COMP 472 MP1 - Analysis

Professor:
Leila Kosseim

TeamName:
COEN & SOEN

Members:
Brandon Takli (40124336)
Facundo Obeid (40130234)

Submission Deadline:
October 22, 2022

We certify that this submission is the original work of members of the group and meets the Faculty's Expectations of Originality

Table of Contents

1. Dataset Analysis:

Our dataset, which was a modified version of the *goemotions* dataset of Reddit posts and associated classifications of *Emotions* and *Sentiments*. The following pie charts show the distribution of these labels for each classification category:

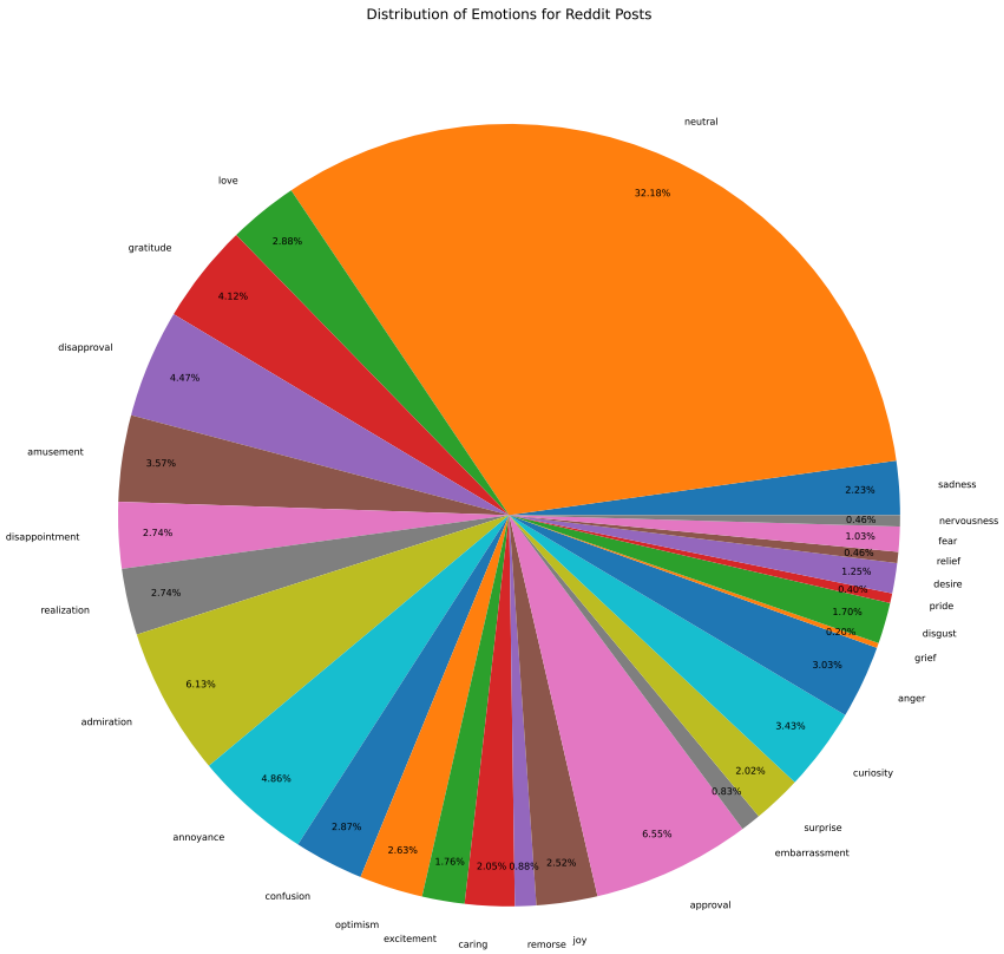

Figure 1.1: Distribution of Emotions for Reddit Posts

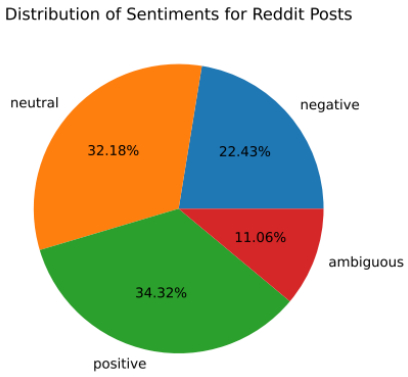

Figure 1.2: Distribution of Sentiments for Reddit Posts

As can be seen in *Figure 1.1* and *Figure 1.2*, the dataset was shockingly imbalanced for emotions, with 32.18% of the 28 classes being *neutral*. This is problematic, as *neutral* is now overrepresented in the dataset which led to problems when it came to classifying the other classes. *Sentiments*, on the other hand, was more balanced than *Emotions*, but still could have been better distributed. For this reason, we came to the following conclusions regarding the metric to use for better comparisons of model performance for the Top models using Gridsearch:

- Accuracy is not a good option since our class set is imbalanced so we will falsely get high accuracy.
- Precision and recall are better options due to not being affected by class imbalance.

Thus, we will use F1 score as our evaluation metric since it is a good metric for imbalanced datasets since it comes from precision and recall. We will show a graph at the beginning comparing the averages (macro and weighted) of the f1-score and accuracy, the latter of which is not a great metric for our imbalanced set.

In general, the classification performance of the *Sentiments* models performed better than the *Emotions* models, we believe this is due to the class imbalance being less pronounced in the *Sentiments* model.

2. Analysis of Results:

For each of the following models, *train_test_split* was used to create an 80/20 training/test set split. The same seed was used throughout the project to allow for a more proper comparison among all the models and feature extraction techniques. All models were pickled (technically saved using joblib, which uses pickle under the hood) in order to not have to repeat training every run of the notebook, which would take many hours.

Furthermore, MLP runs were limited to 80 iterations due to the amount of time it took per run at the default 200 iterations (~8 hours for gridsearch). Even at 200 iterations, MLP models did not converge. Comparing the previously done 200 iteration runs with the 80 did not yield that significant of a difference (typically +/- 0.05 at most) so we are not too concerned with this iteration difference. An additional important note is that one of the predetermined values of alphafloat for our Top MNB model was 0. Sklean does not allow a value of 0 and instead automatically sets it to a very small value near 0. This was discussed with the professor and deemed acceptable. The tradeoff in iterations allowed us to experiment with more values and improve our results a bit for the Top models.

As mentioned before, we will compare the averages (macro and weighted) of the f1-score and accuracy of each model here, just note that it is not the best metric for this unbalanced dataset:

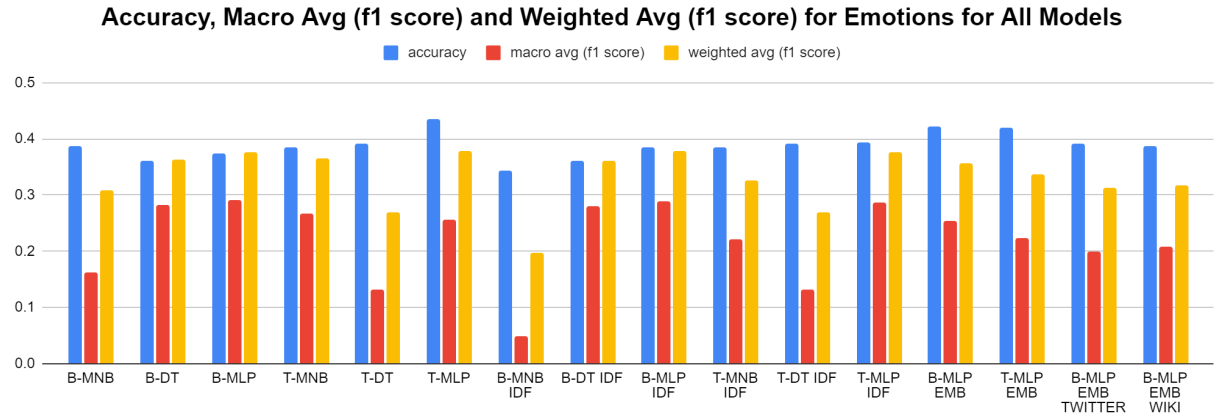

Figure 2.1: Accuracy, Macro and Weighted Average of F1 Score for Emotions All Models

A quick note regarding our shortened naming convention: B = Base, T = Top, IDF = tf-idf. EMB = embedding. This is maintained throughout the report.

From this figure we notice a trend: Accuracy is generally at or near the top metric for all models. Even in our worst performing model: the B-MNB IDF (Base-Multinomial Naive Bayes using tf-idf) model, we see an accuracy around 34%, which coincides closely with the 32.18%

size of the *neutral* emotion in our dataset. This shows us why accuracy is a bad measure for unbalanced datasets: Generally even for quite poor models, we tend to get at least an accuracy equal to the proportion of the largest unbalanced set.

The F1 score is a better measure and by looking and the weighted average of it, which is proper once again due to the unbalanced dataset, We can see that overall MLP was the best model for each of our feature analyzing techniques. Overall, the best model in terms of weighted average of F1-score was B-MLP IDF with a weighted average of 0.3783. It is worth noting that the top model was closely followed by T-MLP, T-MLP IDF and B-MLP who each had weighted averages of 0.3774, 0.3767 and 0.3753 respectively. It can be said that the tf-idf and embedding models performed similarly here, and our averaged embeddings ran *substantially quicker* than our tf-idf and count vectorizer models (4x quicker or more). So in terms of quality per unit time, embeddings wins by far.

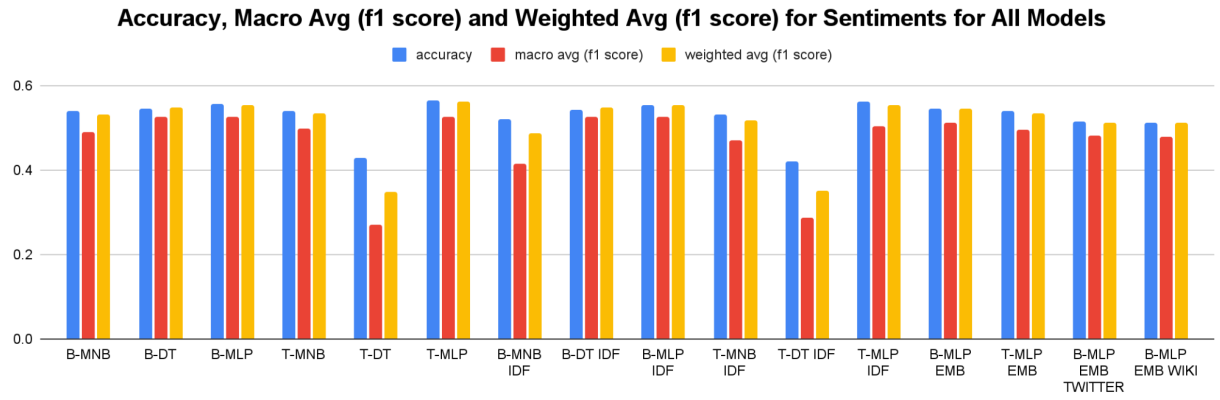

Figure 2.2: Accuracy, Macro and Weighted Average of F1 Score for Sentiments All Models

Looking at the averages and accuracy for *Sentiments* reveals a much higher on average performance than *Emotions* did. As mentioned prior, we believe this is due to the strong imbalance present in the emotions dataset, whereas the imbalance in sentiments is far less pronounced. This could also be due to the sheer number of classes in emotions (28) versus a much smaller 4 in sentiments. We can see here that in general, all 3 values of accuracy, macro and weighted average for f1-score are fairly close to each other compared to the wild differences observed for some of the models for emotions (notably T-DT). What is interesting to note is that our previous worst model (B-MNB IDF) performs quite alright here, with the new worst being T-DT IDF, whose performance is nowhere near as bad as B-MNB IDF was for emotions.

Our best performing model here in terms of weighted average of f1 score is T-MLP, with 0.5616, slightly edging out B-MLP which had 0.5543 and B/T-MLP IDF which hovered around the same. Overall it is plain to see that the MLP models tend to perform the best across both emotions and sentiments. Given their far longer compute times than any other models, this makes sense. The poor performance of some of our Top models is due, in part, to us not having

enough time to play around with different variables given the extremely long compute times of the MLP models and the deadline of the project.

2.1 Using Word Frequencies

The following models were trained by converting each post into word frequencies by using the *CountVectorizer()* from *sklearn.feature_extraction.text*. A total of 30449 tokens were found in the vocabulary. Our hypothesis was that the word frequencies would perform worse overall than tf-idf and embeddings, but this was not the case and the results are more mixed overall than initially anticipated. The best classifier for sentiments was Top-MLP which used count vectorizer. Furthermore, the Base-MLP results were not far off from the tf-idf Base-MLP which was the best for emotions.

2.1.1 Base-MNB

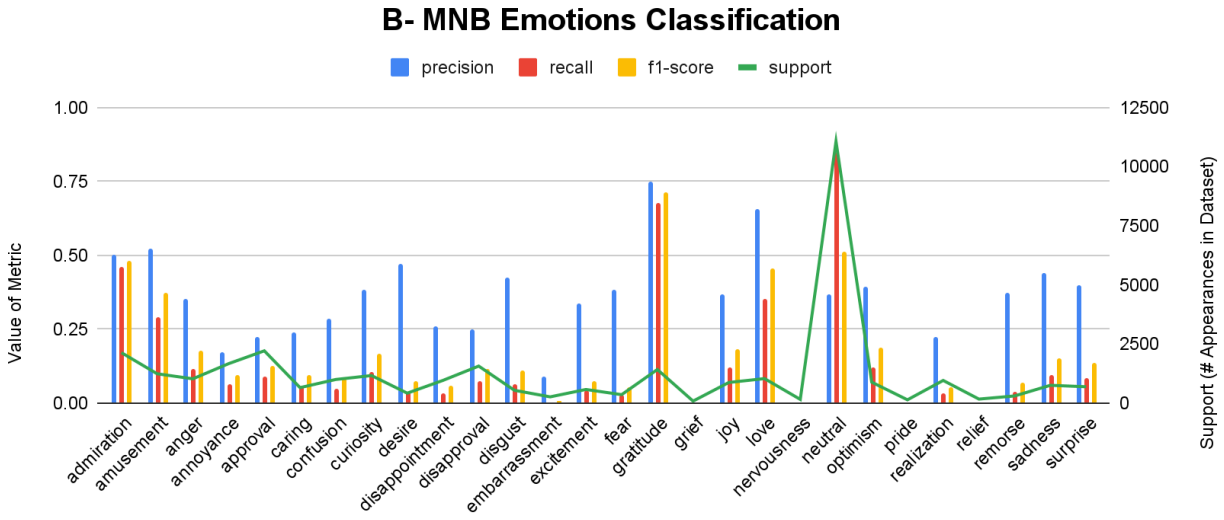

Figure 2.1.1.1: B-MNB Emotions Classification

These figures tell an interesting story based on the correlation of the *support* value and the values of the other metrics. *Figure 2.1.1.1* shows some emotions such as grief and relief have values of 0 for all their metrics. By looking at the support value, we can see that there are relatively extremely few of these emotions in the dataset: Out of 34364 posts, only 87 (or 0.2531%) are classified with grief. This explains the 0 values for all the metrics: It is essentially an insignificant classification. This trend of low/no metric values for low occurrences of a class continues for all our models, to varying degrees of severity. There are a couple outliers however, particularly gratitude, which has a relatively low support but very high values for the metrics. This is possibly due to the fact that posts classified as gratitude are fairly distinct from the rest of

the emotions, for example, "*thank you so much!"* is something that would be common in a post classified as gratitude and fairly rare in other types of posts.

Furthermore, precision seems to be substantially higher than all the other metrics in the majority of cases here aside from neutral, where recall is dominant. Given that precision is defined as the proportion of instances labeled with the class of interest that are correct, this makes sense, as due to our imbalanced dataset, for the model to classify an instance as something other than neutral, it has to be fairly certain that that instance is not neutral, so the number of errors is lower. Neutral has a lower precision since a lot of other classes that should be a class other than neutral are labeled as neutral. Recall is high because evidently due to the high number of neutral instances, the odds of an instance being labeled correctly is high.

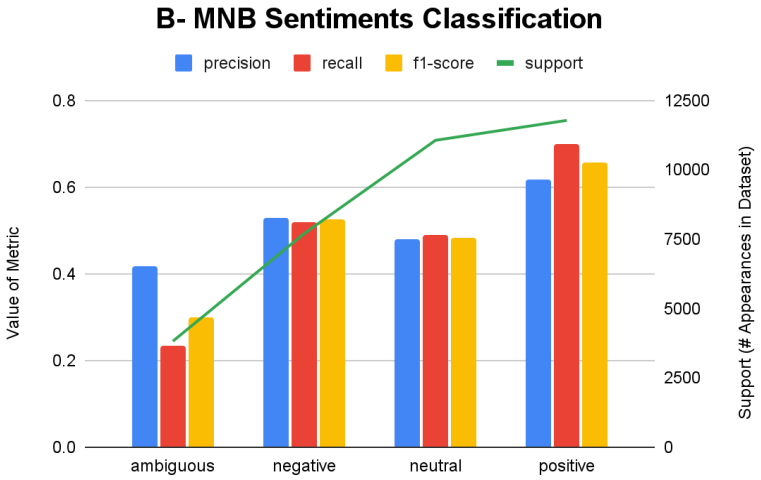

Figure 2.1.1.2: B-MNB Sentiments Classification

Figure 2.1.1.2 shows us that for *Sentiments*, the correlation is similar: Generally, the higher the support, the higher the value of the metrics. Since the sentiment set is more balanced, the issue is not as apparent, but it still visibly has an effect on the performance of our models. Again, due to the more balanced nature of these labels, the metrics are all about even, which is in stark contrast to the emotions labels.

2.1.2 Base-DT

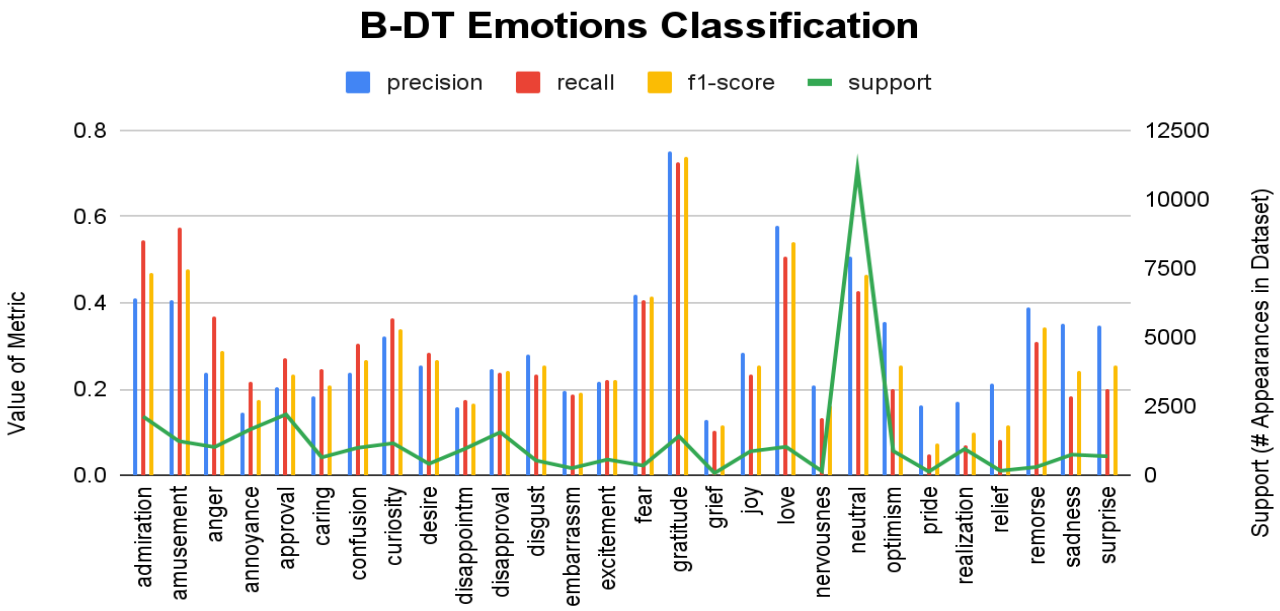

Figure 2.1.2.1: B-DT Emotions Classification

This decision tree model shows a surprising shift from the standard: The labels with the highest values for all 3 metrics are love and gratitude, which have a far lower support than neutral. This could be due to these types of posts being fairly unique in their word choice and appearance, and also due to the fact that while not high in support, their values are not insignificant either, giving the model more posts to train on than something like pride or grief. Compared to *2.1.1 Base-MNB* the 3 metrics are far closer to each other on average, with precision no longer being dominant. Furthermore, while still low, labels such as grief and pride no longer have 0 values for the metrics. This shows us that the decision tree did a far better job of picking up on the specifics of each label and was better able to classify each post.

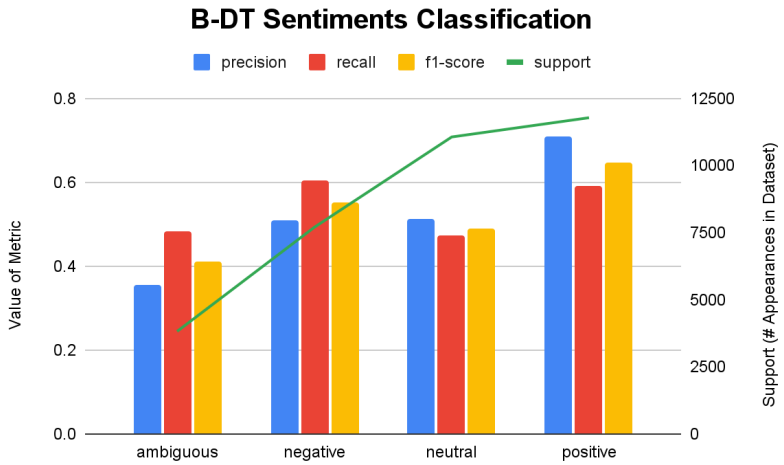

Figure 2.1.2.1: B-DT Sentiments Classification

There is not much difference between this figure and the one in the previous section. The basic trend is the same but with an overall improvement for the metrics. Precision became less pronounced in a relative sense for the ambiguous class, but an increase in the f1 score was observed as well.

2.1.3 Base-MLP

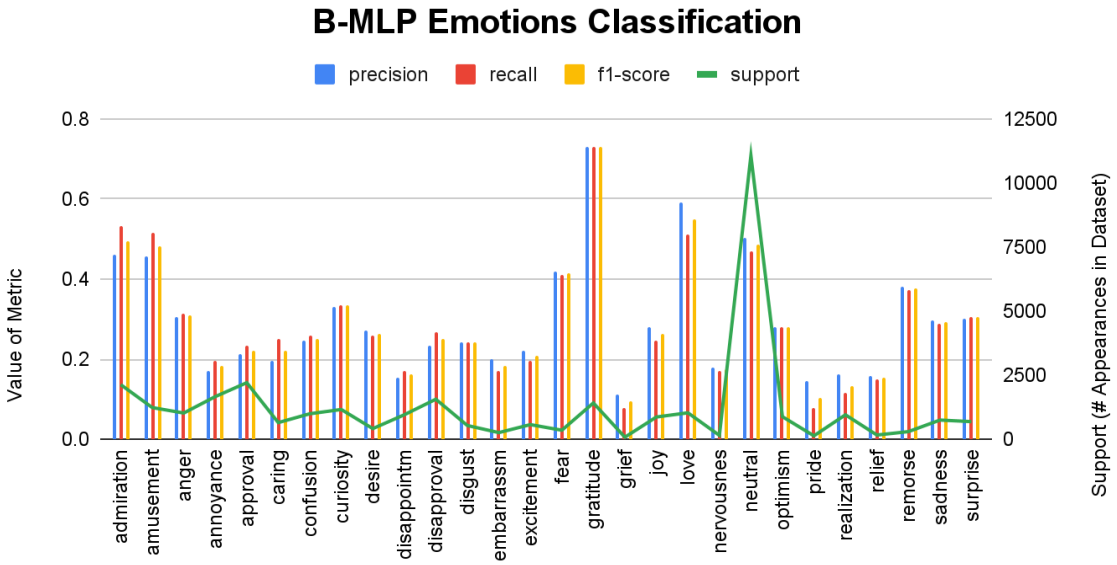

Figure 2.1.3.1: B-MLP Emotions Classification

The Multi-Layered Perceptron model demonstrates the same general shape as the previous B-DP model, where the top labels are still gratitude and love and the support curve looks practically identical. An interesting observation is that in this model, the precision, recall

and f1-score values of each label are much closer to each other compared to previous models. This means that the current model is classifying an approximately- equal amount of false positives and false negatives for every label.

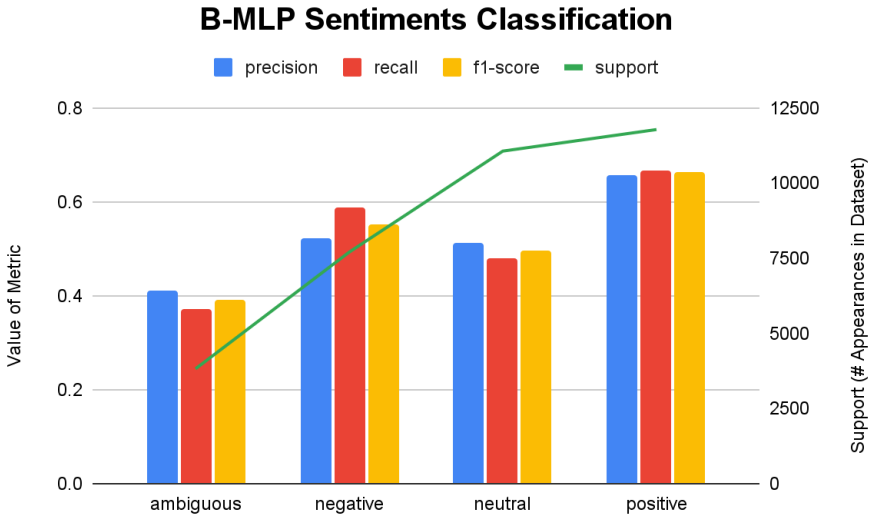

Figure 2.1.3.2: B-MLP Sentiments Classification

As for the sentiment classification of this model, it seems the recall values for the first three labels have decreased, while the value for the "positive" label has increased, which could be due to a higher number of true positive cases. Like the emotions classification, each label in this model has similar precision, recall and f1-score values.

