FeelsAmazingMan: Combining contextual with static representations to infer Twitch chat sentiment

Ryan Jun Wang[1] **Feifei Li**[1] **Victor Trinh**[1]

[1]Department of Computer Science
[1]University of Toronto
{ryanjun.wang, ff.li, v.trinh}@mail.utoronto.ca

Abstract

Sentiment analysis provides a way of understanding the semantics of text, especially important in the context of online platforms that rely on communication like Twitch.tv. Twitch serves as a rich pool of text data that is dynamically growing with the inclusion of new word tokens, or emotes. Understanding this unique domain can provide insight into how online communities behave and what rhetoric they employ. Previous state-of-the-art (SoTA) models use traditional machine learning like Support Vector Machines and Random Forests, however SoTA deep learning models have not been applied in recent literature. We perform various combinations of BERT and static embeddings. The fine-tuned BERT architecture attached to a classifier which aggregates the BERT encoding with static Word2Vec embeddings via a feed-forward network improves on previous SoTA by 4.5% in accuracy and 6.1% in Macro F-1 score. Combining the fine-tuned BERT with a classifier made of a layer-normalised GRU cell also improves by 2.8% in accuracy and 5.4% in Macro F-1 score. These results show the power of combining transformers with static embeddings in a novel domain with a low amount of data.

1 Introduction

Online spaces provide platforms for any individuals to share their messages or opinions. These opinions provide a vast source of data, useful in decision making at the individual, or corporate level. The act of understanding and analyzing these data is known as *sentiment analysis* [1]. Sentiment analysis on general platforms like Twitter [2] have been extensively studied to understand a tweet's mood [3]. However, more niche areas like video game spaces have not been explored as thoroughly. Twitch is an increasingly popular live-streaming platform and provides a chatroom for users to communicate in during streams [4] . Twitch is unique in that most users communicate through *Twitch emotes* [5]. These are keywords that represent custom images, consider keyword-emote pair "(PepeHands,)", which represents sadness. A unique challenge is posed by Twitch emotes, in that there is no constant lexicon or huge labeled data set to encompass their sentiment. Understanding the sentiment of Twitch chat can provide insight into how we analyze small amounts of unique text data, as well as how different online communities react to different entertainment. These analyses can be used by both Twitch or it's streamers to produce more engaging content and improve channel growth.

With the advent of transformer architectures [8] like BERT, the natural language processing world has advanced greatly. We seek to introduce BERT and combine it with classic natural language processing (NLP) techniques like [12]'s *Word2Vec* to the unique domain of Twitch and improve on traditional machine learning methods. We hypothesize that combining BERT with traditional embeddings will aid in generalisation with our small amount of data.

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

2 Related Works

2.1 Sentiment Analysis

Usage of deep learning for sentiment analysis has been explored extensively, for example, [7] show that LSTMs (Long Short-Term Memory) have very high prediction accuracy on movie review sentiment. More recently, with the development of transformer architectures [8], BERT fine-tuning (and other BERT-based modeling) has been used for training classifiers, achieving very high accuracy on English datasets [9].

2.2 Twitch Chats and Emotes

Although social media platforms like Twitter have been explored extensively for sentiment analysis, the Twitch landscape still remains sparse. Previous sentiment analysis of Twitch chat primarily relies on traditional machine learning techniques (Support Vector Machines, Random Forests, Logistic Regression, etc. [10], [6]) and little deep learning exploration (convolutional text classification networks [5]). All three sentiment analyses use [5]'s "emote-controlled" data set with [6] achieving the best accuracy at 71.16% using a Bi-gram Random Forest approach. This is the highest performing model in this niche, to the best of our knowledge.

