the source forum from which the post was retrieved. The result of this process is the input dataset $\{X,Y\}_N$, where X is the high-dimensional representation of the post and Y is the target category label.

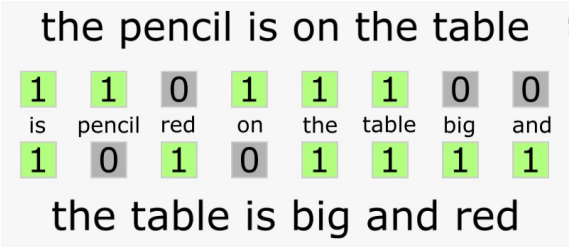

Figure 2: Vector representation of two phrases.

Model training

In this stage, a neural network is trained using the input dataset previously obtained through the Data Preprocessing stage. This neural network is able to perform classification of new forum posts. For each post, the classifier maps each category to a probability, which indicates the likelihood of a post belonging to a that category. The model then yields the labels ordered by their probability. An example is shown in Table 1 with an input post from the Fitness forum category: *"how would dividing interval training into multiple workouts actually affect fitness gains lets assume that the time spent in the training zones remains the same"*.

Forum	Probability
Fitness	0.17
Robotics	0.14
Esperanto	0.13

Table 1: Example of model training output for a post from the Fitness forum category.

Therefore, an effective assessment of categories can be performed on the posts through our method, in order to provide

an automatic means of organizing content within a forum board.

Experiments

We used a dataset made available by StackExchange [6], consisting of posts from 347 different forums stored in XML format. A total of 15 forum categories were randomly selected, consisting of a total of 52500 posts.

The following forum categories were selected: 3d Printing, Chess, Esperanto, Fitness, Freelancing, Health, Law, Pets, Robotics, RPG, Sports, Startups, Vi, Windows Phone and Wood working.

Data preprocessing was performed as follows. For Step 1 or our method, the first 3500 posts of each forum were selected from the input files and the `<body>` attribute of the text was parsed. Then, every punctuation mark and special character was removed. At the same time, a word dictionary was created for each category, which contained a list of every word on the forum, and the number of times it appeared. Then, the information was stored in two binary files per forum using a Python library called *MessagePack* [7]. Then, we merged every word dictionary into a single dictionary which contained the words from the 15 selected forum categories and their corresponding occurrences.

Once the word dictionary was unified, for Step 2 of our method, non-representative words were filtered. First, we analyzed the data and found a highly biased distribution with mean in 28, but a standard deviation of 1228, as shown in Figure 3.

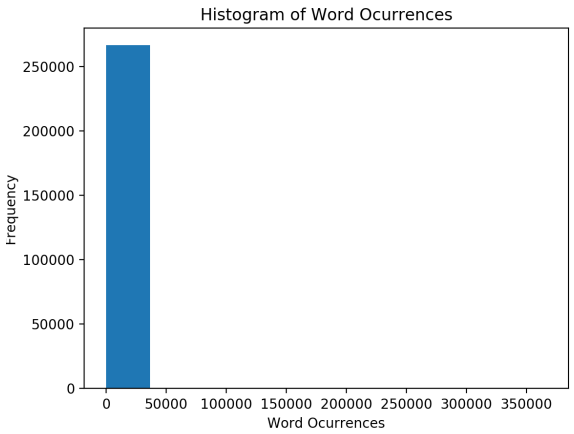

Figure 3: Histogram of word occurrences for the 15 forums.

We observed that the least frequent words —less than 3 occurrences, mostly spelling mistakes— represented more than the 84% of total amount of words. Therefore, these words were removed from the training set.

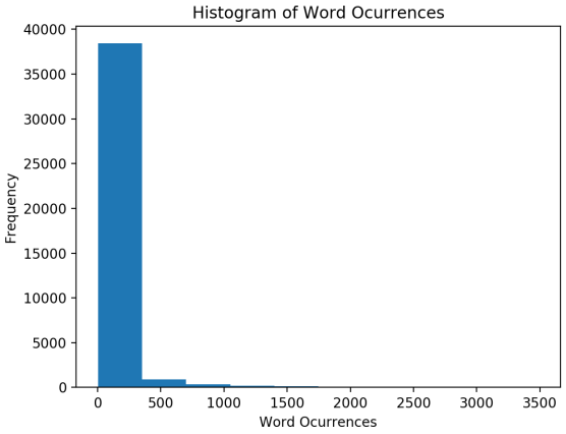

Figure 4: Relative frequency of words in the corpus.

