Finding Human Emotions from Text Comments

CS4442B Artificial Intelligence 2 Final Project (Undergraduate)

Jeongwon Song
Department of Electrical and Computer Engineering
Western University
London, Canada
jsong336@uwo.ca

Jason Koo
Department of Electrical and Computer Engineering
Western University
London, Canada
jkoo26@uwo.ca

***Abstract*—Texted-based emotion detection has many use-cases as it allows the digitization of human emotion from massive text data generated from social media. This experiment aims to apply transfer learning by leveraging BERT [1], a transformer-based [2] neural architecture which extracts contextual embedding from texts, to detect 27 fine-grained human emotions. The models were trained with one of the largest fined-grained emotion text datasets, GoEmotions [3]. The authors apply two methods to fine-tune pretrained BERT-small [4] [5], a smaller variant of BERT: adding DNN layers that take the pooled output and adding bidirectional GRU layers [6] that take sequential embedding. Both models performed very similarly, achieving the same scores up to the third decimal digit. The final models obtained an average ROC score of approximately 0.80 and a macro-F1 score greater than 0.39, thus showing moderate success in detecting appropriate emotions from texts.**

***Index Terms*—NLP, Emotion, Emotion-detection, Transfer learning, BERT, GRU, PyTorch, HuggingFace**

I. Introduction

With the emergence of deep learning, there have been significant achievements in various sentimental analysis tasks from both research and industrial applications. However, practitioners often simplify the sentiment space, i.e., binary sentiments, preventing sophisticated analyses from text-based data. Fine-grained emotion detection allows for more profound analyses of human sentiments by extracting rigorously categorized emotions from the texts, bringing new opportunities for researchers and engineers in various applications, including chat-based systems, comment analyses, or as a feature to other problems.

The BERT architecture has gained popularity with its ability to extract contextual embedding using a self-attention mechanism [1] [2]. Unlike traditional word embedding techniques, the feature vectors from the BERT are relative to the sentence rather than representing the global word meaning. Many variants of pre-trained BERT models are publicly available, enabling practitioners to build advanced NLP models by simply adding custom layers on top of the pre-trained BERT encoders. In the project scope, we fine-tune BERT-small [4] [5], a smaller variant of BERT, with custom classifier layers which predict human emotions based on the BERT's feature embedding.

In this experiment, we formulate the emotion detection problem as one versus one, multi-label classification across the 27 different emotions as found in the GoEmotions dataset [3], one of the largest labelled emotion datasets extracted from Reddit comments. This experiment focuses on two approaches to fine-tuning BERT models: 1) adding DNN classification layers that take the pooled output of BERT and 2) adding bidirectional GRU [6] layers that take sequential embeddings of input sequences, then forwards to dense classifier layers. Both models achieved similar results in which the GRU-based network scored a slightly higher micro-F1 compared to the DNN based network. The best model scored an average AUC-ROC over 0.80 and macro-F1 over 0.39, indicating success in detecting emotions from texts.

II. Related Work

Dorottya Demszky et al. [3], the authors of GoEmotions, compared the performance of fine-tuned BERT and the traditional bidirectional LSTM [7] with word embedding using the GoEmotions dataset. Dorottya Demszky et al. showed that BERT performed significantly better than LSTM with word embedding by scoring macro-F1 of 0.46 across 27 emotions.
.

Similarly, Alvarez-Gonzalez et al. [8] compared models' performances based on multiple feature extraction methods with the GoEmotions [3] and Vent [9] datasets. Their results showed that the BERT model achieved the best scores among other embedding methods like FastText [10], TF-IDF and Bag of Words. Alvarez-Gonzalez et al. applied two architectures, first a stacked bidirectional LSTM on top of different embedding layers and second, DNN layers that take the sequential inputs in parallel, pooling the output after the DNN forwards. The BERT models in [8] scored slightly higher F1 than in the original GoEmotion paper [3].

