Classification of User Comments into Social Media Groups with Opposing Views

Ozge Lule
Computer Engineering
Middle East Technical University
Ankara, Turkey
luleozge@gmail.com

Baris Ozkuslar
Computer Engineering
Middle East Technical University
Ankara, Turkey
ozkuslar.baris@gmail.com

Esref Ozturk
Computer Engineering
Middle East Technical University
Ankara, Turkey
esrefozturk93@gmail.com

***Abstract*—The goal of this project is classifying user comments into social media groups with opposing views. In the scope of this project, comments on Turkish and English pairs of Facebook pages are classified with Recurrent Neural Networks and Convolutional Neural Networks. Resulting trained models can classify user comments into Facebook pages better than humans.**

***Index Terms*—LSTM, GRU, CNN, RNN, Word Embedding, t-SNE, Binary Classification, Facebook**

I. Introduction

In the scope of this project, Facebook pages of political parties with opposing views are used. Two Facebook political party pages whose opinions are opposite are selected and the comments on these pages are labeled with the page opinion. Previously, we intended to use only the comments from people who liked the page, because if a person has liked a page this would mean that their opinions are likely to be aligned with the opinions of the page, so their comments in this page could also be used for classifying the ideological stance of the user. However, we had difficulties getting like information of a page since Facebook Graph API does not allow getting personal like information. Since classifying in which page a comment is made was also an important problem, we continued with this task. Therefore, this project classifies in which page a certain comment was made.

II. Background and Related Work

A. Background

Before reading this project, one should review the fundamental concepts of the machine learning and also binary classification concepts. Also, recurrent neural networks and convolutional neural networks should be reviewed properly. High-level explanations of them are given here.

Recurrent neural network (RNN) is a type of a neural network that has internal memory that enables it to process sequence input. RNNs have shown great results on many NLP tasks. Long short-term memory (LSTM) block is itself a recurrent neural network with a memory cell that can store information for long-term and is better than a vanilla RNN because it handles vanishing gradients problems better. Gated Recurrent Unit (GRU) is similar to LSTM but do not use separate memory cells.

Convolutional neural network (CNN) is a type of neural network which is space invariant. Unlike artificial neural networks, neurons are locally connected. CNN has been proven to be effective on NLP. Word embedding is a technique used in NLP where words from vocabulary are represented as real number vectors. We have used pre-trained word embeddings as the first layer of CNN to initialize the Embedding Layer of our network.

The main source of the project was the comments on Facebook pages which are crawled through Facebook Graph API. [1] Facebook Graph API is a low-level HTTP-based API that you can use to programmatically query data and perform a variety of other tasks on the Facebook's platform.

B. Related Work

In this classification task, we try to find in which of the two opposing groups a comment was posted. There are no significant research on this certain task but since our task is highly related to understanding stance of a comment, related work for this problem are mentioned in this section.

Over the past years, there have been many research on classifying the stance of the text. But these research are mostly in congressional debates or forums. [2] [3] [4] [5] [6] [7] There is a benchmark dataset of stance detection on Twitter but this research only utilizes vanilla recurrent neural networks. [8] Lastly, a user-topic-comment neural network on Facebook posts to understand comments' stance on a post was proposed in 2016 which only uses English texts. [9]

III. Dataset

Our dataset consists of comments on 2 pairs of political party pages. The first pair is Turkish ones, AKP and CHP. Other is English ones, Republican and Democrat. Total of 400k comments is downloaded, 100k from each page.

To collect the comments, an architecture with 4 servers and 1 database located in West Europe is used. Each server downloaded 100k comments from a page and saved them to the database.

Crawling the comments were done using Facebook Graph API. Firstly, posts of pages are downloaded using "posts" API. Then, the comments of these posts are downloaded using

”comments” API and labeled with the policitical party the page represents.

Facebook comments can be in various formats other than just texts. In this study, we used only text comments. In fact, 100k comments we crawled only includes these text comments.

In order to simplify things while classifying comments, we filtered only letters, dots, and spaces. Moreover, we lowered all letters. For Turkish letters, we converted them to their English equivalents.

We have used word2vec method to get word embeddings for visualizing our datasets. t-SNE method is used for reducing the dimensionality to 2 and visualizing the embeddings to see if relationships of the words were visible by eye. Note that plottings can be different at different runs as loss function of t-SNE is not convex.

