Computational Politics: Advancing Sentiment Analysis through Natural Language Processing in Election Studies

Members:
Mundhir Al Bohri (B00858907) - mn409130@dal.ca
Harsahib Preet Singh (B00850322) - hr6446@dal.ca
Brijesh Hota (B00844617) - br747252@dal.ca

Problem statement:

The Canadian federal elections are organized every 4 years where different candidates are elected amongst the various political parties to secure a party seat in Canada's House of Common where they represent their riding as Member of Parliament (MP). Social media (like Twitter, Reddit, etc.) provide voters an open platform to convey different opinions about competing candidates and political parties. Understanding the sentiments of the voters is as important to other voters in the electorate as it is for different contesting political parties. Natural Language Processing (NLP) techniques can help us to effectively decipher sentiments expressed by the electorate during election campaigns using data from various social media platforms.

List of possible approaches:

i. **Probabilistic Models (Naive Bayes Classifier):**

Bayes' theorem is a fundamental model for sentiment analysis, known for its simplicity, efficiency, and effectiveness in producing reliable results. We could say it is the Swiss Army Knife of machine learning algorithms, it is versatile and particularly adept at handling small to medium-sized datasets. For our purposes, we will employ the Naive Bayes model as a foundational layer for sentiment analysis. However, it is important to note that our utilization of this model will be confined to basic sentiment analysis and simple analytical tasks.

Bayes' theorem can be expressed in two forms [1]:

1) First form

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

2) The second form is based on breaking the set B into disjoint sets $B = \bigcup_{i=1}^{n} A_i$

$$P(A_i|B) = \frac{P(B|A_i) \cdot P(A_i)}{P(B)} = \frac{P(B|A_i) \cdot P(A_i)}{\sum_i^P (B|A_i) P(A_i)}$$

The prediction formula for our model, where C represents the probability of being political or non-political and W represents the words, is shown below:

$$conclusion = \arg\max_{C} \; P(C|W) = \arg\max_{C} \frac{P(W|C) \cdot P(C)}{P(W)}$$

$$= \arg\max_{C} \; P(W|C) \cdot P(C)$$

Where $P(W \mid C) = \prod_{i=1}^{n} P(w_i \mid C)$

ii. Transformer Models:

Transformer models, introduced in the paper "Attention is All You Need" [2], have revolutionized the field of natural language processing. They have laid the groundwork for large language models such as BERT and GPT. In contrast to past neural network models like LSTMs (Long Short-Term Memory Networks) and RNNs (Recurrent Neural Networks), which were constrained by sequential data processing and required significant computing power, transformers are much more efficient. The main strength of transformers lies in their ability to process text in parallel, significantly reducing training time while delivering superior performance. This allows capturing dependencies in long texts with which traditional neural networks struggle [2].

A transformer model is composed of two main parts: an encoder and a decoder. Each consists of a stack of layers that process the input sequence in parallel, as opposed to sequentially. The pivotal components of a transformer model include the self-attention mechanism, multi-head attention, and positional encoding. These elements work together to capture the intricacies and relationships within the input data, facilitating a deeper understanding of the text [2].

1. Self-Attention Mechanism

The self-attention mechanism allows the model to weigh the importance of each word in the input sequence based on its relationship with other words. The self-attention score is calculated using the following equations:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where:

- Q , K , and V are the query, key, and value matrices, respectively.
- d_k is the dimension of the key vector.

2. Multi-Head Attention

Multi-head attention consists of multiple self-attention layers running in parallel, each with its own set of parameters. The output of each self-attention layer is concatenated and linearly transformed to produce the final output. The equation for multi-head attention is as follows:

$$MultiHead(Q, K, V) = Concat(head_1, head_2, \dots, head_h)W^O$$

$$\text{Where } head_i = Attention\left(QW_i^Q, KW_i^K, VW_i^V\right)$$

where:
- $head$ is the number of self-attention heads.
- $\left(W^O, W_i^Q, W_i^K, W_i^V\right)$ are parameter matrices.

3. Positional Encoding

Since the transformer model processes the input sequence in parallel, it does not have any sense of order. To address this, positional encoding is added to the input embeddings to give the model information about the position of each word in the sequence. The positional encoding is defined as follows:

$$PE(pos, 2i) = \sin\left(pos/10000^{(2i/d_{model})}\right)$$
$$PE(pos, 2i+1) = \cos\left(pos/10000^{(2i/d_{model})}\right)$$

where:
- pos is the position of the word in the sequence.
- i is the dimension.
- d_{model} is the dimension of the model's embeddings.

Transformer models have demonstrated significant effectiveness across a variety of NLP tasks. Presently, numerous pre-trained models are freely accessible. Utilizing these models is a compelling approach for political analysis, as it necessitates a comprehensive foundational knowledge to identify names, phrases, and the behavior of individuals behind the tweets.

Project plan for the rest of the term:

- **Data pre-processing**

  We plan to kick-off the project with the dataset pre-processing tasks. We have finalized our datasets which will be currently open datasets from Kaggle covering the Tweets of 44th Canadian General Elections. The data columns will be analysed for the different correlation

amongst them. We will also plot different Histograms, Bar plots and Stacker'd charts to understand the dataset better.

- **Model Selection**

  As we finish pre-processing our dataset, we begin working on the NLP model. We would construct a basic model using Naïve Bayes approach for basic sentiment analysis. As we get a basic idea of the model's performance, we would move forward to develop a Transformer based model using Google's BERT-base model. We would be using Hugging Face model repository to import our base model and fine-tune it to our needs based on the initial Naïve Bayes model.

- **Model Evaluation**

  We'd understand our model's performance based on different evaluation metrics like F1 score, accuracy, and precision. As we finish analysing our model, we'd fine-tune our Transformer model's hyperparameters and plot different curves to showcase if the model is overfitting or underfitting.

References:

[1] V. Keselj, "Introduction to Probabilistic NLP Notes," in CSCI 4152/6509 - Natural Language Processing, Faculty of Computer Science, Dalhousie University, Halifax, NS.

[2] A. Vaswani et al., "Attention is all you need," arXiv.org, https://arxiv.org/abs/1706.03762.

Possible References for Future Work:

[1] Mr. V. Chandra Sekhar Reddy, K. Manvith Reddy, CH. Vachan Sai, K. Suraj, A. Abhinash, "A Survey on Automated Sentimental Analysis of Twitter Data using Supervised Algorithm", International Journal of Advanced Research in Science, Communication and Technology, pp.196, 2022.

[2] C. Ziems, W. Held, O. Shaikh, J. Chen, Z. Zhang, and D. Yang, "Can Large Language Models Transform Computational Social Science?" arXiv.org, https://arxiv.org/abs/2305.03514.

[3] S. Palakodety, A. R. KhudaBukhsh, and J. G. Carbonell, "Mining Insights from Large-scale Corpora Using Fine-tuned Language Models," 2019 https://www.cs.cmu.edu/~akhudabu/ECAI-2019-BERTElection.pdf

Declarations: we utilized ChatGPT, to assist in clarifying and refining the content presented in this document.