III. Data & Exploratory Data Analysis

A. Emotion Taxonomy

Fine-grained emotion detection is considered a challenging problem due to the ambiguity and complexity of human emotions. The GoEmotions dataset follows a relatively new emotion taxonomy developed by Cowen et al. [11] Cowen et al. proposed 27 different emotions found with advanced statistical methods which capture complex semantic space by analyzing human responses from over 2000 video clips. The

27 emotion groups could be further grouped into three more generic emotions, positive, negative, and ambiguous [3].

B. Dataset Subsets

The Go-Emotions dataset contains 58k selected Reddit comments labeled with 27 different emotions in addition to neutral, where three to five separate raters annotated individual comments providing a total of 221K rows [3]. In this experiment, we removed any ambiguous comments that did not belong to any emotion classes. Any comments with multiple raters were aggregated into a single row. Next, the dataset was reformatted so that comments that did not belong to any emotion class were considered as neutral. After random ordering, the dataset was divided into training, validation and tuning, and test sets.

C. Imbalance Class

It is vital to understand the class distribution in the datasets before choosing the metrics to measure the models' performance. Figure 1 shows the proportion of positive labels for each emotion. There are less than 15% of positive labels for most classes, which must be considered when analyzing the results or setting the classification threshold. Due to a high imbalance in each class, we mainly use F1 score to interpret the models' performance, described in detail the later section.

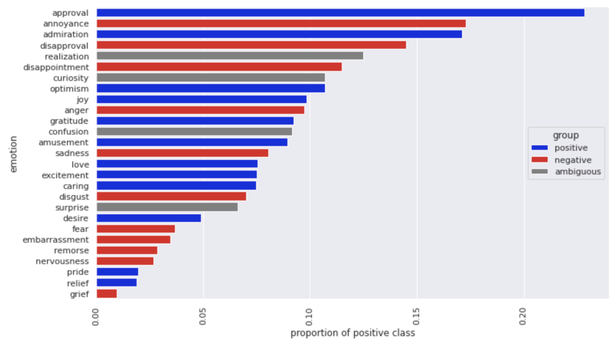

Fig. 1. Proportion of positive labels for each emotion

D. Relationship between Emotion

Another interesting factor to consider in the experiment is the correlation between each emotion. Even with advanced emotion taxonomy, a sentence can still covey multiple emotions which could be interpreted differently by both the reader and writer. Through the labelled GoEmotions dataset, we can represent the relationships between different emotions in a more quantifiable way. The high co-occurrence between two emotions in the comments indicates two emotions are more likely to be similar. Hence, we could build a co-occurrence matrix where each column vector of the matrix is normalized except for the diagonal elements resulting in 26-dimensional embedding vectors of each emotion. Figure 4 shows the linear projection of the embedding in the 2D space using Principal Components Analysis. [3] describes similar analyses.