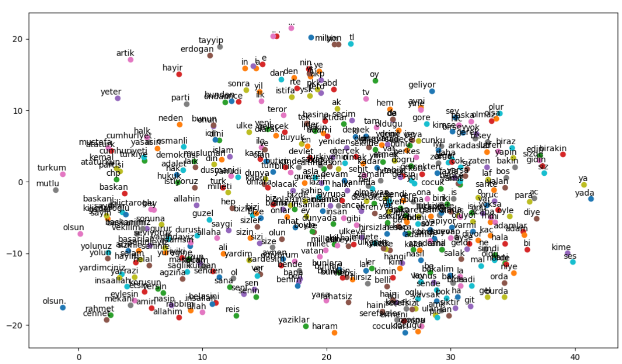

Fig. 1. t-SNE representation of word embeddings from turkish dataset. Only words with minimum count 500 in the dataset are shown here for visualization purposes.

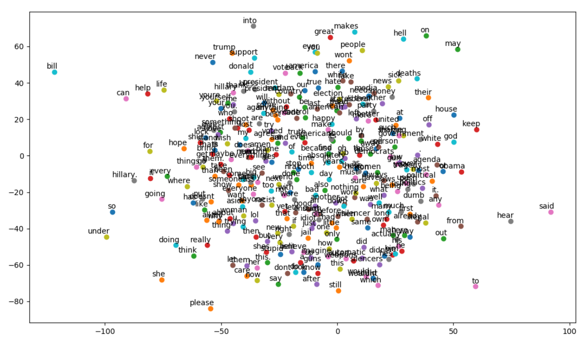

Fig. 2. t-SNE representation of word embeddings from english dataset. Only words with minimum count 500 in the dataset are shown here for visualization purposes.

IV. Methodology

A. CNN with Word Embedding

CNN architecture used in this classification task is as in the figure which is similar to that explained in (Kim, 2014). [10] In order to use word embeddings in this CNN architecture, comments are first turned into word sequences. Since traditional convolutional neural network implementations do not consider variable size inputs, we put a 200-word constraint on sequences and apply zero padding if there are fewer words in a comment. Then we initialize corresponding word vectors from pre-trained word vectors. We use pre-trained word vectors because training data is not sufficient. We used Fasttext English word vectors (300 dimensions) for comments on English pages and Fasttext Turkish word vectors (300 dimensions) on Turkish pages.

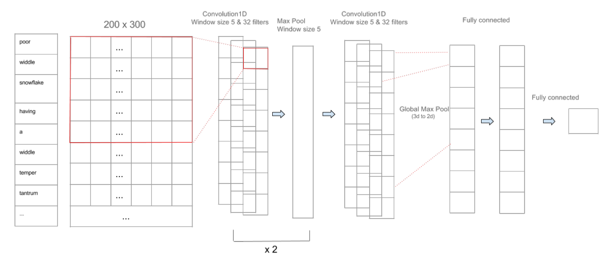

Fig. 3. Architecture of CNN with Word Embeddings

As can be seen from Figure 4, first we have an embedding layer. Since we initialized weights with vectors from pre-trained word vectors, we set this layer frozen (non-trainable) to keep embeddings fixed. One dimensional convolutional layers have kernel size 5 and there are 32 filters in these layers. After convolution, there is a max pooling layer with filter size 5. We have two conv1d-max pool layers and after there is again a convolutional layer. After the last convolutional layer, we have global max pool to reduce dimension to 2 from 3. After reducing to features to one dimension, we have a fully connected layer with dropout. Dropout here is to prevent overfitting and it has a rate 0.2. At the end, we have a sigmoid function for one output. In this network, Adam optimizer is used.

B. Character Level RNN

Recurrent models are very suitable for text data because of their ability to keep state information through time. Considering that different groups(mainly political parties in our case) might have different follower distributions with respect to geography, education, dialect, we hypothesized that there can be differences of writing the same word on character level. For that reason, we decided to use character level recurrent networks.

We used two variants of recurrent neural networks, LSTM and GRU. However, in our initial tests, LSTM usually performed better than GRU by a little bit, so we mainly used LSTM in our following experiments.

Considering the tradeoff between temporal memory depth and training time, we started using 80 timesteps in our recurrent models. This configuration took around 1 hour to train in a GTX1070, and it's theoretically able to memorize 80 characters. The average length of comments was around 70 for dataset 1 and 77 for dataset 2, so the model was sufficient for most of the data. Shorter comments were 0-padded at the end, longer comments were trimmed to 80 characters. However, we have decreased timesteps from 80 to 40 in our following tests and didn't see any decrease in performance. Since the training time decreased to nearly the half, we settled on using 40 timesteps in the end.

