



Report

Assignment 3: SemEval task (graded part)

Important: Please, when writing the report, follow the principle that less is more. Provide all the information needed for us to understand your work, and not more.

Name and fruit 1: Eva Richter, Orange 

Name and fruit 2: Nora Graichen, Apple 

Contributions of each member to the assignment (brief): Note: this is compulsory!
Submissions that do not include this information will not be accepted.

Data collection/Notebook/Experiments/Description - Nora
Quality Evaluation/Analysis/Data Presentation - Eva

1 Task and data (with examples)

SemEval-2016 Task 4 consists of five subtasks, from which we picked subtask A: Message Polarity Classification. Given a tweet the model has to predict whether the tweet has a positive, negative, or neutral sentiment. Due to the privacy policy the tweets are assigned an ID with their corresponding sentiment label. To obtain the text of the tweet one has to crawl Twitter and the relevant tweets (given by ID) will be downloaded. This subtask is a rerun and provides data from previous years e.g., *SemEval-2013* edition, which we used for training/development. Source: <https://www.kaggle.com/azzouza2018/semevaldatadets>
Our test set is the official one provided in https://www.dropbox.com/s/byzr8yoda6bua1b/2017_English_final.zip?
Path: /2017_English_final/GOLD/Subtask_A/twitter-2016test-A.txt

Please regard our [jupyter notebook](#) for the data representation. We listed samples and the sentiment label distributions. Some example tweets:

Positive: One Night like In Vegas I make dat Nigga Famous

Neutral: @ashaLadawn yeah, Imma be in vegas from 18th-23rd so I just want a dinner in dallas with friends

Negative: I'm stuck in London again... :(I don't wanna spend the night in McDonald's!

While working with the dataset we noted some irregularities: some tweets (*for instance* *"Hey, you're missing Beauty and the Beast, Vampire Diaries & Grey's Anatomy right now on TV. Saturday is Merlin" - @costelloek15 #sisterlife*) were assigned different sentiment labels (e.g., neutral and negative) for exactly the same sentence. Moreover, the tweet messages seem very challenging in comparison to the more transparent movie reviews of the first assignment, and we could not always follow some specific gold label decisions.

2 Method

- Pre-processing:

A label translation from -1 to negative (0), 1 to positive (1) and 0 to neutral (2), was necessary such that the data could be processed by the model. We excluded all text messages that had more than 280 characters (because twitter only allows that many characters) Normalization and anonymisation of twitter user references as "@USER" and URLs as "@URL".

We used a pre-trained BERT Tokenizer from bert-base-cased model, which worked best with the tweet texts in English. The uncased model might miss some IMPORTANT information encoded. This pre-trained tokenizer also brings the input into the correct BERT format with added [CLS], [SEP] and [PAD] tokens.

- Evaluation metrics:

To gain insights into our model we calculated after each epoch

Test Accuracy, Recall, Precision, F1_Score, F2_Score, displayed the Confusion Matrix and looked closely into each class accuracy and which label was in which manner wrongly assigned.

- Model.

We used a tutorial for [Text Classification with BERT in PyTorch](#) to tackle this challenge: We build a multiclass classifier model that uses BERT embeddings to predict whether the tweet has a positive, negative, or neutral sentiment. BERT, Bidirectional Encoder Representations from Transformers, introduced by Devlin et al. (2019), belongs to the transformer-based, fairly new family of neural network architectures. It uses multiple encoders, which are stacked upon each other, with attention to predict words and sentences. We used the BERT model to get an enriched input representation: an embedding vector of size 768 for each of the text tokens. These representations are fed as input for the Message Polarity Classification, using the [CLS] classification token to predict the sentiment of the given tweet. Fine-tuning: The obtained representations from the BERT tokenizer are input into the BERT model. After several experiments, we found out that the following architecture with (hyper)parameters works best: (final model: on Train data set CLEANED @USER, @LINK, drop: 0.1, batch 7 lr 1e-5) final model after: 2 epochs,

learning rate: 1e-5

dropout rate: 0.1

adam optimizer (learning_rate= 1e-5, beta1=0.9, beta2=0.999, epsilon=1e-8)

and gradient clipping

Include information about model/feature improvement if you do more than one iteration.

We tried out different architectures for the model and reported (in the Appendix) our experiments. Due to the unbalanced data set (more neutral and positive messages than negatives) one can observe switches of learning: in one epoch the model may classify neutral or positive labels very well, but then "unlearns" to do so; or to be more precise gives more attention to the negative labels by reshifting its weights, resulting in poorer results for the other classes. Overfitting occurred approximately after two or three epochs. We noted a stagnation of the development testset around 75% accuracy. To decide on the final model, we evaluated the epochs and class-label assignments.

One of our criteria was that the negative and positive labels were assigned with an accuracy of more than 75%, which was fulfilled after 2 training epochs (`finetuned_BERT_epoch_1.model`). We believe, although this is application dependent, a false assignment of a neutral statement as negative or positive is less fatal than a negative, labelled as neutral or a positive one.

