Twitter Troll Detection: *A Language Model Approach*

Atamson Atam
Norman Paterson School of International Affairs
atamsonatam@cmail.carleton.ca

Mohamed Elbeltagi
Physics Department
mohamedelbeltagi@cmail.carleton.ca

Deepro Sengupta
Sprott School of Business
deeprosengupta@cmail.carleton.ca

Supervisor: **Dr. Ahmed El-Roby**
School of Computer Science

Carleton University

Introduction

Foreign digital interference via online social media "*trolling*" may serve as a strategic tool to spread misinformation. This practice could influence public reactions to social events, policies, and even democratic processes such as election outcomes.

Research Question

- Can language (specifically transformer) models be trained to detect and consequently used to flag such troll activity?
- How do different approaches using language models compare against each other?

Dataset & Limitations

- Our full dataset comprises of **100K** confirmed **troll tweets** from the 2016 US election[1] (tweets from Russian agency accounts), and **100K** general political tweets from the 2020 US election[2], acting as **non-troll** examples.
- We ran into some computational resource limitations, more on this in the **Results** section.

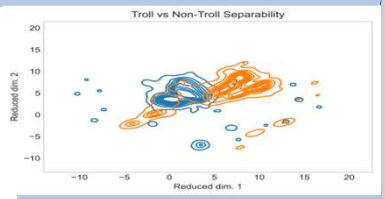


Figure1: Feature dimensionality reduction showing separability

Methodology

1st Approach: *Frozen Model* → (Improving Classification) → 2nd Approach: *Fine-tuned* → (Using Less Data) → 3rd Approach: *SetFit*

- Distilled pretrained transformer (distilBERT)
- Weights of transformer are frozen
- Transformer produces text features
- Classifier head trained on features and labels

- Fine-tune a distilled pretrained transformer
- Gradients calculated for transformer as well
- Weights of transformer & classifier updated
- Idea is to produce better features for classifier

- Brand new technique called SetFit[3]
- Efficient few-shot learning using 16 examples
- Uses contrastive learning to fine-tune model
- Classifier head trained on the features

Figure2: 3 different approaches involving (distilled versions) of large language models. 80%-20% train-test split for first 2

Classification Pipeline

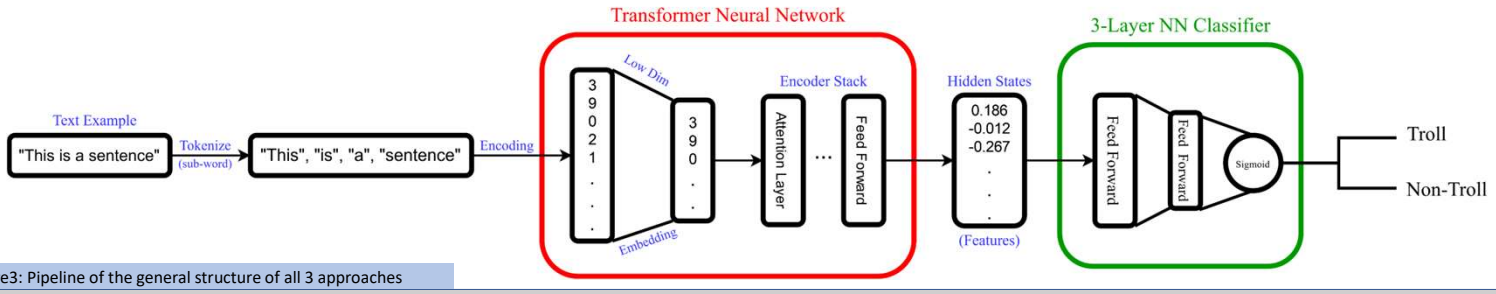
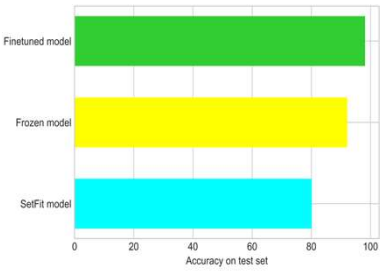


Figure3: Pipeline of the general structure of all 3 approaches

Results & Conclusions

- The most performant model was the **fine-tuned transformer**. However, there was a **surprise**: due to GPU memory constraints we were not able to fine-tune the same pretrained transformer as in the frozen model, instead we used an **even smaller model** (a version of TinyBERT), despite that it **still outperformed** the larger frozen model. This is the **recommended approach** given computational resources availability.
- The **SetFit** model achieved ~80% accuracy, despite the minuscule number of examples used for training compared to the other 2 approaches! This is incredible, and **recommended if data is scarce**.
- The **frozen model** achieved very good accuracy. We trained this model "manually", defining the training loop in PyTorch. **Training loss had not yet plateaued**, but stopped due to time constraints. If data is available but computational resources limited, this approach is recommended.



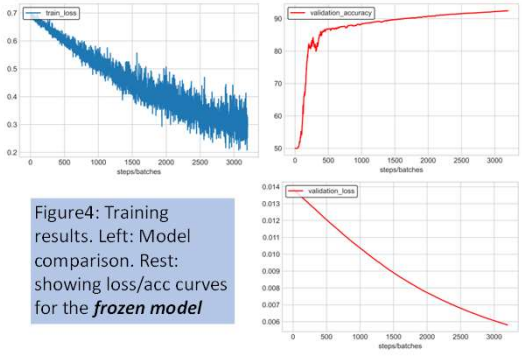


Figure4: Training results. Left: Model comparison. Rest: showing loss/acc curves for the ***frozen model***

References

[1] VIKAS. 2018. Russian Troll Tweets. Retrieved from https://www.kaggle.com/datasets/vikasg/russian-troll-tweets
[2] Manch Hui. 2020. US Election 2020 Tweets. Retrieved from https://www.kaggle.com/datasets/manchunhui/us-election-2020-tweets
[3] Unso Eun Seo Jo et al. 2022. SetFit: Efficient Few-Shot Learning Without Prompts. Retrieved from https://huggingface.co/blog/setfit

Carleton University

Acknowledgments

We would like to thank the instructors of the DATA5000 course, especially Dr. Ahmed El-Roby from the School of Computer Science for his helpful feedback and guidance. We also extend our thanks to the organizers of Data Day for all their effort.