After this first filter, the resultant word frequencies were evaluated and an upper boundary was defined to eliminate the most frequent words, such as prepositions or articles. This chart is shown in Figure 4. We obtained a less biased distribution, with a mean of 61 and a standard deviation of 241. The final dictionary consists of 50376 words.

For Step 3 of our method, being the final step of the data preprocessing stage, we created, using the BoW model from the previously described files, the input data used for training the model. Following a classical machine learning approach, the data was di-

vided via cross-validation into 90% for training and 10% for testing.

In order to classify a post into a certain discussion category, we created an artificial neural network and trained it using the previously processed data. We used Tensorflow, an open source library for Machine Learning [8]. The model consisted of a simple sequential neural network architecture as shown in Figure 5.

The architecture or the network consist in 1 fully-connected ReLu hidden layer of 500 neurons and a Softmax output layer of 15 neurons. The ReLu activation function improves training performance for large and complex datasets and reduces drastically the possibility of gradient vanishing [9]. Predictions were optimized using Adam optimization, due to its high effectiveness in the update of sparse gradients with minimal memory requirement as discussed in [10]. We chose Softmax, a cross-entropy based algorithm, as the loss function because of the probabilistic nature of the classification problem.

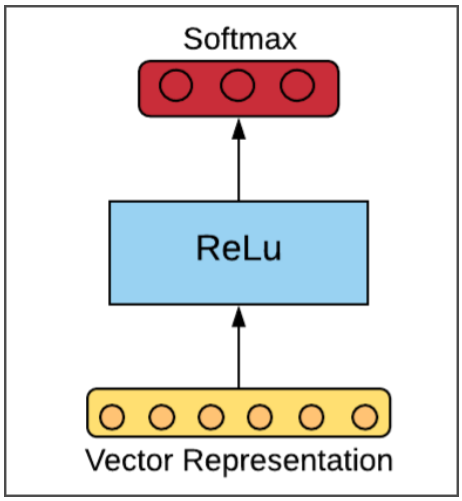

Figure 5: Neural network architecture.

The weights and biases of the network were initialized using random values from a normal distribution with mean 0 and standard deviation 1 [11].

We performed the entire data preprocessing and the neural network training with a computer running 16GB of DDR4 RAM, an Intel® Core™ i7-6700 processing unit at 3.40GHz and a NVIDIA® GeForce GTX 1050 GPU with 2GB of VRAM.

In our experiments, we found that increasing the batch size reduces the overall training time, but reduces the final accuracy during training, despite the loss decline. The batch size was tested with values between 10 and 100 with a step of 10, as shown in Figure 6. Each configuration was ran by training for 100 epochs. The figure shows that as the batch size is increased, the accuracy of the model decreases.

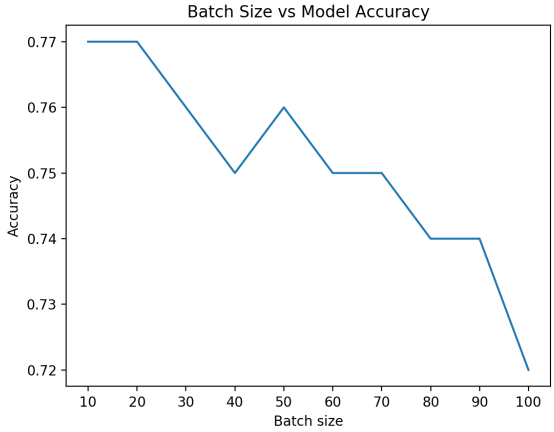

Figure 6: Batch size vs Accuracy of the model.

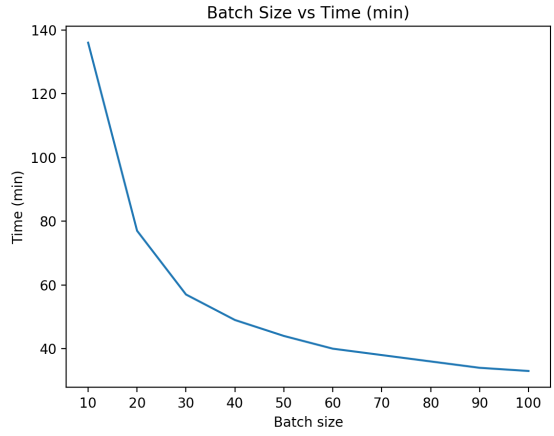

Figure 7: Batch size vs Training time in minutes.