3 Methods

3.1 Data

We used a pre-labeled Twitch sentiment data set [5] to train and evaluate our proposed methods. The labeled data set consists of 1922 Twitch comments from the five most commented English channels in May 2018. $404(21.02\%)$ data points were negative, $748(38.92\%)$ neutral, and $770(40.06\%)$ positive. We are specifically interested in the message and sentiment fields. We performed an 80%/20% (1538/384 data points) train-val split in accordance with the same methods used in [5] and [6] for performance evaluation. We also performed a 60%/20%/20% (1153/384/385 data points) train-val-test split in accordance to [10]. We report the best performing accuracy from both setups. Models will be evaluated with accuracy and macro F1-scores.

3.2 BERT Fine-tuning and Model Architectures

We explored 3 deep learning sentiment classification models to fine-tune [8]'s pre-trained base BERT (retaining letter case using [13]'s HuggingFace Python library) and using a feed-forward network and default Adam [11] optimizer with weight decay of 0.01 (0.1 on the GRU model). We trained with batch size of 64 on Tesla P100's. *For supplemental model architectures and figures, see **App. 6.2**; see **App. 6.3** for code repository.*

3.2.1 BERT-W2V: Combining Static Embeddings and BERT

We learned static 768-dimension embeddings of the labeled data using *Word2Vec* described in [12]. We then normalised the BERT pooler output and the Word2Vec embedding where token embeddings were summed element-wise. The two embeddings are then aggregated together through element-wise summing. Prior to being used as input to a feed-forward network, dropout is performed for the combined input with probability 0.5. The input is then passed into a feed-forward network comprised of a 128-unit linear layer, a swish activation function, another 128-unit linear layer, and outputs logits (MLP model) to a standard 3-class cross entropy loss (**Algo. 1**).

3.2.2 BERT-W2V-GRU: Selective Semantic Extraction from Static Embeddings

We first send the BERT pooler output to a GRU cell via the hidden state entry and then the Word2Vec embedding via the input entry. Layer normalisation is performed after each linear operation in the GRU cell. Dropout at a probability of 50% is performed for the GRU cell output, and it is then passed into a feed-forward network that is the same as the second one in BERT-W2V-MLP (**Algo. 2**). Prior to fine-tuning BERT, we froze the first four transformer blocks and reinitialised the weights of the last six blocks to obtain more stable predictions.

Algorithm 1 BERT-WORD2VEC(message x_i, label t_i)

$z_i \leftarrow$ sum(Word2Vec(x_i))
$h_i \leftarrow$ Fine-tuned BERT-BASE-CASED(x_i)
$\tilde{h}_i \leftarrow$ normalise(h_i) + normalise(z_i)
Dropout($\tilde{h}_i$)
$y_i \leftarrow$ FEED-FORWARD-NETWORK($\tilde{h}_i$)
$\mathcal{L}_i \leftarrow \mathcal{L}_{CE}(y_i, t_i)$
return $(y_i, \mathcal{L}_i)$

Algorithm 2 BERT-W2V-GRU(message x_i, label t_i)

$h_i \leftarrow$ Fine-tuned BERT-BASE-CASED(x_i)
$z_i \leftarrow$ sum(Word2Vec(x_i))
$\tilde{h}_i \leftarrow$ LAYER-NORM-GRU-CELL(h_i, z_i)
Dropout($\tilde{h}_i$)
$y_i \leftarrow$ FEED-FORWARD-NETWORK($\tilde{h}_i$)
$\mathcal{L}_i \leftarrow \mathcal{L}_{CE}(y_i, t_i)$
return $(y_i, \mathcal{L}_i)$

3.2.3 Word2Vec-MLP: Classifier Using Static Embeddings Only

We compared transformer performance to a standard fully-connected neural network that uses the same feed-forward networks as in BERT-W2V-MLP, except that only the Word2Vec embedding is passed into this model without aggregating the BERT pooler output (**Algo. 3**).

