COMP 551 Assignment 3: Classification of Textual Data

Okyanus Gumus[1], Mihail Calitoiu[2], Eren Gurhan[3]

[1]McGill ID: 260900481.
[2]McGill ID: 260972537.
[3]McGill ID: 260973884.

Contributing authors: okyanus.gumus@mail.mcgill.ca; mihail.calitoiu@mail.mcgill.ca; eren.gurhan@mail.mcgill.ca;

Abstract

In this assignment, naive Bayes and BERT models were implemented for emotion detection using the dataset Emotion. For the naive Bayes method, a pre-processing pipeline was designed where the unstructured text data was converted into numerical features. For the BERT model, on the other hand, the transformer package was used to tokenize the input text where the input tokens were then converted into numerical features. The developped models demonstrate the expected behavior and performance levels.

Keywords: Emotion dataset, Naive Bayes, BERT

1 Introduction

Emotion recognition allows computer to have some semblance of emotional intelligence and is an important part of human computer interaction Cowie et al. (2001). Emotion recognition has been investigated in prior works, such as in Tzirakis et al. (2017). A variety of different approaches have been used for diverse emotion recognition needs, ranging from convolutional neural networks to deep neural networks to multi view convolutional neural networks Parthasarathy and Tashev (2018); Khorrami et al. (2016); HE and Ferguson (2020). In this assignment, a deep learning approach with a text based dataset will be used.

This report is structured as follows: In Section 2, more information about the datasets is provided. The obtained results are found in Section 3. Discussion and conclusions are provided in Section 4, followed by the statement of contribution in Section 5. Relevant screenshots and plots are provided in Appendix A.

2 Datasets

The Emotion dataset is a dataset of Twitter (now known as X) messages in English, capturing six basic emotions: anger, fear, joy, love, sadness, and surprise. The dataset has a total of 417,000 rows of text. Out of these 417,000 rows a subset split of 20,000 rows of text is available. This 20,000 rows are portioned into a training subset of 16,000 rows, validation subset of 2,000 rows, and a test subset of 2,000 rows. There are two datafields; one field for the text and another field for the label, with the label being one out of six possible emotions. The distribution of the emotions are provided in Figure A1.

For this assignment, while implementing the Naive Bayes method, the text was converted into numerical features using the SciKit learn function

CountVectorizer1. On the other hand, for BERT, the PyTorch-Transformers library was used to tokenize the rows of text and, similarly, achieve numerical features.

3 Results

3.1 Task 1

Naive Bayes implementation:

For this assignment, we were asked to use the Emotion dataset of Twitter where every line of data is separated into "text" and "label". To make use of the data, we needed to use CountVectorizer from SkLearn to turn the unstructured text into numerical features. The steps taken to complete this step are as follows:

1) Download the dataset

For the initial step, we downloaded the dataset with the help of the dataset library which was installed into the Google Colab Notebook.

We separated the entire data into training and test data while ignoring validation since there are no hyperparameters that need tuning.

2) Vectorizing the data

To vectorize the data, we created an instance of CountVectorizer and fitted (and transformed) our training and test data. The output we got was similar to a matrix where the x-axis holds features (words) while the y-axis hold sentences of our dataset. When we convert the matrix into an array using the *.toarray()* function, we get a one-hot-encoded matrix which is ready to be fed into our model

BERT implementation:

The first step in processing the data for the BERT model was to convert the input strings into tokens. We decided to use the tokenizer that is provided by the pretrained model "bert-base-uncased," which is based off of Google's WordPiece tokenizer. WordPiece has a hybrid approach, specifically it combines both word-level and character-level tokenization, and is particularly useful for handling rare/unknown words. The main benefits of WordPiece are:

- **Efficient Language Representation:** Breaking words into sub words lowers model complexity but still captures the nuances of word morphology.
- **Robust Unknown Word Handling:** Unknown words are broken down into known subwords or characters, allowing handling of out-of-vocabulary words.
- **Reduced Vocabulary Size:** Model's vocabulary size is manageable, both lowering memory requirement and required computation. Once the strings were tokenized, we removed the original strings from the dataset and converted the dataset into PyTorch tensors (our models are PyTorch based).