Training time was also measured with every batch configuration tested, as shown in Figure 7. As the batch size was incremented from 0 to 100, the training time was reduced from 136 minutes to 33 minutes. Regarding accuracy, in our particular case, as we increased the number of epochs and the batch size, the prediction accuracy did not exceed

79.9%, which was achieved with 20 epochs and a batch size of 1 in 5 hours of training. However an accuracy of 77% was obtained using a batch size of 10 in more appropriate training time. These results are shown in Table 2.

Batch size	Execution time (H:M:S)	Accuracy
1	4:43:15	0.791
2	1:53:50	0.799
4	1:04:25	0.785
6	0:48:00	0.779
8	0:32:21	0.786
10	0:27:55	0.773
12	0:23:23	0.765
14	0:20:36	0.776
16	0:18:31	0.769
18	0:16:59	0.768

Table 2: Batch size effect in execution time and accuracy.

In the table, the second and third columns show the training time and the obtained accuracy for each batch size in the rows, respectively. In this set of runs the number of epochs was maintained fixed with a value of 20.

Taking into account this information, we selected a batch size of 14 samples, given the excellent combination of accuracy (77.6%) and training time (20.36 minutes). With this configuration, we achieved a training time improvement of 93% while lowering the accuracy by 3%, in comparison to the non-batched execution.

Using the selected batch size, a series of experiments were conducted as shown in Figure 8. Epoch values higher than 16 lead to little or no improvement in the accuracy. Therefore, we selected 16 as the optimal number of epochs.

With this combination of batch size and number of epochs, we ran the model 100 times obtaining an average accuracy of 77%. This is shown in Figure 9. Accuracy values vary between 0.768 and 0.772 along the 100 runs, with a mean of 0.771.

We also discovered that the amount of data used per category is critical for increasing the accuracy of the predictions. In our experiments we started with an initial amount of data consisting of 2 forums with 1000 posts each, and increased its size gradually up to 15 forums with 3500 posts each. In each experiment, we kept the same neural network architecture. The results of the testing is summarized in Table 3. Different datasets and the accuracies they produced when training the model are represented in each row. Compared to the first configuration, the prediction capability of the model was improved by 11%.

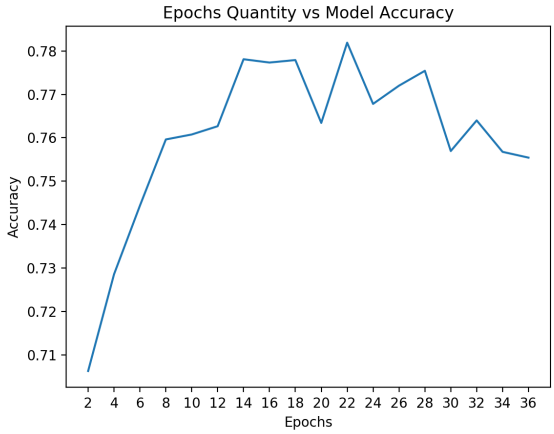

Figure 8: Number of epochs vs. accuracy of the model.

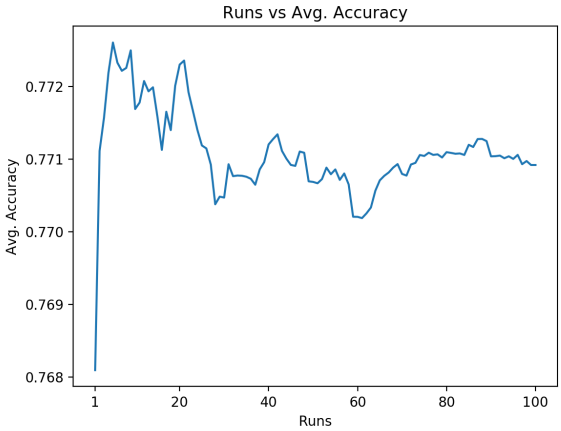

Figure 9: Averaged accuracy of 100 trainings.

Configuration	Accuracy
5 forums 625 posts	0.66
10 forums 1250 posts	0.66
15 forums 1825 posts	0.73
15 forums 3500 posts	0.77

Table 3: Impact of dataset size in model accuracy.