2.1.4 Top-MNB

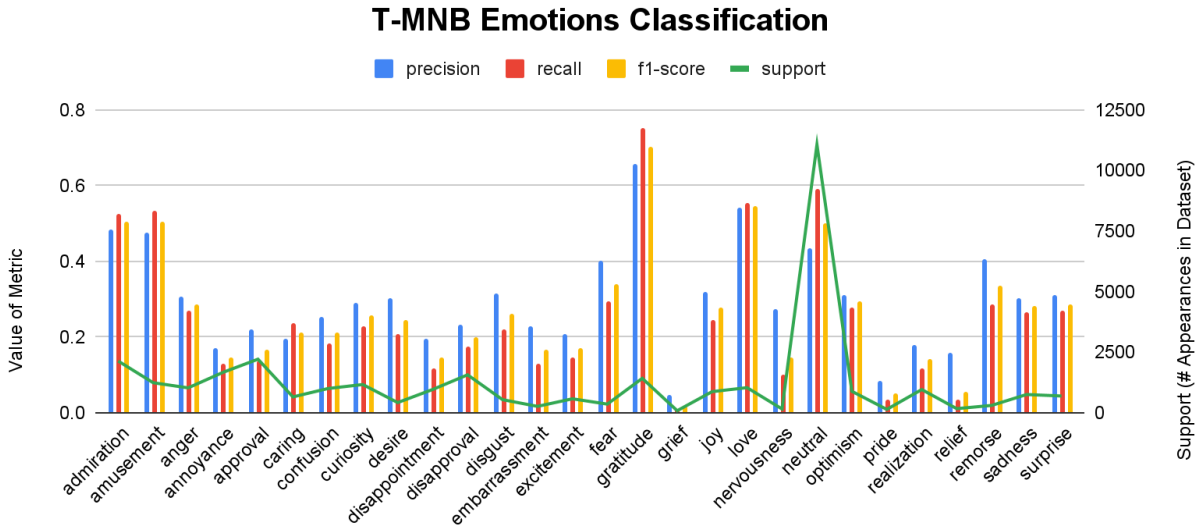

Figure 2.1.4.1: T-MNB Emotions Classification

The Multinomial Naive Bayes Classifier model is now set up using “GridSearchCV”, which resulted in a hyper-parameter “alpha”=0.21 being used for this model. From comparing the results seen in Figure 2.1.4.1 to the ones in Figure 2.1.1.1, it can be seen that the recall and F1-score values for seemingly every label increased, while the precision decreased or remained the same. From such results, it can be concluded that the selected hyper-parameters were beneficial in making the model more successful overall.

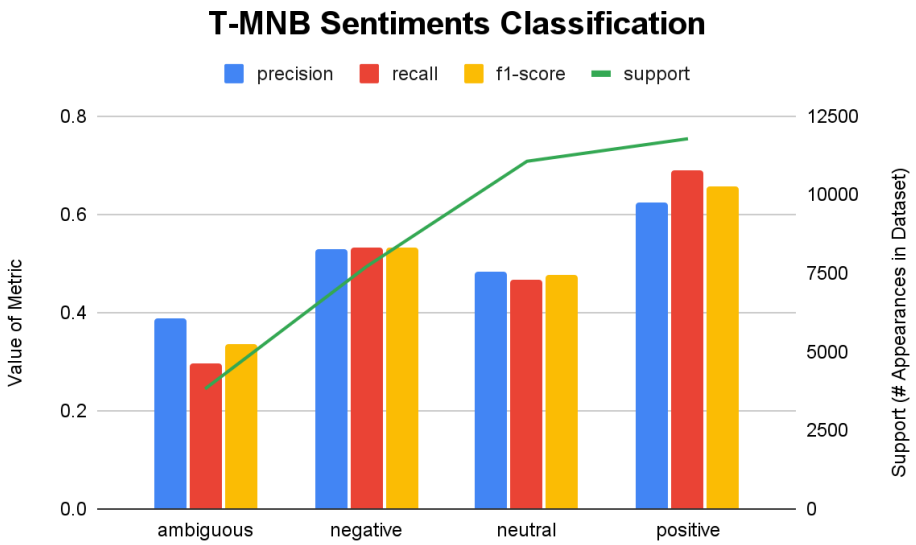

Figure 2.1.4.2: T-MNB Sentiments Classification

For the sentiment model, a hyper-parameter “alpha”=0.48 was used. When comparing the sentiment classification for T-MNB, it can be noted that there is not much difference between Figures 2.1.1.2 and 2.1.4.2, with slight differences in values, but the overall shapes and distributions remaining very similar.

For Multinomial Naive Bayes Classifier models, GridSearchCV’s hyper-parameter optimization seems to lead to more successful models in terms of F1-scores.

2.1.5 Top-DT

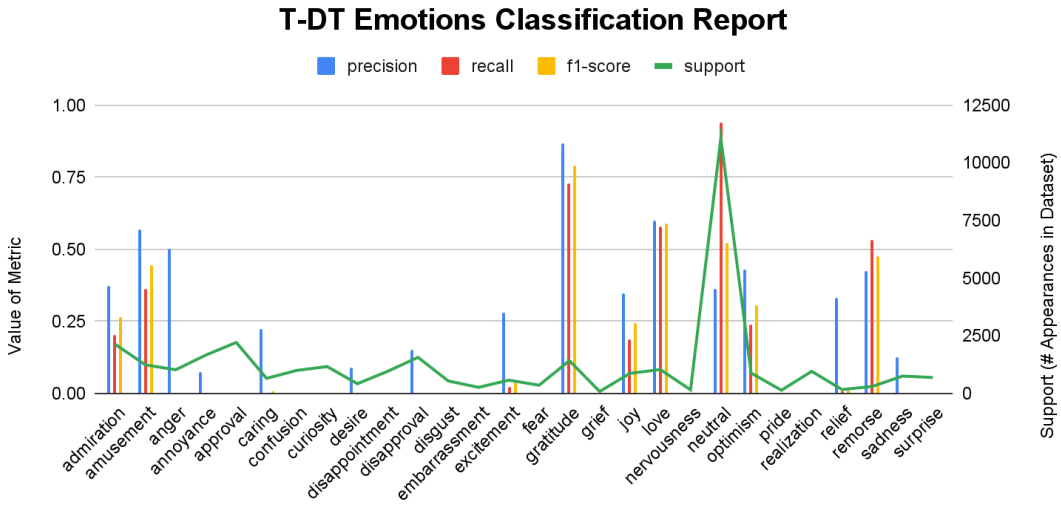

Figure 2.1.5.1: T-DT Emotions Classification Report

Having applied GridSearchCV on a Decision Tree model can be seen to have an undesirable effect on the classification report. The determined hyper-parameters for this model were: “criterion” = “entropy”, “max_depth” = 10 and “min_samples_split”=8. When comparing Figure 2.1.5.1 to Figure 2.1.2.1, it can be noticed that the recall and F1-score values for a great majority of the labels are very small or non-existent. The precision values seem to also have decreased. One interesting observation occurs when looking at the recall value for the “neutral” label. This value more than doubled in value, meaning there was a significant increase in true positives or a decrease in false negatives for that specific label. The confusion matrix *(Figure 2.1.5.3*) also shows that neutral dominated and “stole” classification from many of the other empty classes.

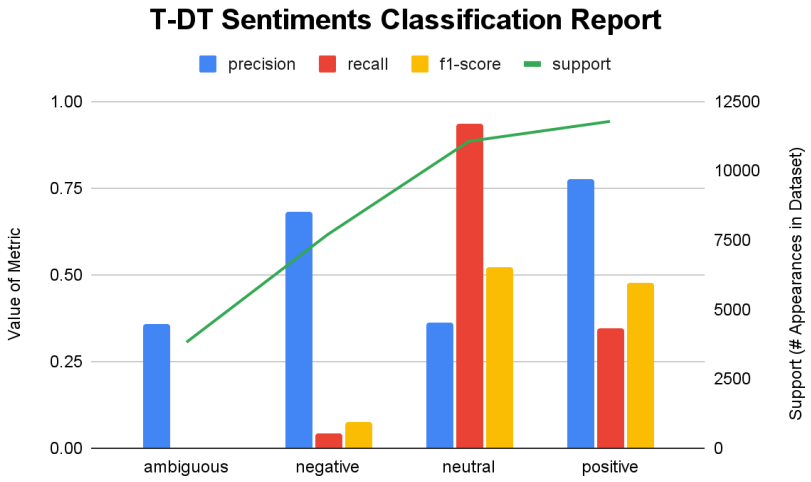

Figure 2.1.5.2: T-DT Sentiments Classification Report

For the sentiment model, the “min_samples_split” was determined to be 12. When comparing Figure 2.1.5.2 to 2.1.2.1, similar patterns to the emotions classification report can be observed. Both the recall and f1-score values of every label with the exception of “neutral” are comparatively low when using Grid Search CV on decision trees.

From both of these reports, it can be assumed that, when looking at F1-score’s value as a metric of model validity, using GridSearchCV to define the hyperparameters of a Decision Tree model leads to less successful models. The confusion matrix (*Figure 2.1.5.4)* is also extremely skewed towards neutral, with ambiguous and negative barely being classified.

2.1.6 Top-MLP

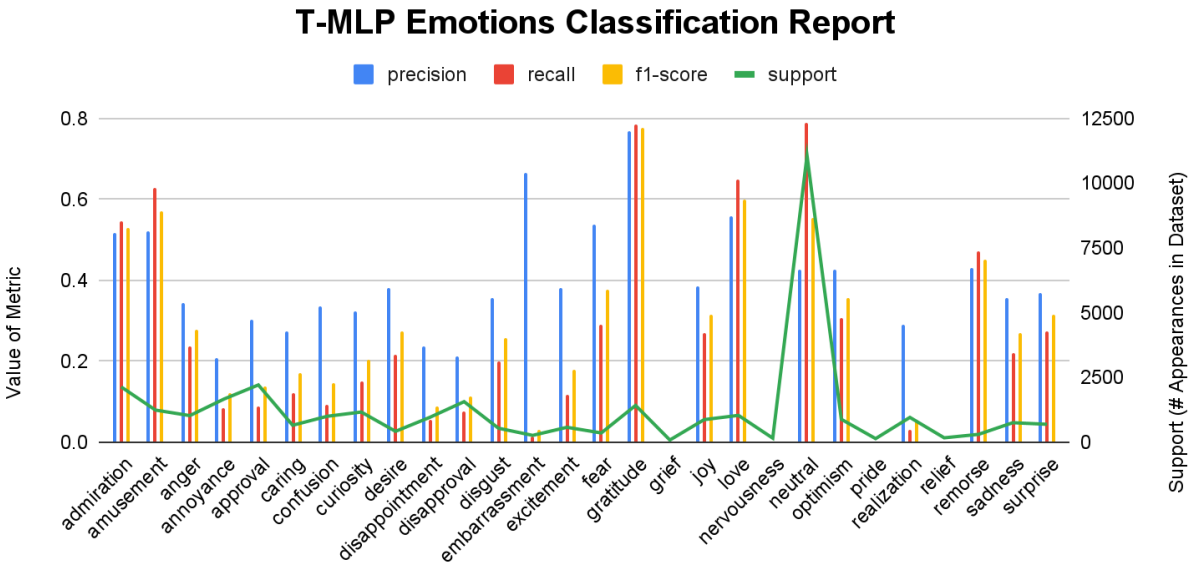

Figure 2.1.6.1: T-MLP Emotions Classification Report

The determined hyper-parameters for this model were: “activation” = “tanh”, “hidden_layer_sizes” = [30,50] and “solver”=”sdg”. As seen when comparing the above Figure 2.1.6.1 to Figure 2.1.3.1, applying GridSearchCV’s hyper-parameter optimization to a Multi-Layered Perceptron model seems to decrease the recall and F1-score values while increasing precision values on almost all labels in the model. Some labels like “grief”, “nervousness” and “pride” seem to have a complete lack of support. The decrease in F1-score could well indicate the the hyper-parameters for this model weren’t optimal.

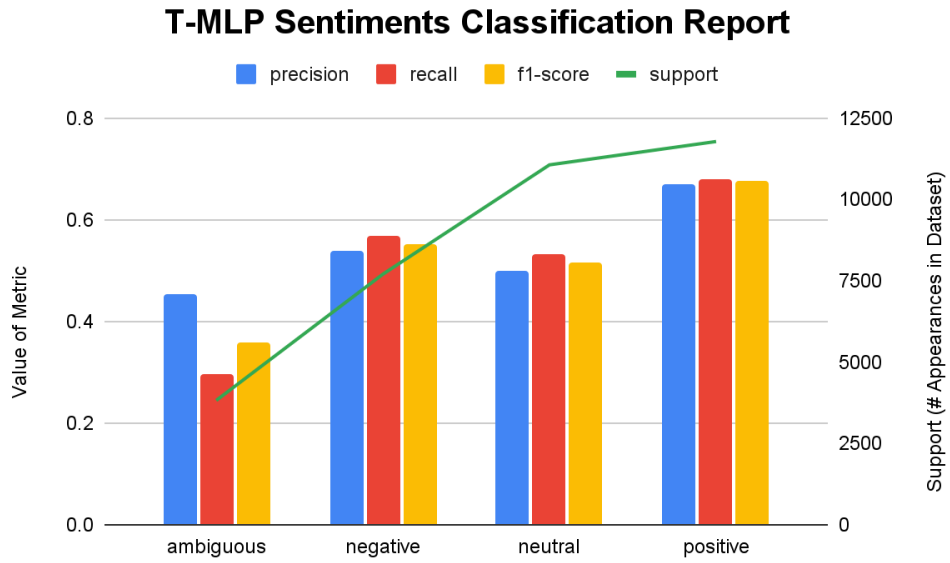

Figure 2.1.6.2: T-MLP Sentiments Classification Report

Regarding the sentiment classification reports seen in both Figures 2.1.6.2 and 2.1.3.2, it can be said that no significant change can be noticed. Apart from the "ambiguous" label's drop in recall and F1-score values, everything else, including the support shape seems to have remained almost identical.

From these comparisons, it can be established that using GridSearchCV's optimized hyper-parameters does not seem to render the model much more successful than it already was without using such techniques.

2.2 Using tf-idf Transformer

Term-frequency times inverse document-frequency (or tf-idf) is a common term weighting scheme used in the retrieval of information and document classification. This scheme smoothens the impact of token words that appear in higher frequencies (deemed less informative) in order to favor the impact of those that appear less often, which in many cases could be more informative than the latter. We hypothesized that using tf-idf would improve our results due to the relative weights of high frequency tokens being reduced and the increased weight of more specialized tokens. In the end, our best model for emotions was Base-MLP with tf-idf so the tf-idf weighting scheme seems to have worked quite well with the imbalanced emotions labels.

2.2.1 Base-MNB

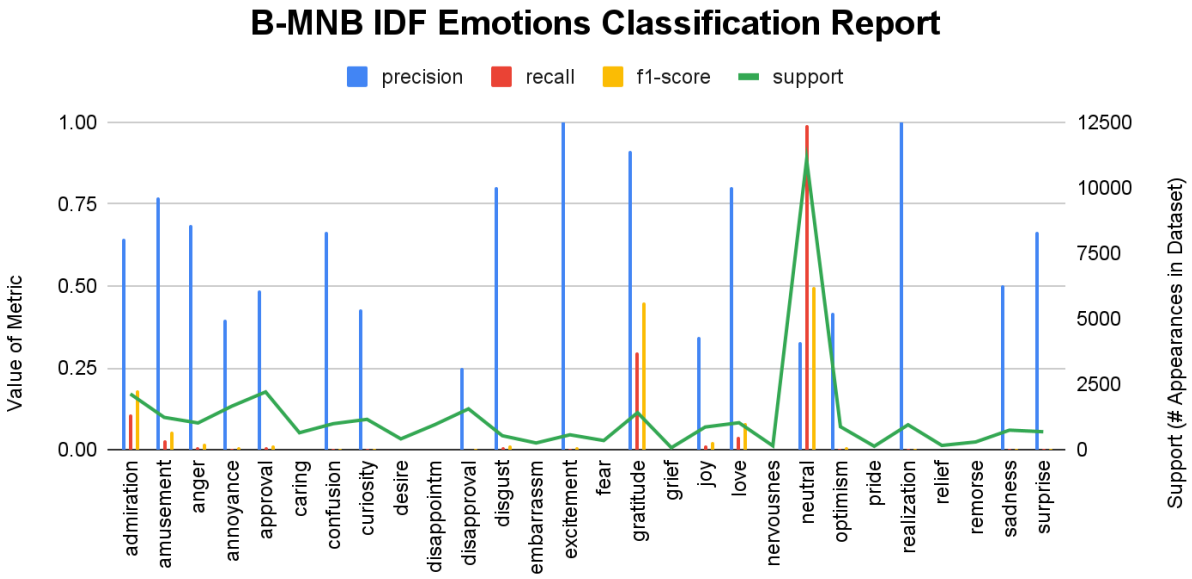

Figure 2.2.1.1: B-MNB Emotions Classification Report

Comparing the above figure to the previous ones for the B-MNB models, it can be noticed that the recall and consequently the F1-scores of every label are remarkably low and, in most cases, non-existent. The precision on the other hand, has increased drastically for almost every label, which can only mean that there are a lot less false positives being classified for each label. Interesting observations to mention would be how the "realization" and "excitement" labels seem to have 1.00 precision values, meaning there are close to no false positives for those labels. On the other hand, the "neutral" label seems to have close to 1.00 recall value which would mean there are almost no false negatives for the label.

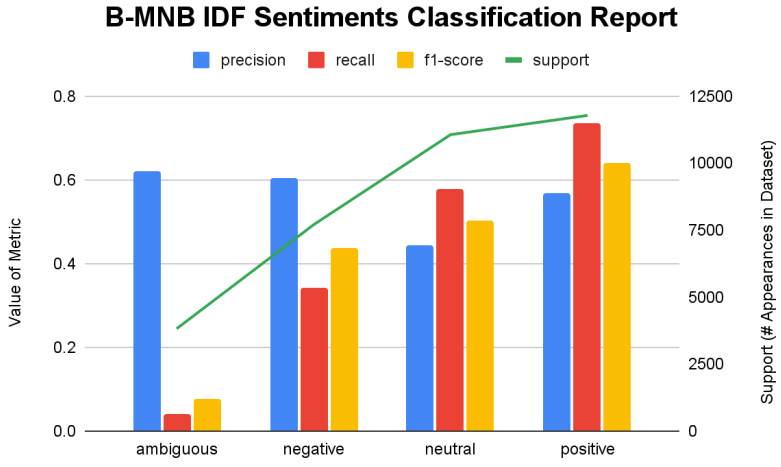

Figure 2.2.1.2: B-MNB IDF Sentiments Classification Report

When comparing the sentiments report for B-MNB's Figures 2.2.1.2 and 2.1.1.2, it can be noticed that the labels's metrics remain somewhat similar with the exception of the "ambiguous" label has a much higher precision value while also having a much lower recall value which in turns lowers the F1-score. From this observation, it can be said that there are much less false positives, but a lot more false negatives for that label.

2.2.2 Base-DT

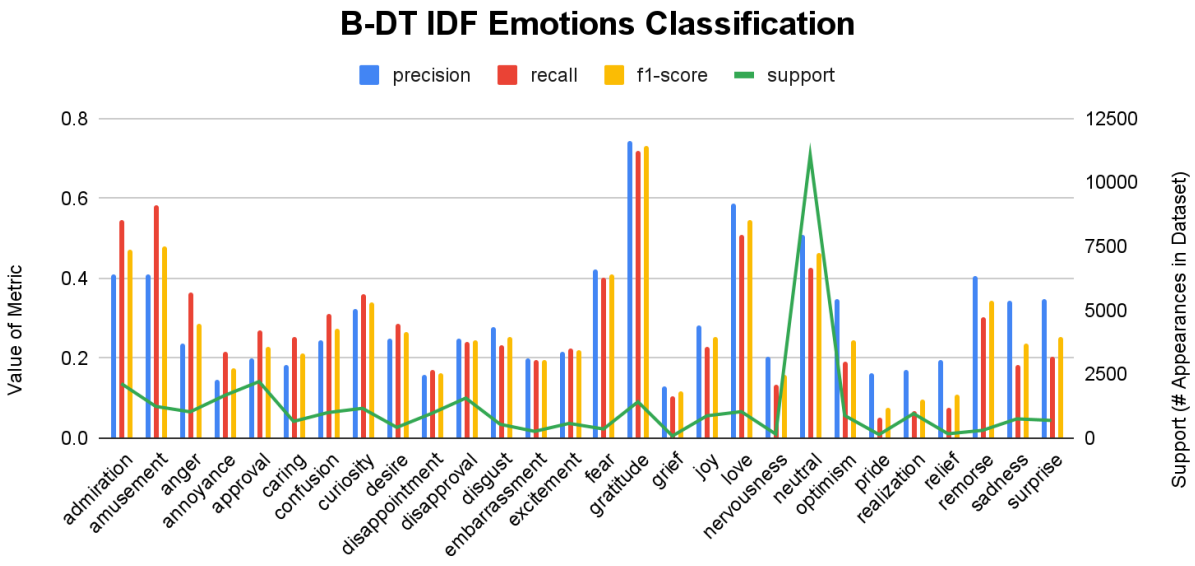

Figure 2.2.2.1: B-DT IDF Emotions Classification

For the decision tree model, it seems like using tf-idf doesn't seem to have changed any metric in particular as both Figures 2.1.2.1 and 2.2.2.1 look practically identical. When comparing the performance files, it can be seen that there is a slight difference in metric values for each label, but they are statistically insignificant.

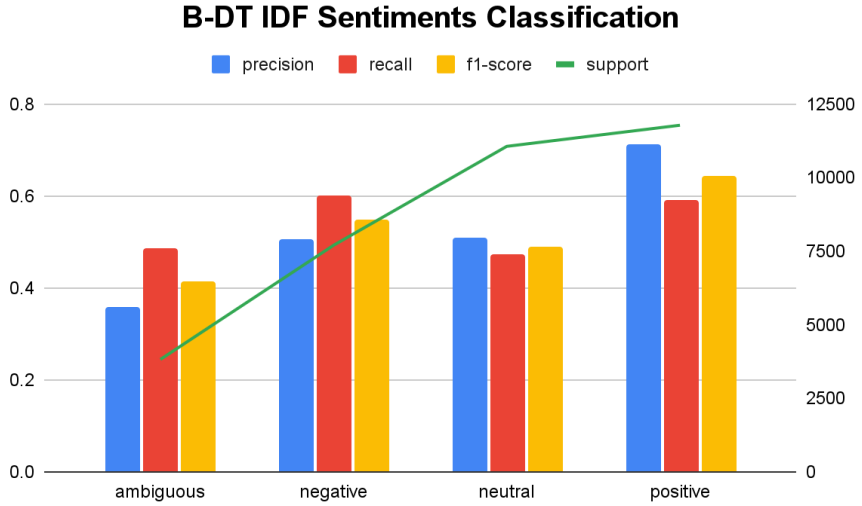

Figure 2.2.2.2: B-DT IDF Sentiment Classification

For the sentiment model, the same can be said, where Figure 2.2.2.2 seems to be practically identical to Figure 2.1.2.2, where the actual difference is not big enough to be noticeable in the classification report.

When looking at the Confusion matrices for both the emotion and sentiment models, it can be noticed that the patterns are also the same, while the values are slightly different, further supporting the previously mentioned claims.

2.2.3 Base-MLP

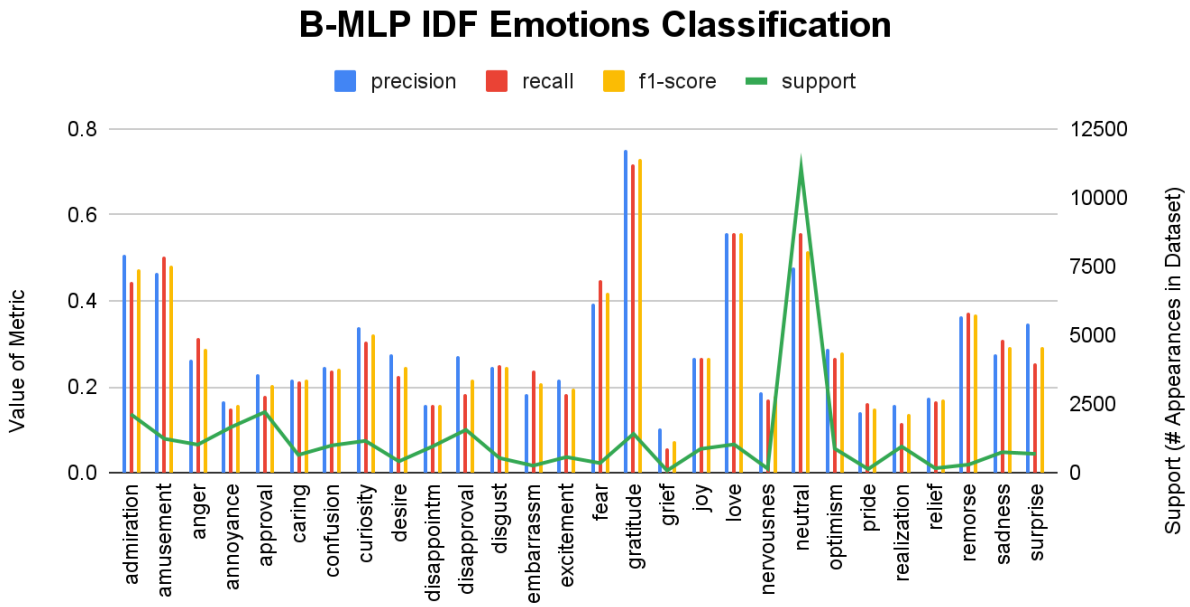

Figure 2.2.3.1: B-MLP IDF Emotions Classification

Regarding the Base-MLP model using tf-idf, it can be noticed that when comparing Figure 2.2.3.1 to Figure 2.1.3.1 almost every metric for every label seems to have remained roughly the same, with a few being slightly higher and other slightly lower than in previous models. From this it can be concluded that tf-idf doesn't affect a Base-MLP model significantly when using the reddit dataset and default MLP parameters.