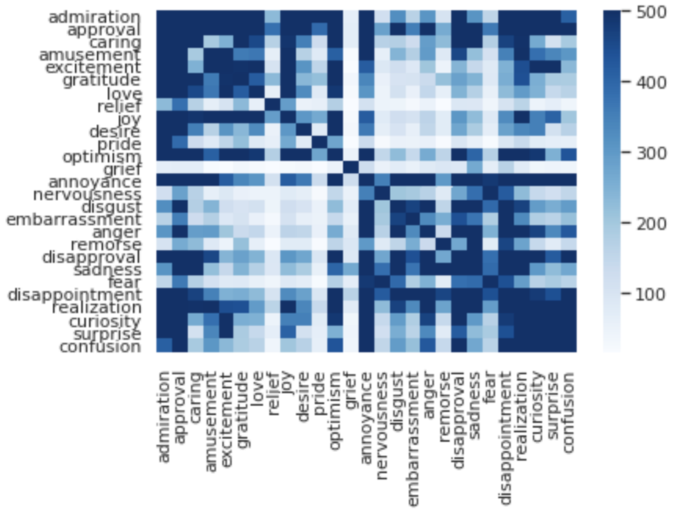

Fig. 2. Co-Occurrence matrix of emotion groups.

```
Cosine Similarity with "love"
gratitude        0.98
excitement       0.95
admiration       0.92
pride            0.90
amusement        0.90
relief           0.90
approval         0.86
joy              0.86
caring           0.83
optimism         0.80
desire           0.76
surprise         0.66
realization      0.64
curiosity        0.50
annoyance        0.48
grief            0.46
sadness          0.46
disapproval      0.44
embarrassment    0.44
fear             0.43
remorse          0.43
nervousness      0.43
confusion        0.43
disappointment   0.42
disgust          0.37
anger            0.32
```

Fig. 3. Cosine similarity between emotions and "love"

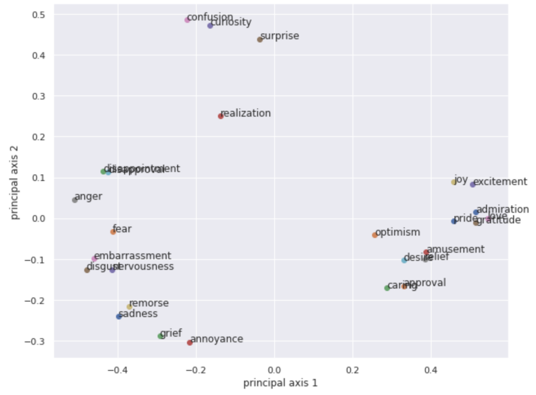

Fig. 4. Linear Projection of Co-Occurrence matrix of emotion groups.

IV. Data Processing and Feature Engineering

Data preparation is often the most essential and time-consuming task in many machine learning systems. Especially in text-based features, which require multiple data preparation steps to make text understandable by mathematical models. In this experiment, we utilized the tokenizer published along with the pre-trained BERT-small model [4] [5] which provides robust tokenization tasks through common text processing steps. However, to maximize the benefit of transfer learning, it is still essential to ensure that the text data fed to the model does not deviate much from the original text format in which the model was trained.

A. Text Cleaning

Since the GoEmotions dataset was extracted from Reddit comments, it contains non-alphanumeric characters, URLs, and spelling mistakes. Below summarizes the text processing steps taken to improve the quality of text before being fed the models.

1) Removing non-alphanumeric or punctuation characters and mapping differently encoded characters. i.e. UTF-8 xE2x80x99(’) and x27(’).
2) Fixing popular spelling mistakes. The list of popular spelling mistakes is found in Birkbeck spelling error corpus [12].
3) Removing URLs, Hashtags, and Emojis with Tweet Preprocessor package [13].

Furthermore, the pre-built tokenizer performs common text cleaning tasks, including converting to lowercase, splitting punctuation, separating punctuations/contraction, and removing white spaces.

B. Padding

BERT [1] requires fixed-length padded sequences and attention masks that indicate whether each token is padding or not. Attention masks allow the model to exempt applying attention to the padding tokens [2]. The input sequences are padded with the maximum sequence length. The comments from the GoEmotions dataset have relatively short corpora, where the majority of the texts sequences lengths are below 30.

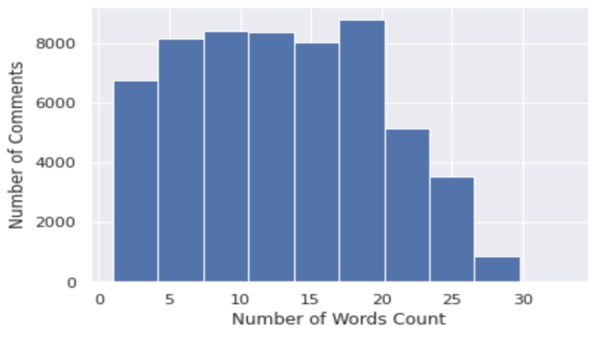

Fig. 5. Distribution of Words Counts

C. Tokenization

The BERT tokenizer applies the WordPiece algorithm introduced in [1] [14] to build the vocabularies by iteratively training the language models with the subwords and combining them with the highest likelihood. This approach allows subword-based tokenization, which mitigates common problems with word-based tokenization, such as large unknown tokens, contractions, and multiple forms of the same words.

