Liberté, Égalité, Fraternité
Stance Detection in French Language Tweets

Michael Baltz: mbaltz@usc.edu Rachita Pradeep: rachitap@usc.edu
Stephen Magro: smagro@usc.edu Ted Monyak: monyak@usc.edu

I. Division of Labor

All team members worked on collecting and labeling data for the project. Stephen and Ted worked concurrently to develop and test the recurrent Neural Network. Stephen was responsible for the development of the fastText model and designing the RNN structure. Ted evaluated performance and debugged issues with this code. Michael and Rachita worked together on the word2vec implementation. Rachita gathered tweets and created the pre-computed vector models for the classification algorithm. Michael wrote the logistic regression classifier which used this model. All team members wrote similar-length segments of the final report.

A. Word Count

Word Count: 2005

II. Introduction

One of the most talked-about stories from the 2016 U.S. Presidential Campaign was the emergence of fake news across the internet. The current French Presidential campaign, which will conclude in May 2017, resembles the U.S. campaign in several ways, most notably with the rise of a populist right-wing candidate in Marine Le Pen, and the prevalence of news sharing on the internet.

Most efforts to combat fake news have been focused on English, including the system recently built using the Emergent data-set [1]. However, there have been few systems tailored for French, and due to the timing of their election, this is where we chose to focus. Due to the complexity of the fake news detection task, we investigated a sub-task: stance detection. Stance detection is the determination of an authors opinion on a topic, and is a helpful first step for general fake news detectors. Our goal was to classify the opinion expressed by a body of text relative to a topic as being in favor of, against, neutral, or unrelated.

In late 2016, the Fake News Challenge was announced [2], to foster the development of tools that can assist in fake news detection. The first phase of this competition is aimed at the stance detection of the relationship between an article headline and an article body. However, due to the absence of a reliable data-set for fake news articles in French, we focused on determining the stance of standalone tweets on a given topic. Twitter contains primarily opinionated bodies of text, and a high volume of sharing, so it makes an ideal target for our system.

In this paper, we propose two methods for stance detection: a logistic classifier trained on a word2vec model, and a recurrent neural network that uses transfer learning, as inspired by an entry [3] to the SemEval-2016 competition on stance detection, which demonstrated a successful recurrent neural network for English stance detection even with a small amount of training data.

III. Materials

A large volume of French language tweets were collected from Twitter using the Twitter Streaming API. 250 MB of tweet text was collected, which consisted of around 1.8 million tweets - many regarding the 2017 French presidential elections. The properties to create a good labeled stance data-set, is to use a target that is commonly understood by a wide range of people and get significant amount of data for all four classes - agree, disagree, neither and unrelated. The collected data was annotated for the topics “Je soutiens Marine Le Pen” (“I support Marine Le Pen”), “Je soutiens François Fillon” (“I support François Filon”) and “Je soutiens Emmanuel Macron” (“I support Emmanuel Macron”). It is important to note that the target need not be explicitly mentioned in the tweets. The tweets that depicted favorable stance were labeled as agree, those that depicted unfavorable stances were labeled disagree and the tweets lacked evidence for favor or against were labeled “neither”. A neither stance of a tweet does not indicate that the tweeter is neutral towards the target, it just means the stance could not be deduced from the tweets. Tweets that were unrelated to the target were labeled as “unrelated”. Retweets and tweets which consisted of only a link to an external page were excluded from the data-set. The final labeled data consisted of 254 agree, 260 disagree, 142 neither, and 65 unrelated tweets. For this experiment we decided against uniformly distributing labeled data by class in favor of keeping the class distribution we mined from Twitter.

IV. Procedures

We present two stance detection implementations - a word2vec Logistic Regression implementation and Recurrent Neural Network implementation - as well as a baseline classifier. As a general tokenization of tweets used in this experiment we used the nltk tweet tokenizer to segment our data [4].

