[Team 07] A Comparison of Traditional, Neural, and Transformer Approaches to Emotion Labeling

Aaron Casey
amrothem@ncsu.edu

Xavier Genelin
xgeneli@ncsu.edu

Alexej Lozevski
aglozevs@ncsu.edu

I. Motivation

Emotions are an integral aspect of human experience, cognition, and communication. Computer scientists, however, have struggled to develop models which capture the complexities of human emotion. Many sentiment classification studies simplify the classification of phrases into binary (positive, negative) or ternary (positive, neutral, negative) sets. While this simplification and quantification makes the task more accessible and is suited for many applications, other applications such as empathetic chat bots may require a more nuanced understanding of human emotion.

To attempt to expand the number of emotions which can be recognized in natural language text, we have opted for a discrete emotion model which classifies emotions into more fine-grained classes. We seek to classify texts into one of five emotions: mad, sad, joyful, powerful, and scared [9]. While far from representing the full web of human emotion, we feel that these categories provide a useful expansion of recognizable emotions.

II. Data

A. Dataset

In our project, we use the Empathetic Dialogues dataset to train our model to determine the emotion behind responses. The dataset consists of 25,000 conversations between a speaker and listener, and each conversation is human-labeled with one of 32 different emotions, including afraid, hopeful, lonely, and surprised [1]. Each label is assigned to between 425 and 1200 conversations, with between 3 and 8 utterances per conversation. We use only the first utterance from each conversation, as this utterance is a direct response to the question "Describe a situation when you felt {emotion}".

In order to evaluate several different methods of classifying emotion, we elected to build 3 models: a support vector machine classifier, a bidirectional long short-term memory neural network, and a partially retrained Bidirectional Encoder Representations from Transformers (BERT) model.

B. Combining Labels

Initially, we attempted to perform classification using all 32 emotions. However, all of our models failed to achieve an F1 Score greater than 0.40. We suspect that this failure owed to both data uncertainty in the differences between various labels (e.g. annoyed, angry, and furious), as well as a small number of examples per dataset.

Therefore, to simplify our task, we combined labels according to an emotional hierarchy developed by Gloria Wilcox [9]. Five emotion labels–Sentimental, Surprised, Nostalgic, Anticipating, and Impressed–did not translate neatly into the new categories, and thus were dropped from the dataset.

New Label	Original Labels
Mad	Angry, Jealous, Annoyed, Furious, Disgusted
Sad	Sad, Devastated, Ashamed, Guilty, Disappointed, Embarrassed, Lonely
Joyful	Joyful, Excited, Trusting, Caring, Grateful, Hopeful, Content
Powerful	Confident, Prepared, Proud, Faithful
Scared	Anxious, Apprehensive, Terrified, Afraid

TABLE I
Mapping from original labels to new labels

From the original 32 emotions in the dataset, the table below has the top 5 and bottom 5 emotions that were present. As we can see there are not many observations for the top emotions for a datasize of 25,000. There may not be enough data for each category for the model to be able learn enough information to give an accurate classification. There is also a notable difference between the top emotion, surprised, and the bottom emotion, faithful (table II).

Label	Count	Label	Count
surprised	1196	caring	614
excited	867	trusting	579
annoyed	819	ashamed	576
proud	806	apprehensive	565
angry	796	faithful	424

TABLE II
Original Label Counts

After combining the emotions, we get a slightly more even distribution of emotions as shown in the table below (table III). There are also more observations for each label, which will give more data for the model to learn from and more accurately classify emotion.

Label	Count
mad	4923
powerful	4879
sad	3725
scared	2756
joyful	2661

TABLE III
New Label Counts

III. Methodology

A. Training, Validation, and Test Data

Data was split into training, validation, and test sets before any experimentation was begun. The test data comprised 20% of the dataset, the validation set 16%, and the training set 64%. In order to preserve the integrity of our model, the test set was never used until the final evaluation.

B. Models

1) Support Vector Machine: Our baseline model was a support vector classifier. First, we used Stanford's GloVe word embeddings to create a list of 200-dimensional embeddings for each word in the dataset. We used GloVe embeddings trained on 6 billion tokens with a vocabulary of 400,000 words [5], with 9839 distinct tokens successfully mapped to our dataset.