3.2 Task 2

Naive Bayes implementation:

Naive Bayes is a probabilistic classification algorithm that calculates and compares the observed probabilities to test data for the purpose of classifying new test data. Naive Bayes is "naive" because it does not consider the order of words. As a result, Naive Bayes can not distinguish the difference between two sentences using the same set of words if the ordering of each sentence is different. While ordering of words are crucial to train a model that have a higher accuracy when classifying data, in our case, as can be observed, Naive Bayes was accurate enough. We were able to reach a 76.6% accuracy. To attain such accuracy levels, we have written a Naive Bayes class that includes a fit, predict, and evaluate functions.

1) Fit function:

Within the fit function, we divided the training data into six arrays corresponding to their labels. This step was taken to determine the probability of a word appearing in a single class later in the code. Afterwards, using an alpha value of 1, the priors have been calculated. We have selected an alpha value of 1 to ensure that there are no words with 0% probability. This precaution ensures that we never get a 0% class prediction to due multiplication by 0.

2) Prediction:

In the prediction function, we have separated the training data based on their labels, and determined the probability of a word appearing in each label. We stored these probabilities in an array to use it later in the code. Secondly, we take the logarithm of prior values and add it to the logarithmic sum of probability of words. The result of this summation is an array with six values. The index of the maximum element in this array corresponds to the predicted value for a sentence in the test data. This resulting prediction is also saved for use in determining the accuracy of our model.

3) Accuracy:

To determine the accuracy, we have set up a FOR loop which counts the instances when the prediction and true-labels match. This count is divided by the length of the label array and multiplied by 100 to get a percentage value. The Naive Bayes Class can be called just like any other class where we first fit the X train and Y train into the model, then we get the predictions with X test and compare the accuracy with Y test.

Bert implementation:

The first step in detecting emotion from a sentence is to understand how the BERT model works. BERT stands for "Bidirectional Encoder Representation from Transformers" and is often used for machine translation, question answering, sentiment analysis, text summarization etc.

BERT - MLM and NSP objectives:

The BERT model used for this assignment was taken from the website Hugging Face, specifically the "bert-base-uncased" model. The model is trained with two objectives, specifically masked language modeling (MLM) and next sentence prediction (NSP). The MLM process involves masking 15% of a sentence's words (i.e "What a wonderful day" becomes "What a wonderful [MASK]", where the word "day" is replaced with [MASK]), passing the sentence to the model and receiving BERT's prediction for the masked words. This allows BERT to learn the bidirectional representation of the dataset's sentences. The NSP process involves concatenating two sentences, passing it to the model, and receiving the model's prediction of whether the two sentences follow each other (i.e. "What a wonderful day. The sun is out."). After accomplishing both objectives, the model learns the inner representation of the English language, and can be used for downstream tasks (in this assignment's case, emotion classification). Additionally, it is important to note that the chosen model is large, specifically with 110M parameters, and is uncased, meaning that the words have been made lower case.

Emotion model architecture and training overview:

To adapt the BERT model for emotion classification, a final layer needs to be added, with six output neurons, representing the probabilities of the sentence showing the emotion of "sadness," "joy," "love," "anger," "fear" or "surprise." The newly added neurons have attached weights which need to be modified via training. Several options exist when training the model, particularly 1) the modification of all weights, including those inside the pre-trained BERT, and 2) the adjustment of weights in the final layer while freezing all of BERT's weights. Additionally, the Hugging Face website provides several pretrained, BERT-based, emotion classification models, with the same shape as described before, but fine-tuned to yield optimal accuracy. We will investigate all three models and compare them.

Pre-trained emotion model:

Firstly, we will explore a fine-tuned model for emotion classification. From the Hugging Face website, we imported "bhadresh-savani/bert-base-uncased-emotion," which is a BERT-based model with an additional layer. The model is fine tuned using a learning rate of 2e-5, batch size of 64 and 8 epochs, and achieves an accuracy of 92.65%. We verified this claim using a while loop that iterates through the test dataset and counts the number of correct predictions.

Weight freezing for pretrained BERT + classification layer:

Next, we will train a new emotion classification model, expanding from a pre-trained BERT model. As mentioned previously, we will explore the difference between updating all weights vs. final layer weights. Using the Hugging Face Trainer, we froze the weights inside the BERT layers (by setting, requiredGradient to false) and trained the model. We achieved 85.5%. Next, we trained the model without freezing any weights,

and achieved 93.2%. This suggests that the BERT layers also need to be adjusted for emotion classification. This finding is logical given that, although BERT is well trained, classifying sentences into specific emotions is difficult. One layer of customized weights (in the last layer) for emotion classification is not sufficient. Instead, leveraging BERT's understanding of language, and performing transfer learning by modifying all the weights is the ideal approach.