A. Baseline Classifier

To establish performance of our various methods we developed a baseline classifier. The classifier is similar to a four

class Naive Bayes classifier in which the word probability priors are calculated from the training set. For the test data, candidate probabilities are calculated for all four classes using the precomputed priors given the probability of each label for the candidate. The label with the maximum probability for a given candidate is taken as the predicted value for a tweet. Laplace smoothing was implemented for previously unseen values.

B. word2vec Logistic Regression

The `word2vec` [5] approach was informed by [6], [7]. The `word2vec` logistic regression model was implemented using the gensim [8] library, a Python library for analyzing plain-text documents for semantic structure. The model uses 4 main parameters for training: `size`, `window`, `min_count` and α. The dimensionality of the feature vectors is set to 700. `window` is the maximum distance between the current and the predicted word in a sentence which is set to 4. The `min_count` is minimum frequency below which the word is ignored - is set to 2 which helped to remove URLs from appearing in our model. The initial learning rate, α, is set to 0.025. The `word2vec` model is trained using the 250MB of French tweets and the final model consisted of around 60000 word vectors.

This model was then used to compute the cosine similarity between a tweet and a topic. A multiclass logistic regression model was trained, using the `scikitlearn` [9] Python library, on the the cosine similarities between the tweet and topic word vectors. This model was used to predict class labels for test data. This method was drawn for the analysis of new articles in which cosine similarity is computed between news headline and news articles, however, in our experiment we used a tweet and a topic statement and computed the cosine similarities between their respective cosine similarities. The logistic regression utilizes the `lbfgs` solver and a `multinomial` loss fit for the across the probability distribution. In the the event of a token not appearing in our vector model, this data point is treated as an error by the classification and is ignored by the cosine similarity, however, this is often only the problem for tokens with one occurrence in the entire data-set. All tweets have tokens that appear in the vector model.

C. Recurrent Neural Network

Our final approach is a Recurrent Neural Network built using the `Keras` [10] Python library on top of TensorFlow [11]. The RNN is built using two LSTMs (one for the topic input and one for the tweet text input). The input to the topic LSTM is a string describing the topic, such as "Je soutiens Marine Le Pen" ("I support Marine Le Pen"). This allows the model to extract some meaning from the topic so that the resulting model can classify tweets for multiple topics. Before each LSTM is an embedding layer where each word in the topic/tweet is converted to a 100-dimension word vector using a fastText [12] model trained on a large corpus of French language tweets that we collected. There is also a convolutional layer that operates on the tweet embedding, that is able to extract some more information from the tweet. The outputs of the topic and tweet LSTMs and the convolutional layer are concatenated and connected directly to a densely connected layer of 4 neurons. This final layer represents a one-hot encoded class that classifies the tweet's stance. This structure was arrived at experimentally and proved to be the best performing architecture that we attempted. Other attempts included training a separate network for each topic, but that resulted in lower accuracy because there was less data for the model to learn from. The neural net was trained using a dropout rate of 0.2 on all layers to prevent the model overfitting our training data.

V. Evaluation

A. Classification Evaluation

For our results we used a k-fold cross validation scheme with k value of four, reporting the accuracy, precision, recall, and f1 for each fold as well as the average and standard deviation across all folds. These methods are common to evaluate the efficacy of supervised classification algorithms and in referenced sources similar metrics were also used.

B. Labelling Evaluation

Our data-set was pulled tweets from the Twitter API [13] using Tweepy [14], and the data was labeled manually by group members. Using this API we were able to collect tweets targeting specific topics in the French language that were tweeted from the geographical bounding box around France. For manual evaluation tweets were run through Google Translate and team members used this English translation and the context of the tweet to provided an label for each data point. Each tweet was reviewed by three team members and only tweets that had a majority consensus regarding the label were included in the final data-set. Retweets and duplicate posts were excluded from the labeled data-set to ensure that data points were unique. We considered using hashtags as a noisy labeling method, but ultimately these results were excluded from this assessment for accuracy reasons as hashtags we found were often used ironically (or simply as a way to categorize a tweet to a topic, not to take a stance) and thus are not a good indicator of stance.