V. Method

The project aims to build one versus one multi-label classification models that detect 27 fine-grained emotions using two different neural architectures by fine-tuning BERT-small [4] [5]. Each model outputs 27-dimensional vectors where the sigmoid of each element represents the estimated probability of the associated emotion and negative in all categories representing neutral. The results are compared to the performance of similar architectures trained with larger BERT models from [3] and [8].

A. Pre-trained BERT and BERT-small

BERT [1] has gained popularity by providing robust contextual embedding through the self-attention mechanism [2]. The original BERT model contains over 12 transformer encoder layers with 768 hidden dimensions and 110M parameters. In this experiment, we use BERT-small with four encoding layers and 512 hidden dimensions [4] [5].

B. Fine Tuning BERT-small With Pooling

The original implementation of the BERT model outputs a sequential contextual embedding of the input sequences and the pooled output which is the embedding of the [CLS] token [1]. The [CLS] is a special token added to the beginning of every sequence. The embedding of [CLS] captures the aggregated representation of the whole sequence and is often sufficient in many classification tasks. Thus, we added dense layers with ReLU activation functions on top of the pre-trained BERT model to take the pooled output as features to the classifier units.

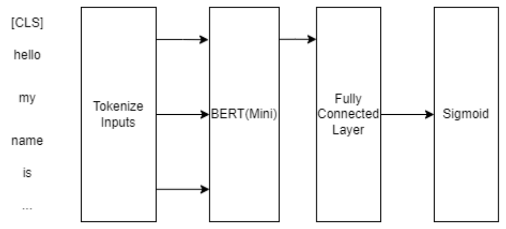

Fig. 6. BERT with Dense

C. Fine Tuning BERT-small With GRU

Another approach uses sequential embeddings for whole input sequences with bidirectional GRU layers [6]. GRU is a variant of RNN published much later than LSTM and is more compactly represented than LSTM [7]. The performance of

LSTM and GRU is often similar, but GRU is considered more efficient and fast trained, whereas LSTM tends to remember hidden states longer than GRU. By stacking two GRU cells where each one takes a sequence from opposite directions, the models learn the hidden state from both directions of the input sequence. The last hidden states of the GRU layers from both directions are forwarded to dense layers, which work as a classifier like previous methods.

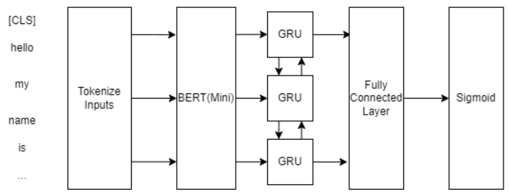

Fig. 7. Tuning BERT with GRU

D. Optimization and Loss Function

Since each emotion class is considered an individual binary classification task, we use binary cross-entropy as the optimization function. The models are trained with the PyTorch implementation of binary cross-entropy with a log-sum-exponent trick which provides better numerical stability [15].

Adam [16] is an iterative gradient-based optimization algorithm that combines momentum techniques with RMSProps and is considered one of the most popular optimization techniques for deep learning. Adam was often used with L2 regularization to penalize large parameters in the past. However, Ilya Loshchilov et al. showed that applying L2 regularization with Adam is different from weight decaying and is less effective [17]. Hence, Ilya Loshchilov et al. proposed AdamW [17], a variant of Adam that decouples weight- decaying from optimization steps to apply weight-decaying more effectively on the original Adam algorithm. In this experiment, We used HuggingFace's implementation of AdamW with linear warm-up, which gradually increases the learning rate up to the predefined point and decreases to zero linearly [18].

