IDS703 Final Project Report

Anna Dai, Satvik Kishore, Moritz Wilksch

December 12th, 2021

1 Introduction

In this project, we have worked on tweet classification as a Natural Language Processing problem, more specifically, as a document classification problem. Twitter is a microblogging service where users post publicly visible "tweets", that are essentially texts with less than 280 characters. These tweets may also contain other media objects which are discarded for the purposes of this project. These tweets most often serve as discussion pieces as part of larger conversations. They are relevant to any number of topics under discussion. These "topics" are also often explicitly highlighted by the user using a "hashtag", i.e. text with '#' followed by the topic name, or a commonly used shorthand for it. In our project, we treat these hashtags as "topics" for our document classification model, where each tweet is an instance of a document.

2 Data

We have manually selected 7 topics, or hashtags for classification: crypto, tesla, championsleague, formula1, thanksgiving, holidays, covid19.

These topics were intentionally selected to have topics that have some amount of overlap between them (holidays and thanksgiving) while also having topics easier to differentiate (crypto vs formula1). We scraped data using the python library twint **twint**, scraping approximately 10,000 tweets for each of the seven topics. We also applied a few pre-processing steps before the next steps. This included tokenization of the tweets into words. [TODO: elimination of common words?]. Conversion of emojis into tokens (each appearance of an emoji is a token. Multiple emojis strung together are treated as different tokens in sequence). We also removed punctuation marks. Following these steps, we split our data into three parts using a 60:20:20 split to form a training dataset, a validation dataset, and a test dataset.

3 Methodology

3.1 Workflow

The collected corpus of almost 70,000 tweets is randomly split into a training, validation, and test set in a 60/20/20 ratio. We will train models on the train set, find optimal hyperparameters using the validation set, and report all performance metrics in the *Results* section on the test set, which is not used for any other purpose than evaluating each model once.

3.2 Generative Model

We use a Latent Dirichlet Allocation (LDA) model as a generative model to learn from the corpus we have collected. This type of model does not require any hyperparamter tuning, and thus is trained using a combination of the training and validation dataset. We used the LDA implementation from scikit-learn (Buitinck et al. 2013). It is fit to the corpus to find seven separate topics, as this is the

number of actual topics in the training set. Subsequently, for each inferred topic, we manually inspect the top 50 words that are associated with it to assign it a name. This manual step is necessary, as the order of topics is not preserved. In fact, the LDA model does not even guarantee to find the same topics that were collected in the original data set. This shortcoming will be discussed in the *Results* and *Conclusion* sections. To use the LDA for document classification, we let it infer the topic distribution of each document in the test set and use the $argmax(\cdot)$ function to assign each document the topic that is most prevalent according to the LDA. Finally, the LDA is used to generate synthetic data. For each topic, we sample 10,000 artificial documents, the length of which is sampled from the empirical distribution of tweet lengths each time. Similar to the actual data set, the synthetic data set is also split into a train, validation, and test set.

3.3 Discriminative Model

3.3.1 Model Architecture

- network description - Adam optimizer - Learning Rate found using the Learning Rate range test (= 10 ** -2.5)(Smith 2018)

3.3.2 Training Process

1) hyperparameter tuning on synth data 2) benchmark on synth and real 3) continue training on real data 4) benchmark on synth and real 5) Completely new model: Hyperparametertuning only on real data 6) benchmark on synth and real

4 Results

4.1 Benchmarking on Synthetic Data

	precision	recall	f1-score	support
thanksgiving	0.916	0.969	0.942	1979
formula1	0.986	0.897	0.939	2172
covid19	0.960	0.966	0.963	2002
championsleague	0.957	0.965	0.961	1926
crypto	0.945	0.901	0.923	2017
tesla	0.943	0.985	0.963	1967
holidays	0.924	0.954	0.939	1937

Table 1: Benchmark results of neural net (trained on synthetic data only) on synthetic data

4.2 Benchmarking on Real Data

	precision	recall	f1-score	support
thanksgiving	0.499	0.401	0.444	1920
formula1	0.033	0.037	0.035	1800
covid19	0.183	0.454	0.261	801
championsleague	0.484	0.503	0.494	1899
crypto	0.717	0.317	0.440	4514
tesla	0.326	0.495	0.393	1332
holidays	0.099	0.150	0.119	1403