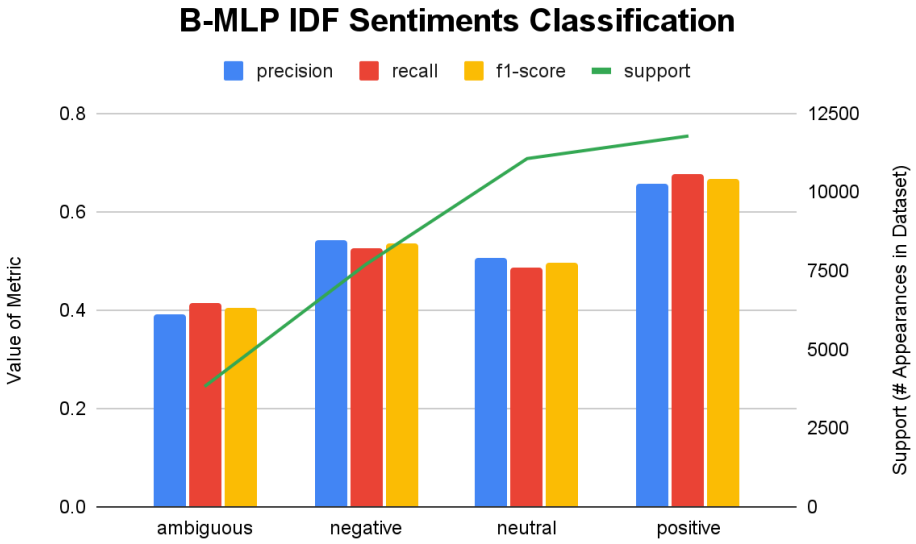

Figure 2.2.3.2: B-MLP IDF Sentiments Classification

The same comments stated above can be said regarding the sentiments model. The metric values seem to have remained roughly the same when comparing Figure 2.2.3.2 and Figure 2.1.3.2.

From the collected results, it can be said that for Base Multi-Layered Perceptron models using tf-idf does not seem to change any of the analysis metrics when compared to normal models. This means that these models are neither more or less successful than normal ones.

2.2.4 Top-MNB

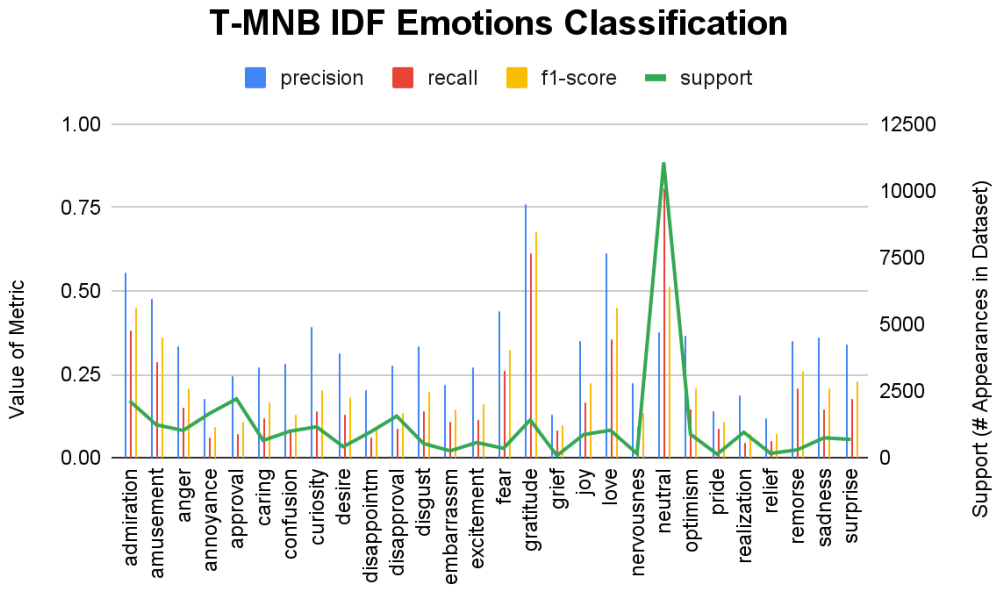

Figure 2.2.4.1: T-MNB IDF Emotions Classification

From the Figure 2.2.4.1 above, it can be noticed that while precision values seem to increase for almost every label, their recall and F1-score values seem to be lower. This could mean that there is a greater increase in false negatives than decrease in false positives being classified.

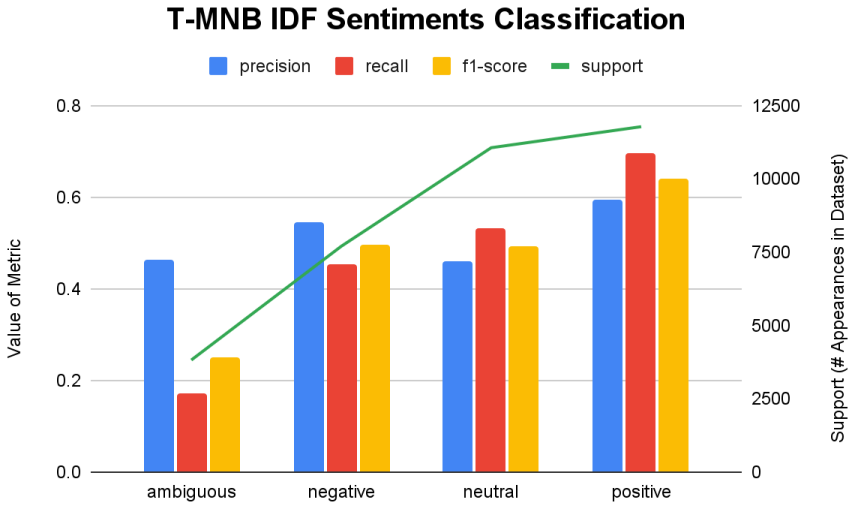

Figure 2.2.4.2: T-MNB IDF Sentiments Classification

Comparing Figure 2.2.4.2 to Figure 2.1.4.2, it can be seen that the recall value for the "ambiguous" label has decreased significantly and as mentioned for the emotions model, the precision values increased while the F1-score decreased slightly.

Regarding the confusion matrix, it can be said that even though the values vary slightly, both the emotion and sentiment matrices kept the same general patterns as the non tf-idf models' matrices.

From these observations it can be said that when "alpha" is set to be 0, the tf-idf model is in general less successful than the normal T-MNB model.

2.2.5 Top-DT

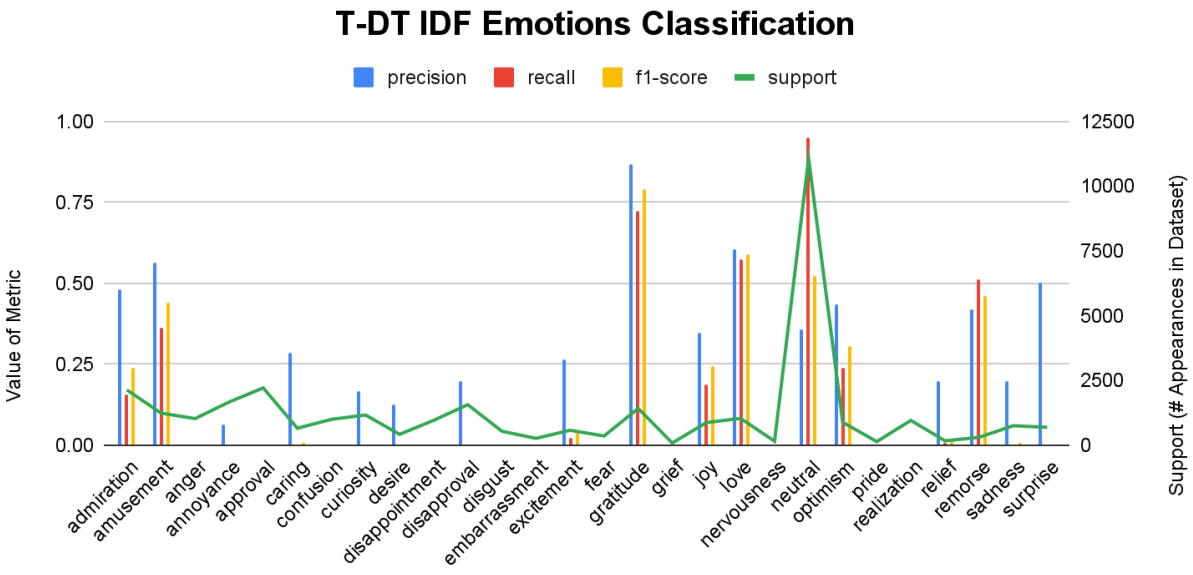

Figure 2.2.5.1: T-DT IDF Emotions Classification

For this model, the hyper-parameters were determined as: "criterion"="entropy", "max_depth"=10 and "min_sample_size"=9. For Figure 2.2.5.1, it can be noticed that it is very similar to Figure 2.1.5.1, with the exception that a few labels like "caring", "curiosity", "disapproval" and "surprise" have increased precision values, while "anger" seems to have zero precision.

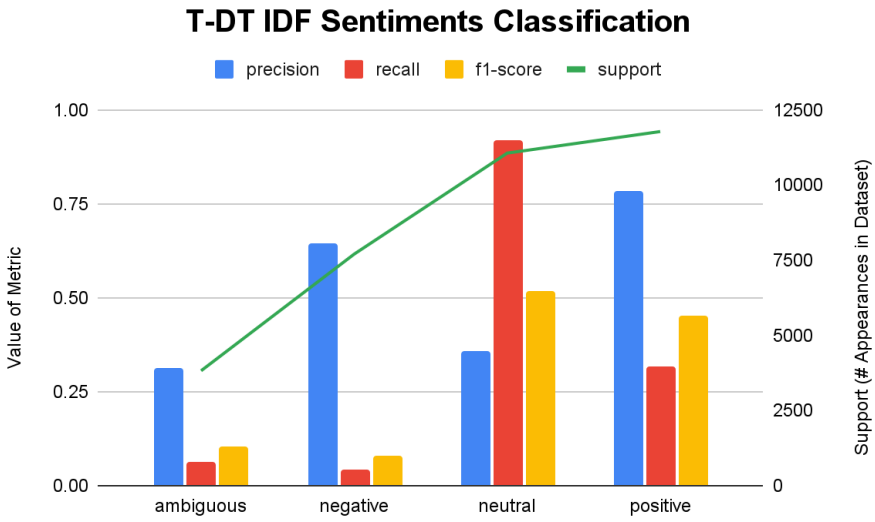

Figure 2.2.5.2: T-DT IDF Sentiments Classification

For the sentiments model, the classification report seems to have the same general results as the one seen in Figure 2.1.5.2. It can be noticed that the “ambiguous” label’s recall value is not zero, therefore generating a non-zero F1-score as well.

Looking at the confusion matrix for the sentiment model seen in Figure 2.2.5.4 it can be seen that almost every word is being classified as “neutral”, which is an unnatural result. This could be due to the decision tree over classifying everything as “neutral” since it is the most common “label” in all other models and datasets.

2.2.6 Top-MLP

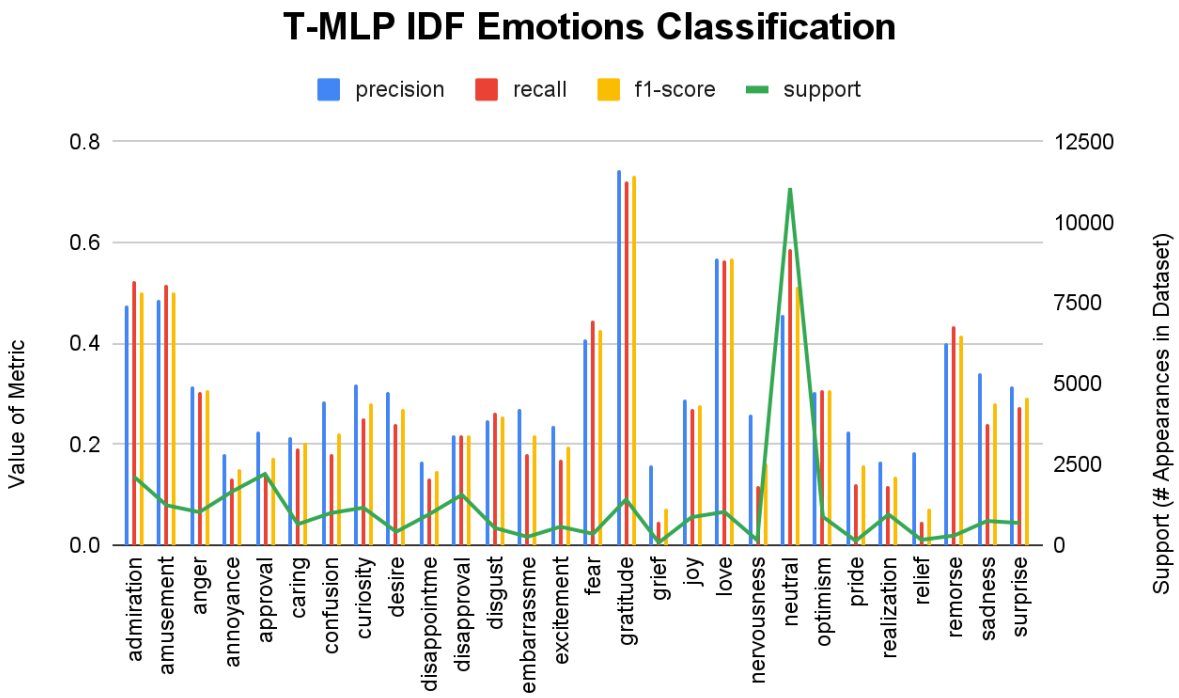

Figure 2.2.6.1: T-MLP IDF Emotions Classification

For this model, the hyper-parameters were determined to be: “activation”= 'logistic' and “hidden_layer_sizes”= [30, 50]. When comparing Figures 2.2.6.1 and 2.1.6.1, it can be noticed that for a majority of lesser labels the recall value and F1-score have increased to the point where they are equal or close to the precision value. This model is most likely the best demonstration of how tf-idf works, where the smaller labels were able to obtain higher F1-scores.

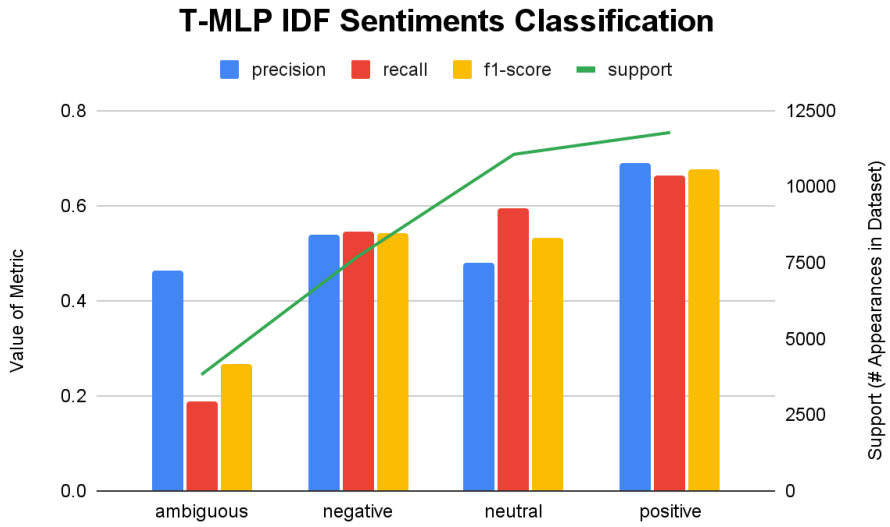

Figure 2.2.6.2: T-MLP IDF Sentiment Classification

For the sentiment model, the different hyper-parameters were: "activation"='tanh'. Looking at Figure 2.2.6.2 and 2.1.6.2, it can be noticed that there is not much difference for most labels' metrics, except for a drop in "ambiguous" 's recall and F1-score metrics.

From the observations and analysis seen above, it can be said that, for models with a higher quantity of labels, using tf-idf will result in a more successful model. On the other hand, not much change can be noticed if there are fewer labels.

2.3 Embeddings

In this section, the Multi-Layered Perceptron models were trained using Word2Vec embeddings and compared to pretrained embedding models such as "word2vec-google-news-300", "glove-twitter-200" and "glove-wiki-gigaword-300". The "twitter" model was chosen because of its popularity and as well as their similarity to the "reddit"-based trained model as both twitter and reddit are known as social media blog-posting forums. Similarly, the "wikipedia" model was chosen for its popularity, but also because of the presence of very diverse articles within the resource's database, which would bring up interesting points of comparison. The hit rate for the models were as follows: Google: 77.46% train, 77.42% test. Twitter: 84.53% train, 84.57% test. Wikipedia: 85.34% train, 85.37% test. So while twitter and wikipedia did indeed have higher hit rates as we predicted, they performed worse. This is most probably due to the fact that the embeddings themselves were flawed for these models (at least for our dataset). We hypothesized that the embeddings model should perform better than word frequency and tf-idf, which was not the case. They did however perform fairly close to the other two methods and in much less time, which is a benefit in its own right.

2.3.1 Base-MLP

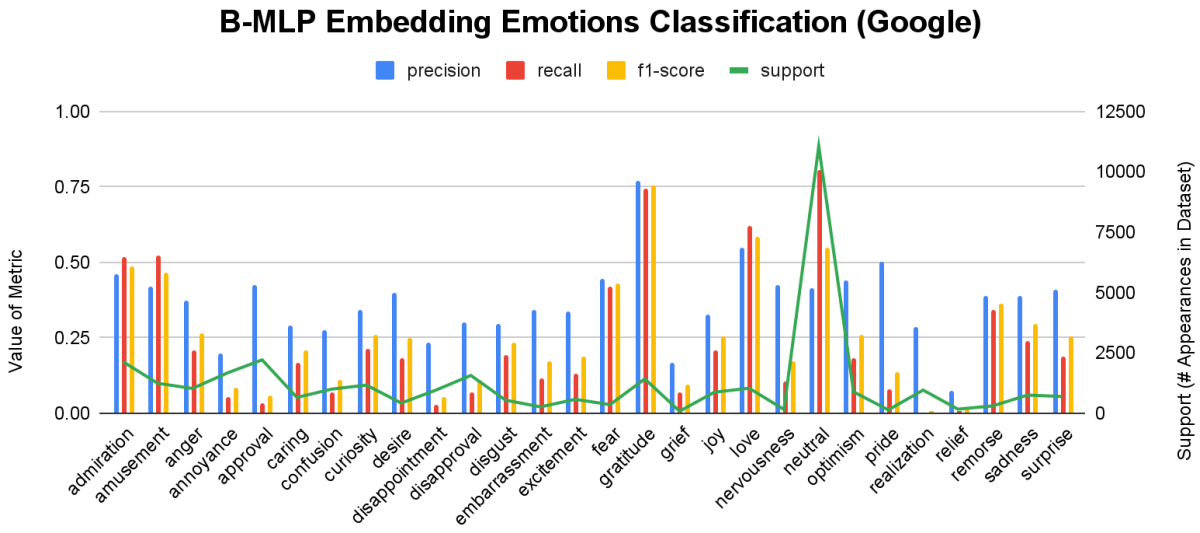

Figure 2.3.1.1: B-MLP Embedding Emotions Classification (Google)

Using the B-MLP model on the word2vec Google dataset as seen on Figure 2.3.1.1 resulted in a graph trend that is similar compared to the one shown in Figure 2.1.1.1. This time however, lower recall and higher precision values can be noticed for many of the lesser supported labels, proving to be a slightly less successful model when trying to use the google dataset.

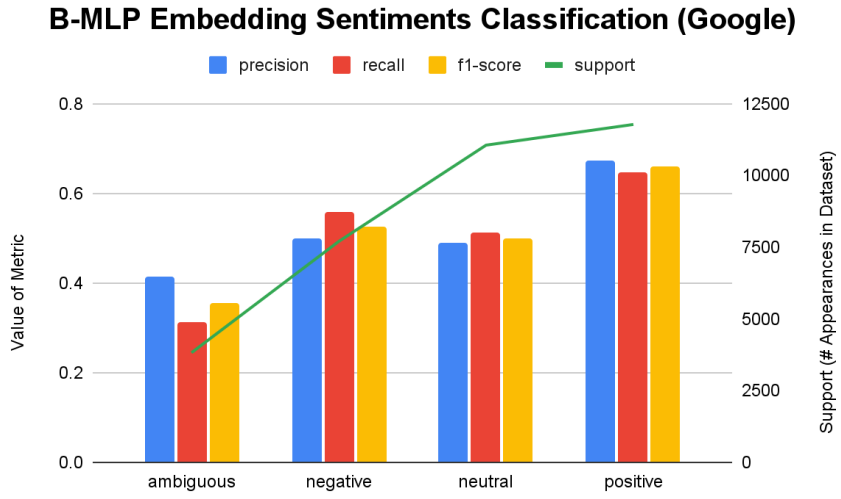

Figure 2.3.1.2: B-MLP Embedding Sentiments Classification (Google)

In terms of the sentiment model, it can again be said that the general shape of the graph shown in Figure 2.3.1.2 is similar to the one in Figure 2.1.1.2. The difference again being lesser recall values for the “ambiguous” label.

In terms of the confusion matrices, both emotion and sentiment models show similar patterns to the models in section 2.1.3, which support the results seen above. From such results,

it can be said that B-MLP's model used on the Google dataset does not perform as well as the original Reddit one, but is still considered to be generally successful.

2.3.2 Top-MLP

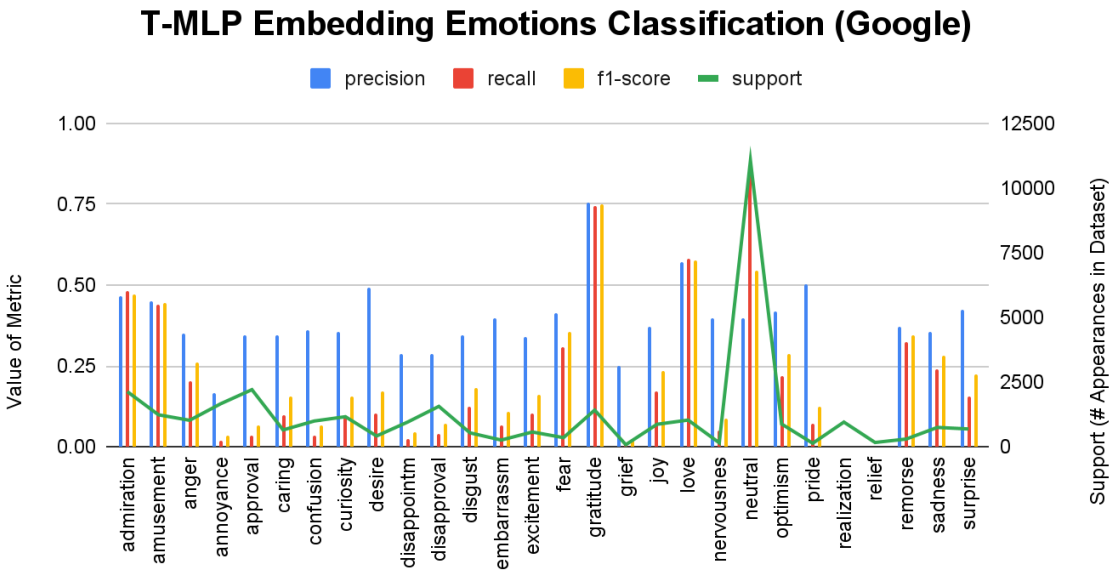

Figure 2.3.2.1: T-MLP Embedding Emotions Classification (Google)

Using the T-MLP model on the google embedded dataset proved to have higher overall values of precision, but lower recall and F1-score for almost every label. For many of the classes such as annoyance or approval, the recall values are very low. Far lower than in tf-idf and word frequencies. This could be due to the fact that the embeddings method of vectorizing posts does not respond as well to the possible values we tried with gridsearch. However, the B-MLP model for embeddings performed worse for recall as well, showing us that perhaps embeddings was not a great fit for our reddit dataset.

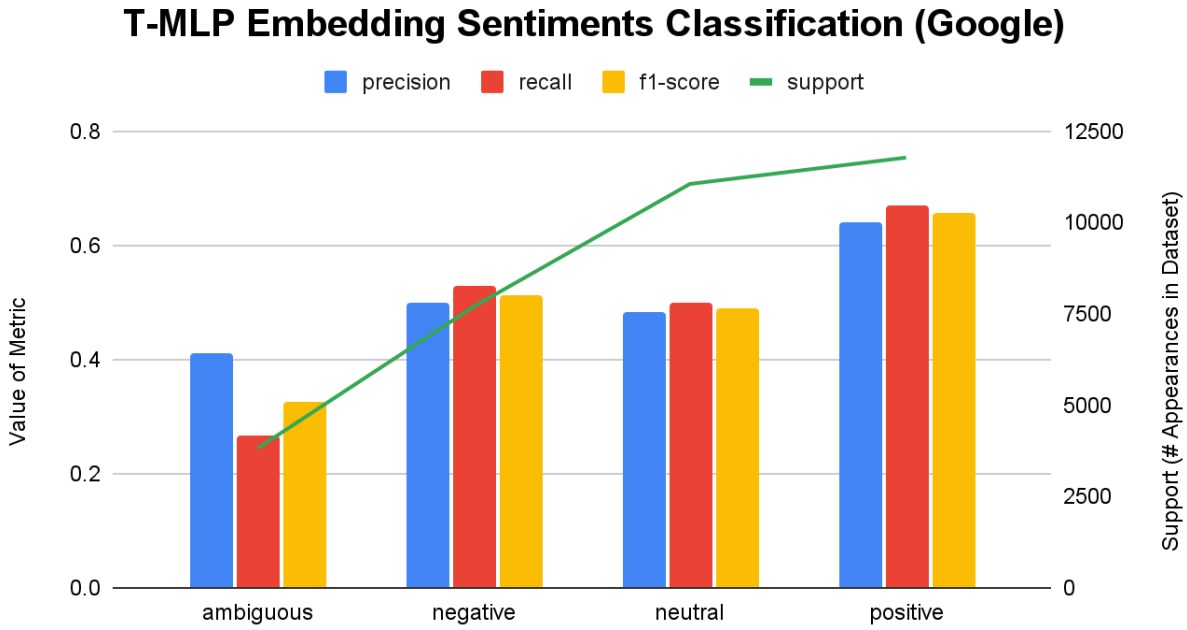

Figure 2.3.2.2: T-MLP Embedding Sentiments Classification (Google)

For the sentiment model, the graph shown in Figure 2.3.2.2 is very similar to the one in Figure 2.3.1.2, with the exception that the metric values for every label have decreased slightly. This shows that the embeddings are not ideal for our dataset.