Algorithm 3 W2V-MLP(message x_i, label t_i)

$z_i \leftarrow$ sum(Word2Vec(x_i))
$\tilde{z}_i \leftarrow$ FEED-FORWARD-LAYER 1 (z_i)
Dropout($\tilde{z}_i$)
$y_i \leftarrow$ FEED-FORWARD-NETWORK 2($\tilde{z}_i$)
$\mathcal{L}_i \leftarrow \mathcal{L}_{CE}(y_i, t_i)$
return $(y_i, \mathcal{L}_i)$

4 Results

Model	Accuracy	Macro F-1
[5] Sentence CNN	63.8%	62.6%
[6] Bi-gram Random Forest	71.2%	N/A
Fully-connected MLP + Labeled-Word2Vec (Batch Size 64)	60.3%	45.2%
BERT (Batch Size 64)	73.9%	68.1%
BERT + Labeled-Word2Vec (Batch Size 64)	**75.7**%	**68.7**%
BERT + Labeled-Word2Vec + GRU (Batch Size 64, Weight decay 0.1)	**74.0**%	**68.0**%

Table 1: Best recorded model accuracy and F1 score (80/20 setup) of the original Twitch sentiment paper's model (CNN), the previous best performing model (Random Forest), and our approaches.

We show the performance of the various architectural approaches we took compared to previous best performing models in recent literature for this specific application (**Table 1**). We found that when we followed the same 80/20 splits, we improved on the best performing Bi-gram Random Forest model by $\approx 2.7\%$ using base BERT. When summing the normalised BERT pooler output with normalised Word2Vec embeddings of the corresponding data point, we achieved $\approx 4.5\%$ improvement on the previous best validation accuracy. These fine-tuning were not very stable though, potentially due to the lack of layer freezing. When these are applied however, we see relatively good and stable performance from the GRU model. All transformer models outperform the standard MLP.

With [10]'s 60/20/20 setup, just the BERT base model achieves 71.4% test accuracy at the point with the best validation accuracy. However, we only achieve a macro F1 score of $\approx 64\%$ while [10] reports a 67.3% macro F1 score. This difference may be due to extra text filtering [10] performs that may change data set balance, which can affect F1-score.

5 Discussion

5.1 BERT and Static Embeddings

While BERT on its own already out-performed SoTA models, the inclusion of static Word2Vec embeddings further improved the model performance. We suspect that the Word2Vec embeddings are serving as an informed regularization. That is, the Word2Vec embeddings are learned from the data and provide an extra source of token information that can be used for classification that does not change, hence pushing the BERT fine-tuning to focus on other potential information for classification. Furthermore, contrasting with traditional modeling [6], [10], our method does not require very intensive feature engineering, which can be a benefit in production environments.

5.2 Comparing and Improving Deep Learning Model Performance

Word2Vec-MLP's (No BERT) poor performance reflects the effect contextual representations can play in improving classification tasks. Word2Vec embeddings only focus on semantic and syntactic similarity among words in Twitch messages. This suggests that a sole MLP may fail to learn the syntactic patterns of token sequences contributing to sentiment, which may be captured better by BERT. Previously we discussed the combination of BERT and Word2Vec. To further incorporate knowledge carried by these two embeddings, we borrowed the mechanism of a GRU cell's update gate, which could selectively integrate information from the Word2Vec embeddings into BERT's pooler representation, and its reset gate which can filter out information irrelevant from the Word2Vec embeddings. However, the classifier using a GRU cell does not perform as well as BERT+Word2Vec which directly aggregates the two normalised embeddings. This could be attributed to the partial information of Word2Vec embeddings discarded at the update gate, which was fully-preserved during summation in the BERT+Word2Vec model. Although we have achieved results that surpass the previous SoTA methods, the validation accuracy of our models drop to 60% when training for more epochs (> 25 epochs) with validation loss diverging from training loss. This implies potential over-fitting and a decrease in the generalisation performance of our models. Like previous literature [15], freezing BERT weights (first four Transformer encoders) and re-initialising the BERT weights (final six blocks) stabilises validation accuracy from 70% to 74% when training longer (> 50 epochs). However, these improvements only hold if we perform both freezing and re-initialisation.