Inv. Regularization Strength (C)	Kernel	Gamma	F1
1	Linear	NA	0.597
10	Linear	NA	0.605
100	Linear	NA	0.608
1000	Linear	NA	0.610
1	RBF	.001	0.192
1	RBF	.0001	0.081
10	RBF	.001	0.431
10	RBF	.0001	0.192
100	RBF	.001	0.578
100	RBF	.0001	0.432
1000	RBF	.001	0.601
1000	RBF	.0001	0.578

TABLE IV
Results for Cross-Validation of Support Vector Classifier

In order to address class imbalance, we over-sampled in our training dataset from the minority classes. We tuned using cross-validation with 12 different hyperparameter combinations table IV. The highest performing model achieved a macro-averaged F1 score of 0.610.

2) Bi-LSTM Network: Our second model was a bi-directional LSTM (bi-LSTM) model. We used the same GloVe embeddings as used for the support vector machine. However, instead of using oversampling to address class imbalance, we instead used a weighted cross-entropy to prioritize under-represented classes.

Our sequences were lower-cased and tokenized using the Sacremoses tokenizer [10]. Before being passed into the network, each token was mapped to the index of its matching vector in the embedding layer (or to the zero vector, if not found in the embedding). Then, the sequences were padded to a standard length for efficient parallelization.

Our model consisted of an embedding layer, a bi-directional bi-LSTM with 2 layers each with 512 hidden units per direction, and a 2-layer fully-connected network.

After the embedding layer, the padded sequences were packed, such that the output from padding tokens would not affect the bi-LSTM's output.

Then, the sequence was fed to the bi-LSTM. The final hidden layer output for each sequence (both the forward and backwards direction) was used as the input to a fully-connected layer, a dropout module, and a second fully-connected layer before outputs were passed to our cross-entropy loss function. Every model used Adam for optimization.

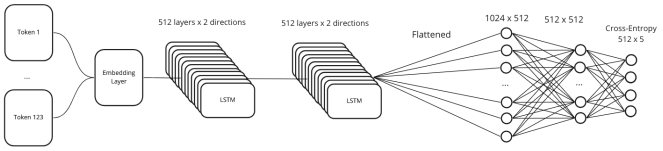

Fig. 1. LSTM Architecture

We used our validation set to optimize 8 sets of hyperparameters. All models were trained for 25 epochs. However, the reported score is for the best-performing epoch.

Epochs	Learning Rate	LSTM Layers	Dropout	F1
5	.001	2	.1	.717
5	.001	2	.2	.721
14	.0001	2	.3	.708
7	.0001	3	.3	.705
10	.01	3	.2	.695
11	.001	3	.3	.689
9	.001	3	.4	.700
8	.001	3	.5	.699

3) BERT Transformers: For our third model, we retrained, to varying degrees, a BERT transformer released by Google [6]. The BERT transformer model for classification is a transformer which uses attention mechanisms, but leaves out the decoder component of the transformer architecture, making it easier to re-train for transfer learning. The model consists of a 768 layer word embedding layer, 11 encoder layers, and a pooler layer (besides various sub-layers and encodings). In total, the BERT models contained 342,620 trainable parameters, although we froze many parameters due to the small size of our dataset and hardware constraints.

We used both the "bert-base-uncased" [7] model and the "twitter-roberta-base-sentiment" [8] models from HuggingFace using PyTorch and the Transformers library. For each base transformer, we appended a single dense layer to map the transformer outputs to our class labels. Then, we trained five different models, successively compounding the following additional layers for retraining: the dense layer, the transformer's pooling layer, the final encoder layer, the penultimate encoder layer, and finally the third-from-the-last encoder layer (such that the final model included retraining of all five mentioned layers, for example).

We expected that the "twitter-roberta-base-sentiment" model, having been based on the larger RoBERTa architecture and pre-trained for sentiment analysis, would outperform the base BERT model. However, the BERT model consistently outperformed the RoBERTa model. With 5 unfrozen layers, the BERT model achieved an F1 score of 0.798 (table V). We used the Adam optimizer with a learning rate of 10^{-5}, and no weight decay. Each model was trained for 25 epochs and validated using our held-out validation set.