Table 2: Benchmark results of neural net (trained on synthetic data only) on real data

	precision	recall	f1-score	support
thanksgiving	0.386	0.524	0.445	1539
formula1	0.009	0.012	0.010	1468
covid19	0.797	0.518	0.628	3100
championsleague	0.857	0.542	0.664	3071
crypto	0.543	0.751	0.630	1389
tesla	0.838	0.567	0.676	3036
holidays	0.032	0.164	0.054	397

Table 3: Benchmark results of neural net (trained on synthetic data and real data) on synthetic data

	precision	recall	f1-score	support
thanksgiving	0.814	0.921	0.864	1362
formula1	0.742	0.889	0.809	1691
covid19	0.811	0.798	0.804	2021
championsleague	0.878	0.673	0.762	2578
crypto	0.764	0.782	0.772	1950
tesla	0.798	0.709	0.751	2273
holidays	0.824	0.974	0.893	1794

Table 4: Benchmark results of neural net (trained on synthetic data and real data) on real data

	precision	recall	f1-score	support
thanksgiving	0.395	0.482	0.435	1714
formula1	0.052	0.046	0.049	2217
covid19	0.519	0.350	0.418	2989
championsleague	0.616	0.708	0.659	1689
crypto	0.566	0.553	0.559	1969
tesla	0.636	0.546	0.588	2391
holidays	0.048	0.092	0.063	1031

Table 5: Benchmark results of neural net (trained on real data only) on synthetic data

	precision	recall	f1-score	support
thanksgiving	0.885	0.822	0.852	1659
formula1	0.825	0.810	0.817	2063
covid19	0.865	0.769	0.814	2236
championsleague	0.796	0.844	0.819	1863
crypto	0.862	0.710	0.779	2423
tesla	0.680	0.855	0.758	1608
holidays	0.838	0.979	0.903	1817

Table 6: Benchmark results of neural net (trained on real data only) on real data

	precision	recall	f1-score	support
thanksgiving	0.946	0.979	0.962	2000
formula1	0.986	0.977	0.982	1942
covid19	0.986	0.986	0.986	1976
championsleague	0.986	0.969	0.977	2092
crypto	0.986	0.987	0.987	2014
tesla	0.986	0.976	0.981	2054
holidays	0.967	0.969	0.968	1922

Table 7: Benchmark results of LDA classification on synthetic data

	precision	recall	f1-score	support
thanksgiving	0.150	0.107	0.125	2122
formula1	0.558	0.562	0.560	1975
covid19	0.034	0.022	0.026	2026
championsleague	0.339	0.443	0.384	1541
crypto	0.333	0.304	0.318	1988
tesla	0.412	0.319	0.359	2021
holidays	0.387	0.670	0.490	1996

Table 8: Benchmark results of LDA classification on real data

4.3 Learned Word Embeddings

To verify that the neural network model has learned meaningful word embeddings, we feed it 7 manually chosen words (one for each topic, but never the name of the topic itself) and find the ten closest word in the embedding space using cosine similarity. We find that the word embeddings capture an impressive amount of meaning: The name of Formula1 driver Lewis Hamilton is closest to other driver's names as well as "formula" and "race". "Barcelona" maps to other sports clubs that played in the championsleague. "Vaccine" is close to health- and covid-related words while "Christmas" is similar to "december" and "festive". For "Elon" and "BTC" the closest words are coherent and reffer to Tesla and Cryptocurrencies. Only "turkey" is not clearly close to thanksgiving, presumably as the topic "thanksgiving" has been hard to identify throughout the experiments. TODO: - Do the accuracies support this view?

hamilton	barcelona	vaccine	christmas	turkey	elon	btc
max	bayernbara	immunity	december		musk	cryptos
ocon		vaccines	nicholas	nick	fsd	coins
lewis	zenit	deaths	festive	cotton	teslas	eth
vsc	barca	tests	visit	flex	ev	ssfeed
championship	bayern	measures	snowman	gratitude	supercharger	tether
abu	ucl	boris	pack	nov	binance	io
fia	liverpool	booster	wreath	rosemary	cointrade	opportunity
formula	milan	vaccinated	rainbow	chronicles	giga	dump
race	matchday	vaccination		skyrocket	autopilot	analyzing

5 Conclusion

References

Buitinck, Lars et al. (2013). "API design for machine learning software: experiences from the scikit-learn project". In: *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pp. 108–122.

Smith, Leslie N (2018). "A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay". In: *arXiv preprint arXiv:1803.09820*.