Political Alignment Prediction

Kobee Raveendran
University of Central Florida
kobee.raveendran@knights.ucf.edu

Abstract

This work is an attempt to incrementally improve on a similar work related to classifying users of social media sites by political alignment. Unlike some similar previous approaches, this work attempts to use the context of a user's speech to generate predictions that align more accurately with their true political leaning. In this report, all experiments are carried out using data scraped from public Reddit posts in a variety of forums denoting the target political audience. A wide range of recent and foundational natural language processing methods, from multinomial naive Bayes to transformers such as XLNet (Yang et al., 2019), are evaluated for classification performance on real user comments.

1 Introduction

In recent years, social media has built a monopoly over the space of ideas. This is no less true in the realm of politics. Unlike in past decades, when much of a person's political ideology could be traced to popular, mainstream media outlets, many people now derive their thoughts on politics from more diverse and unfiltered sources. Often, political news and ideas spread farther and faster from social media sites like Twitter, Facebook, and Reddit than the established news outlets. This explosion of information dissemination that can spawn from almost any source has led to a change in the political climate and the *range* of political ideas. Categorizing the alignment of the populace has thus become a much trickier, yet potentially more necessary task, especially with the influence of targeted online advertising on voter opinions.

In its implementation, a task such as this is not new. In the realm of natural language processing, text classification is a well-known and established task. Many methods exist for categorizing documents into any number of classes, drawing on a variety of contextual features from the training documents to make these decisions. However, from my exploration, political classification resources that could accurately determine a person's political leaning using the context of their messages were scarce or incomplete. In particular, those that could go beyond misleading or surface-level "keyword" features that may signal the alignment of the user did not seem prevalent.

For example, in the approach that spawned this work (0xTiger, 2020), the method takes as features the *number* of comments made by a user on Reddit within certain "subreddits," and uses a logistic regression classifier to predict their political alignment. Methods that consider indirect features such as these are often prone to sampling bias resulting from misleading assumptions of the environment in which the user participates in. Such assumptions usually lead to mis-classifications purely on the basis of a user's beliefs straying from those of the rest of the subreddit's, yet still being classified as if their beliefs aligned.

Even methods that take direct input from the user, such as sites that employ "decision tree"-like questionnaires, base their classifications on generalized factors that have varying degrees of correlation to political alignment, such as income, career prospects, location, and lifestyle. The goal of this work is to focus classification on the context of a user's speech, taking advantage of textual features that can more accurately reveal their true political alignment.

2 Related Works

2.1 Reddit Stance Classifier

The Reddit Stance Classifier (0xTiger, 2020) is a recent project that brought political alignment classification to the social media forum Reddit. In that work, the author scrapes the comment

histories of over 20,000 users in the subreddit `r/politicalcompassmemes`. In this subreddit, users can self-assign tags, called "flairs" that denote their political alignment. The flairs capture a range of ideologies on a two-dimensional grid, ranging from "left" to "right" on the x-axis, and from libertarian (abbreviated "lib") to authoritarian (abbreviated "auth") on the y-axis.

2.2 Predicting the Political Alignment of Twitter Users

This work (Conover et al., 2011) tackles a task similar to that of this report, but in a different social media platform, which carries with it differences in the intricacies of textual data. In one approach described in (Conover et al., 2011), several models like support vector machines are trained on content-based features from the metadata of tweets from 1,000 Twitter users.

The aim of their method is much simpler in complexity than (0xTiger, 2020), and aligns much more closely to my experiments, as the models are only tasked with classifying tweets as "conservative" or "liberal," rather than in ranges between the two leanings like in (0xTiger, 2020).