Fine tuning investigation:

Investigating the effect of the trainer's parameters (finetuning) is important to better understand the model behaviour. We chose to investigate two parameters, namely learning rate and batch size. Learning rate is a critical parameter that determines the extent of how much a weight is updated. Batch size is also important, a small batch size requires less memory but also increases the noise in gradient estimates. Large batch sizes are more beneficial, but require more memory. A balance in between is key.

For learning rates, we chose the learning rates of 16, 12, 8, 4. For batch size, we investigated the sizes 2E−3, 2E−4, 2E−5, 2E−6. Results of the fine tuning are provided in Table 1.

Looking at Table 1, we see that batch size has a minimal effect on overall accuracy. However, loss decreases as batch size increases. For learning rate, accuracy significantly decreases for the large learning rates. The ideal combination of batch size and learning rate is 16 and 2e-5 respectively.

3.3 Task 3

Naive Bayes and Bert comparison:

As can be seen from Table 2, BERT performs much better than Naive Bayes. This is because Naive Bayes makes predictions only based on the probability of a word appearing in training set while BERT is capable of understanding the context behind a phrase. Moreover, BERT is also capable of bidirectional processing. Bidirectional processing is the ability to consider words on the left and right side of the current word. This processing allows the BERT model to capture more complex relationships in sentences.

All of the reasons listed above are the reasons behind BERT's higher accuracy rates. It is important to note, however, that BERT is a complex model which requires larger datasets, more computing power, and longer training time.

As a result, if our tasks requires classification of complex relationships such as classifying the emotions conveyed in a sentence, it would be better to stick with BERTs. However, if we are dealing with simpler classifications such as detecting spam and non-spam emails Naive Bayes can also be a good contender.

BERT attention matrix exploration:

Looking at an attention matrix offers an insightful way into determining how different words/tokens of a specific phrase contribute to the overall emotional context. We created an attention matrix using the library "bertviz" from the already trained emotion model "bhadreshsavani." From the assignment requirements, two phrases were used, one correctly classified and one incorrectly classified. We chose the following two phrases because they have similar length.

- **Incorrect phrase:** "i feel if i completely hated things i d exercise my democratic right speak my mind in what ever ways possible and try to enact a change" with predicted emotion of sadness and actual anger
- **Correct phrase:** "i never make her separate from me because i don t ever want her to feel like i m ashamed with her" with predicted and actual emotion of sadness.

Looking at Figure A3 and Figure A4, we see the attention matrices for the two phrases.

We investigated the 5th layer's attention since it offers a balanced representation that is neither too raw nor too abstract. Looking at the phrases, the incorrect phrase should give more attention to the key words "never,"separate," "dont," "want,"feel," "ashamed" to be correctly classified.

We selected each head and determined that the dark grey head has meaningful attention. The model gives has highest attention for the words "never", "want" and "ashamed", which likely leads to the classification of "sadness." While the model gave a wrong prediction, it is important to note that the sentence is ambiguous even for humans. Our team members also had trouble classifying the sentence as "anger." As a result, this

Table 1 Fine Tuning: Comparison of Batch Size and Learning Rate

Batch Size	Accuracy(%)	Loss	**Learning Rate**	Accuracy(%)	Loss
4	93.5%	0.26	2e-3	34.75%	1.56
8	93.7%	0.20	2e-4	34.75%	1.56
12	93.3%	0.17	2e-5	92.95%	1.19
16	93.5%	0.16	2e-6	77.65%	0.70

Table 2 Accuracy comparison of different models

Models	Accuracy(%)
Naive Bayes ($\alpha = 1$)	76.6
BERT (Compleletely Pre-Trained)	92.68
BERT (Pre-Trained BERT + All Layer training)	93.2%

error shows that the emotion conveyed by the sentence is not exactly clear and that it would be wrong to claim that the model was at fault to misclasify the sentence.