E. Metrics of Interest

In this experiment, we mainly considered F1 based metrics, including individual F1 score, macro-F1, and micro F1. F1 is the standard metric for measuring the performance of detecting positive classes. F1 scores in multi-classes could be represented in either macro-F1 or micro-F1 in multi-class classification. Macro-F1 is simply averaging the individual F1 scores over 27 emotion classes, whereas micro-F1 calculates F1 by considering the contribution of all classes while computing the metrics.

VI. Experiment Results

A. Comparing models

Models were trained with 20 epochs and were saved every half epoch. For each model, the best-performing state was used to compare the performance. Both models performed very similarly up to two decimal places in macro-F1 and micro-F1, shown below.

	Parameters	M-f1	m-f1	auc-roc
Baseline(BERT-base)*	110M	0.46	0.51	-
CLS Pooled(BERT-small)	28.8M	0.395	0.453	0.798
GRU-Based(BERT-small)	28.9M	0.394	0.459	0.799 height

Results as reported by [3]

Both scored a macro-F1 over 0.39 and a micro-F1 over 0.45, which is reasonable considering similar models trained with the original BERT architecture scored a macro-f1 of near 0.46 with much more parameters in [3] [8]. The GRU-based model scored slightly higher than the DNN based model in micro-F1; however, vice versa in macro-F1.

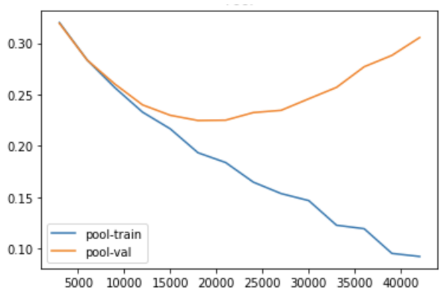

Fig. 8. Train vs Validation Loss, BERT-small, Pooled

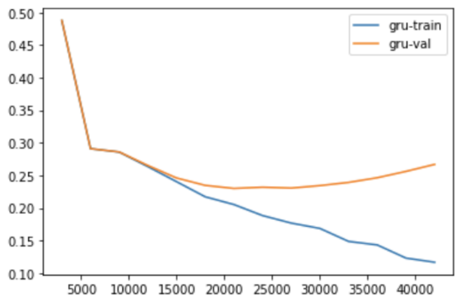

Fig. 9. Train vs Validation Loss, BERT-small, GRU

B. Positive Prediction Ratio during Training

It is also interesting to look at recall and the ratio of positive prediction over all predictions i.e. $(TP+FP)/Total$, during training. The ratios of positive labels were low in the dataset, i.e. there are more negatively labelled data than positively labelled. By plotting the ratio of positive predictions and the macro-recall, we see that the models blindly predicted the negative class at the beginning of training until they began to learn and predict with better recall.

C. ROC Curve and AUC

The GRU-based model scored an average AUC-ROC over 0.80, indicating that the models were able to distinguish the positive and negative classes for most emotion classes. Figure 11 shows the ROC curves for all 27 emotion classes. We see that all of the curves are concave over the diagonal line.