We compared these results with a random based classification, which used the same input data used for training the neural network model but assigned forums taken from a uniform probability distribution. It successfully categorized 6% of the posts into the correct forum. We performed an analysis of variance test (ANOVA) between the residuals of the neural network model and the ones of the random model. Our null hypothesis was that for both models the means were equal, and that they were different as the alternative hypothesis. The ANOVA test yielded results for the p-value of less than 0.001. Therefore we accepted the alternative hypothesis that there is effectively a 400% difference between the means of the models, as shown in Figure 10.

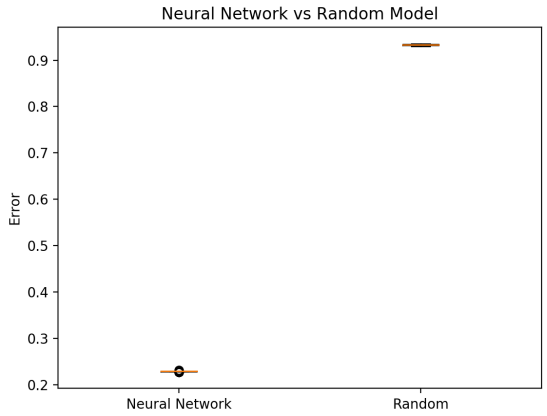

Figure 10: Error means comparison between both models.

We started this work facing the preprocessing of the initial dataset in order to get a clean input for a neural network. We experimented with many types of configurations and different data amounts, and found appropriate parameters through a series of experiments. These were used on a simple neural network architecture which successfully performed classification tasks due to the adequate preprocessing of the dataset and the effective definition of its parameters.

Related work

This work was developed within the project "Minería de Datos aplicada a problemáticas de Big Data". In this context, two other researches have been published. They are "Un Nuevo Método para Clustering de Tweets Basado en Métodos de Ensambles y Técnicas de Hashing" and "Comparative Analysis on Text Distance Measures Applied to Community Question Answering Data".

Conclusions and future work

In this work we presented a method for classifying publications into discussion forum categories based on a deep learning approach. Our method achieved excellent accuracies in a considerably short training time, without the need for large computational capabilities.

For future work we would like to test different neural network architectures, which are very prominent in the text classification field, for example Recurrent Neural Networks with Long-Short Term Memory (RNN LSTM).

We also would like to test this same structure in a Big Data environment, using cluster-computing techniques and an architecture which may allow to reach more accurate results by utilizing higher magnitudes of data.

References

[1] Cerulo, L., & Distante, D. (2013, March). Topic-driven semi-automatic reorganization of online discussion forums: a case study in an e-learning context. In Global Engineering Education Conference (EDUCON), 2013 IEEE (pp. 303-310). IEEE.

[2] Distante, D., Fernandez, A., Cerulo, L., & Visaggio, A. (2014, October). Enhancing online discussion forums with topic-driven content search and assisted posting. In International Joint Conference on Knowledge Discovery, Knowledge Engineering, and Knowledge Management (pp. 161-180). Springer, Cham.

[3] Delort, J. Y., Arunasalam, B., & Paris, C. (2011). Automatic moderation of online

discussion sites. International Journal of Electronic Commerce, 15(3), 9-30.

[4] Shin, Y., Gupta, M., & Myers, S. A. (2011, March). The Nuts and Bolts of a Forum Spam Automator. In LEET.

[5] Lieke, K. Search Engine Optimization. Henry Joutsijoki (toim.), 32.

[6] Stack Exchange (September 2018). Files for stackexchange.https://archive.org/download/stackexchange

[7] Sadayuki Furuhashi (September 2018). MessagePack: It's like JSON, but fast and small.https://msgpack.org

[8] Tensorflow (September 2018). An open source machine learning framework for everyone.https://www.tensorflow.org

[9] Srivastava, R; Masci, J; Gomez, F; Schmidhuber, J.(2014) Understanding Locally Competitive Networks. arXiv eprint arXiv:1410.1165

[10] Kingma, Diederik P.; Ba, J.(2014). Adam: A Method for Stochastic Optimization. arXiv eprint arXiv:1412.6980

[11] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

[12] McTear, M., Callejas, Z., & Griol, D. (2016). The conversational interface: Talking to smart devices. Springer.