Website Classification

Michael Allemann, Dr. Quentin Rossy
EPFL / UNIL, Switzerland, May 2019

***Abstract*—Website classification is a ubiquitous problem. There exist webshops selling illicit substances and contraband openly in the internet. Starting with a dataset containing links and short descriptions of websites scraped from the web, we build a classifier capable of identifying illegal webshops. We do so by labeling data for training, preprocessing the data using natural language processing and classifying using a multilayer perceptron.**

I. Introduction

This report is not meant to be a standalone text. To fully understand the project, the Google tutorial, the jupyter notebook and the slides must be consulted. Text classification is at the heart of many information systems. When a query is sent to a modern search engine, such as Google, often thousands of links are returned. In this project we try to fine tune the process of identifying webshops, that sell illicit merchandise, by classifying the snippet below the link, returned from the search engine. In the following sections of this report, the applied method will be described, followed by the results. The classification method closely follows the following Google tutorial on text classification [1].

II. Labelling

A. Read json

In a first step the given json file is opened as a pandas object. The rows, where the snippet is 'NaN' are dropped and the title and the snippet are merged. The arguments, which were used to query the search engine are split and only the first two are kept. The link is kept as an index and the display link is kept as a column. When this is done, the file is written to csv.

B. Unique display link

In order to label a good proportion of the given 510192 datapoints the fact that many links can have the same display link, but different snippets, was used. For example 'https://www.epfl.ch/schools/' and 'https://www.epfl.ch/research/' are different links, but have a common display link, namely 'epfl.ch', however their snippet will be different. The above created csv file was thus grouped by the display link, a counter was added as a column, which counts the number of occurences of the display link. This procedure reduced the dataset to the size of 14'586. Finally the dataset was sorted by the count, assuming that websites that match the query best, were collected most often.

C. Removing popular websites

Upon inspection of the thus created dataset it can be seen that the display links with the highest counts are largely big websites such as 'youtube.com'. The assumption is made, that big companies have an interest in protecting themselves and thus do not allow illegal activities on their websites. Using a dataset containing the 1mio most popular websites according to Amazon Alexa, the names of the 1000 most popular websites are extracted and datapoints, whose display link contains one of the words in the 1000 most popular, are removed. This further reduces the dataset to the size of 13284 datapoints.

D. Labelling

Next the 500 remaining websites with the highest count were labelled by hand. The label '0' was given for false positives, the label '1' was given for true positives and the label '2' was given for dead links. 357 were labelled with 0, 92 with 2 and 50 with 1.

E. Expansion

50 samples are not enough to train a natural language classifier. To leverage this weakness, the column containing the label of the labelled data was added as a column to the original dataset. The merge was performed on the display link column, so suddenly each link has a label, instead of each display link! After removing all duplicate samples this

resulted in 23510 labelled data samples for learning and 71074 unlabelled samples for prediction.

III. Preparation for classification

The 23510 labelled are split into a training and a test set, with a ratio 80%, 20%. Next the proposed preprocessing from the Google tutorial is applied. The proposal is well in line with what is taught at EPFL [2]. First the tf-idf transform is fit to the train set, removing stopwords, creating ngrams of size 1 and 2 and dropping occurences of less than 2. The term frequency-inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus [3]. This results in a matrix with tens of thousands of features. The authors of the google paper conclude, after empirically investigating the importance of features, that the added information after around 10'000 features plateaus. Thus a further step of feature selection was performed extracting the 10'000 most important features according to their ANOVA f-value [4]. The train, test and unlabelled data all underwent the tf-idf transform followed by the feature extraction using the f-value, whereby the statistics were found using only the train data.

IV. Classification

Now that the data is ready, the classification is performed. In the Google tutorial they recommend calculating the ratio between the number of labelled samples and the average number of words per sample. For the current dataset this ratio is 922.16, which is smaller than 1500, so a simple fully connected multilayer perceptron with 3 hidden layers with each 64 nodes is used. As optimizer Adam [5] is chosen with a learning rate of 0.001. The last layer consists of one node to which the sigmoid function is applied [6]. As a loss function binary cross entropy [7] is used. For regularization dropout [8] is used. Batches of size 100 are fed forward and 100 epochs are performed for training. The training takes around 10 minutes on a 2012 macbook air.

V. Results

The sigmoid function projects all numbers to the interval $[0, 1]$. When the training is done, the test set is fed forward. If the resulting number is larger than 0.5, the label 1 is given and compared to the actual label. The training and test accuracies are both over 99%. There were 15 false positives and 19 false negatives in the test set classification. We can confidently say, that the classification performs well.

VI. Postprocessing

Finally the unlabelled data is fed forward through the network, predicting a label for each link. Again, what actually interests us is the display link, so as before the resulting dataframe is grouped by the display link and counted. Further the ratio between the count and the number of labels '1' per display link is calculated and the resulting dataframe is sorted by this ratio. A high ratio implies high confidence that it is an illegal webshop. 602 display links were classified as being illegal webshops. The 50 which were labelled by hand to be illegal webshops, were appended to these 602 display links and saved to disk.

VII. Discussion

Starting with a dataset of around 500'000 samples we created a datapipeline for processing, labelling, feature extraction and classification, which results in 652 suspicious display links. These display links contain many dead links, which could be removed by hand. The display links could further be inspected by an expert, which would result in a refined dataset. This refined dataset could then be pushed through the pipeline again to get an even preciser model.

References

[1] google.com. Text classification. https://developers.google.com/machine-learning/guides/text-classification/, 2018.

[2] Karl Aberer. Distributed information systems. http://edu.epfl.ch/coursebook/en/distributed-information-systems-CS-423, 2019.

[3] wikipedia.org. tf-idf. https://en.wikipedia.org/wiki/Tf%E2%80%93idf, 2019.

[4] wikipedia.org. F-test. https://en.wikipedia.org/wiki/F-test, 2019.

[5] wikipedia.org. Stochastic gradient descent. https://en.wikipedia.org/wiki/Stochastic_gradient_descent#Adam, 2019.

[6] wikipedia.org. Sigmoid function. https://en.wikipedia.org/wiki/Sigmoid_function, 2019.

[7] wikipedia.org. Cross entropy. https://en.wikipedia.org/wiki/Cross_entropy, 2019.

[8] wikipedia.org. Dropout. https://en.wikipedia.org/wiki/Dropout_(neural_networks), 2019.