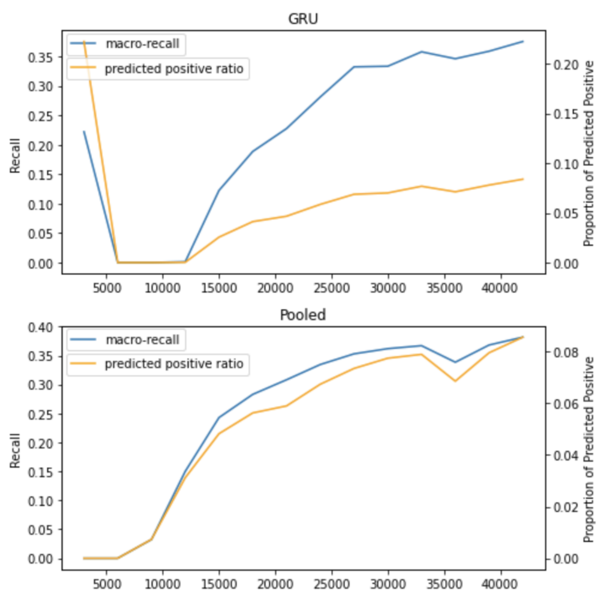

Fig. 10. Macro Recall vs Predicted Positive Ratio

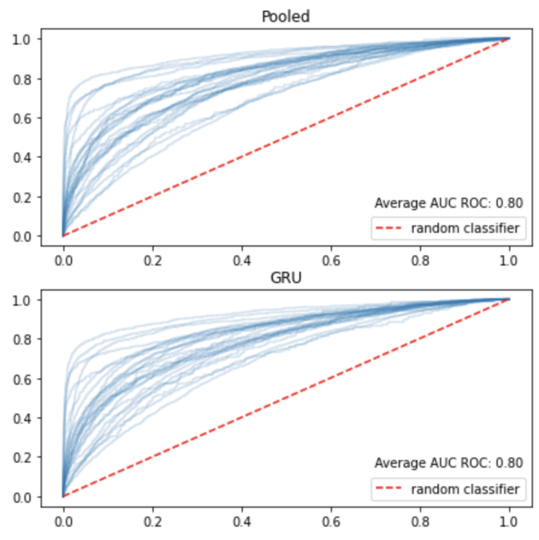

Fig. 11. Area under the curve ROC

D. Performance Over Emotion Taxonomy

Looking at the models' performance on individual emotions, we see that F1 scores differ severely across the emotions classes. For example, the models achieved F1 scores over 0.7 in love and gratitude, but they completely failed to detect grief or relief from the text. The Pearson coefficient values between the F1 scores and the percentage of positive labels are 0.46 and 0.43, respectively, for GRU-based and pooled models, indicating models tend to perform better with the more common emotions. However, it is not always the case since few emotions including remorse scored relatively high while rarely occurring in the dataset.

Emotion	Pooled F1	GRU F1	Positive Ratio
pride	0.00	0.02	0.02
grief	0.02	0.06	0.01
relief	0.16	0.03	0.02
embarrassment	0.16	0.09	0.04
nervousness	0.22	0.20	0.03
realization	0.25	0.26	0.14
desire	0.28	0.30	0.05
disappointment	0.29	0.31	0.13
excitement	0.32	0.34	0.08
disgust	0.37	0.38	0.08
caring	0.38	0.37	0.08
annoyance	0.40	0.44	0.19
confusion	0.40	0.43	0.10
optimism	0.42	0.43	0.12
disapproval	0.42	0.43	0.16
surprise	0.43	0.42	0.07
sadness	0.43	0.43	0.09
fear	0.46	0.45	0.04
joy	0.46	0.44	0.10
approval	0.46	0.44	0.25
anger	0.47	0.46	0.10
remorse	0.53	0.54	0.03
admiration	0.59	0.61	0.18
curiosity	0.64	0.65	0.11
amusement	0.68	0.70	0.10
gratitude	0.71	0.70	0.10
love	0.74	0.73	0.08

VII. Conclusion

We present two approaches to text-based emotion detection with the GoEmotions dataset [3], one of the largest annotated fine-grained emotion datasets. Our experimentation shows how two different fine-tuned BERT-small [4] [5] models behave. The DNN-based model performed slightly better with a macro F1 score of 0.395 than the GRU-based model with a macro F1 score of 0.394. Overall, the models performed very similarly. The DNN-based model tends to converge quicker than the GRU-based model. Although our model performed relatively well, it is still far from the macro F-1 scores achieved by larger models in [3], and [8].