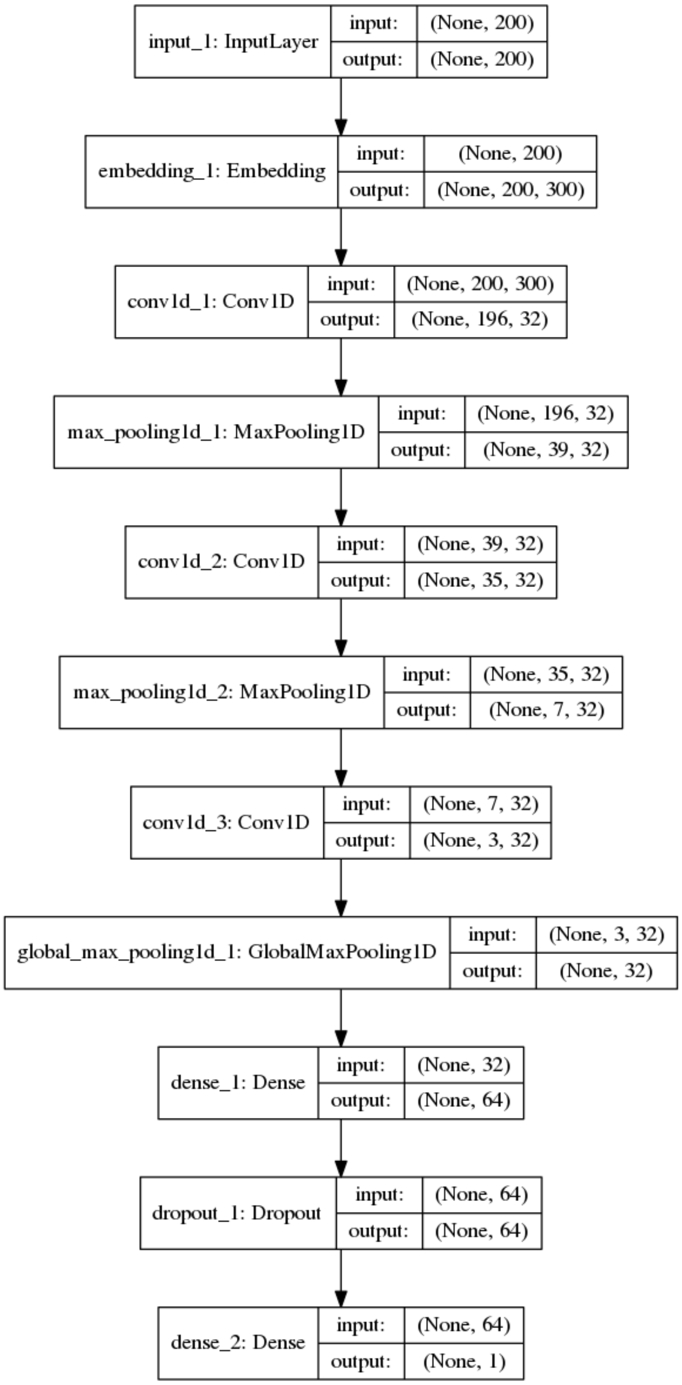

Fig. 4. Keras generated architecture of CNN with Word Embeddings

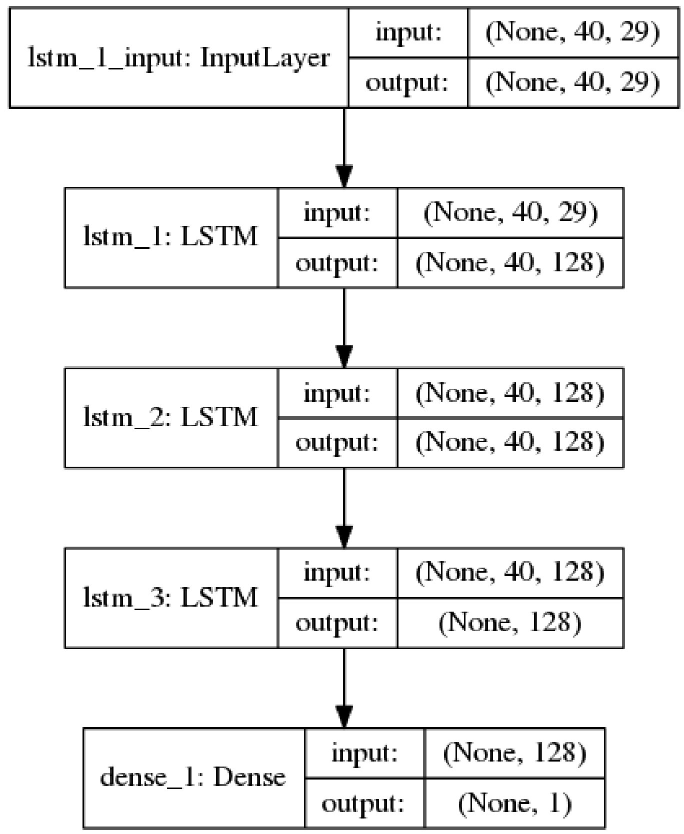

Fig. 5. Architecture of LSTM model

As can be seen from Figure 1, this model consists of 3 LSTM layers with 128 neurons each and one fully connected layer with one neuron. We saw that shallower LSTM nets were not able to reach 80% accuracy. They quickly converged to a point in 70-75% range, then started to overfit with more epochs. Higher layer counts than 3 weren't successful either(probably due to the size of the dataset being insufficient), so we decided on 3 layers for our experiments.

First two LSTM layers pass sequential output to the next layer, the last LSTM layer passes the output from the last step to the fully connected layer. Fully connected layer outputs one value with sigmoid activation. GRU architecture is the same except the usage of GRU unit instead of LSTM.

100K comments from both sides in the dataset were given as input to the network, each of which included 40 characters.