5.3 Limitations and Future Directions

One of the major limitations of our study is exceedingly small and 4 year old data set 4 years old. We found the data set to be very noisy with the same text having different sentiment, potentially due to labeling variation [5]. Furthermore, we lacked the compute to try more 60/20/20 splits and intensive models that have already been pre-trained on more data like RoBERTa [14]. We also restricted ourselves to an 80/20 data split for the majority of training due to keeping consistent comparisons with previous SoTA ([5], [6]). These serve as areas to explore in the future with access to more compute and less noisy, labeled data. Furthermore, [6] found emotes to be very influential on classification, so a contrastive learning approach could be interesting to look at text similarity with emotes and using resulting representations for downstream tasks.

6 Conclusion

We introduced and explored SoTA deep learning architectures in the unique domain of Twitch chat messages. These data show the promising results when combining transformers with classical static embeddings in the context of unique, low volume data, improving on previous traditional machine learning models. In addition to the usefulness of this modeling approach in this very common scenario (lacking large data sets), these data show that we have relatively good performance for use in commercial applications and further downstream analysis.

References

[1] Ligthart, A., Catal, C., & Tekinerdogan, B. (2021). Systematic reviews in sentiment analysis: a tertiary study. Artificial intelligence review, 54(7), 4997-5053.

[2] Go, A., Huang, L., & Bhayani, R. (2009). Twitter sentiment analysis. Entropy, 17, 252.

[3] Giachanou, A., & Crestani, F. (2016). Like it or not: A survey of twitter sentiment analysis methods. ACM Computing Surveys (CSUR), 49(2), 1-41.

[4] Johnson, M. R., & Woodcock, J. (2019). 'It's like the gold rush': the lives and careers of professional video game streamers on Twitch. tv. Information, Communication & Society, 22(3), 336-351.

[5] Kobs, K., Zehe, A., Bernstetter, A., Chibane, J., Pfister, J., Tritscher, J., & Hotho, A. (2020). Emote-controlled: obtaining implicit viewer feedback through emote-based sentiment analysis on comments of popular twitch. tv channels. ACM transactions on social computing, 3(2), 1-34.

[6] Dolin, P., d'Hauthuille, L., & Vattani, A. (2021). FeelsGoodMan: Inferring Semantics of Twitch Neologisms. arXiv preprint arXiv:2108.08411.

[7] Li, D., & Qian, J. (2016, October). Text sentiment analysis based on long short-term memory. In 2016 First IEEE International Conference on Computer Communication and the Internet (ICCCI) (pp. 471-475). IEEE.

[8] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[9] Batra, H., Punn, N. S., Sonbhadra, S. K., & Agarwal, S. (2021, September). BERT-Based Sentiment Analysis: A Software Engineering Perspective. In International Conference on Database and Expert Systems Applications (pp. 138-148). Springer, Cham.

[10] Chouhan, A., Halgekar, A., Rao, A., Khankhoje, D., & Narvekar, M. (2021, September). Sentiment Analysis of Twitch. tv Livestream Messages using Machine Learning Methods. In 2021 Fourth International Conference on Electrical, Computer and Communication Technologies (ICECCT) (pp. 1-5). IEEE.

[11] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[12] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

[13] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... & Rush, A. M. (2019). Huggingface's transformers: State-of-the-art natural language processing. arXiv preprint arXiv:1910.03771.

[14] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

[15] Zhang, T., Wu, F., Katiyar, A., Weinberger, K. Q., & Artzi, Y. (2020). Revisiting Few-sample BERT Fine-tuning. https://doi.org/10.48550/arXiv.2006.05987

Appendix

6.1 Data Augmentation

The language used by the Twitch community is rather different from common English, characterised by excessive use of abbreviations, duplicated phrases, emotes, and new slang terms constantly being developed. Hence, conventional data augmentation methods for natural language processing such as synonym/acronym replacement or back translation are not feasible for this task. Here we adopted an emote-based data augmentation method: emotes in labeled messages were replaced by different emotes, and the labeled sentiment values of the augmented messages were reweighed by the labeled sentiment scores of the replacing emotes. This implicitly shifts more weight contributing to the sentiment of a message towards emotes.