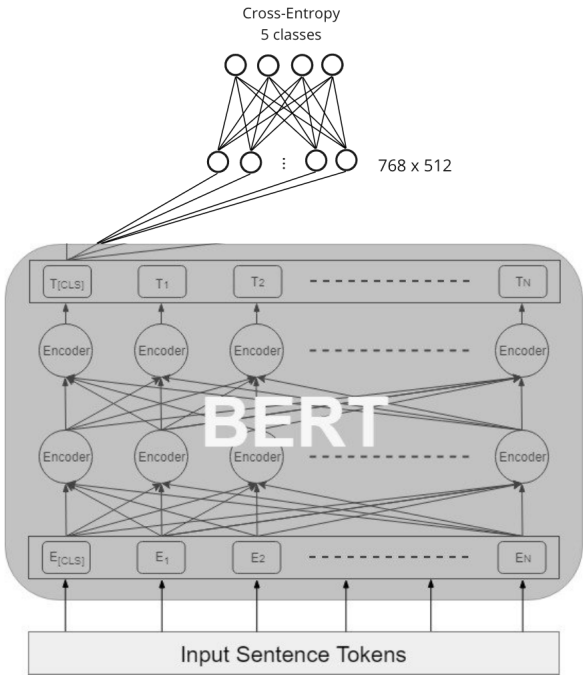

Fig. 2. Simplified Bert Architecture [11]

Model	Layers (Re-)Trained	Validation Macro F1
BERT	1	0.433
BERT	2	0.680
BERT	3	0.778
BERT	4	0.787
BERT	5	0.798
RoBERTa	1	0.250
RoBERTa	2	0.342
RoBERTa	3	0.484
RoBERTa	4	0.547
RoBERTa	5	0.553

TABLE V
MACRO F1 SCORES FOR EACH TRANSFORMER MODEL

IV. EVALUATION

In order to evaluate our final models, we retrained each on the combined training and validation sets. Each model was trained in the same way as its best performing precedent during tuning. As expected based on our validation results, the BERT model performed best, followed by the bi-LSTM (table VI).

Model	Test Macro-Averaged F1 Score
Bert	0.813
LSTM-CNN	0.707
SVC	0.613

TABLE VI
TRAINING PERFORMANCE FOR ALL THREE MODELS

A. Baseline SVC

The support vector machine was trained with a linear kernel and a regularization parameter of $C = 1000$. It achieved a macro-averaged F1 Score of .613.

B. bi-LSTM

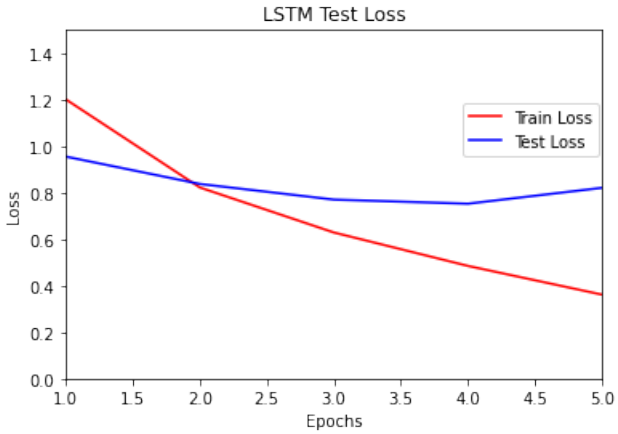

Fig. 3. LSTM loss as calculated on test data

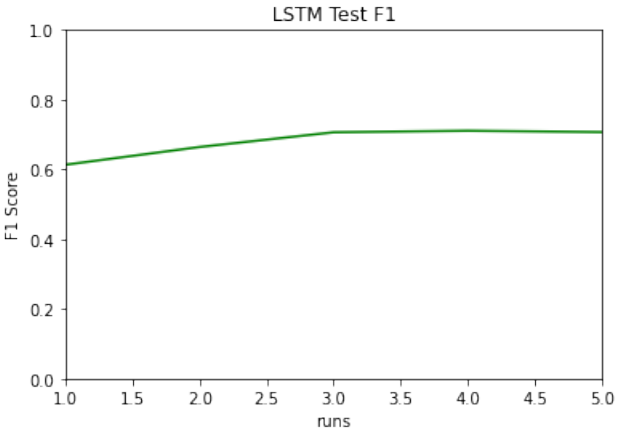

Fig. 4. LSTM macro-averaged F1 scores as calculated on test data

The bi-LSTM was retrained for 5 epochs with a learning rate of 0.001, 2 LSTM layers, and dropout of 0.2. It achieved a macro-averaged F1 Score of 0.707.

C. BERT

The base BERT model was retrained, with all 5 of the final layers retrained for 25 epochs. It achieved a macro-averaged F1 Score of 0.813 (table VI). It is observed that the test loss increases after the eighth epochfig. 5. However, the best F1 score was achieved in the final (25th) epoch fig. 6. This may reflect either an increase in model uncertainty (embodied as cross-entropy scalars for the true class further from 1) or the model learning the imbalanced classes.