One hot encoding was used to represent each character was represented with a 29-length binary vector.

V. Result and Discussion

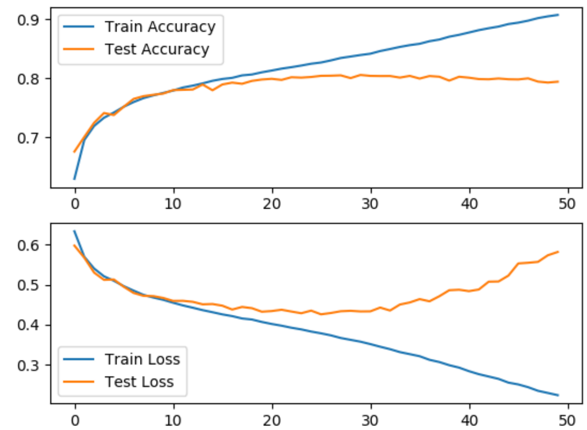

Fig. 6.

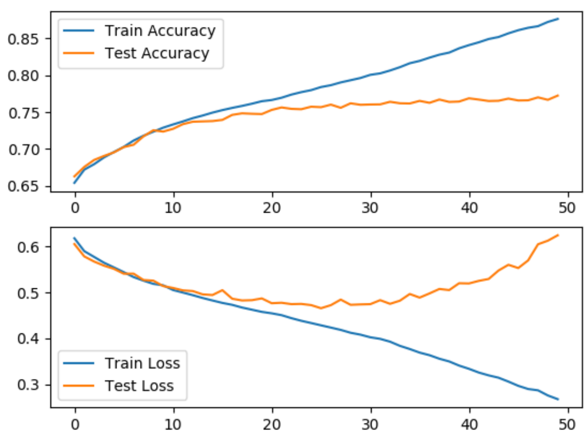

Fig. 7.

TABLE I
Accuracy

Dataset	LSTM	GRU	CNN	Human
AKP/CHP	82	80	75	75~
Republican/Democrat	76	-	78	-

Note: Human accuracy was calculated for a small subset of the data.

80~% success may appear low for a binary classification problem. However, it's important to note that human accuracy in this classification task is around 75%. Therefore, it's fair to say that trained models are successful in the task compared to human performance.

Dataset	Method	F1 Score	Precision	Recall
AKP/CHP	CNN	0.7144	0.7953	0.6512
AKP/CHP	LSTM	0.7963	0.8160	0.7778
Republican/Democrat	CNN	0.7715	0.8052	0.7464
Republican/Democrat	LSTM	0.7599	0.7706	0.7498

As mentioned above, GRU wasn't used in all our tests. It reached 80% accuracy in Turkish dataset, it's also included in the table above.