For the correctly classified phrase, we used the same head as before. The words given most attention are "feel," "hated," "democratic," "way," "possible" and these words suggest a feeling of sadness, leading to the correct classification.

Pre-training on external corpus:

Using a pretrained BERT model, which was trained on an external corpus, offers significant advantages for the task of emotion prediction. Specifically, "bert-base-uncased" is trained on 74 million phrases, allowing it to develop a thorough understanding of the semantic and syntactic structure of language. The addition of a layer (6 neurons), and performing transfer learning, fine-tunes the model for specific emotion prediction. This approach achieves 93% accuracy, which shows that the model is indeed able extend its understanding of language to a classification task.

Deep learning vs traditional machine learning methods:

When comparing deep learning (specifically BERT) to traditional machine learning methods, several conclusions can be made. Deep learning are capable of detecting complex patterns in language, allowing many tasks to be completed, from machine translation to emotion detection. Large data sets are handled correct and can be used to train. Traditional machine learning relies on hand-crafted features, which are usually difficult to extract and thus often limits its usefulness. The main downside of deep learning, which is directly a cause of good it is, is the intense computation required for training.

3.4 Bonus Section

In addition to the above sections, we decided to explore the confusion matrix of the pretrained BERT model to identify which emotion labels are often misclassified. Looking at Figure A2, we can see that most misclassification occurs between the class "love" and "joy," with 56 incorrect occurrences. This is understandable because the difference is nuanced, and can be challenging even for humans. Additionally, following a similar pattern, "fear" and "suprise" are the next pair of misclassifed class pairs, with 22 occurrences. Again, the difference is nuanced; for example, a sudden surprise can generate fear. Alternatively, when experiencing fear, there can be an element of surprise (for example a jump scare in a horror movie).

Although the BERT-based model shows some difficulty for similar emotions, it is able to classify between the similar pairs significantly more times than it does wrong. As such, we can conclude that BERT is indeed capable of identifying the subtle nuances of language, and be used for many language applications.

Additionally, we have tested the accuracy of Naive Bayes with 5 alpha values ranging from 0.01 to 50. We have concluded that an alpha value of 1 corresponds to the highest accuracy level. We believe when the alpha value is too small the model over fits, while for large alpha values it leads the model to under fit. As a result, it is important

to pick the best alpha value that strikes a balance between under fitting and over fitting.

4 Discussion and Conclusions

In this assignment, we have implemented naive Bayes and BERT models and compared their accuracy. From our results, we have concluded that BERT models perform better since they are trained on larger datasets, are capable of understanding complex relationships due to bidirectional processing, and have the ability to handle ambiguities compared to Naive Bayes.

As a result, for complex tasks, we have determined that BERT is a good method. For simple tasks that do not require a model to understand complex relationships, Naive Bayes can also be considered due its faster training time and lower computation power need.

5 Statement of Contribution

The contributions of the group members are as follows:

- Okyanus Gumus: Writing the assignment report.
- Mihail Calitoiu: Implementing the ML models, BERT.
- Eren Gurhan: Analyzing the datasets, naive Bayes.

Appendix A Screenshots

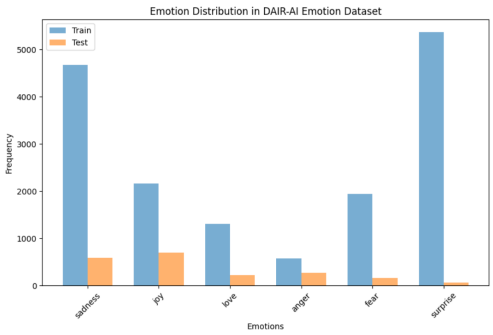

Fig. A1 Distribution of the emotions in the Emotion dataset.

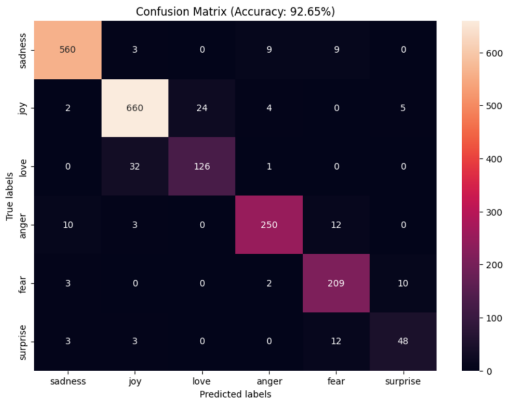

Fig. A2 Confusion Matrix for Pretrained BERT-based emotion classifcation model.