VI. Results

K-Fold CV Result (k = 4)

Metrics	Baseline	Logit Regression	RNN
Accuracy	46.88% +/-1.61%	42.72% +/-1.57%	50.15% +/-4.80%
Precision	50.07% +/-3.07%	35.20% +/-4.44%	52.79% +/-5.52%
Recall	46.88% +/-1.61%	42.72% +/-1.57%	40.31% +/-4.60%
F1 Value	0.45 +/-0.02	0.36 +/-0.01	0.47 +/-0.04

Classification of "Macron est le pire président pour la France" ("Macron is the worst president for France")
on the topic "Je soutiens Emmanuel Macron" ("I support Emmanuel Macron")

Method	Agree	Disagree	Neither	Unrelated
Baseline	3.2e-20	**1.4e-20**	2.1e-19	3.7e-20
word2vec	N/A	**N/A**	N/A	N/A
RNN	0.19	**0.32**	0.30	0.18

Of the methods presented in this paper the results from the recurrent neural network had the best accuracy for tweet stance detection. This result is consistent with trends from SemEval 2016 [3] in which many submissions for a similar stance detection task on English tweets used neural networks. The logistic regression technique drew inspiration from work done with the Emergent Info data-set [1], however, this paper applied this algorithm to stance detection in new articles achieving an accuracy of 73%. We attribute the discrepancy in our results for this method to the shorter lengths of our data points, as tweets are limited to 140 characters. This means for a tweet there were fewer word vectors with which to compute the cosine similarity than the average news article in the Emergent Info data-set [1]. The baseline classifier slightly outperformed the word2vec classification method.

VII. Discussion

Inspired by similar research and systems using English data [3], we have shown that a recurrent neural network using transfer learning is the most effective method out of the three that we proposed at stance detection on French tweets. Therefore, we have shown that this methodology translates well between languages, an important finding in the international efforts to combat fake news independent of language.

The word2vec model was limited in its success, we believe, due to the brevity of individual tweets. We hypothesize that the use of the Emergent data-set [1] was more effective due to the longer word vectors in entire article bodies. It did not surpass the performance of our baseline model, which employs a Naive Bayes implementation to classify stance. However, the baseline performance was still somewhat low, and we hypothesize that this once again has to do with the brevity of tweets creating sparse word probability models.

Due to the manual nature of our annotation methodology, we were limited in the amount of data we could train our models on. With more time and resources, we hope to automate our system for annotating tweets to build a larger corpus. Furthermore, in future research, we intend to adapt our system so it may be used on headline and article body pairs, which was the original problem described in phase one of the Fake News Challenge. This adaptation would make our system most useful for fake news detection, which could enable it to help combat the fake news problem in future French elections. This adds another layer of complexity to the research, in compiling a sizeable French data-set such as the Emergent data-set [1].

References

[1] William Ferreira and Andreas Vlachos. Emergent: a novel data-set for stance classification. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016.

[2] Fake news challenge stage 1 (fnc-i): Stance detection.

[3] Guido Zarrella and Amy Marsh. MITRE at semeval-2016 task 6: Transfer learning for stance detection. *CoRR*, abs/1606.03784, 2016.

[4] NLTK. Nltk 3.0, 2017.

[5] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[6] Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. A dataset for detecting stance in tweets. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, may 2016. European Language Resources Association (ELRA).

[7] Parinaz Sobhani, Saif Mohammad, and Svetlana Kiritchenko. Detecting stance in tweets and analyzing its interaction with sentiment. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics, *SEM@ACL 2016, Berlin, Germany, 11-12 August 2016*, 2016.

[8] Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. http://is.muni.cz/publication/884893/en.

[9] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[10] Franois Chollet. keras. https://github.com/fchollet/keras, 2015.

[11] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[12] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

[13] Twitter. Rest api documentation, 2017.

[14] Joshua Roesslein. Tweepy read the docs, 2009.