3 Results

As we can see in the table below, the best results were obtained on the semeval-2013-test set. The results on the Subset of training data semEval2013 are slightly worse, but overall similar, while the accuracy on twitter-2016test-A is significantly lower. This may be due to the fact that language is changing very fast: many new expressions may have been added that were not used at all in 2013 and therefore do not appear in the training data.

If we look at the accuracy for the classification according to the three different sentiments, we find that, on the subset of training data semEval2013 as well as on the twitter-2016test-A, the highest accuracy was achieved for positive labelled sentences, followed by the accuracy for negative and neutral labelled sentences. Given that the dataset is unbalanced in the sense that we have far more positive and especially neutral labels than negative labels, it is surprising that the lowest accuracy was achieved for sentiments labelled as neutral and a higher accuracy for sentences labelled as negative. For the 2016 set, these results could possibly be explained by the generally lower accuracy due to the size and possibly unknown (new) words, whereas for the subset of training data semEval2013 the reason for this is unclear.

However, regarding semeval 2013 test, the accuracies reflect the unbalanced data better since the highest accuracy was achieved for sentences labelled as positive, followed by sentences labelled as neutral. Given that there is only a minimal difference in the number of positive and neutral labels, especially compared to the other two sets, this is reasonable. The lower number of negatively labelled sentences is also reflected here by a lower accuracy for negative sentiment than for the other two sentiments.

In absolute terms, the highest accuracy was found for sentences labelled as positive and sentences labelled as negative on the semEval2013 subset of training data, and the highest accuracy for sentences labelled as negative on the semeval-2013 test set.

The following table displays the results of all the test runs we made with our final model. The jupyter notebook gives additional insights of the data, however we summarised it for our report:

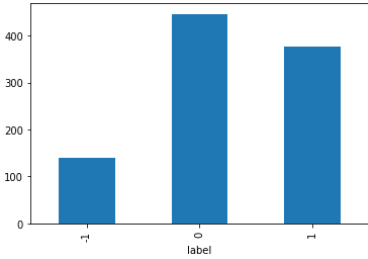
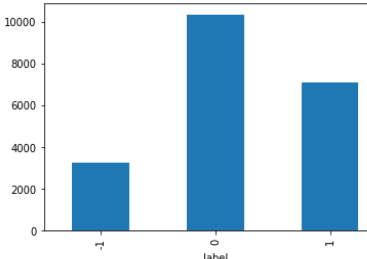
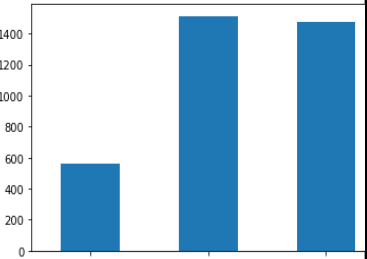
	Subset of training data semEval2013	twitter-2016test-A	semeval-2013-test
Accuracy	0.769	0.654	0.771
Recall	0.769	0.654	0.771
Precision	0.770	0.661	0.771
F1_Score	0.768	0.650	0.771
Confusion Matrix: Neutral, Positive, Negative	[[[505 81] [74 302]] [[434 82] [114 332]] [[763 59] [34 106]]]	[[11602 1971] [2304 4755]] [[7764 2526] [3919 6423]] [[14764 2637] [911 2320]]]	[[1788 284] [333 1140]] [[1656 376] [333 1180]] [[2833 153] [147 412]]]
LABEL: 2 as neutral	Class Accuracy: 0.744 ***assigned as class: 1 %: 15.695 ***assigned as class: 0 %: 9.865	Class Accuracy: 0.621 ***assigned as class: 0 %: 21.243 ***assigned as class: 1 %: 16.651	Class Accuracy: 0.774 ***assigned as class: 2 %: 18.941 ***assigned as class: 0 %: 3.666
LABEL: 1 as positive	Class Accuracy: 0.803 ***assigned as class: 2 %: 15.691 ***assigned as class: 0 %: 3.989	Class Accuracy: 0.718 ***assigned as class: 2 %: 20.489 ***assigned as class: 1 %: 7.707	Class Accuracy: 0.780 ***assigned as class: 1 %: 15.466 ***assigned as class: 0 %: 6.543
LABEL: 0 as negative	Class Accuracy: 0.757 ***assigned as class: 2 %: 16.429 ***assigned as class: 1 %: 7.857	Class Accuracy: 0.674 ***assigned as class: 2 %: 26.406 ***assigned as class: 0 %: 6.233	Class Accuracy: 0.737 ***assigned as class: 2 %: 17.352 ***assigned as class: 1 %: 8.945
Distribution: (-1: neg., 0: neutral, 1: pos.)			

Table 1: Results, testing on held out test set of *semEval2013*, *twitter-2016test-A* and *semeval-2013-test*

4 Quantitative results (comparison with results of systems submitted to SemEval (if applicable))

We only found F-scores for the SemEval-2013 subtask B, the F-scores of 38 teams, who submitted constrained and unconstrained systems (unconstrained means that additional Tweets or additional sentiment-annotated text were used). The average F1-measure was 53.7% for the constrained and 54.6% for the unconstrained systems. The best F-score was achieved by NRC-Canada with 69.02%. These models are probably much more simpler and CPU friendly to run than our model. Nonetheless the board scores are less informative and insightful, and the unconstrained systems are especially difficult to retrace.