Algorithm 4 EMOTE-AUGMENTATION(message (x_i, t_i), emotes $E = \{(e_1, s_1), \ldots, (e_k, s_k)\}$)

1: $(e_j, s_j) \leftarrow$ a randomly sampled labeled emote from $E_{e \notin x_i}$
2: $\hat{x}_i \leftarrow$ replace the emote in x_i with e_j
3: $\hat{t}_i \leftarrow \text{round}(\tanh(t_i + s_j))$
4: **return** augmented message $(\hat{x}_i, \hat{t}_i)$

Incorporating augmentation with BERT and W2V on labeled data results in 63.7% accuracy and a macro F-1 score of 53.0%. This poor performance may be due to the already noisy data set potentially becoming more noisy with augmentation and the over-fitting of our model to said noise.

6.2 Model Architectures

6.2.1 BERT-W2V

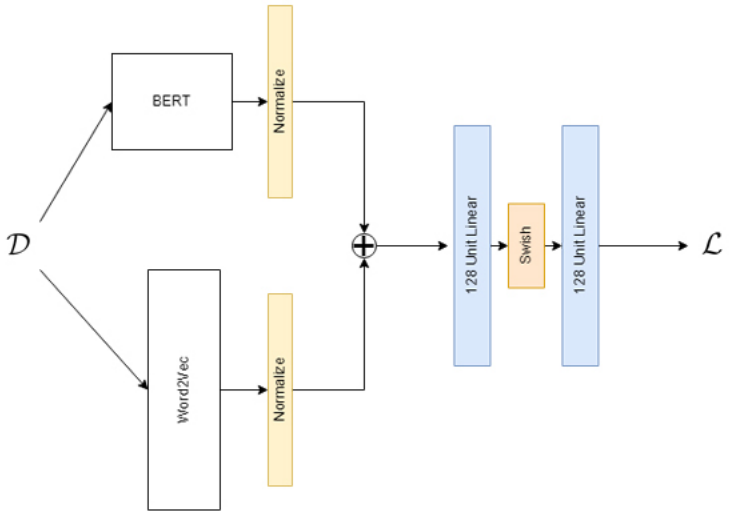

Figure 1: (Supplementary) BERT-W2V aggregated fine-tuning classification model architecture.

6.2.2 BERT-(W2V-MLP)

The Word2Vec embedding is first fed into the first feed-forward network that consists of a linear layer of 768 units and a GELU activation function, and then aggregated with the BERT pooler output vector through element-wise summing. Then dropout at a probability of 50% is performed for the merged embedding, which then is passed to the second feed-forward layer that consists of a linear layer of 256 units, a swish ReLU activation function, a linear layer of 128 units, a swish ReLU activation function, and finally a linear layer of 128 units that outputs logits to the 3-class cross entropy loss.

Layer normalisation is performed before passing the embedding to an activation function in both feed-forward network.

Algorithm 5 BERT-W2V-MLP(message x_i, label t_i)

$h_i \leftarrow$ Fine-tuned BERT-BASE-CASED(x_i)
$z_i \leftarrow$ sum(Word2Vec(x_i))
$\tilde{z}_i \leftarrow$ FEED-FORWARD-LAYER 1 (z_i)
$\tilde{h}_i \leftarrow h_i + \tilde{z}_i$
Dropout($\tilde{h}_i$)
$y_i \leftarrow$ FEED-FORWARD-NETWORK 2($\tilde{h}_i$)
$\mathcal{L}_i \leftarrow \mathcal{L}_{CE}(y_i, t_i)$
return $(y_i, \mathcal{L}_i)$

At batch size 64, achieves 74.0% accuracy and 68.0% macro F1. But is very similar in structure to our better performing BERT-W2V without passing the W2V embeddings into a FFN before classifier network (BERT-W2V **Sup. Fig. 1**).