However, one primary difference between their work and this one lies in the environment from which text data is mined. On Twitter, "tweets" are public posts made by users consisting of text, images, or both. An important property of tweets is that a hard character limit is enforced on them. Tweets with a character count greater than the limit of 140 characters must be split into multiple tweets, and as such most tweets stay below this limit, typically by extensive use of abbreviation (especially common in hashtags). Because of the turbulent, ever-changing nature of such abbreviations and hashtags, building classifiers from the metadata of such tweets would need constant re-training, even moreso than the re-training needed to keep up with a changing political climate.

3 Methods

3.1 Multinomial Naive Bayes

Multinomial naive Bayes (multinomial NB) is a probabilistic text classification model that makes use of word frequency in a document to perform classification. Multinomial NB makes several assumptions of the text features: it assumes that the features are independent of other features present in the document, and that different instances of such features are equal in weight. As such, multinomial NB tends to derive little value from semantic details such as the nuance and complexities of sarcastic speech. Instead, it focuses on the frequency of words in a bag-of-words representation (Zhang et al., 2010) of the document to determine the probability of a document belonging to a certain class.

3.2 Support Vector Machines

The goal of support vector machines is to find the best fitting hyperplane that evenly divides data samples belonging to a set of distinct classes. To find this hyperplane, an SVM uses support vectors. Support vectors are vectors that denote the instances of each distinct class that are closest to the current hyperplane, and the distance between these support vectors is called the margin. The training scheme of an SVM involves adjusting these support vectors such that the maximum margin is achieved, thus optimally dividing the classes.

3.3 Adaboost

Adaboost (Freund and Schapire, 1997) is one of the earliest yet most widely-used practical implementation of a boosting algorithm. Boosting is a machine learning approach in which numerous weak classifiers (i.e. those that may only realistically yield random-level classification performance) are combined and sequentially trained to yield a final, robust classifier. At each iteration of training these weak classifiers, the resulting classifier is more robust to the misclassifications of the previous iteration's classifier. Thus, it avoids some of the crippling issues of other aggregated approaches such as bagging, where mutual misclassifications by several weak learners can lead to those misclassifications propagating up to the final classifier.

As explained previously, Adaboost is a culmination of several weak learners. In this report, the weak classifier chosen is the decisision tree, primarily due to its simplicity (one of the requirements of choosing a classifier to use with Adaboost). Decision trees are simple, rule-based classifiers, making them ideal candidates.

3.4 Long Short-Term Memory and Gated Recurrent Units in RNNs

Recurrent neural networks (RNNs) are a class of artificial neural networks especially suited to natural language tasks, as they have an innate sequential structure that many other types of networks lack. With such a structure, RNNs are able to consider

previous features (words) and even an abstract "history" of an input sequence when predicting the next word in the sequence or classifying a sequence.

However, RNNs tend to suffer from the problem of vanishing or exploding gradients when given longer sequences. By the time the network has processed the end of a long sequence, it will likely have "forgotten" the context and history from earlier parts of the sequence; a direct result of gradients becoming far too small to cause change in backpropagation, or having weight updates from certain later layers become so large that they become dominant over others.

To remedy this, long short-term memory (Hochreiter and Schmidhuber, 1997) cells and other similar approaches, such as gated recurrent units (Bahdanau et al., 2014), were proposed. These methods were better adapted to longer sentences because they keep a memory component within layers, and selectively choose which information in the accumulated history to "remember" and "forget." This property also made them well-suited for modeling long-term dependencies between words in a sentence, leading to better semantic understanding of context and improved results in sequence prediction.

3.5 Transformers

Transformers (Vaswani et al., 2017) are a relatively recent class of encoder-decoder neural networks that share some similarities in purpose to their predecessors, the recurrent neural network. Transformers are also well-suited to handling sequential data, but unlike RNNs, the input sequence need not be set in stone when processing it (more on this later). Additionally, they draw inspiration from the attention mechanisms of LSTMs and GRUs, but again, transformers use attention alone, foregoing sequential processing. The impressive performance of transformer-based methods on a variety of core NLP tasks have shown that RNNs making use of attention can be surpassed using attention alone.