D. Discussion

Based on our validation results, our bi-LSTM only required 5 epochs for training. Therefore, as well as because of its smaller number of parameters, the bi-LSTM trained in 1/22 of

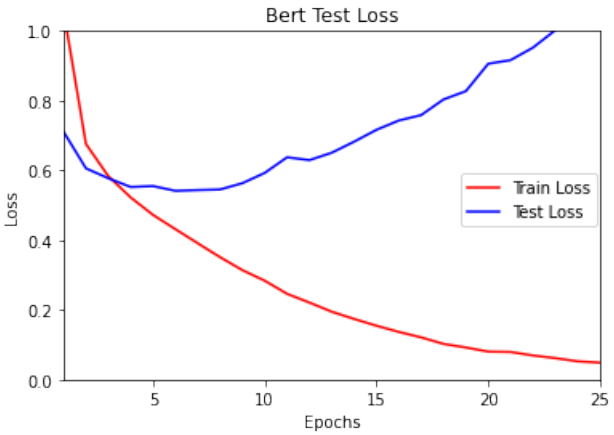

Fig. 5. BERT loss as calculated on test data

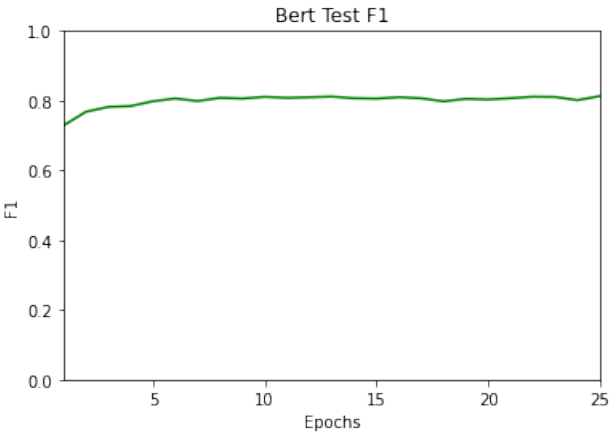

Fig. 6. BERT macro-averaged F1 scores as calculated on test data

the time than the BERT extension. Therefore, if training speed is a higher priority, we recommend considering the simpler bi-LSTM network. To our surprise, the bi-LSTM even trained more quickly than the support vector machine.

Because our F1 scores are still below .85, we recommend further experimentation with retraining the BERT model, as well as curation of a larger dataset, before using our sentiment classification approach in production.

REFERENCES

[1] H. Rashkin, E.M. Smith, M. Li, Y-L Boureau, "Towards Empathetic Open-domain Conversation Models: a New Benchmark and Dataset," Association for Computational Linguistics, 2019.

[2] NI Tripto, ME Ali. "Detecting multilabel sentiment and emotions from bangla youtube comments." 2018 International Conference on Bangla Speech and Language Processing (ICBSLP). IEEE, 2018.

[3] Z Huang, W Xu, and K Yu. "Bidirectional LSTM-CRF models for sequence tagging." arXiv preprint arXiv:1508.01991, 2015.

[4] Devlin J, M-W Chang, K Lee, Toutanova K. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", Association for Computational Linguistics, 2019.

[5] Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532-1543. 2014.

[6] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." In Advances in neural information processing systems, pp. 5998-6008. 2017.

[7] Devlin, Jacob, Ming-Wei Chang, Kenton Lee, en Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". CoRR abs/1810.04805 (2018). http://arxiv.org/abs/1810.04805.

[8] Barbieri, Francesco, Jose Camacho-Collados, Leonardo Neves, and Luis Espinosa-Anke. "Tweeteval: Unified benchmark and comparative evaluation for tweet classification." arXiv preprint arXiv:2010.12421 (2020).

[9] Willcox, Gloria. "The feeling wheel: A tool for expanding awareness of emotions and increasing spontaneity and intimacy." Transactional Analysis Journal 12, no. 4 (1982): 274-276.

[10] Alvations. "Sacremoses." https://github.com/alvations/sacremoses

[11] Eljundi, Obeida, Wissam Antoun, Nour El Droubi, Hazem Hajj, Wassim El-Hajj, en Khaled Shaban. "hULMonA (): The Universal Language Model in Arabic", 07 2019. https://doi.org/10.18653/v1/W19-4608.