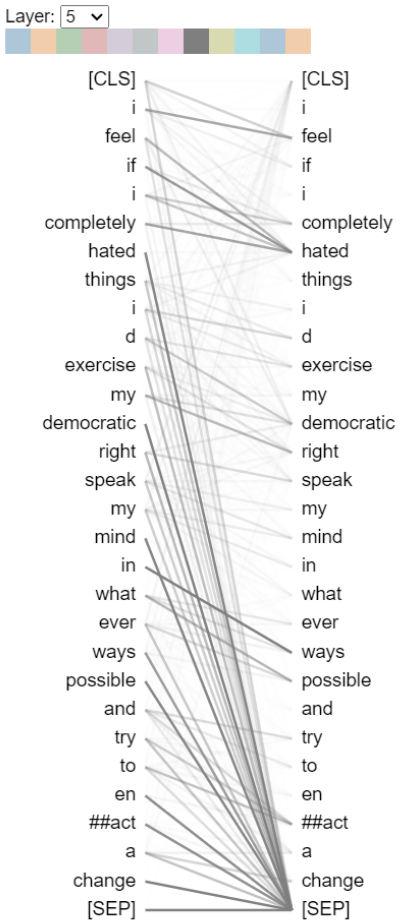

Fig. A3 Attention matrix of correctly classified.

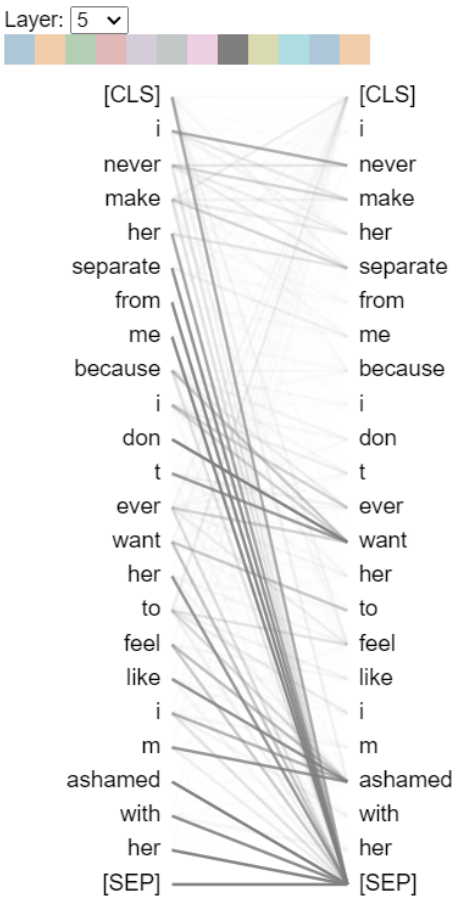

Fig. A4 Attention matrix of incorrectly classified.

References

Cowie, R., Douglas-Cowie, E., Tsapatsoulis, N., Votsis, G., Kollias, S., Fellenz, W., Taylor, J.G.: Emotion recognition in human-computer interaction. IEEE Signal Processing Magazine **18**(1), 32–80 (2001) https://doi.org/10.1109/79.911197

Tzirakis, P., Trigeorgis, G., Nicolaou, M.A., Schuller, B.W., Zafeiriou, S.: End-to-end multimodal emotion recognition using deep neural networks. IEEE Journal of Selected Topics in Signal Processing **11**(8), 1301–1309 (2017) https://doi.org/10.1109/JSTSP.2017.2764438

Parthasarathy, S., Tashev, I.: Convolutional neural network techniques for speech emotion recognition. In: 2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC), pp. 121–125 (2018). https://doi.org/10.1109/IWAENC.2018.8521333

Khorrami, P., Le Paine, T., Brady, K., Dagli, C., Huang, T.S.: How deep neural networks can improve emotion recognition on video data. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 619–623 (2016). https://doi.org/10.1109/ICIP.2016.7532431

HE, N., Ferguson, S.: Multi-view neural networks for raw audio-based music emotion recognition. In: 2020 IEEE International Symposium on Multimedia (ISM), pp. 168–172 (2020). https://doi.org/10.1109/ISM.2020.00037