3.5.1 BERT

BERT (Devlin et al., 2018) is a now-iconic pretraining method for transformer models that takes advantage of bidirectional attention and unsupervised learning. BERT consists of several key steps. First, a large dataset of unlabeled training examples are used to train a transformer for the purpose of learning language representations, and this pretrained model is then fine-tuned on much smaller labeled datasets in a supervised manner.

Of interest in the BERT method is the pretext tasks used for unsupervised training. Pretext tasks are tasks in which the ground truth labels are trivially available, usually because the original input is transformed in some way to introduce artificial ambiguity (i.e. hiding a word in a sentence). In fact, this is exactly what BERT's masked langauge model (MLM) entails: masking a word then using the words before and after it in the sequence to predict the masked word. This bidirectional context was previously not explored or deemed fully possible, and its successful implementation in BERT is partly what led it to outperform previous methods by convincing margins on many supervised NLP tasks.

The other pretext task present in BERT is next sentence prediction (NSP). In NSP, the task of the model is to predict, given a pair of sentences, whether one sentence comes after the other sentence. However, this sentence-level task has been shown to be less influential than MLM, and some works have even removed it altogether (Liu et al., 2019).

3.5.2 XLNet

XLNet (Yang et al., 2019) is a later iteration of transformers that sought to improve upon BERT. Like BERT, XLNet also utilizes bidirectional context, but does so in a different manner. While BERT has the luxury of simultaneously using forward and backward context thanks to its masking approach, these masked tokens are also treated independently of other tokens, with which they may share crucial dependencies. When such dependent tokens take turns being masked, the correlation between them is lost.

To remedy this, (Yang et al., 2019) introduce an autoregressive pre-training method that employs their permutation language model. Autoregressive methods only have access to forward *or* backward context rather than both at the same time. The permutation langauge model (PLM) introduced in XLNet allows for the *simulation* of bidirectional context. The PLM does this by generating permutations of an input sequence, then using forward context as normal (using the previous $i-1$ words as context words for predicting the i^{th} word. This "simulates" bidirectional context because, given the permutations of an input sequence, for every word in the sequence, there exists at least one permutation in which the current word comes both before

and after each of the other words in the sequence.

However, permutation language modeling alone does not handle query and context cases. At some points, a word may need to be predicted, while at other points it may be used as a context word to predict other words. Put another way, there needed to be a way to conditionally access a word's content *and* position (in the context case) or just its position (in the query case). XLNet handles this by using two-stream self-attention, in which two separate attention streams are used. One focuses on content representation (providing word content and position information up to and *including* the i^{th} word), and the other focuses on query representation (providing word content information up to but *excluding* the i^{th} word, and position information up to and *including* the i^{th} word). This allowed XLNet to safely reap the benefits of bidirectional context without the risks of explicit masking.

3.5.3 RoBERTa and DistilRoBERTa

RoBERTa (Liu et al., 2019) is another optimized BERT-like approach that, like XLNet, saw performance improvements over BERT. RoBERTa trained on much more data than BERT, and discarded the other pretext task used by BERT, called next sentence prediction (NSP). In addition to this, RoBERTa changed how the masking pretext task was carried out. In BERT, masks are set before training as a preprocessing step, and these masks persist throughout training, remaining completely *static*. RoBERTa instead takes advantage of *dynamic masking*, in which the masks change between training instances and epochs.

DistilRoBERTa is a knowledge-distilled version of RoBERTa. Knowledge distillation (Hinton et al., 2015) is a method of transferring knowledge representations from a typically large, cumbersome "teacher" model to a smaller, lightweight "student" model while still retaining most of the representational capacity and predictive performance of the larger teacher. DistilBERT (Sanh et al., 2019) is an application of knowledge distillation to the BERT pre-training scheme, which resulted in a smaller, faster, and overall resource-inexpensive version of BERT. DistilRoBERTa follows this principle and distills knowledge from the base RoBERTa model, yielding a model that is twice as fast while maintaining up to 95% of RoBERTa's performance on some benchmarks (Du et al., 2020).