2.3.3 Base-MLP on Twitter data set

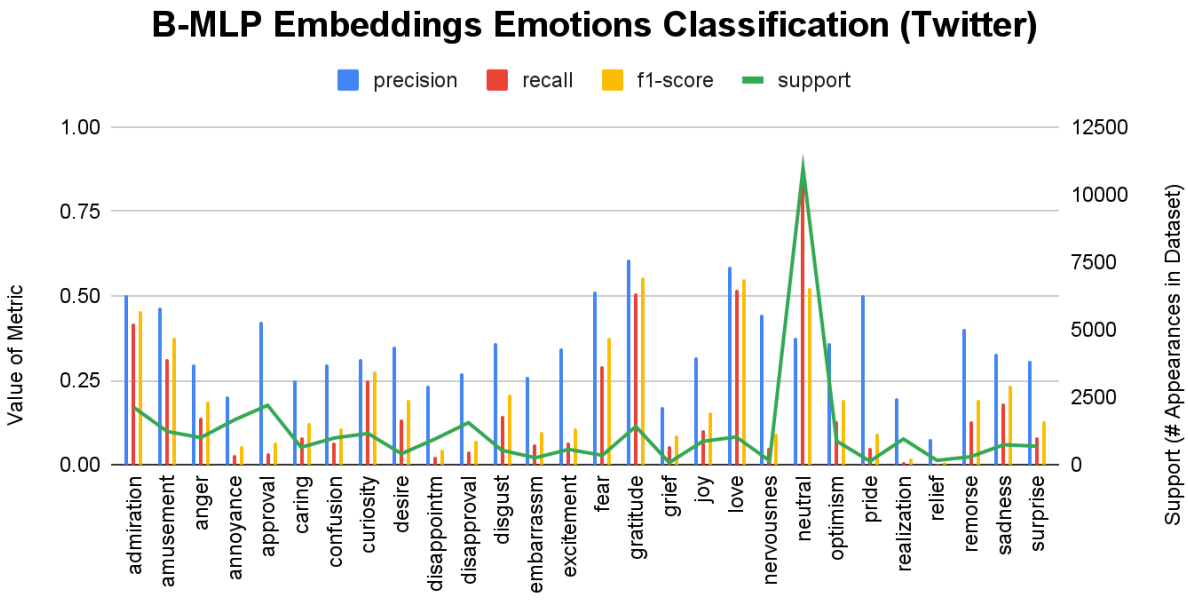

Figure 2.3.3.1: B-MLP Embeddings Emotions Classification (Twitter)

The Twitter model's embeddings clearly work less well for our dataset than the Google model. This is particularly apparent for *gratitude*, which generally has good results in our model so it stands out more. There is a significant visible reduction in the scores of gratitude for all 3 metrics as compared to *Figure 2.3.1.1*.

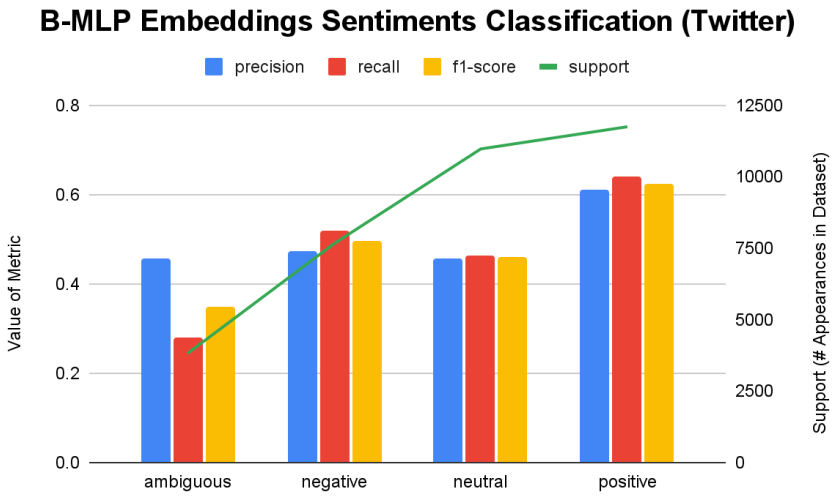

Figure 2.3.3.2: B-MLP Embeddings Sentiments Classification (Twitter)

The differences between the Twitter and Google datasets are less apparent here for Sentiments and perform similarly. There is a visible increase in precision for the ambiguous class, but generally a decrease in recall for negative and neutral. The performance here is better than Emotions, but still generally worse for the f1 score.

2.3.4 Base-MLP on Wiki data set

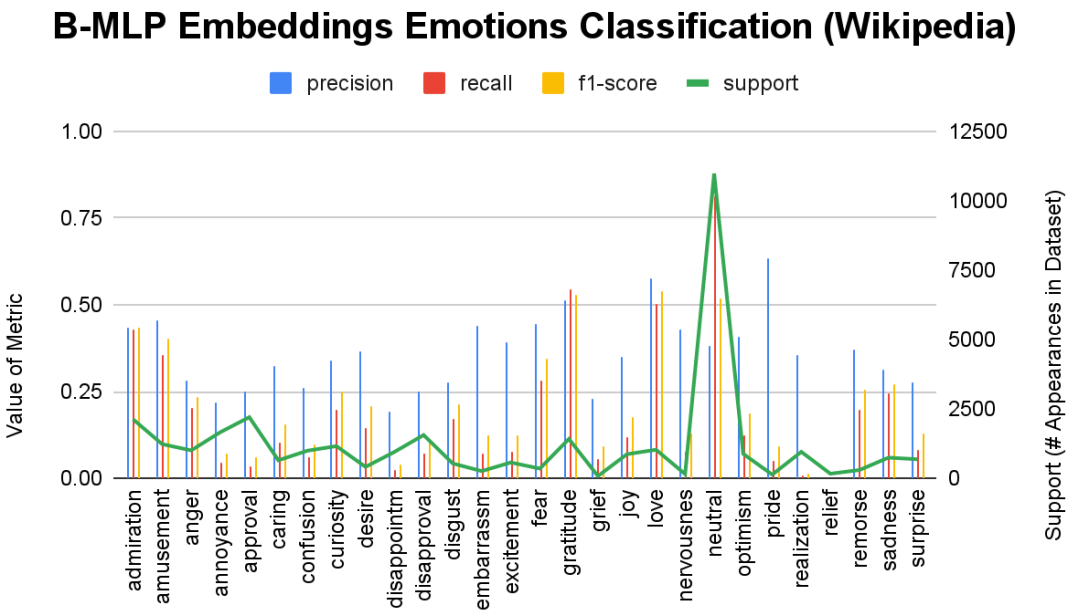

Figure 2.3.4.1: B-MLP Embeddings Emotions Classification (Wikipedia)

The Wikipedia model performs extremely closely to the Twitter model, to the point that we had to double check our code to ensure we did not make a mistake. The relief class here has a value of 0 for all 3 metrics which is worse than the Twitter model. Aside from this, recall seems better in this model than the Twitter model and closer to the Google model. The wider variety of Wikipedia articles perhaps helped recall improve, but it still was not enough to beat the Google model.

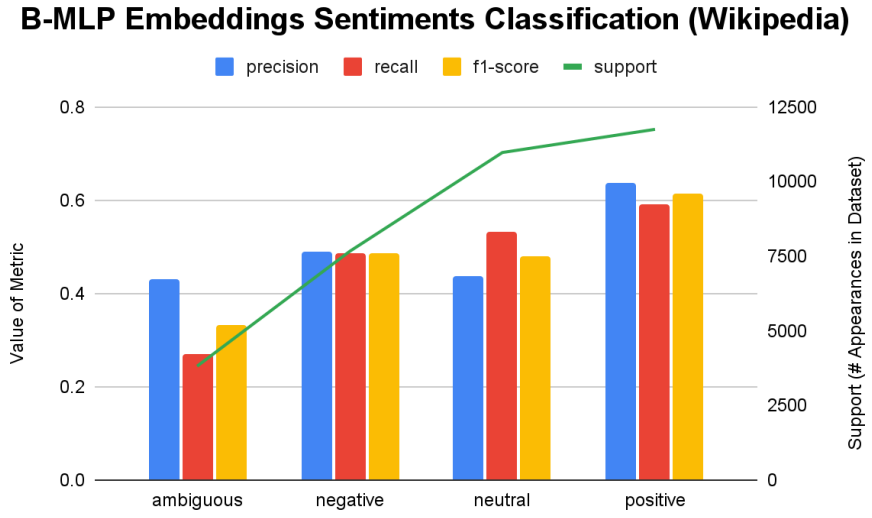

Figure 2.3.4.2: B-MLP Embeddings Sentiments Classification (Wikipedia)

The values here for Sentiments are similar to Twitter's with the exception of neutral performing better for recall and f1-score, and a slightly higher score for precision in the positive class, but the lower recall value leads to a near identical f1-score.

These comparisons help us conclude that the Twitter and Wikipedia models perform overall fairly close to each other. Their worse performance on our Reddit posts show that the types of vocabulary and slang used on Reddit are different from Wikipedia and Twitter. While these words appeared more in these models as is evident by the higher hit rates, likely the way in which they are used is different.

3. Member Contributions:

Work done on the MP1 was distributed fairly equal throughout its duration. For the first few weeks of the project Brandon set up the GitHub repo and the files present in the project, while Facundo did research on the tasks to be accomplished within each section. In regards to coding, most was done in tandem. When coding was performed individually, the other team member usually assisted with additional research and code corrections in a timely manner through discord and git commits. The analysis section of the report was also divided equally, with each member taking his fair share of the work and both discussing and reviewing all that was written.