As the analyses indicate, GoEmotions datasets contain imbalanced emotion sets, significantly influencing the models' performances in each emotion class. The results are also similar in [3], and [8]. Moving forward, we could improve the models by collecting more data on the low-performing emotion classes due to insufficient comments and training the models with specific emotions, i.e. grief and pride.

In our experiment, BERT-small was used due to the project scope. However, we could also leverage BERT with more parameters, i.e. BERT-base, to improve the models' performance as shown in the [3], and [8]. The performances increased drastically from BERT-mini to BERT-small(0.32 - 0.39), and we expect to gain a similar increase with BERT-base [5].

Furthermore, we could consider distinguishing the emotions annotated by readers and the writers as shown in [8]. Previous experimentation with the Vent [9] dataset in [8], showed promising results, surprisingly outperformed humans. The authors of [8] trained the model based on emotions annotated by the writers, in which the model outperformed readers at predicting writer's emotions by 19.9% relative to macro F1 scores. Additionally, the same model performed better at predicting reader annotated labels despite not having trained specifically for this task. These results indicate a step in the right direction for emotion detection classification.

References

[1] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018. [Online]. Available: http://arxiv.org/abs/1810.04805

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017. [Online]. Available: http://arxiv.org/abs/1706.03762

[3] D. Demszky, D. Movshovitz-Attias, J. Ko, A. S. Cowen, G. Nemade, and S. Ravi, "Goemotions: A dataset of fine-grained emotions," *CoRR*, vol. abs/2005.00547, 2020. [Online]. Available: https://arxiv.org/abs/2005.00547

[4] P. Bhargava, A. Drozd, and A. Rogers, "Generalization in nli: Ways (not) to go beyond simple heuristics," 2021.

[5] I. Turc, M. Chang, K. Lee, and K. Toutanova, "Well-read students learn better: The impact of student initialization on knowledge distillation," *CoRR*, vol. abs/1908.08962, 2019. [Online]. Available: http://arxiv.org/abs/1908.08962

[6] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *CoRR*, vol. abs/1409.1259, 2014. [Online]. Available: http://arxiv.org/abs/1409.1259

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[8] N. Alvarez-Gonzalez, A. Kaltenbrunner, and V. Gómez, "Uncovering the limits of text-based emotion detection," 2021. [Online]. Available: https://arxiv.org/abs/2109.01900

[9] N. Lykousas, C. Patsakis, A. Kaltenbrunner, and V. Gómez, "Sharing emotions at scale: The vent dataset," *CoRR*, vol. abs/1901.04856, 2019. [Online]. Available: http://arxiv.org/abs/1901.04856

[10] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," *CoRR*, vol. abs/1607.01759, 2016. [Online]. Available: http://arxiv.org/abs/1607.01759

[11] A. Cowen and D. Keltner, "Self-report captures 27 distinct categories of emotion bridged by continuous gradients," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 114, 09 2017.

[12] "Birkbeck spelling error corpus / roger mitton," oxford Text Archive. [Online]. Available: http://hdl.handle.net/20.500.12024/0643

[13] S. Özcan, "Preprocessor," https://github.com/s/preprocessor, 2020.

[14] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, L. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean, "Google's neural machine translation system: Bridging the gap between human and machine translation," *CoRR*, vol. abs/1609.08144, 2016. [Online]. Available: http://arxiv.org/abs/1609.08144

[15] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[16] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[17] I. Loshchilov and F. Hutter, "Fixing weight decay regularization in adam," *CoRR*, vol. abs/1711.05101, 2017. [Online]. Available: http://arxiv.org/abs/1711.05101

[18] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-Art Natural Language Processing." Association for Computational Linguistics, 10 2020, pp. 38–45. [Online]. Available: https://www.aclweb.org/anthology/2020.emnlp-demos.6