4 Experiments

4.1 Reddit Environment

Reddit is a social media site consisting of smaller forum boards called "subreddits." Subreddits can house posts, which are usually comprised of text, external links, cross-posts (other posts from elsewhere on Reddit), and images. Each post in a subreddit has an associated score, representing the sum of upvotes and downvotes it has received, as well as a set of public comments left by users. A key piece of information about subreddits is that they are self-moderated by community moderators and each have their own set of established rules and guidelines.

To view content in a subreddit, a user must willingly join that subreddit. This is in stark contrast to many other forms of social media, where content from mutual connections, recommendation systems, or a general "front page" is available to users who have not consented to seeing that content. This segregation of content also gives subreddits interesting properties. One of the primary properties of subreddits that I exploit in the data collection process is a combination of community moderation and the tendency to form "echo chambers." An echo chamber is a space in which ideas more closely aligning with the ideas of the rest of the group are celebrated and supported, whereas ideas that challenge the group are often shunned and ignored.

4.2 Text Mining

For this task, in which political ideology classification is the main concern, subreddits are an ideal place to collect data. In a sense, natural operation of a subreddit leads to automatic labeling of content. For example, in a left-leaning subreddit, right-leaning comments and posts will not survive long with positive scores, as the presented ideas disagree with those of the majority. However, posts and comments that are left-leaning and in support of the group ideology will quickly rise in score. This same pattern holds for right-leaning subreddits, and is why I chose Reddit as the environment in which to carry out data mining. Essentially, every comment mined from Reddit is automatically "labeled" as belonging to that subreddit or not, depending on the comment's score.

I gathered comments from multiple subreddits, usually when one was too small to provide enough data on its own. For example, all

of my "conservative" samples were drawn from `r/conservative`, while my "liberal" samples were drawn from both `r/democrats` and `r/liberal`. Since the target audiences of both subreddits have a high degree of overlap, however, it is safe to say that comments from either could be interchangeable and are still safe to use for these experiments.

When collecting comments, I explicitly search for comments that meet certain criteria to minimize the number of mislabeled comments. The first criterion is comment complexity. Many comments on Reddit are incredibly short and made in a joking manner, but such comments are not very useful for classifying political alignment because they often do not include many impactful features that would signal political alignment. If they do, they are often nuanced or may involve figurative elements such as sarcasm, which are difficult for many NLP methods to pick up. Thus, I select comments containing more than 5 words (most containing more than 10-15), to ensure the inclusion of useful, discussion-like content.

The other criterion involves score thresholding. As mentioned previously, the score of a comment is a decent indicator of whether the comment's content "belongs" in that subreddit. Since all comments on Reddit begin with an initial score of 1, testing for only comments with a positive score is infeasible. So, I instead select comments with a score greater than 5, as this encompasses most comments (from top-level to leaf-level comments) that are in support of the general thinking of the subreddit's members.

Once data is collected, I do some initial preprocessing. Many Reddit comments tend to contain links to external webpages, and such links are not useful for classification. I prune out these hyperlinks with a set of regular expressions following Reddit's markdown and non-markdown methods of linking URLs. Additionally, the dataset is modified to expand contractions using the PyContractions library. This is done to avoid ambiguity for the model, as contractions and their constituent words should not have different meanings (and should not be treated differently during training). Finally, some symbols and numerals are removed, as they are typically not useful for classification either.

During preprocessing, depending on the method used, the data has its stopwords removed and tokens extracted using spaCy. Of course, for many of the transformer methods, which use their own special tokenizers, some of these steps are not taken.