4. Additional Figures:

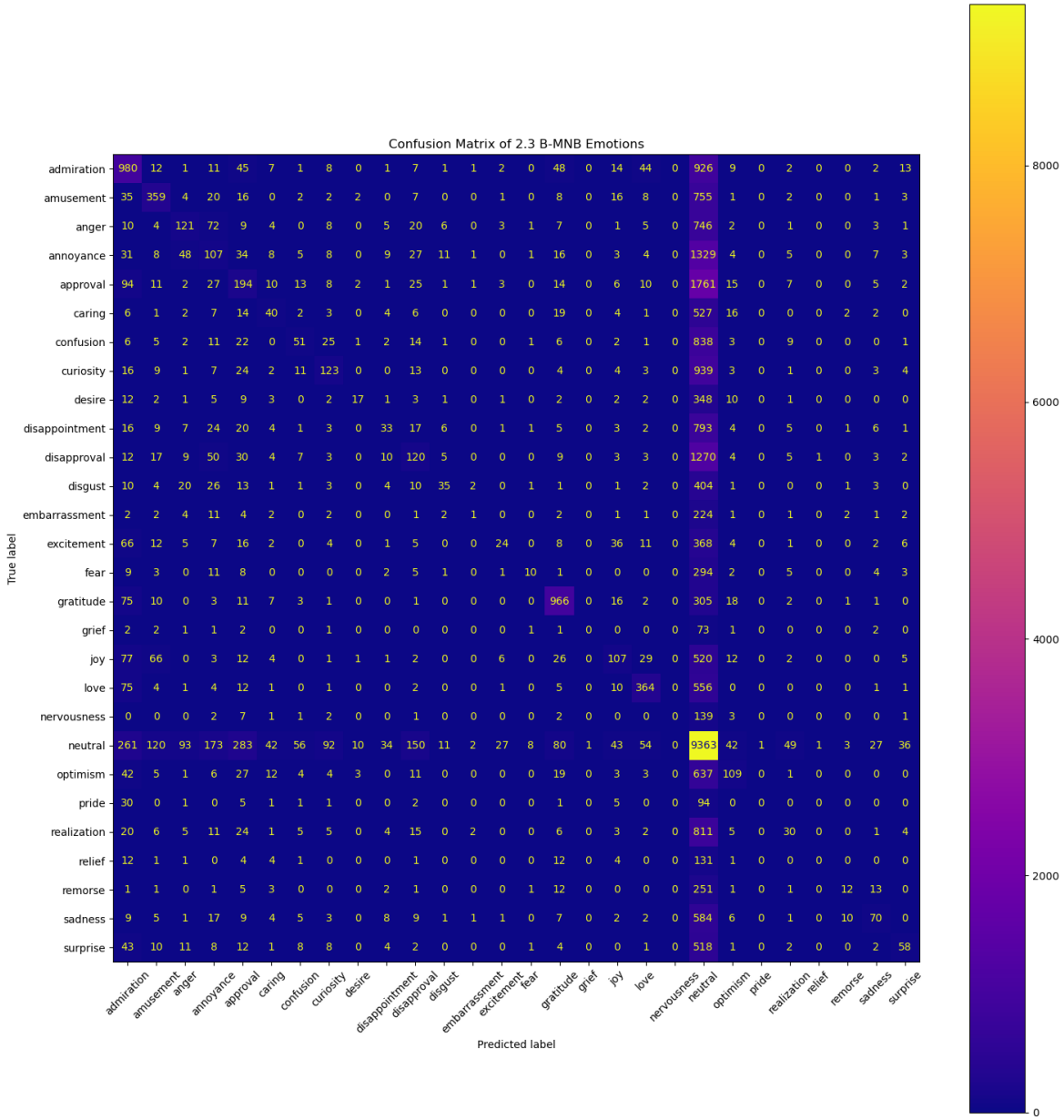

Figure 2.1.1.3: B-MNB Emotions Confusion Matrix

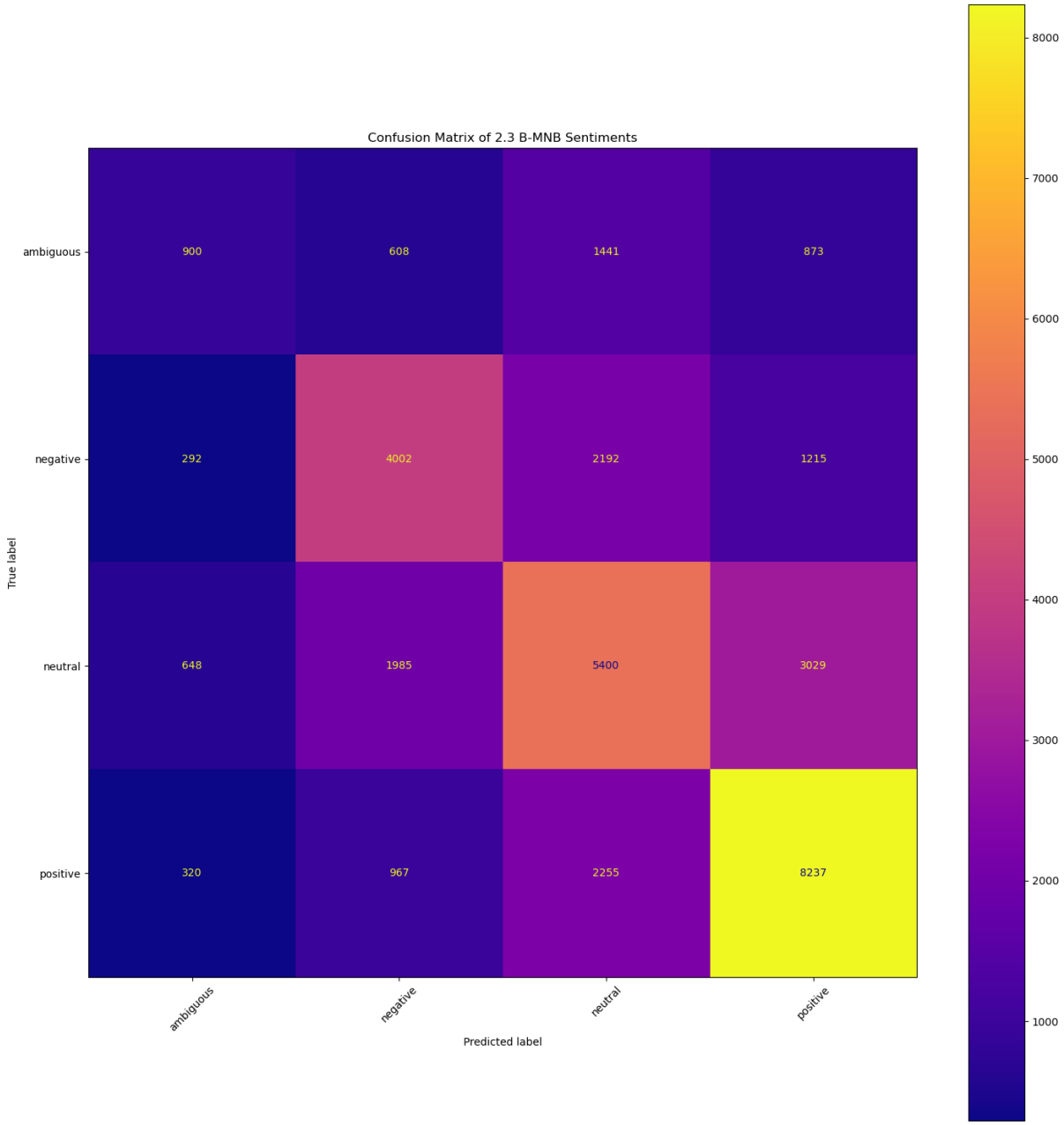

Figure 2.1.1.4: B-MNB Sentiments Confusion Matrix

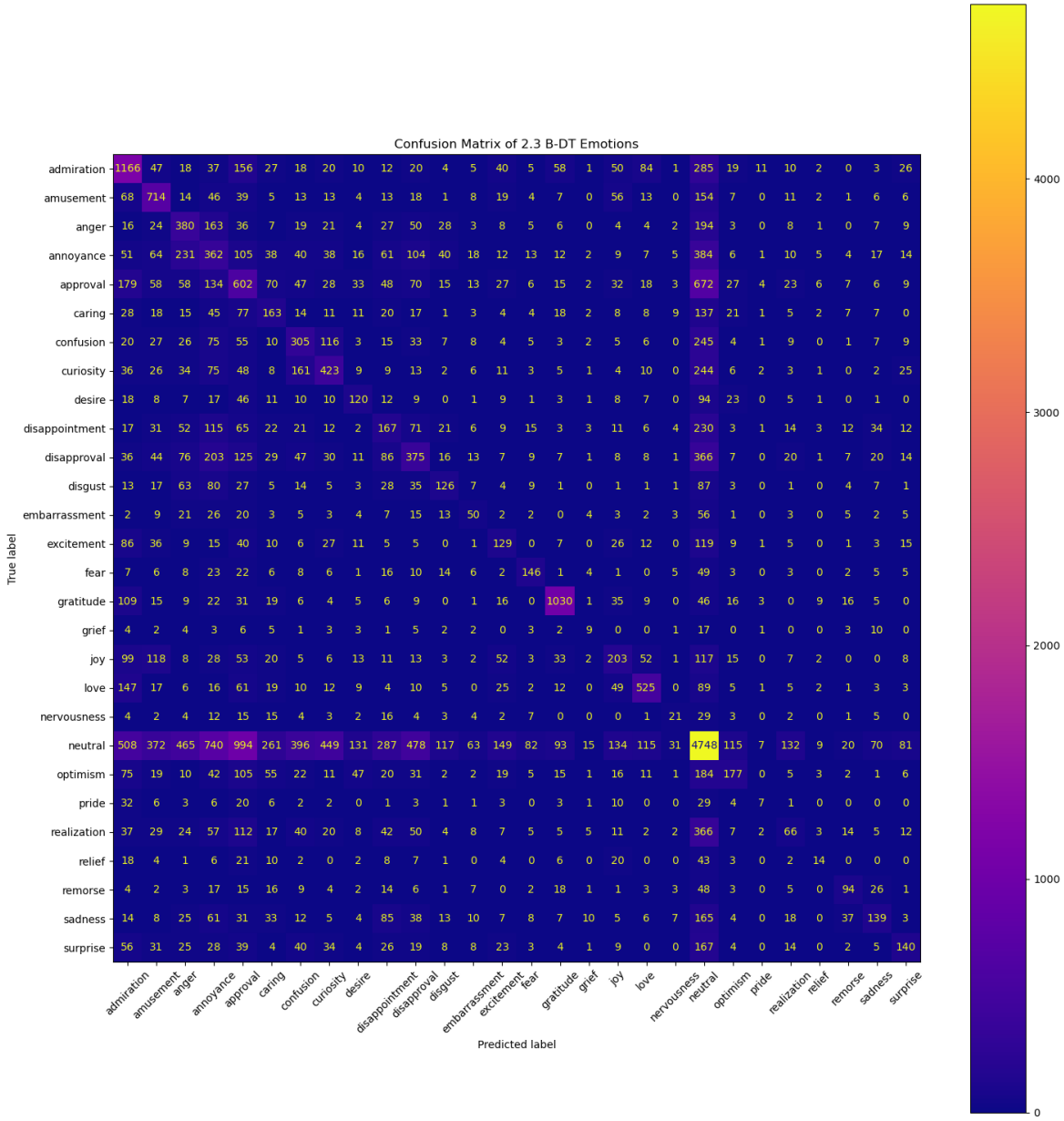

Figure 2.1.2.3: B-DT Emotions Confusion Matrix

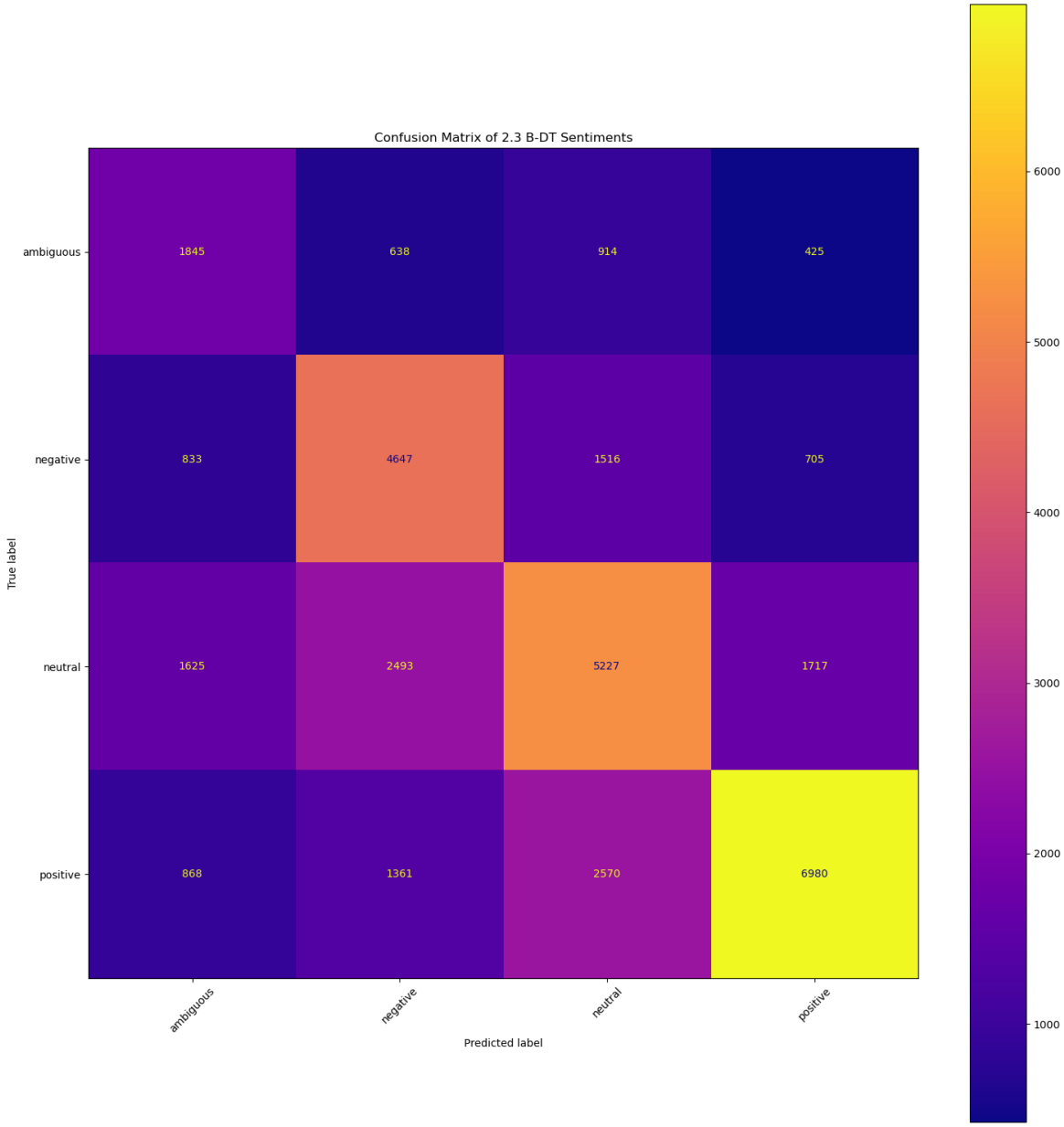

Figure 2.1.2.4: B-DT Sentiments Confusion Matrix

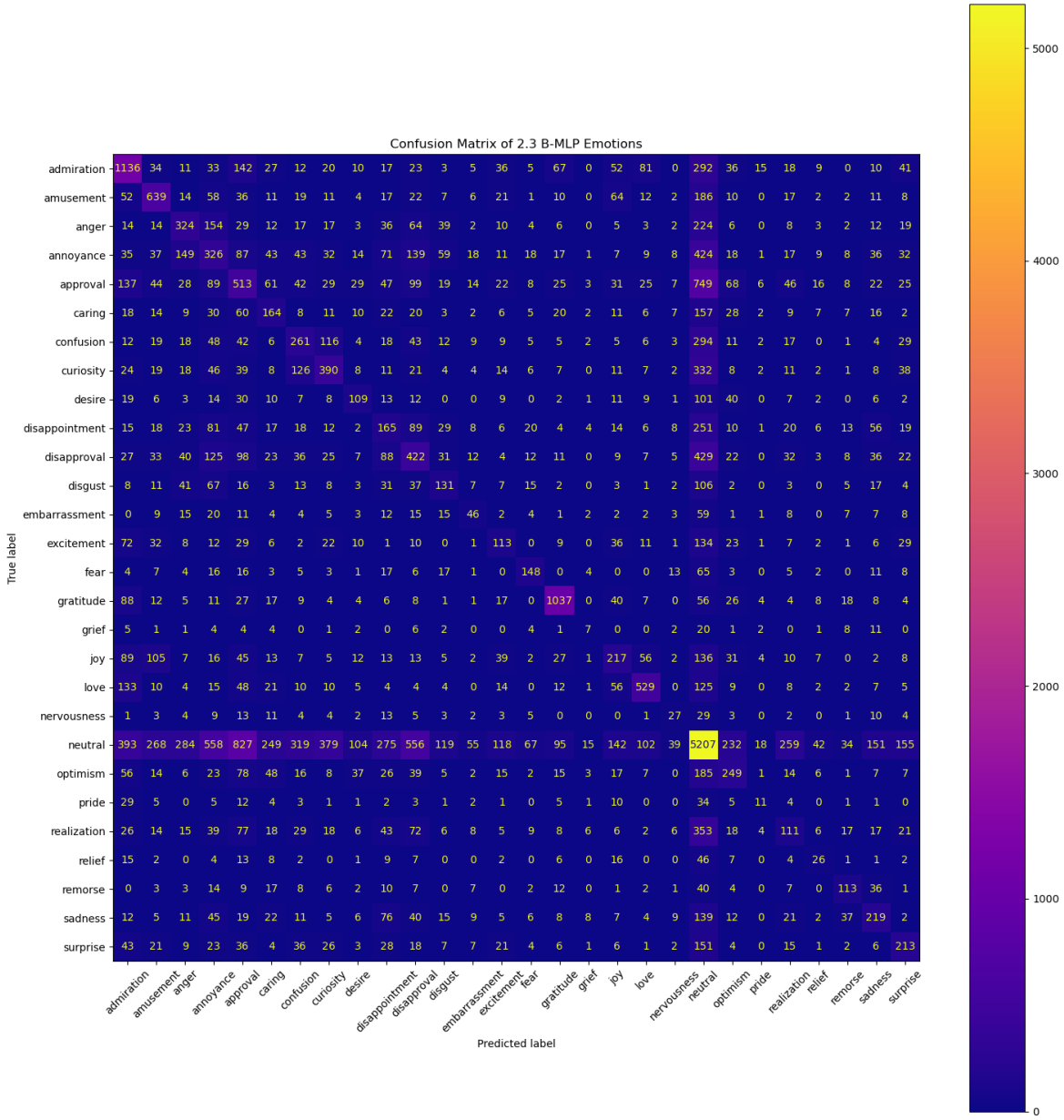

Figure 2.1.3.3: B-MLP Emotions Confusion Matrix

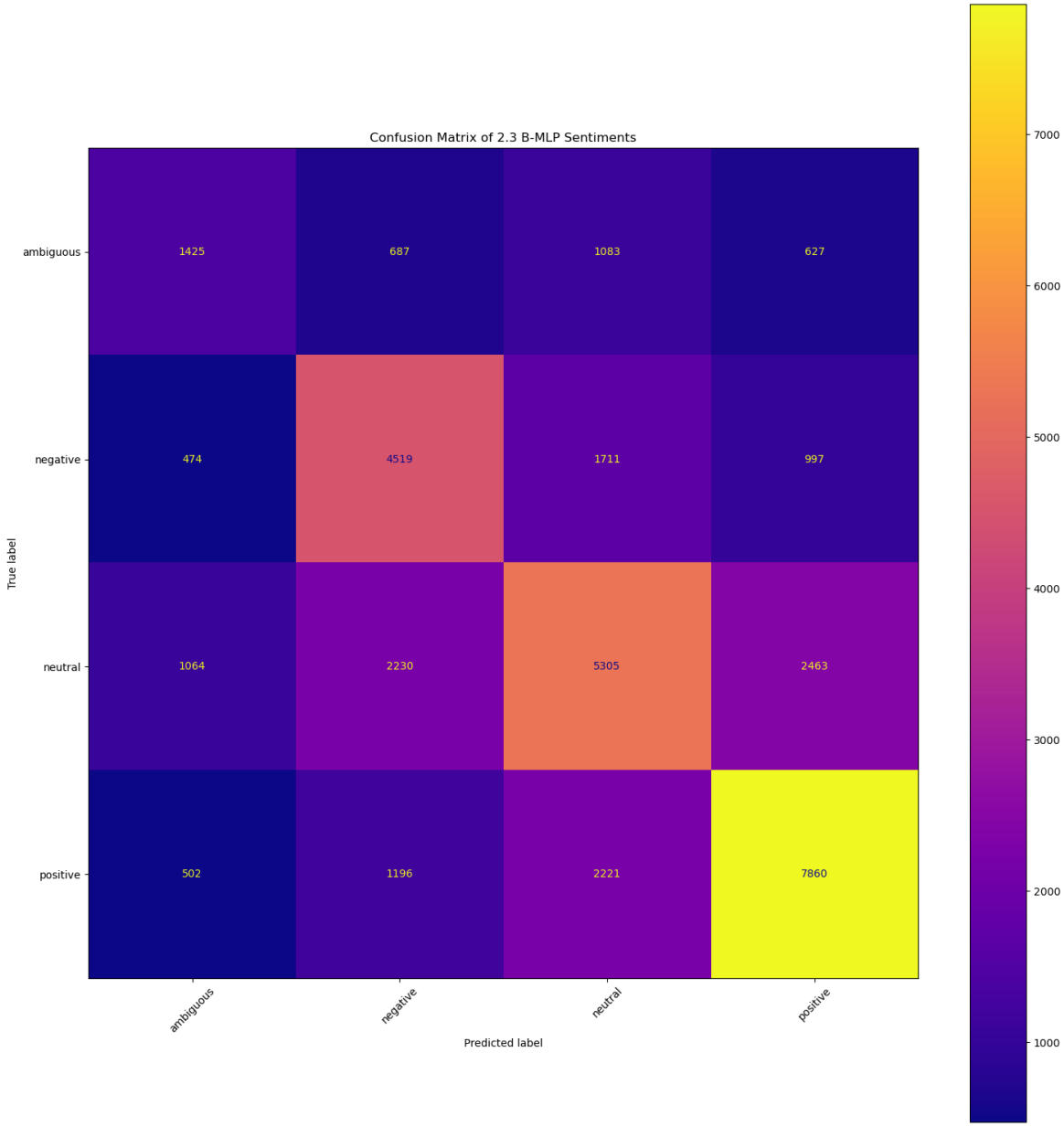

Figure 2.1.3.4: B-MLP Sentiments Confusion Matrix

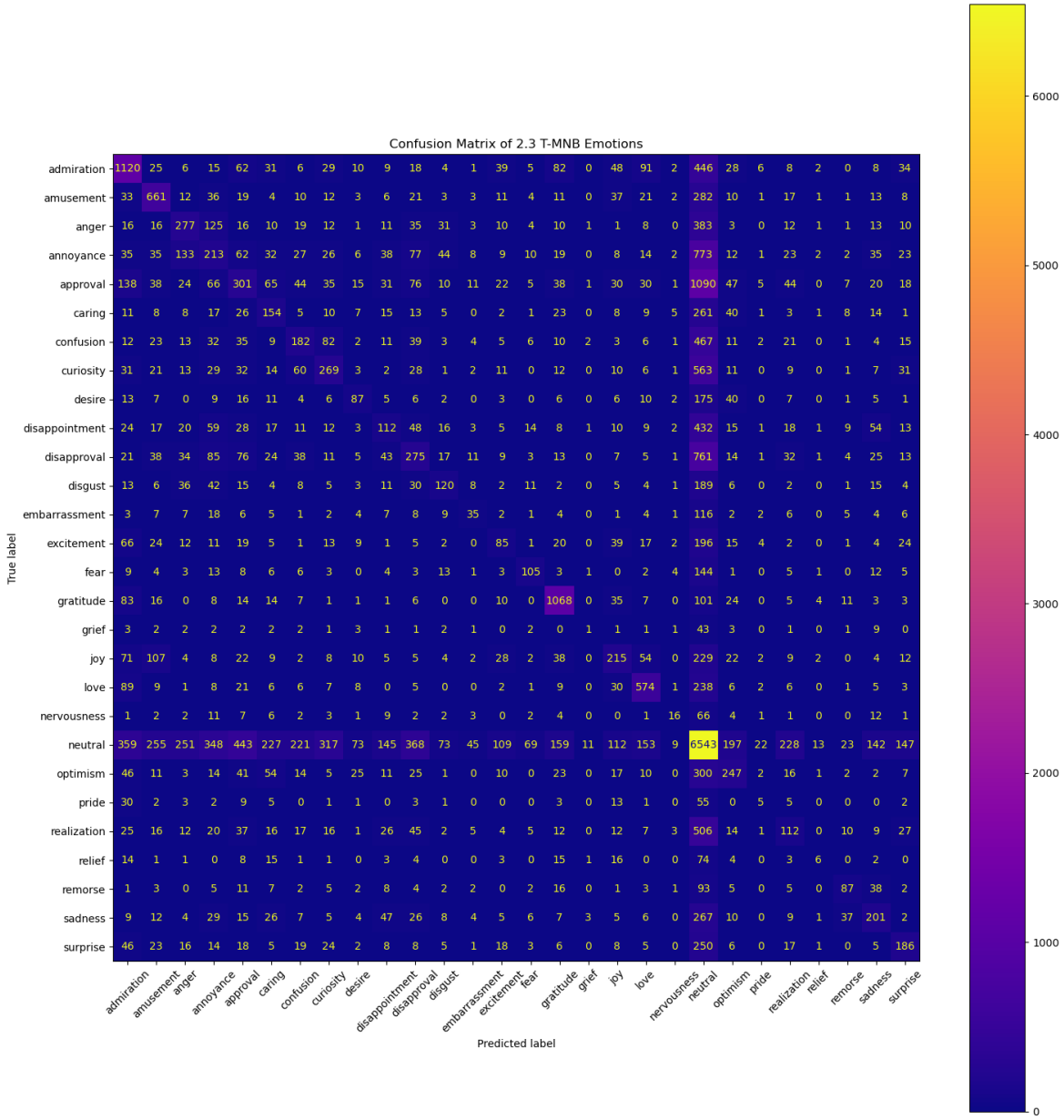

Figure 2.1.4.3: T-MNB Emotions Confusion Matrix

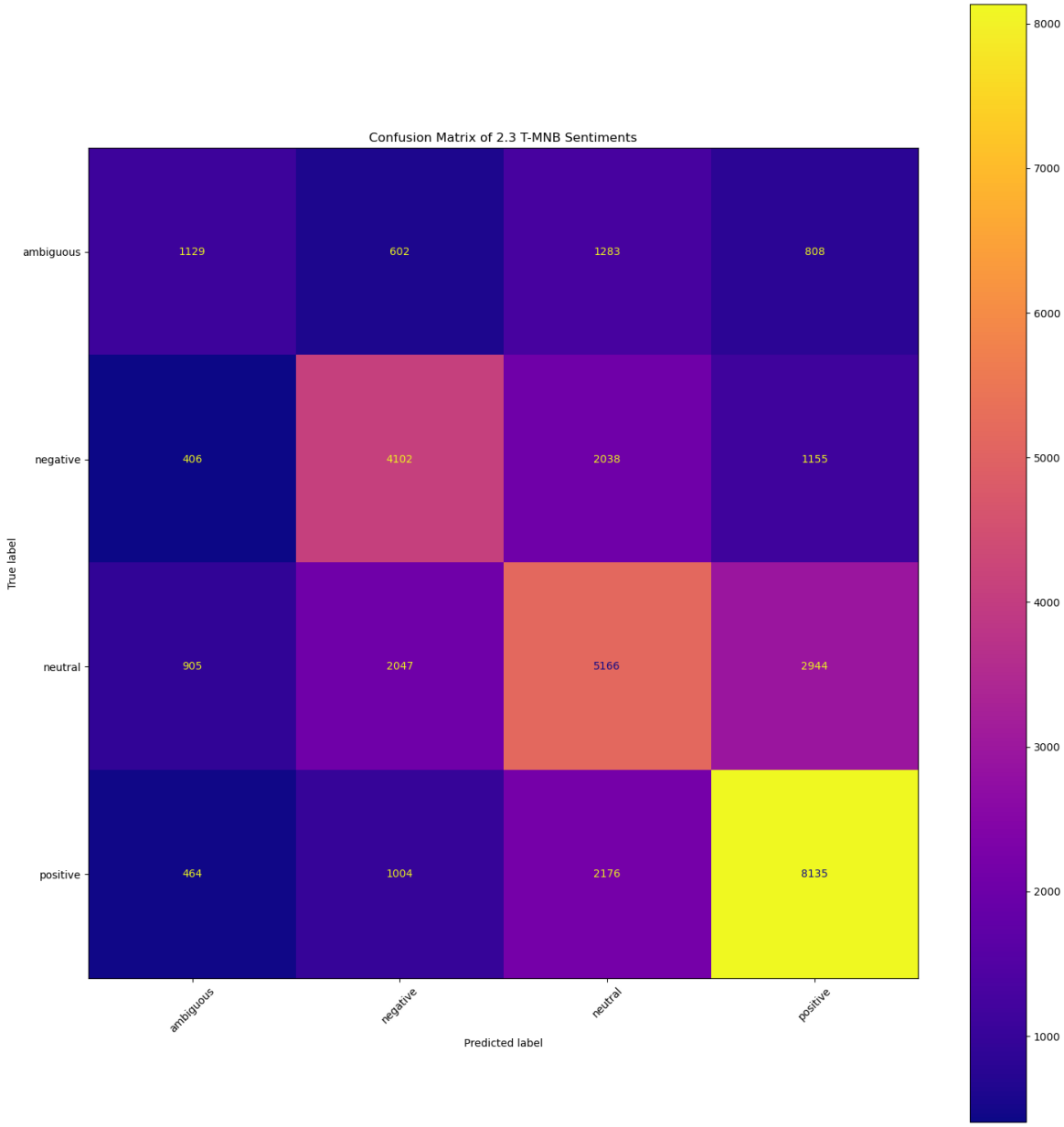

Figure 2.1.4.4: T-MNB Sentiments Confusion Matrix

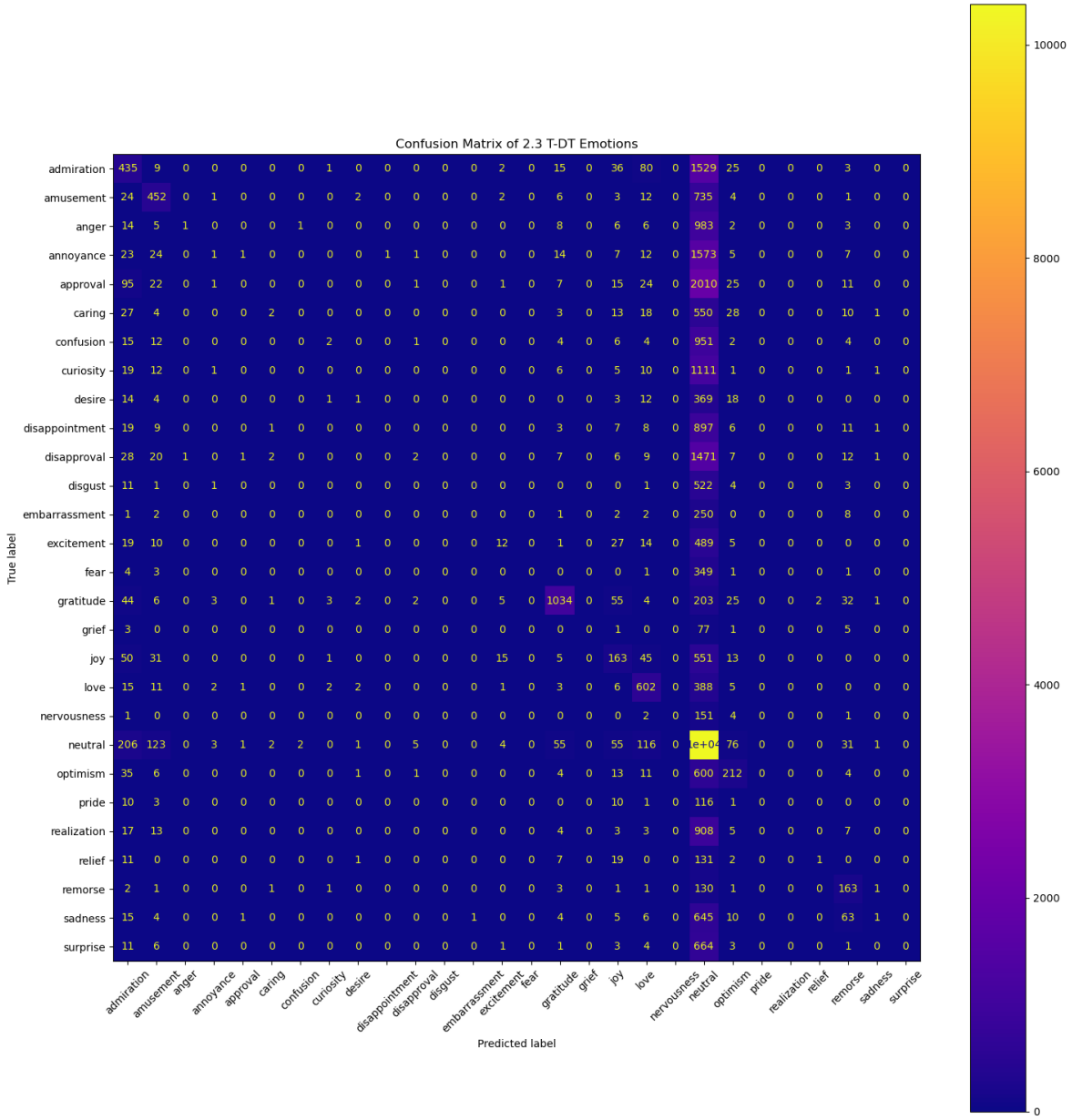