6.2.3 Validation Plots

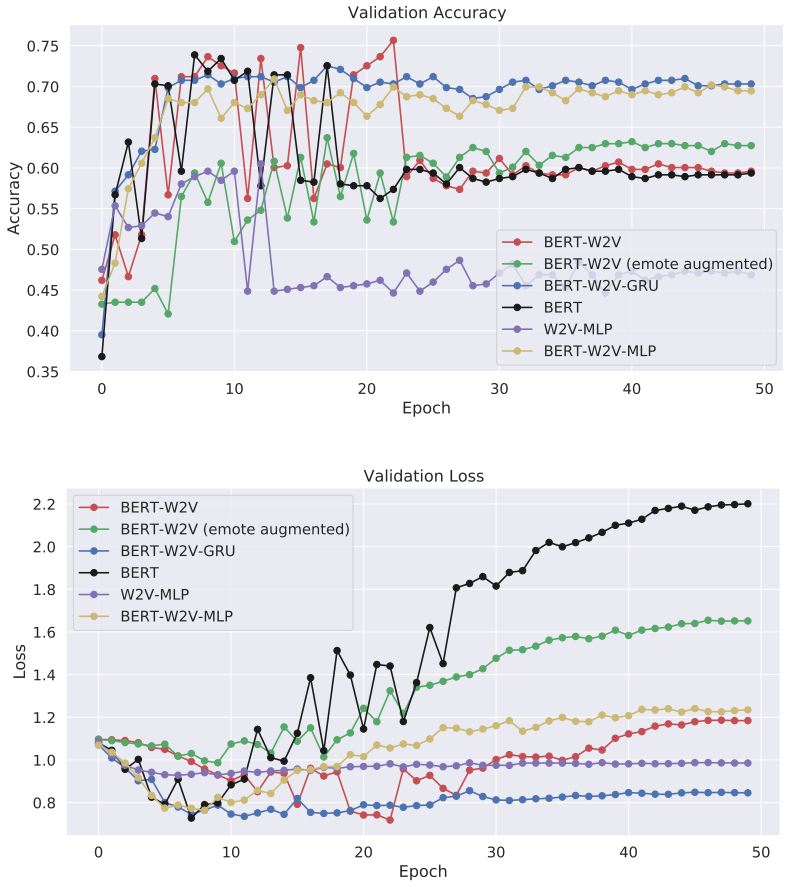

Figure 2: (Supplementary) Model validation performance cut off at 50 epochs. BERT-W2V-GRU and BERT-W2V-MLP continue stable training due to frozen and re-initialised BERT layers.

6.3 Model Repositories

You can find our model repository here and a demo web application here (`https://github.com/rjunw/twitch-sentiment-model` and `https://github.com/Victor-Trinh/twitch-sentiment-webapp`).

6.4 Contributions

6.4.1 Ryan Jun Wang

Developed setup for combining BERT and static representations on previously unexplored domain (deep learning approaches to sentiment analysis of Twitch chats). Specifically explored different potential model approaches and methodology, settling on base BERT and BERT aggregated with Word2Vec (and the feed-forward/MLP classifier). Developed prediction script for applying our models to downstream applications.

6.4.2 Feifei Li

Developed setup for integrating BERT pooler representations and Word2Vec embeddings using a GRU cell in a sentiment classifier for fine-tuning BERT. Performed emote-based augmentation on the labeled training data. Explored the generalisation performance of fine-tuned BERT for downstream classification tasks under various settings of freezing Transformer encoders and re-initialising pre-trained weights.

6.4.3 Victor Trinh

Explored related works and contributed to conceptual discussions regarding project direction. Contributed to inference code for combined Word2Vec and BERT model. Developed fullstack demo to monitor sentiment in realtime on any given twitch channel. Developed API to provide developers with blackbox predictions over the web while deployed.