Model	Test Acc (%)	F1 Score
Multinomial NB	**76.27**	**0.7738**
AdaBoost	66.67	0.7110
Linear SVM	71.67	0.7276
GRU	53.72	0.4900
BERT	69.33	-
XLNet	65.00	-
DistilRoBERTa	71.83	-

Table 1: Evaluation results of the approaches described previously on the Reddit scraped dataset. *F1 scores are not available for some methods as F1 evaluation scores were added after initial runs of these experiments, and transformer models were more costly to retrain.

4.3 Training Environment and Setup

The experiments are carried out using a dataset containing aproximately 6,000 comments scraped from real Reddit posts. The dataset is balanced in that roughly half of the examples are labeled "conservative" while the other half are labeled "liberal." It was sometimes necessary to combine comments from subreddits at different time periods due to a lack of comments meeting my criteria and query limits placed upon my script by the Reddit API.

Hardware-intensive training was carried out on an 8GB NVIDIA RTX 2070 and a 32GB NVIDIA Tesla V100 (when using UCF's Newton cluster). The environment used to run all data collection, preprocessing, and training steps is described in the supplementary material.

4.4 Results and Discussion

Table 1 shows the results obtained from running each of the methods explained above on the same Reddit-scraped dataset. Of note is the surprising performance of the simpler methods such as multinomial NB and linear SVMs, which surpass the performance of some of the more recent, cutting edge methods. This is a clear indication that neural network-based methods are not always the go-to solution, because in use cases where data is limited (such as in my experiments), simpler models can learn quickly and provide more accurate predictions. Even naive Bayes, which has no concept of context, can perform well on this classification task, likely because many of the terms used by Reddit users on both sides of the political spectrum are

prone to repetition among other users, which is ideal for naive Bayes.

Meanwhile, some of the transformer methods still perform decently, which is surprising, becuase I would have expected them to fare far worse. Many of the BERT-based methods were trained on massive datasets, usually above 160GB in size. Although they still are not being trained from scratch in this experiment (only just fine-tuned on new data), it is impressive to see them each pick up features and adjust to the different class distribution when fine-tuned for only 5 epochs and on a much smaller dataset. And regarding this point, I believe the reason for DistilRoBERTa's performance gap between the other two transformers is its smaller, lighter architecture and ability to distill knowledge from a complex teacher model. This "practice" likely made it better adapted to efficiently learning knowledge representations and gave it the versatility to quickly adapt to new data distributions. And while the current results are certainly much more contextually-accurate than (0xTiger, 2020), and certainly better than random guessing, the addition of more data would likely propel this work to achieving classification accuracies in the 90% range.

5 Conclusion

The results of this work are a constant reminder that the cutting edge is not meant for universal application, and a good testament to the continued usage of older methods for modern tasks in any field of machine learning.

Of course, I do believe that the simple models' performance would be easily eclipsed by the RNNs and transformer-based methods if given more data. Part of the reason I believed the GRUs and transformers underperformed was due to a limited amount of training time, and a difference in complexity between the models and data availability. For future work, I would certainly instead collect data incrementally over a long period of time (perhaps every two weeks for one year), to ensure that duplicate examples are not picked up while also maximizing the number of comments available and ready to be used for training.

References

0xTiger. 2020. Reddit stance classifier. https://github.com/0xTiger/reddit-stance-classifier.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. Cite arxiv:1409.0473Comment: Accepted at ICLR 2015 as oral presentation.

M. D. Conover, B. Goncalves, J. Ratkiewicz, A. Flammini, and F. Menczer. 2011. Predicting the political alignment of twitter users. In *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing*, pages 192–199.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jingfei Du, Myle Ott, Haoran Li, Xing Zhou, and Veselin Stoyanov. 2020. General purpose text embeddings from pre-trained language models for scalable inference.

Yoav Freund and Robert E Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119 – 139.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

Yin Zhang, Rong Jin, and Zhi-Hua Zhou. 2010. Understanding bag-of-words model: A statistical framework. *International Journal of Machine Learning and Cybernetics*, 1:43–52.