Figure 2.1.5.3: T-DT Emotions Confusion Matrix

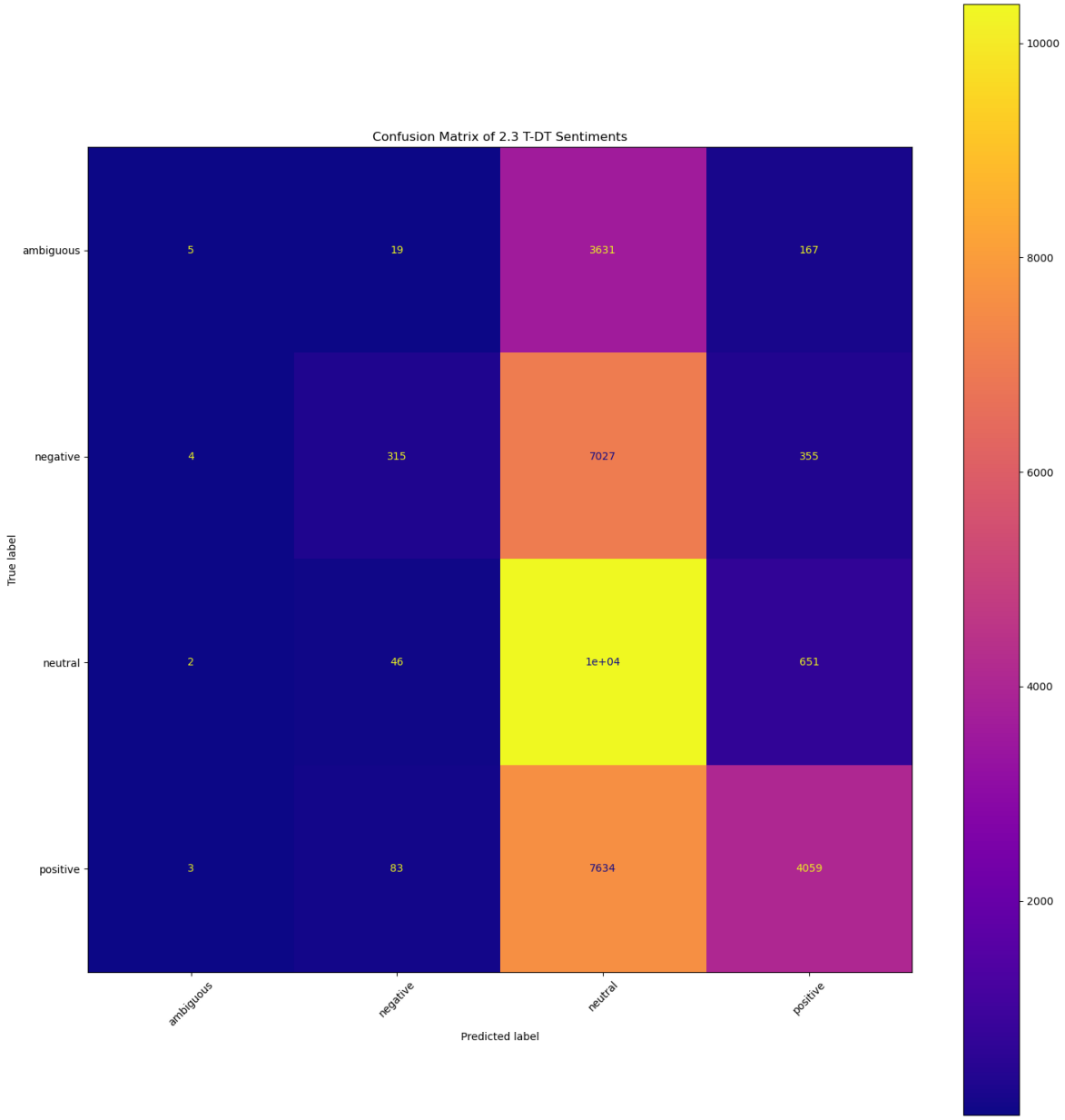

Figure 2.1.5.4: T-DT Sentiments Confusion Matrix

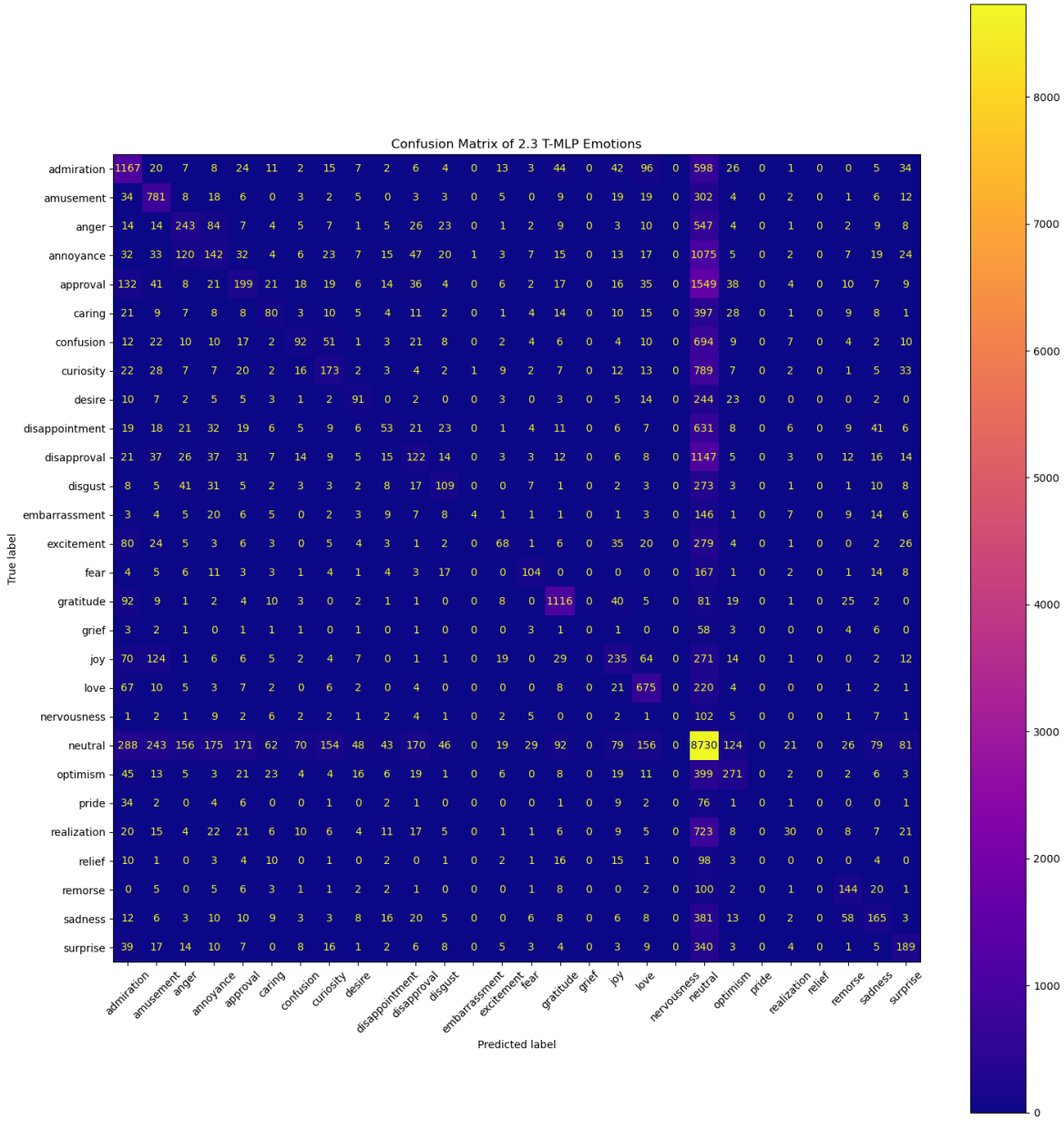

Figure 2.1.6.3: T-MLP Emotions Confusion Matrix

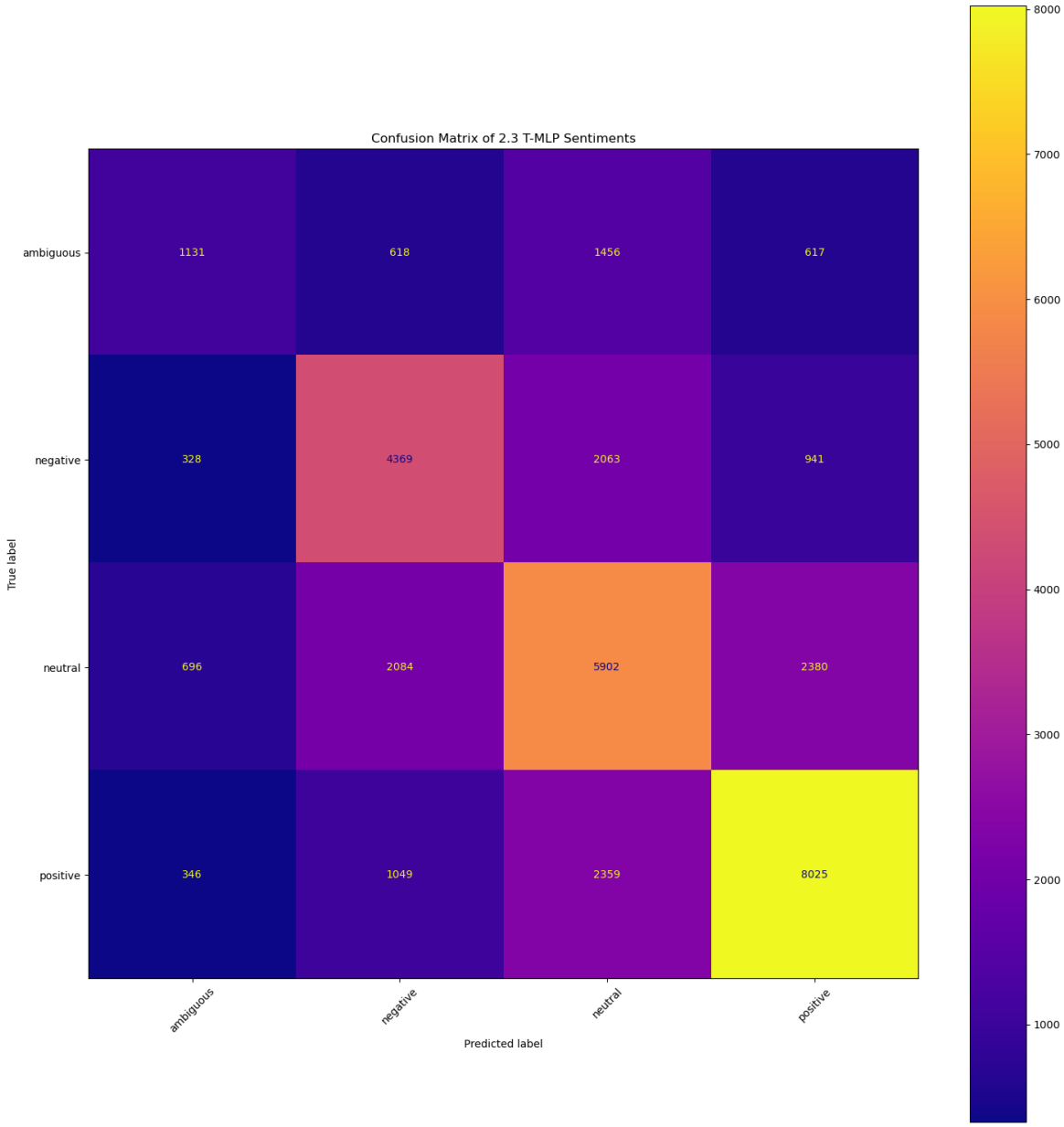

Figure 2.1.6.4: T-MLP Sentiments Confusion Matrix

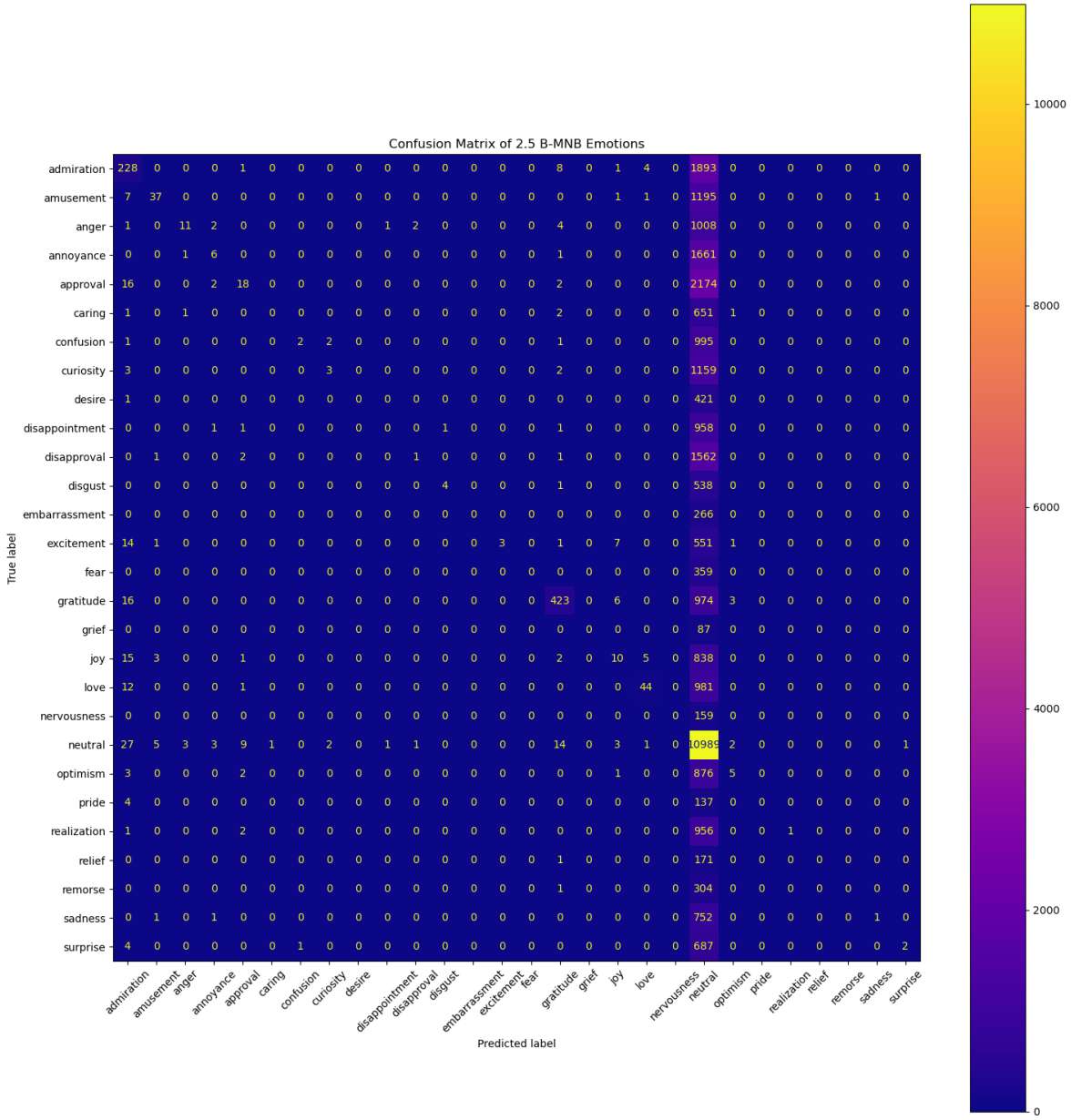

Figure 2.2.1.3: B-MNB Emotions Confusion Matrix w/tf-idf

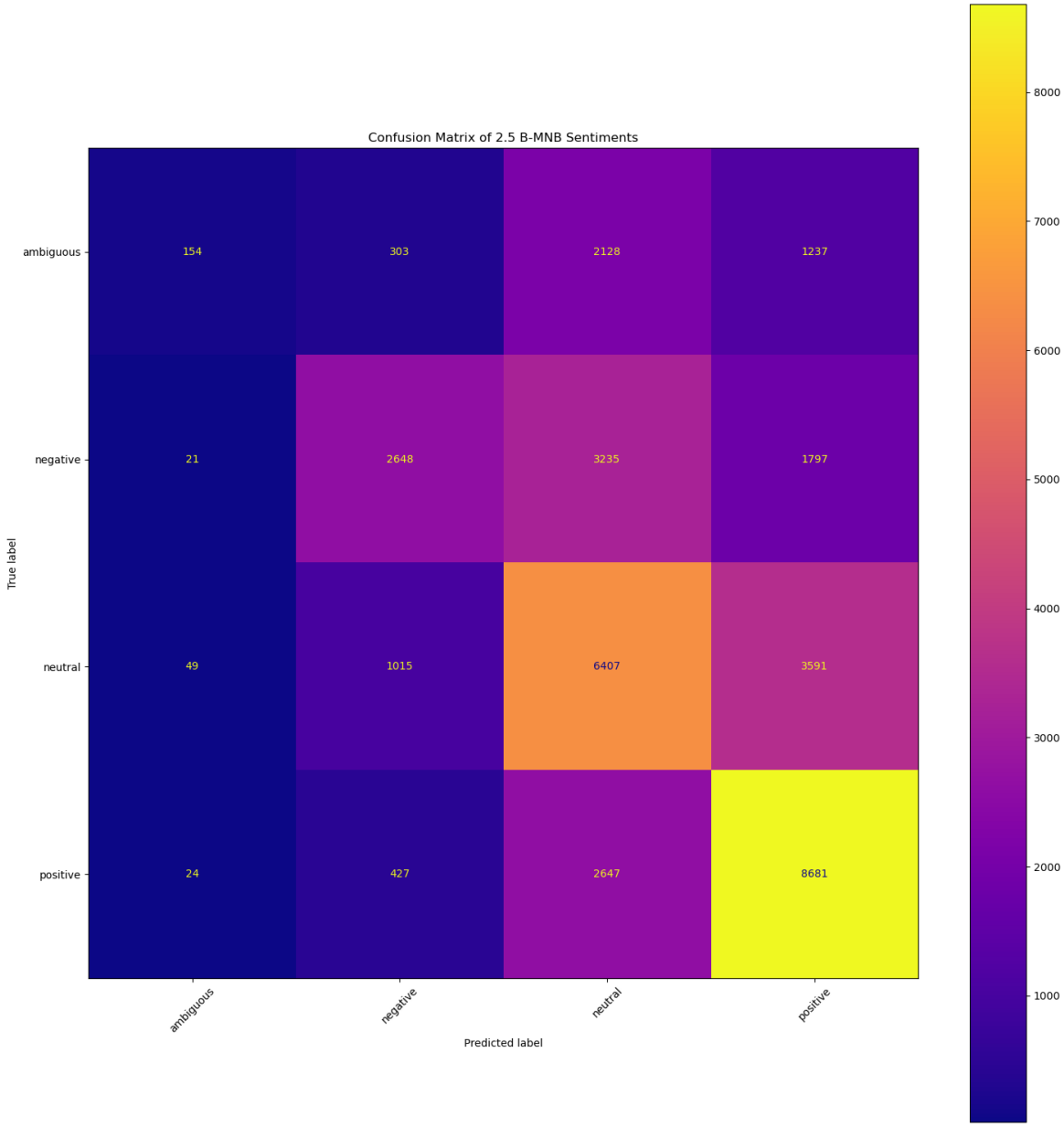

Figure 2.2.1.4: B-MNB Sentiments Confusion Matrix w/tf-idf

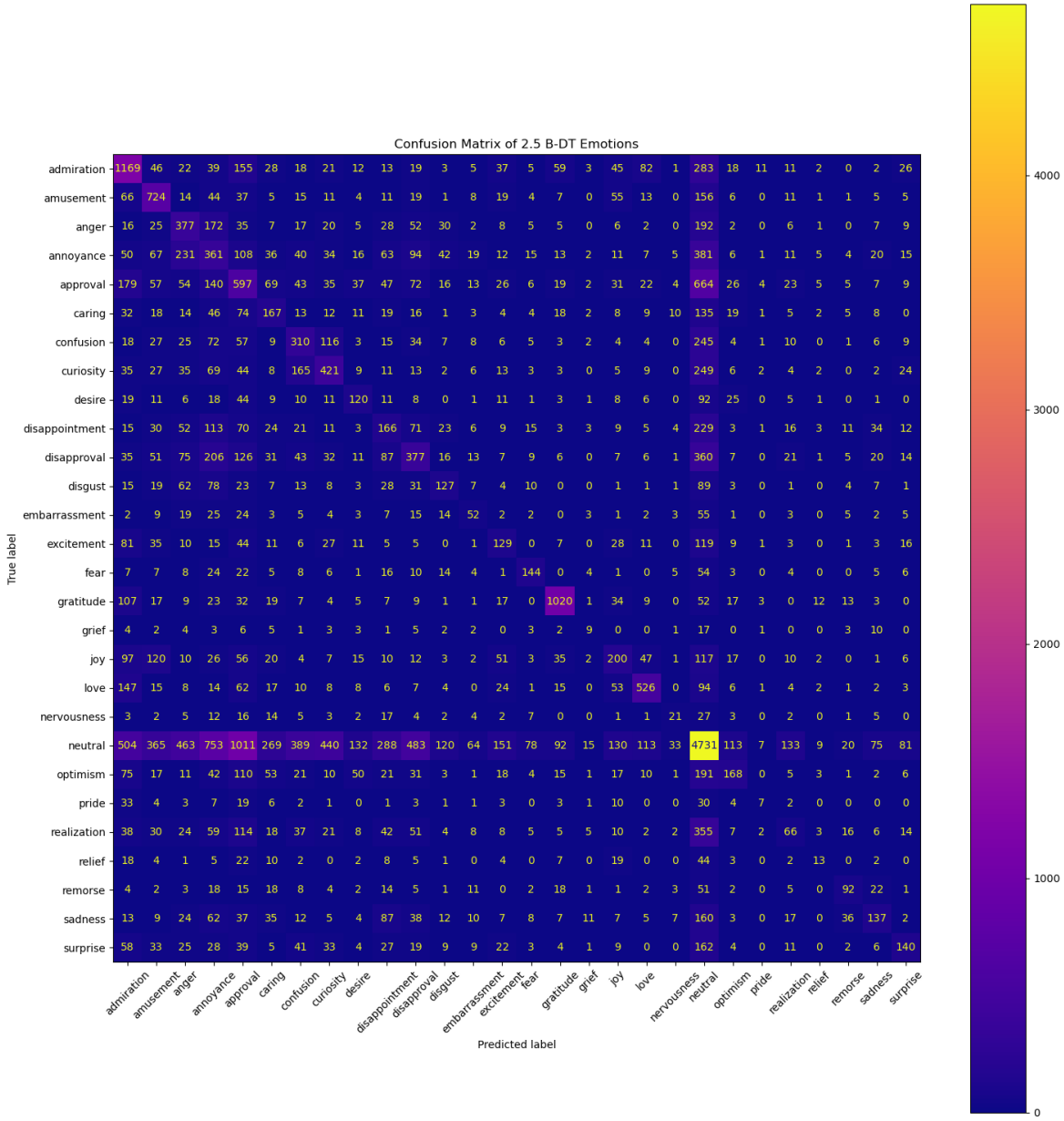

Figure 2.2.2.3: B-DT Emotions Confusion Matrix w/tf-idf

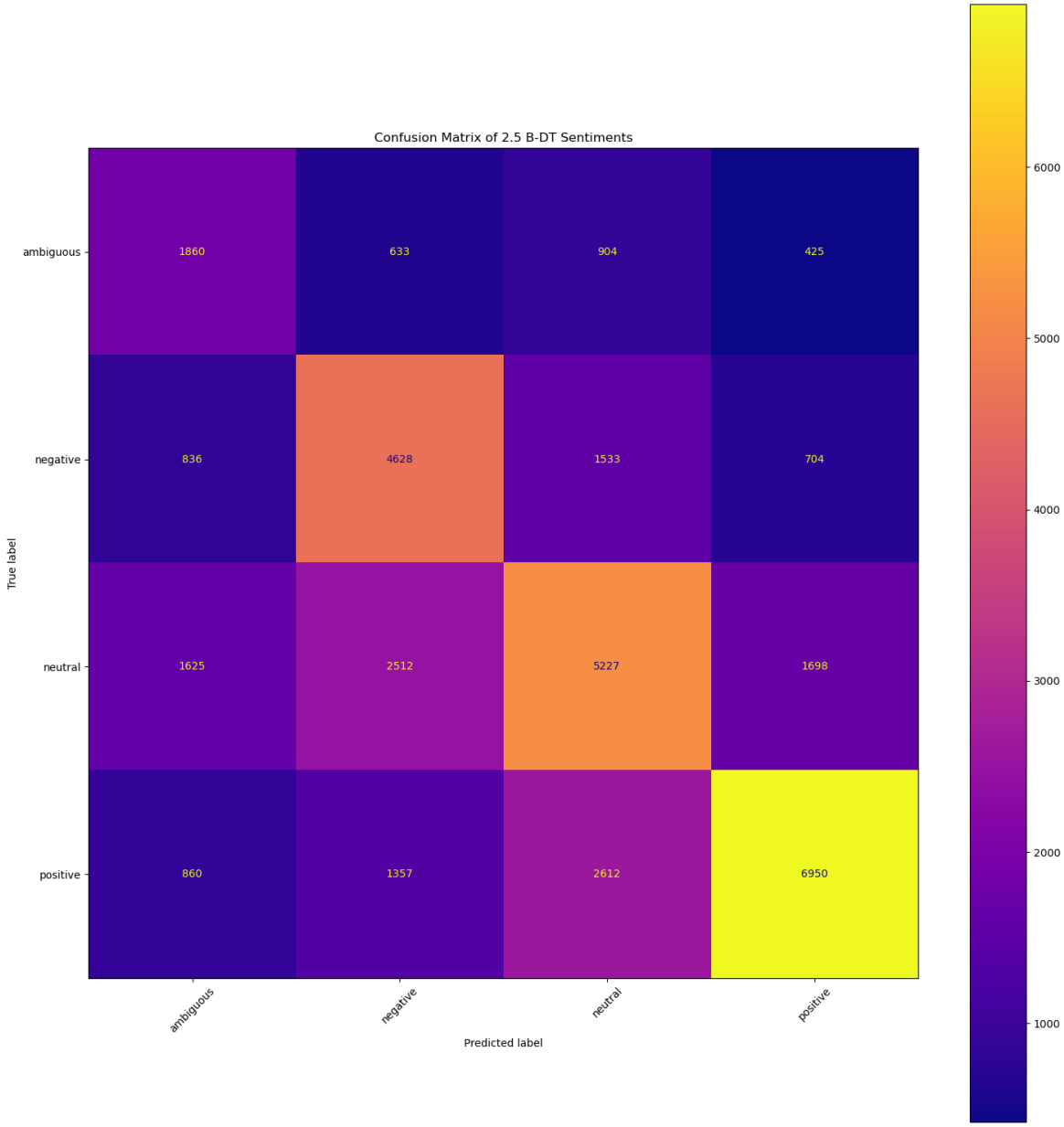

Figure 2.2.2.4: B-DT Sentiments Confusion Matrix w/tf-idf

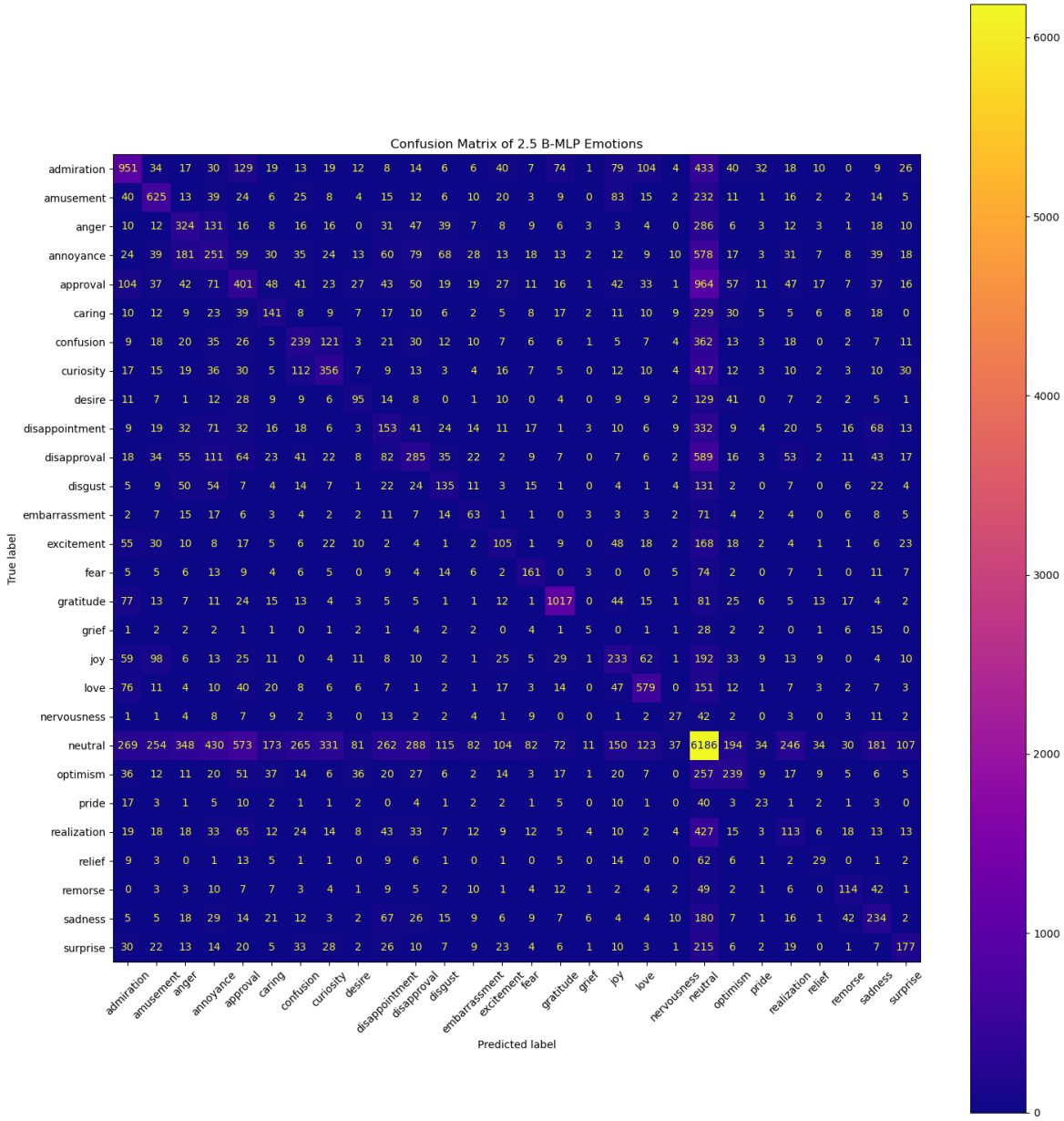

Figure 2.2.3.3: B-MLP Emotions Confusion Matrix w/tf-idf