While LSTM was better in classifying Turkish dataset, CNN did better in English dataset. This result might be related to the highly agglutinative [11] nature of Turkish language compared to analytical English language. A character based model may be able to capture the relation between the different variants of a word better in Turkish since the root-word and morphemes usually stay unchanged in an agglutinative language, which would let a recurrent model share the temporal memory parameters easily for the different inflections or derivations of a word. In contrast, root-words are changed more frequently in English during morphological operations, which could diminish the importance of character level memory.

TABLE II
Examples from human-model comparison

Sentence	Model	Human	Ground Truth
hayir	CHP	CHP	CHP
evet	AKP	AKP	AKP
hirsizlik yapa yapa aldilar yine	CHP	CHP	CHP
insanlarin dini duygularini somurmenin s	AKP	CHP	AKP
cok guzel konustu	CHP	AKP	CHP
ayakkabi kutusu acildi	CHP	CHP	CHP
akp gemiyi batirdi cmhuriyet ha	CHP	CHP	CHP
yasasin sultan recep tayip erdogan. reis	AKP	AKP	AKP
who is voting for this idiot	Rep	Rep	Rep
her ignorance is showing. theyre suppres	Dem	Dem	Dem
every day i thank god that you are our p	Rep	Rep	Rep
yes	Rep	Rep	Dem
yes	Rep	Rep	Rep

Neural Activations in LSTM

After the training, some neurons may gain an "understanding" of certain structures in the language. Here, we are sharing several examples of the images below. The method we used is simply mapping the activations of a neuron hidden state for each character (-1, 1) to red-blue color spectrum. Since these are taken from intermediate layers, the activations can't be interpreted as the classification result, so being blue-ish and red-ish means the same thing in this context. This work was adapted from Karpathy's work. [12]

Important Note: Following images contain randomly sampled text from the dataset. Text content might be offensive.

The figure above is an example to an emergent understanding of a word. Word "osmanli" may be important in this

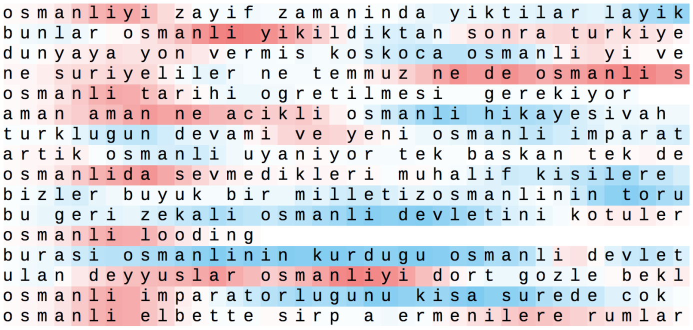

Fig. 8. Layer 2 Neuron 39 - Activations in several comments including "osmanli"

context due to the difference in usage frequency between two parties.

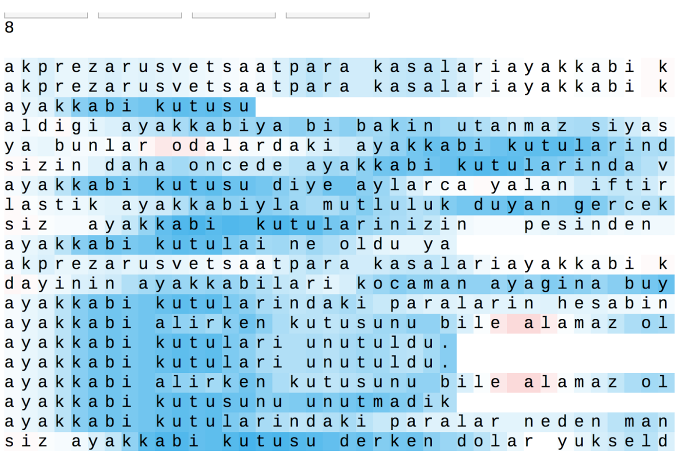

Fig. 9. Layer 2 Neuron 8 - Activations in several comments including "ayakkabi kutusu"
The figure above shows that the neuron gained an understanding of the phrase "ayakkabi kutusu". Words alone doesn't mean anything of significance and the words combined has a different political meaning

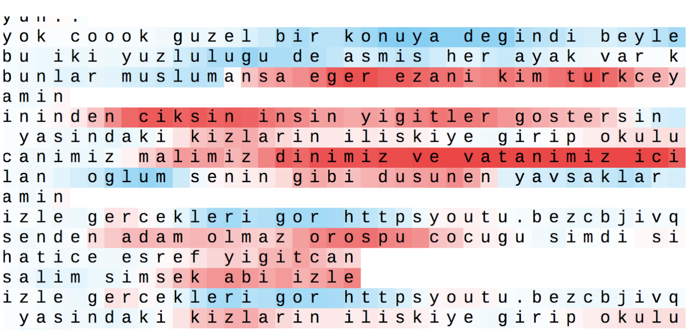

Fig. 10. Layer 2 - Activations of a neuron

As seen from the Figure 10 and Figure 11, layer 1 neurons activate and de-activate quicker in temporal space compared to layer 2 neurons.