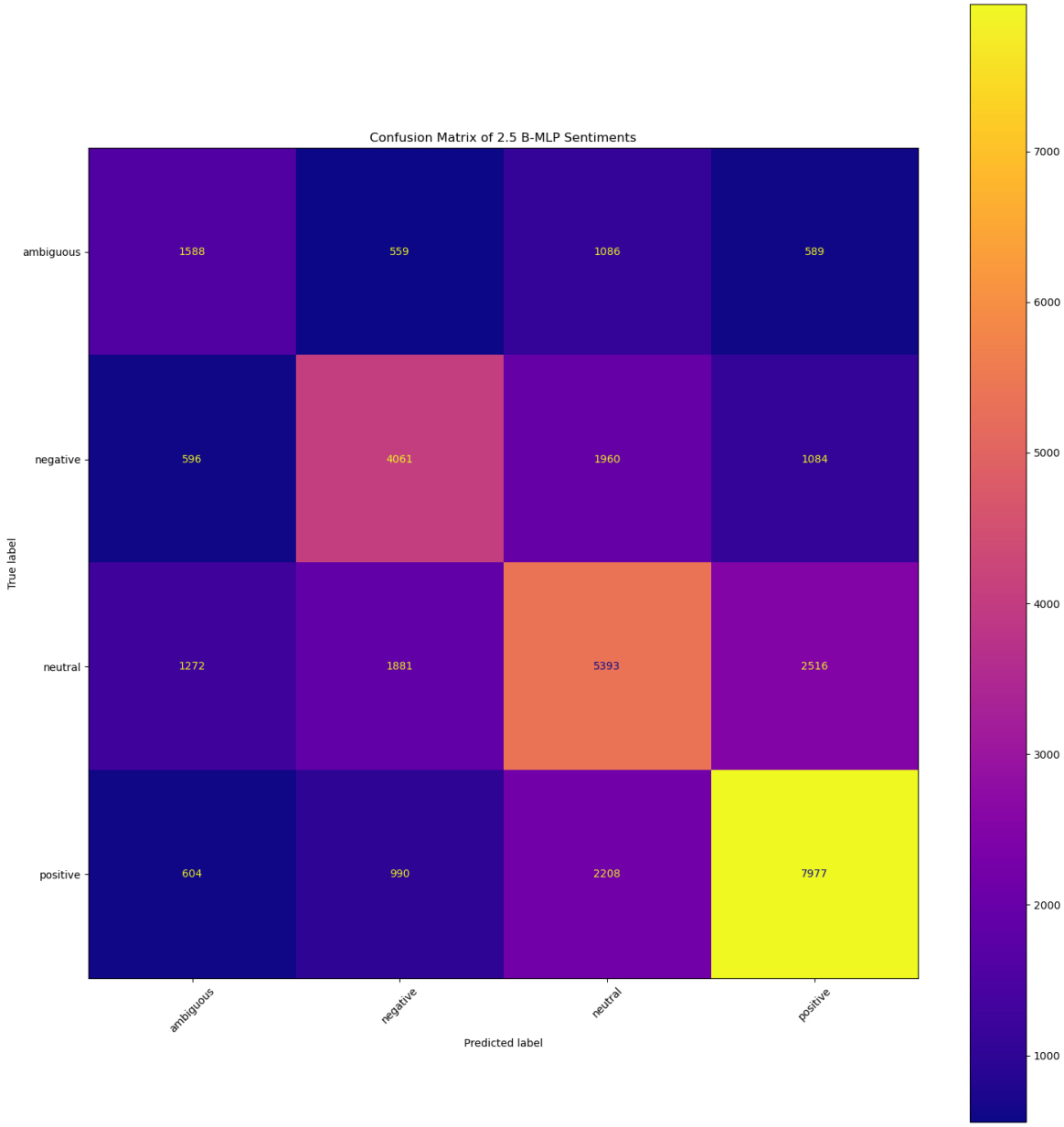

Figure 2.2.3.4: B-MLP Sentiments Confusion Matrix w/tf-idf

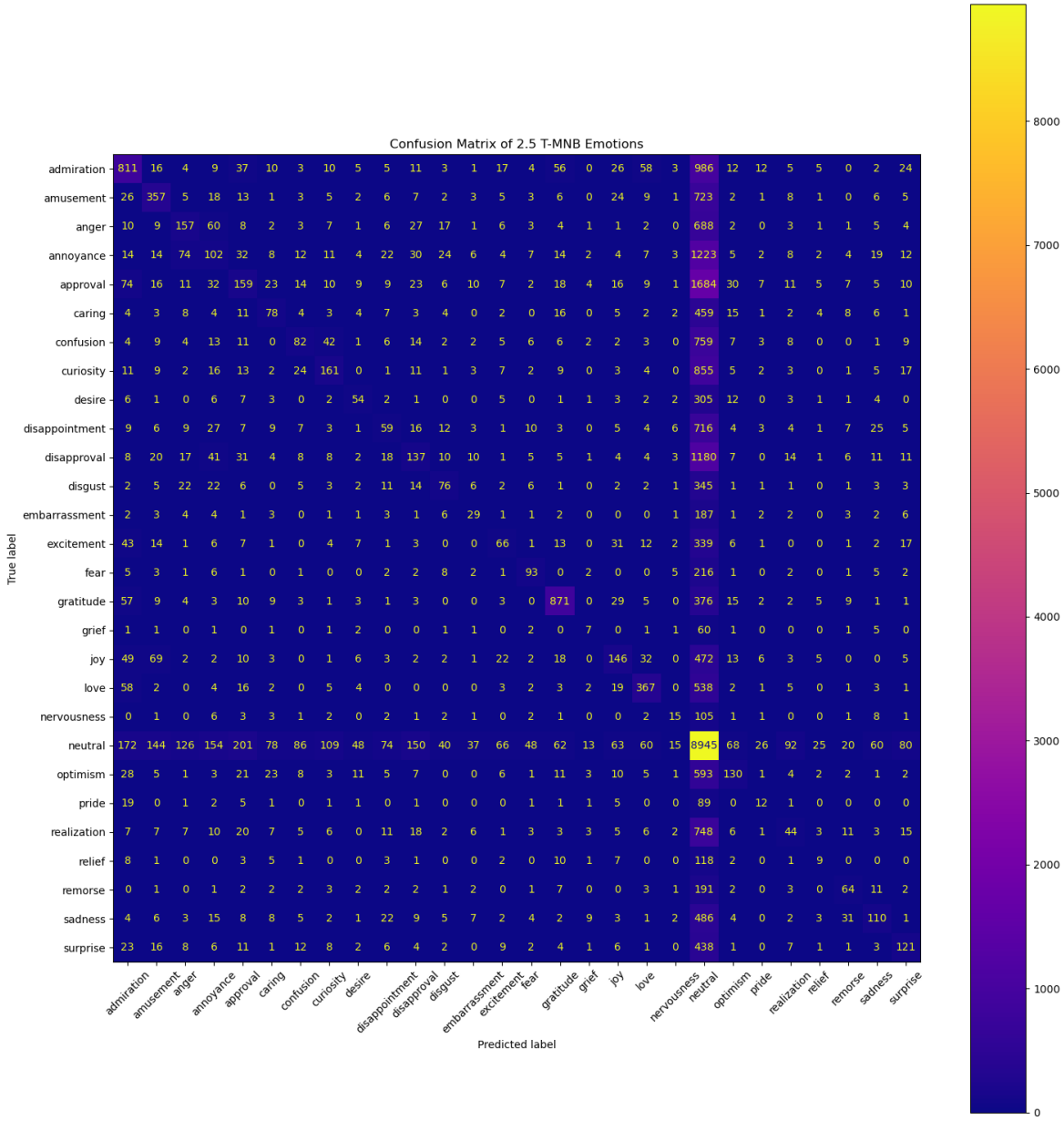

Figure 2.2.4.3: T-MNB Emotions Confusion Matrix w/tf-idf

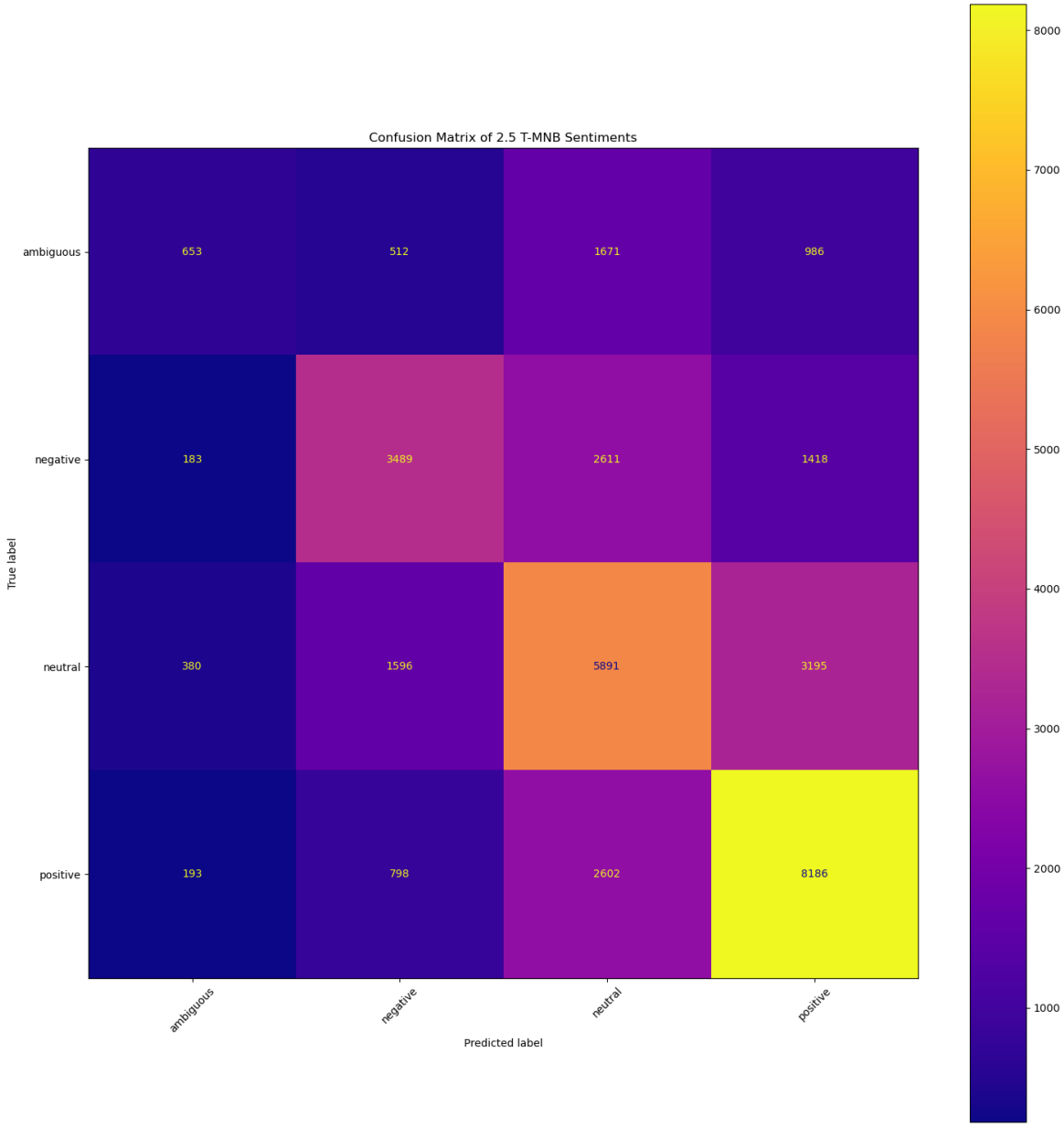

Figure 2.2.4.4: T-MNB Sentiments Confusion Matrix w/tf-idf

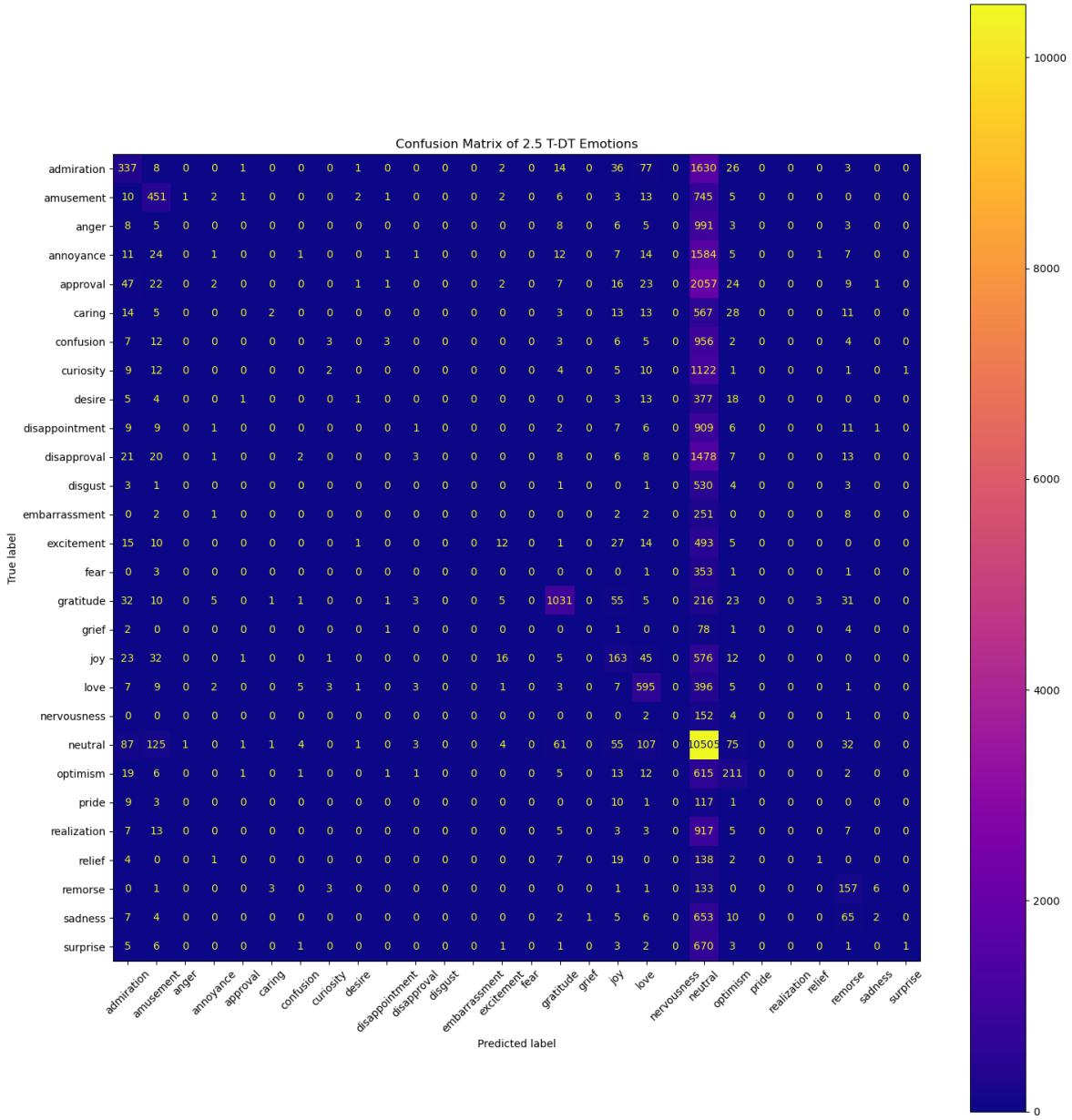

Figure 2.2.5.3: T-DT Emotions Confusion Matrix w/tf-idf

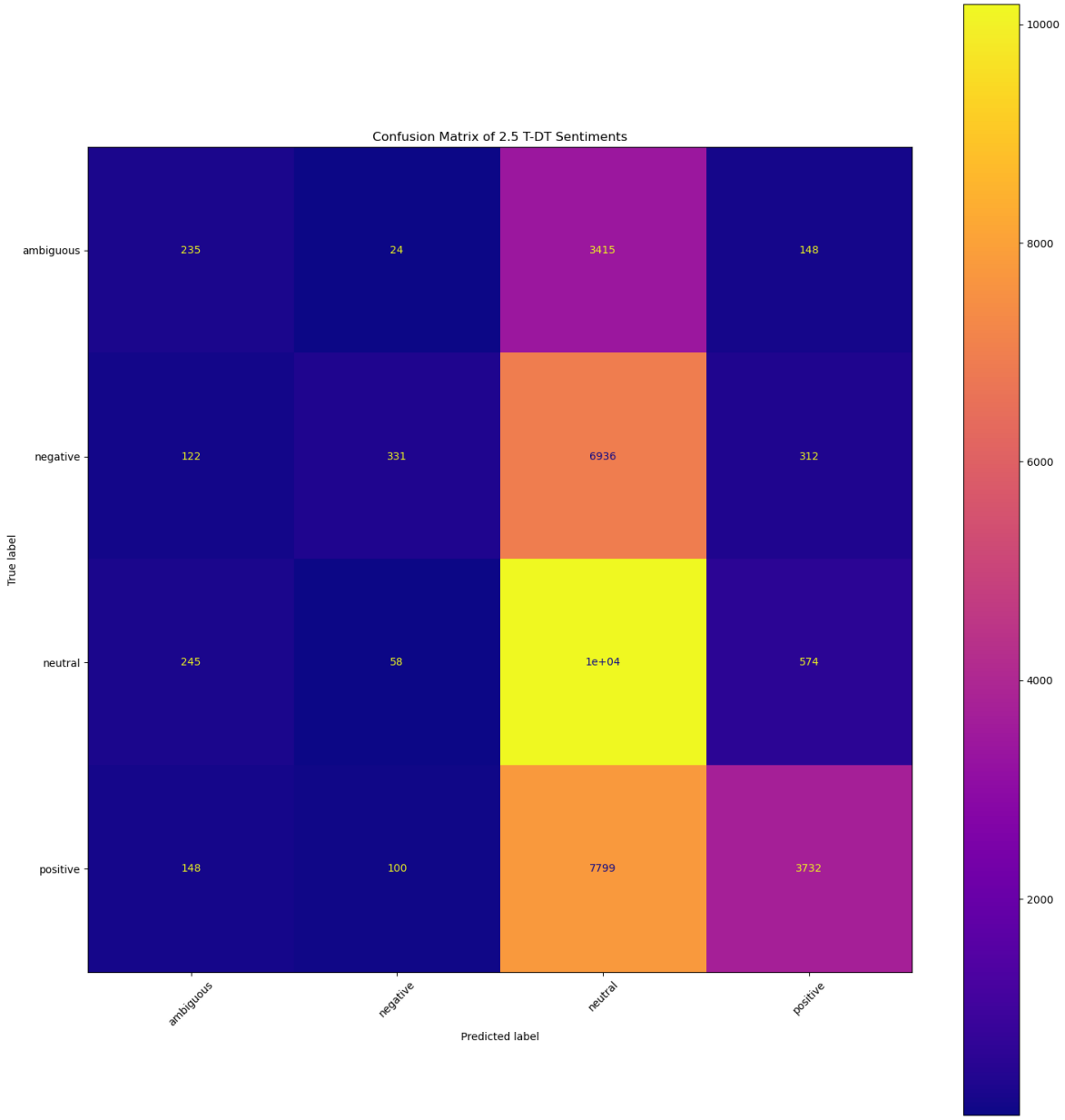

Figure 2.2.5.4: T-DT Sentiments Confusion Matrix w/tf-idf

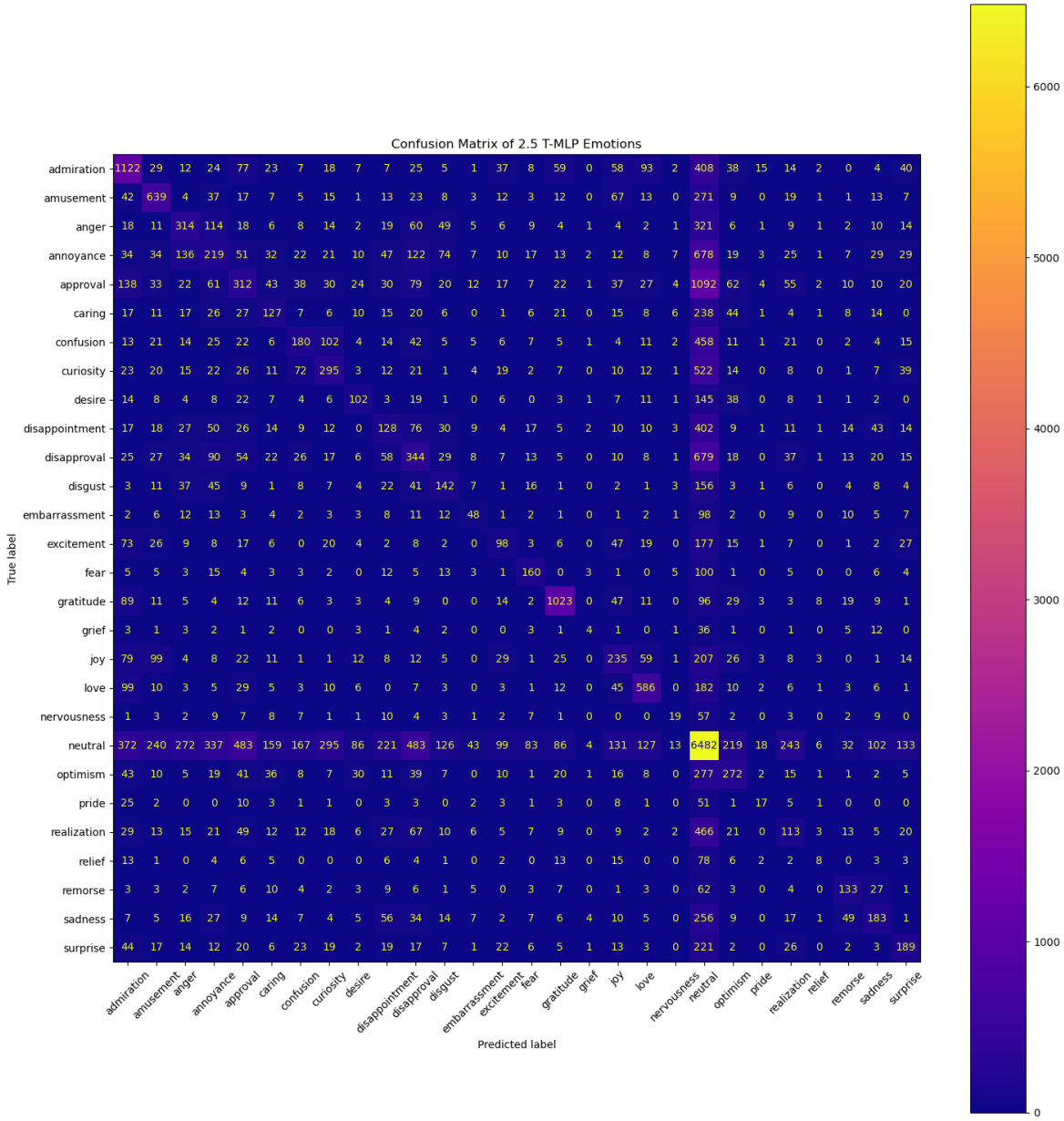

Figure 2.2.6.3: T-MLP Emotions Confusion Matrix w/tf-idf

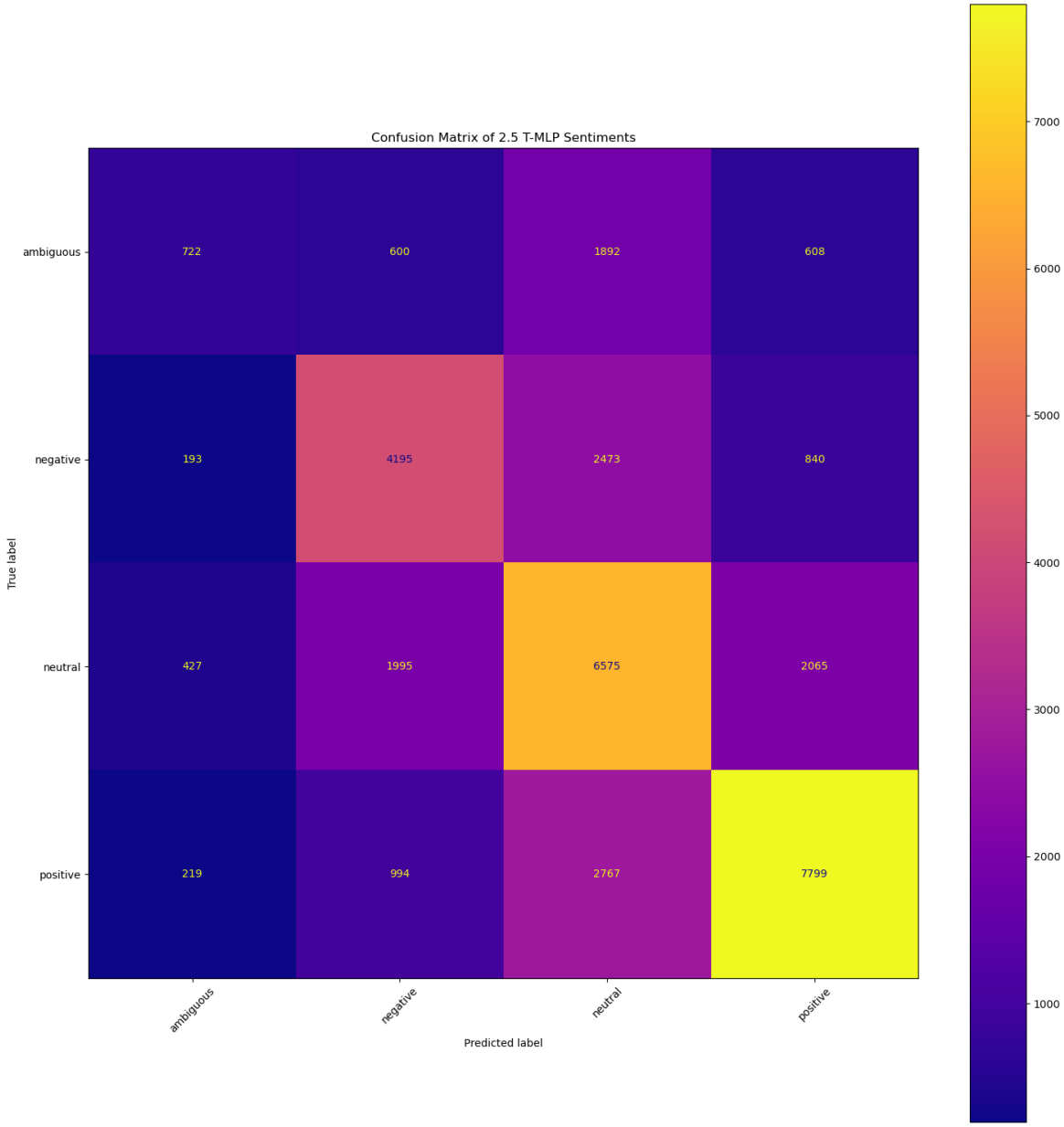

Figure 2.2.6.4: T-MLP Sentiments Confusion Matrix w/tf-idf

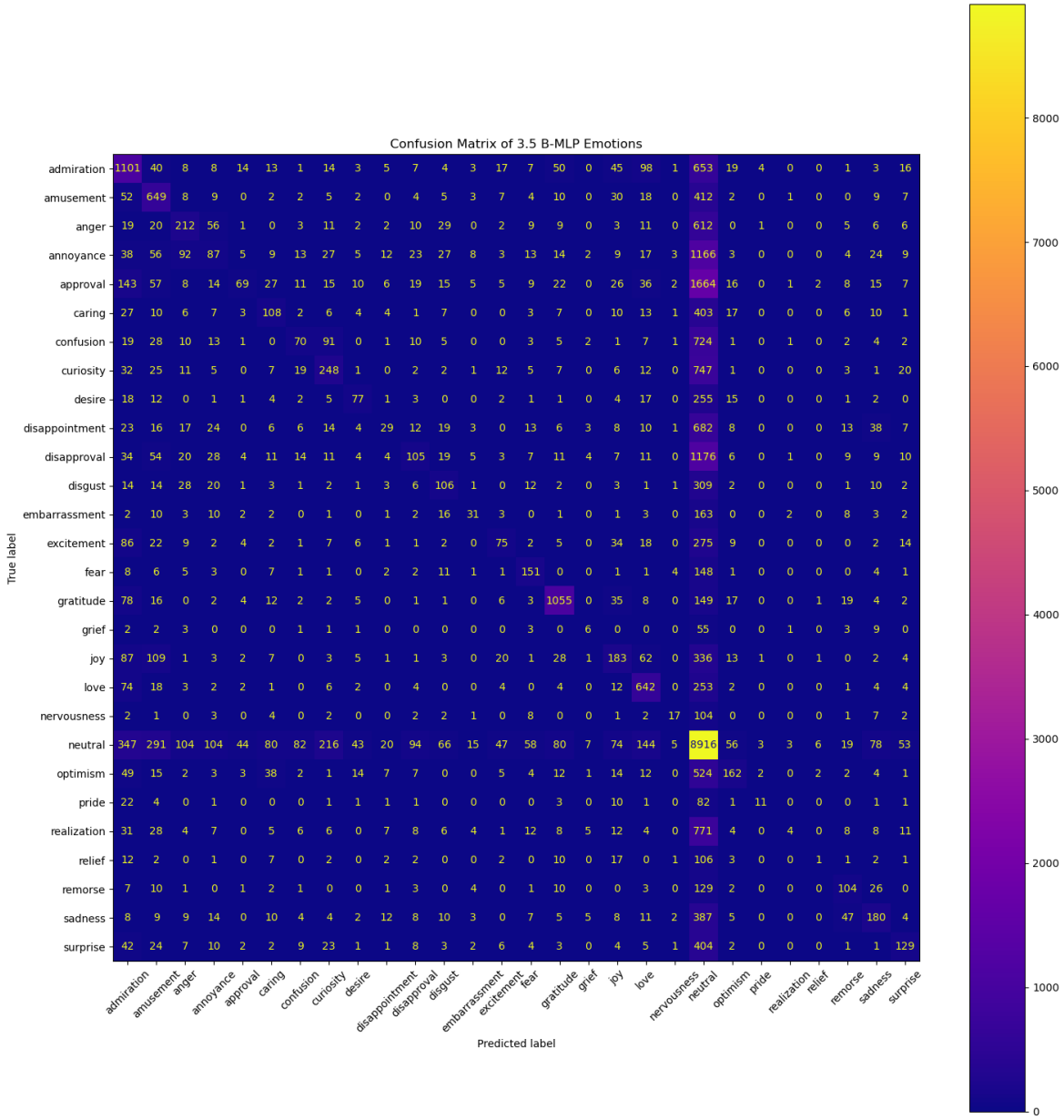

Figure 2.3.1.3: B-MLP Embedding Emotions Confusion Matrix (Google)

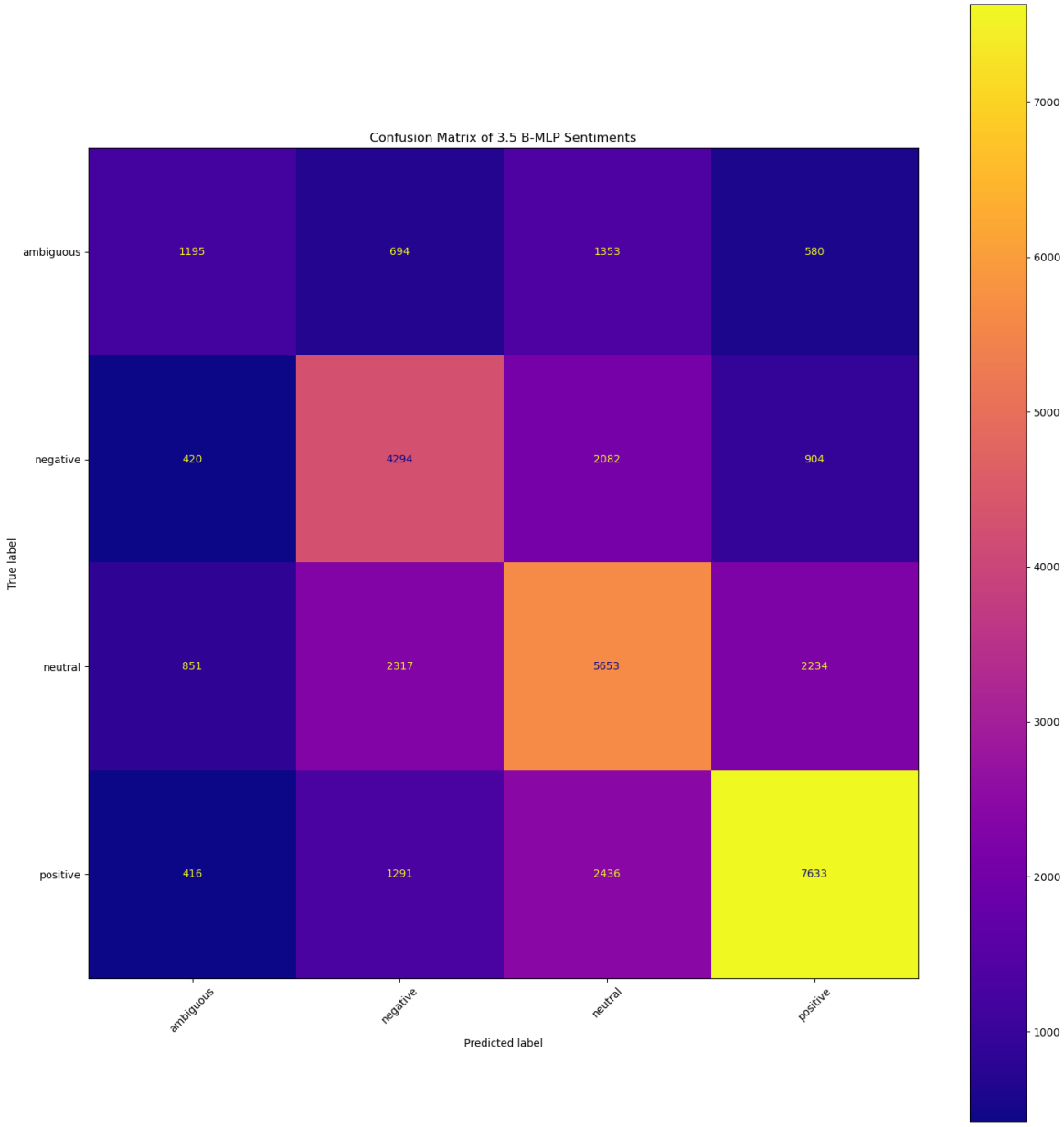

Figure 2.3.1.4: B-MLP Embedding Sentiments Confusion Matrix (Google)

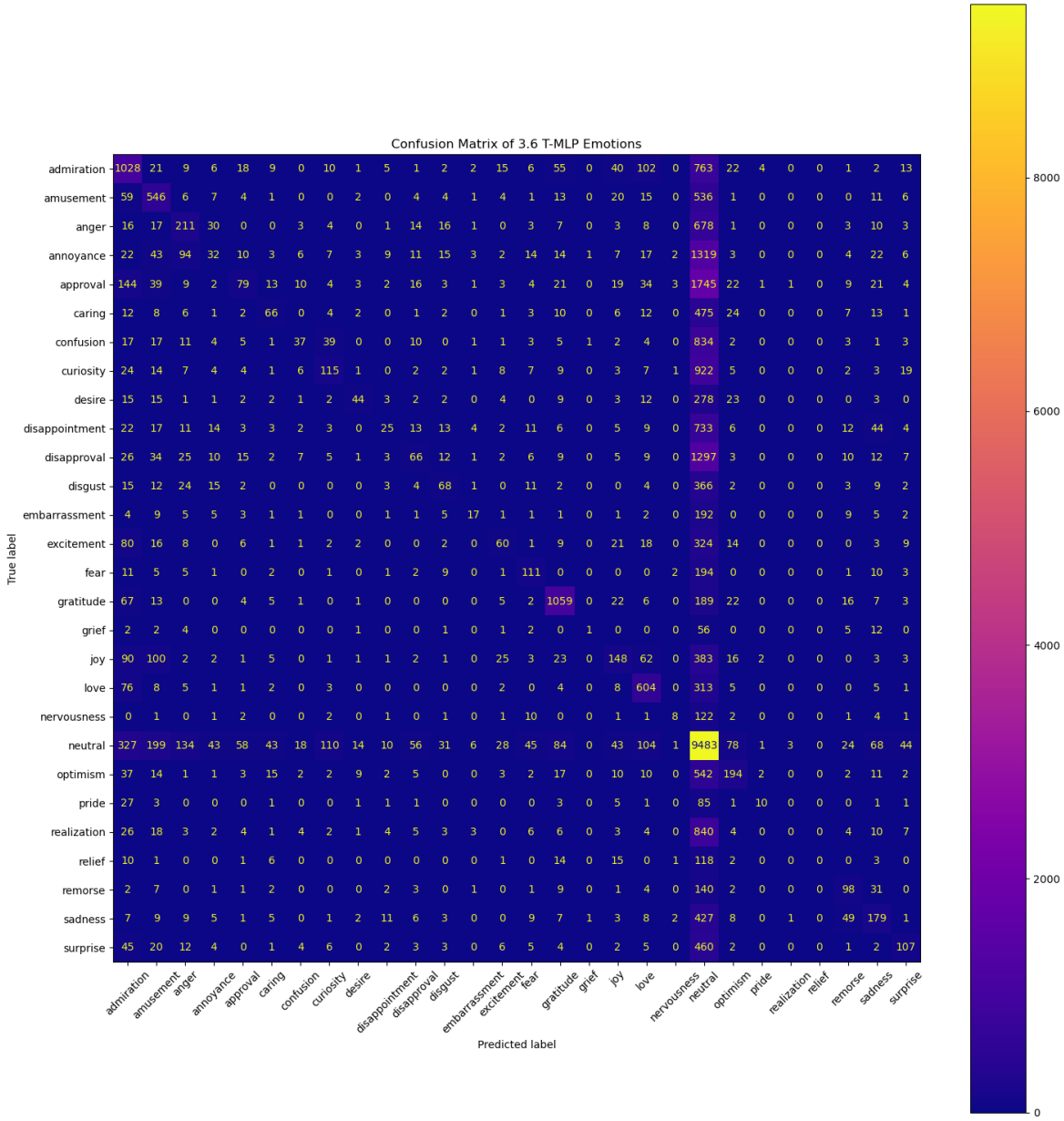

Figure 2.3.2.3: T-MLP Embedding Emotions Confusion Matrix (Google)

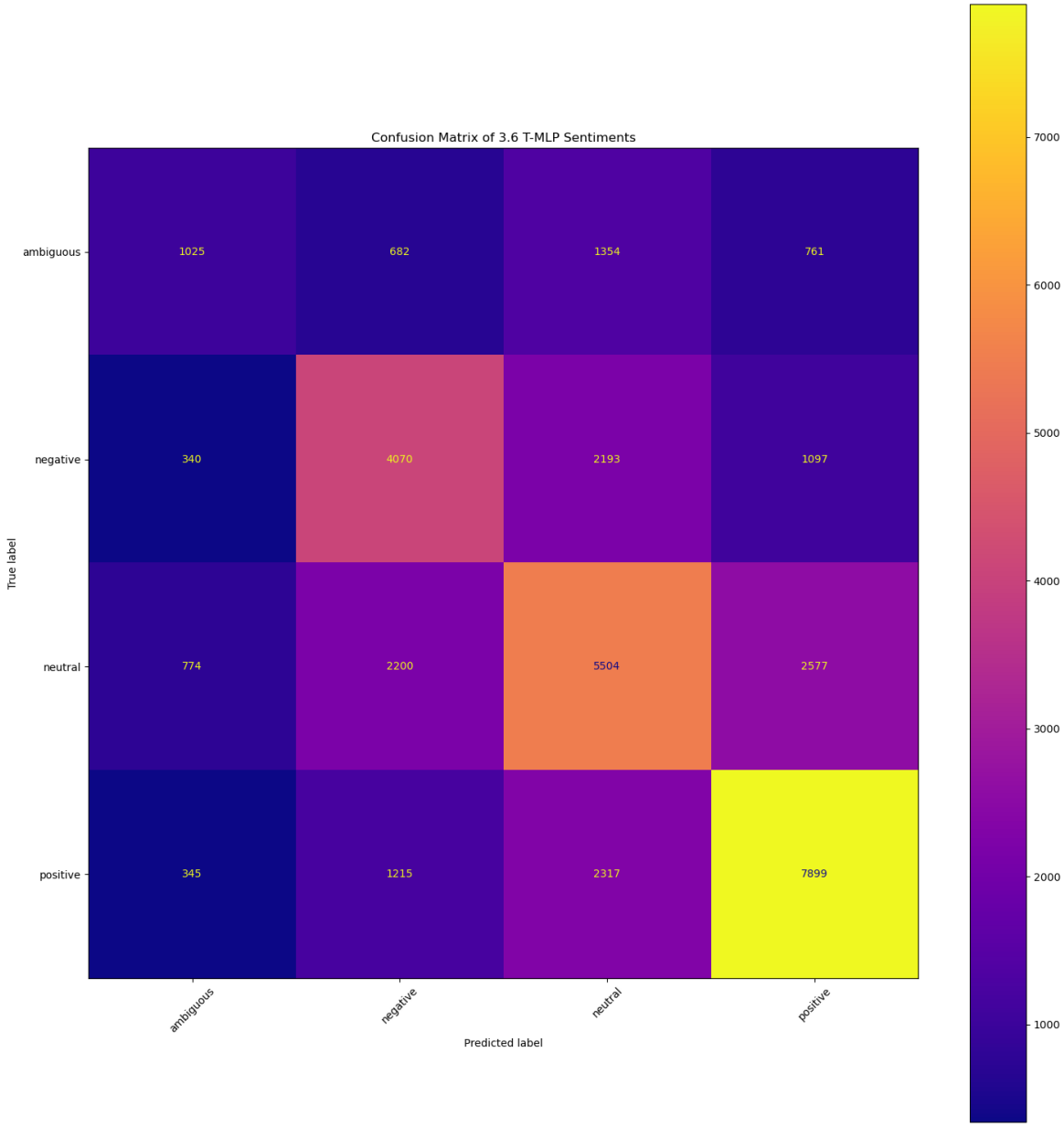

Figure 2.3.2.4: T-MLP Embedding Sentiments Confusion Matrix (Google)

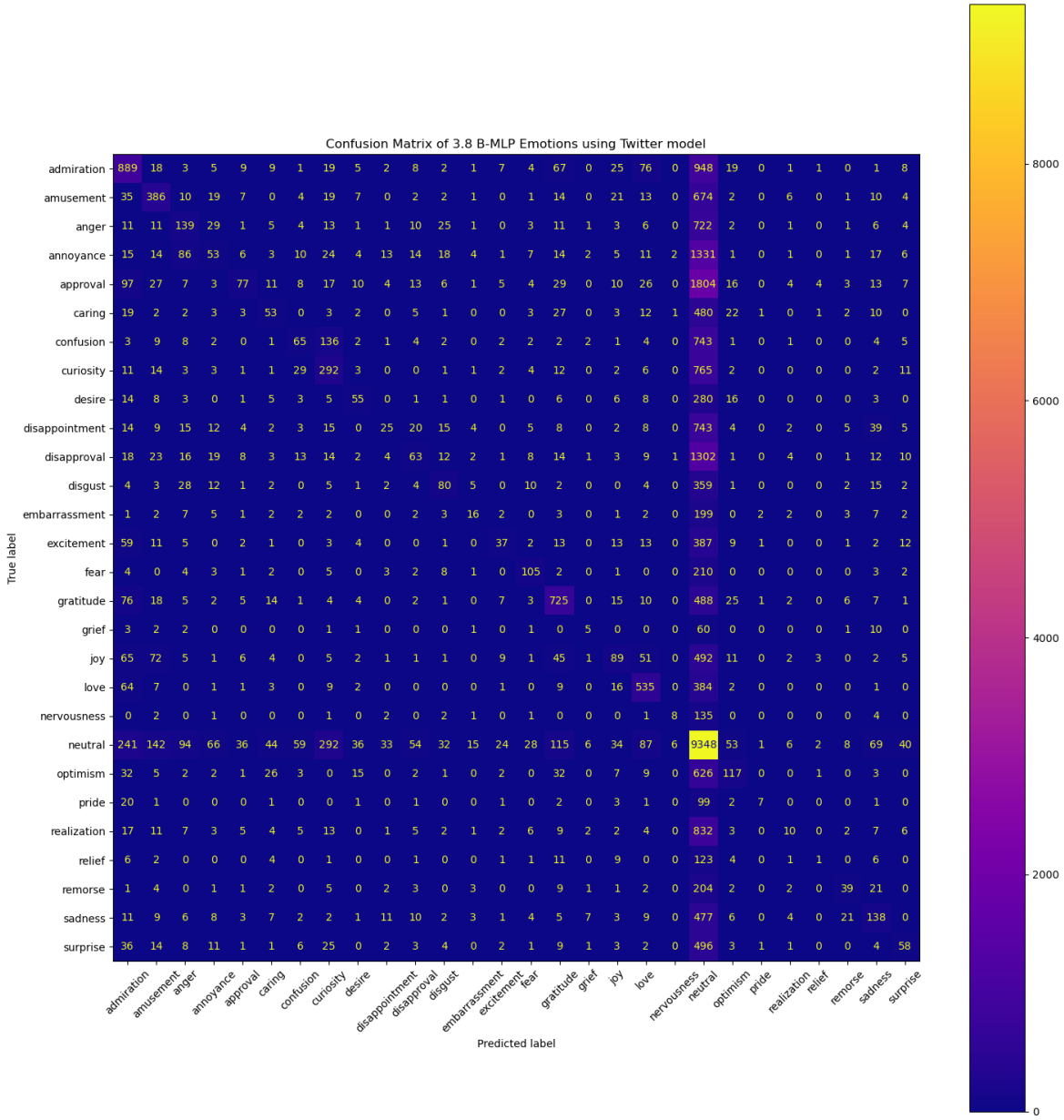

Figure 2.3.3.3: B-MLP Embedding Emotions Confusion Matrix (Twitter)

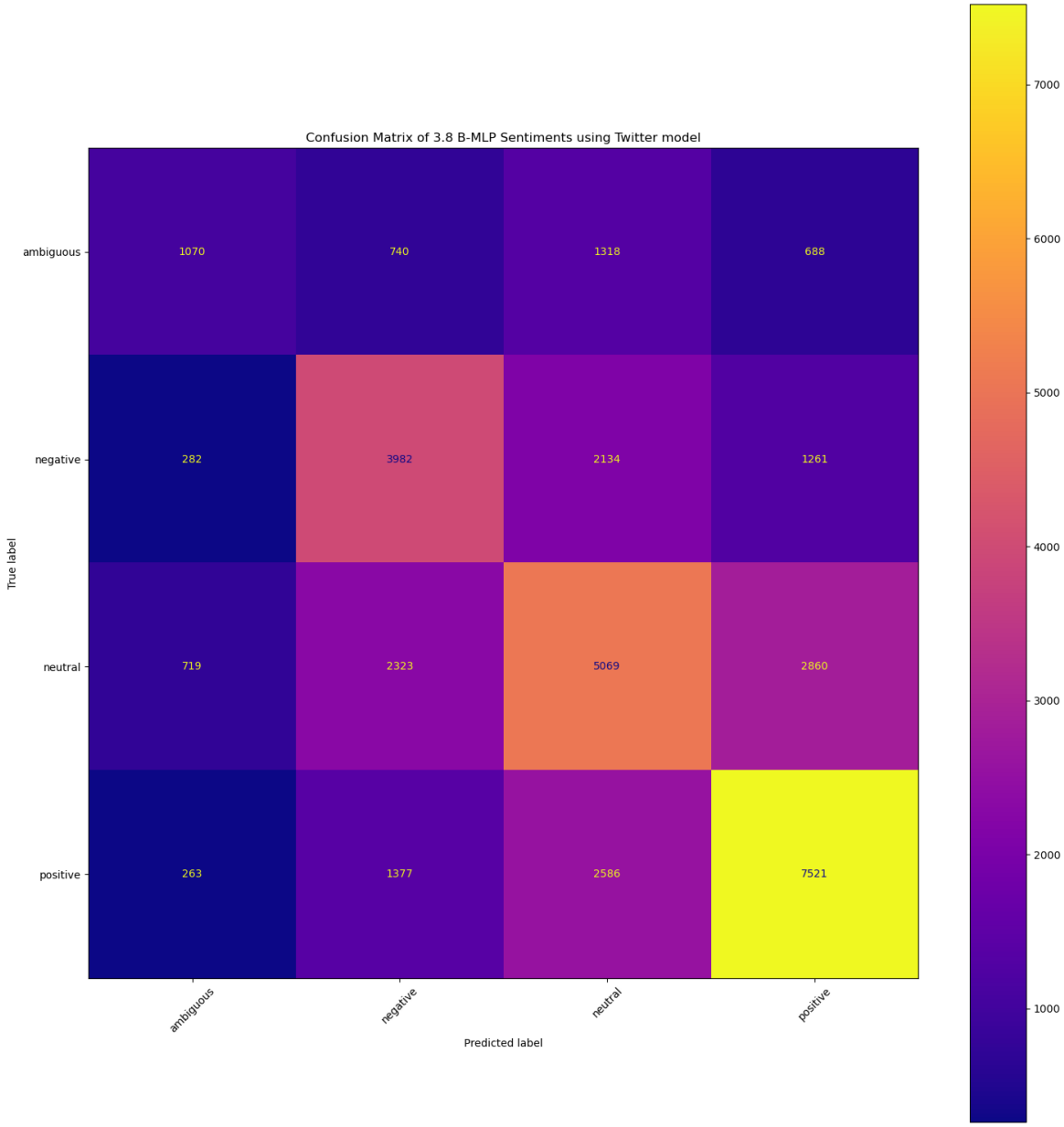

Figure 2.3.3.4: B-MLP Embedding Sentiments Confusion Matrix (Twitter)

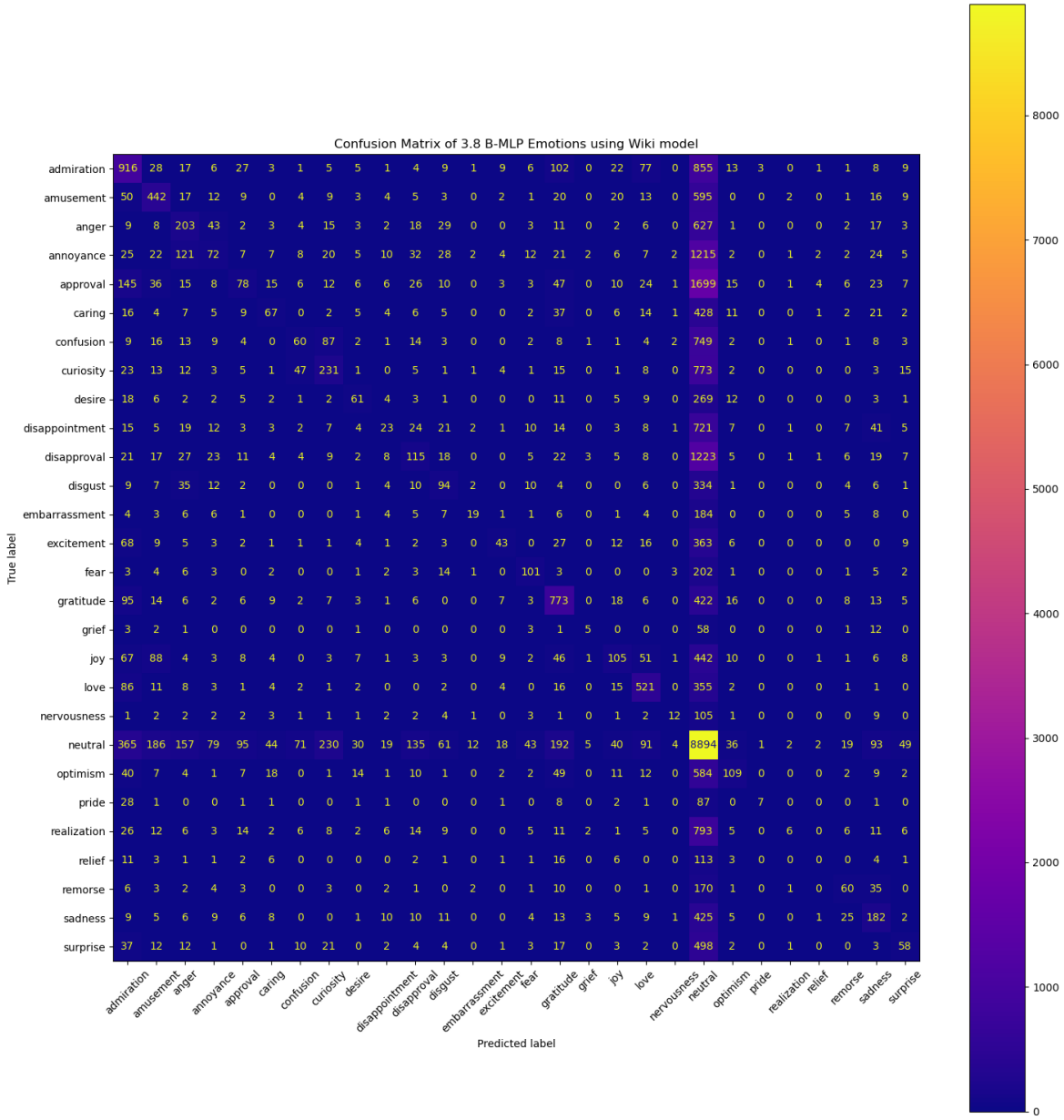

Figure 2.3.4.3: B-MLP Embedding Emotions Confusion Matrix (Wikipedia)

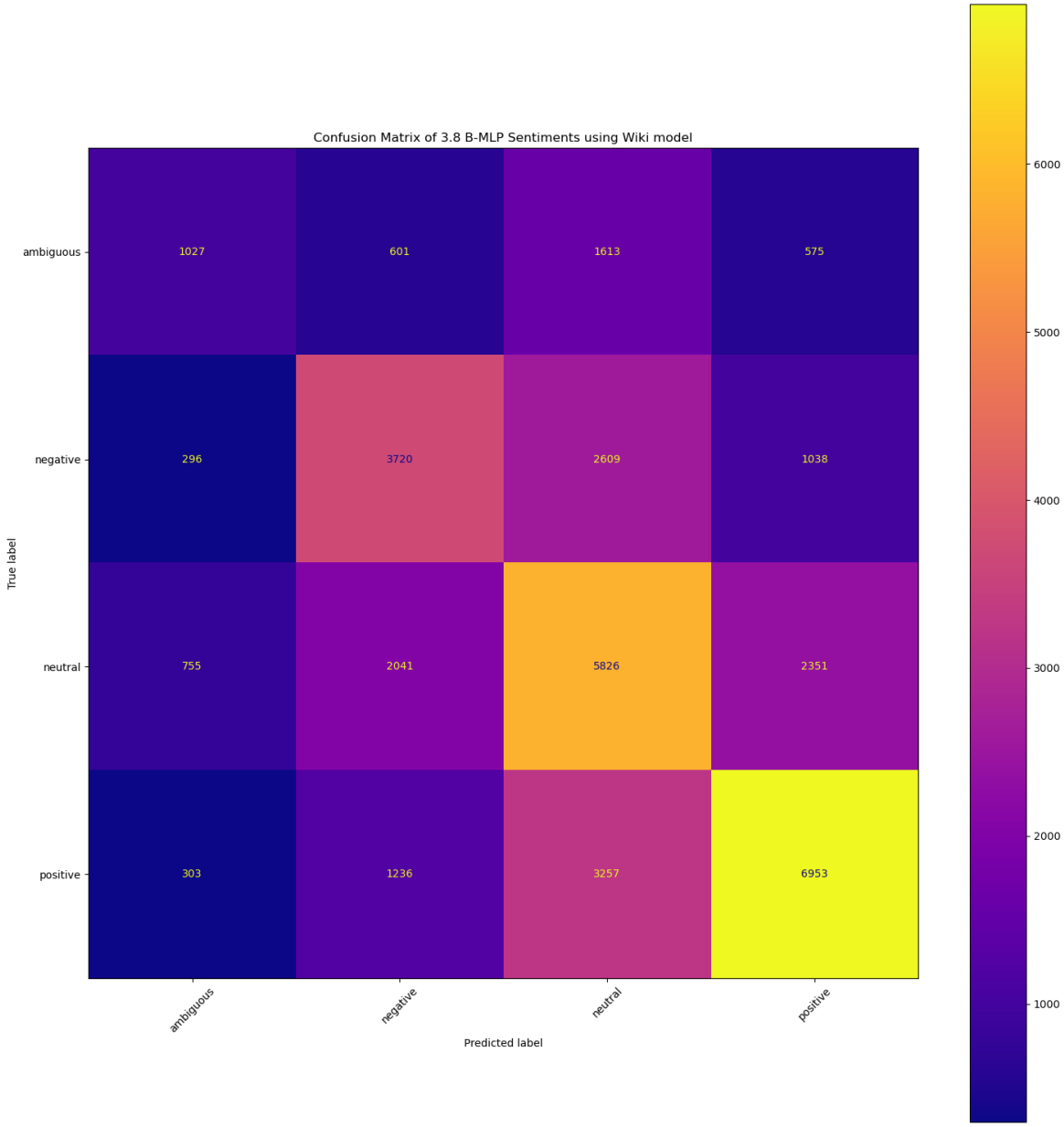

Figure 2.3.4.4: B-MLP Embedding Sentiments Confusion Matrix (Wikipedia)