VI. Future Work

We believe the work we have done in the scope of this project can be extended to many other social media. In fact,

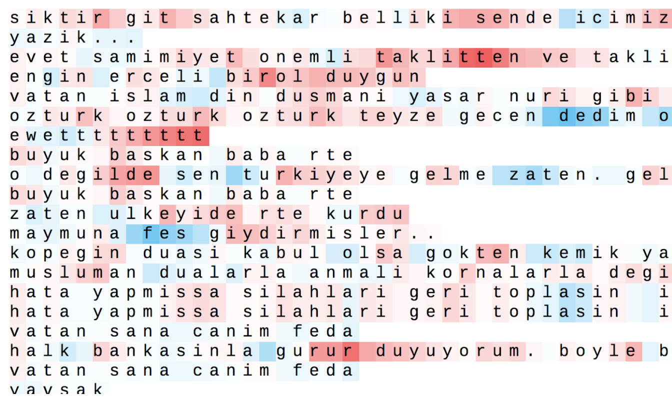

Fig. 11. Layer 1 - Activations of a neuron

utilizing capabilities of other social media can provide us with better information to understand social affinities of people. For example, Facebook Graph API does not give the list of likers but Twitter permits getting the list of followers.

Also, rather than providing only comment information to the model, some other contextual information can be added to classification. These can be a Facebook post that comment was made or information about the commenter. For example, the specific topic of the post text or age, gender of the commenter can be included in the classification.

VII. Conclusion

Comments on Facebook pages have some characteristics that can be distinguished from each other. This holds true for both Turkish and English languages. Due to their high memorization capacity, carefully trained deep learning models can surpass human performance on this prediction task.

All in all, computers can predict the page of a person's comment with high accuracy, better than humans. However, this is still not enough for fully determining the person's ideological stance. In order to enhance the prediction, more information about the person and their activity should be used with neural networks.

References

[1] API Graph. Facebook developers, 2016.
[2] Swapna Somasundaran and Janyce Wiebe. Recognizing stances in online debates. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 226–234. Association for Computational Linguistics, 2009.
[3] Akiko Murakami and Rudy Raymond. Support or oppose?: classifying positions in online debates from reply activities and opinion expressions. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 869–875. Association for Computational Linguistics, 2010.
[4] Pranav Anand, Marilyn Walker, Rob Abbott, Jean E Fox Tree, Robeson Bowmani, and Michael Minor. Cats rule and dogs drool!: Classifying stance in online debate. In *Proceedings of the 2nd workshop on computational approaches to subjectivity and sentiment analysis*, pages 1–9. Association for Computational Linguistics, 2011.
[5] Marilyn A Walker, Pranav Anand, Robert Abbott, and Ricky Grant. Stance classification using dialogic properties of persuasion. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 592–596. Association for Computational Linguistics, 2012.

[6] Kazi Saidul Hasan and Vincent Ng. Stance classification of ideological debates: Data, models, features, and constraints. In *IJCNLP*, pages 1348–1356, 2013.

[7] Dhanya Sridhar, Lise Getoor, and Marilyn Walker. Collective stance classification of posts in online debate forums. In *Proceedings of the Joint Workshop on Social Dynamics and Personal Attributes in Social Media*, pages 109–117, 2014.

[8] Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiao-Dan Zhu, and Colin Cherry. Semeval-2016 task 6: Detecting stance in tweets. In *SemEval@ NAACL-HLT*, pages 31–41, 2016.

[9] Wei-Fan Chen and Lun-Wei Ku. Utcnn: a deep learning model of stance classificationon on social media text. *arXiv preprint arXiv:1611.03599*, 2016.

[10] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[11] George W Stocking. *The ethnographer's magic and other essays in the history of anthropology*. Univ of Wisconsin Press, 1992.

[12] A Karpathy. The unreasonable effectiveness of recurrent neural networks, 2015. *Disponıvel em http://karpathy. github. io/2015/05/21/rnn-effectiveness*, 2016.