We encountered a paper by [Alsaeedi, Abdullah. \(2019\)](#), who investigated the evaluation and comparison issue of Twitter sentiment analysis methods, because researchers evaluate their methods based on different datasets and different evaluation metrics. In this work, the following table is published for the SemEval-2013 dataset: (*Support vector machine (SVM), and naïve Bayes are supervised classifiers, widely used for classification*)

Table 5. Evaluation Results for SemEval-2013 dataset

Classifier	Precision	Recall	F-measure
SVM	0.71	0.68	0.65
Bernoulli naïve Bayes	0.69	0.63	0.58
Multinomial naïve Bayes	0.66	0.65	0.62
Linear regression	0.71	0.68	0.65

We conclude that the enriched representations of the BERT model indeed revolutionised Machine Learning, although our results with F-score 0.77% represent only small improvements made. The gradient search and optimization step could be more extended and the classifier more fine-tuned to the data-set, though it is questionable how generalizable, as seen with the results of twitter-2016test-A, the model's approaches are. Thus, we inspected our model's errors to gain more insights of its inner workings:

5 Model Analysis and Error analysis

In this section, the model is first analysed, as far as this is possible given the fact that our model is like a black box for us and we can not exactly follow and understand what is happening inside, followed by an analysis of the misclassifications.

5.1 Model analysis: BERTViz Model View, Attention mechanism

We tried to examine the attention across all of the layers (rows) and heads (columns) in the model. Since we used *bert-base-cased* with its 12 layers and 12 heads and the specific BERTTokenizer.from_pretrained('bert-base-cased'), we had to adapt the demo version presented in class a bit and of course our input. The attention mechanism was something new for us to work with and is still quite mysterious: It is unclear what kind of information the attention head weights capture; however, it does not resemble human attention intuitively. BERTology research and usage of BERT in many systems has shown its all-rounder nature, probably taking advantage of redundant information encoded and relying on the special

characters ([see](#) Rogers et al. 2020. A primer in bertology: What we know about how bert works.)

Though, the demo gave insights into information used in the model and the quality of the enriched representations. The BERT tokenizer uses byte pair encoding to represent possibly unknown words and reduce the out of vocabulary tokens. The approach is to break down words into sub-words or rather sub-components and piece unknown words together from smaller bits and pieces to obtain a representation. These split representations also share attention connection(Layer 2 Head 1)alignments.

Analyzing the model is, in comparison to the sentiment classifier with a logistic regression model, less transparent and involves more uncertainties. Since this is our first work done with BERT, we believe to have collected enough information to make an informed model choice with the evaluation, though the attention mechanism and the 110 million parameters is still open to explore.

5.2 Error analysis

Since even the annotation of sentences with sentiments can cause problems, and there are often several reasons for and against the assignment of a label that the human annotator has to weigh up, the analysis of why the model assigns a sentiment different from its label is anything but easy. As we did not only have the classification option between positive and negative (like in assignment I), but also included neutral, it made the analysis even more difficult. The degree of difficulty for analysis, however, differs according to the sentence and, as we find, also according to class. We consider the assignment of positive sentiment to sentences labelled as negative and vice versa as “worst” error due to the greater “distance” between the class labels. However, the fact that the greatest contrast lies between these classes facilitated the error analysis for this type of misclassification. Nevertheless, even when classifying positive and negative sentiments, it is important to remember that an affirmative sentence does not always convey positive sentiment and a sentence containing negated words does not always convey negative sentiment.

We found the misclassification of sentences with negative/positive sentiments as neutral or sentences with neutral sentiment as positive/negative less serious, even though it is not correct in itself. However, these types of misclassification are not as straightforward to analyse as between positive and negative sentiment, where a stronger contrast makes the misclassification usually obvious at first glance.

In the following, examples and related interpretations of possible causes are given for each type of miscategorisation that was mentioned. The numbers in brackets after “tensor” represent the classification, with the first one indicating to which sentiment the sentence should have been assigned, and the second one indicating how the sentiment was actually classified.

- Classification of negative sentiments as positive or positive sentiments as negative

```
("[CLS]After the disappointment of the 4Draws ( an og bust it ), I've gone for 7 away wins tomorrow ( including the Mersey side derby ). [SEP]", tensor(1), tensor(0))
```

The sentence above was classified as negative rather than positive, probably because of the words "disappointment" and "wins", the former of which has a strong negative connotation and the latter of which has a strong positive connotation, thus contrasting them. In the end, the sentence was classified as negative, although the sentence itself has a positive meaning in that there have been disappointments in the past, but now there is hope for future wins. This confidence is more evident here than past disappointments, but the model was not able to recognise this correctly.

```
('[CLS] More primary filing mess, Timmons ville is at it again, and more 7th Congressional interviews. Thank God it is Friday! # flosc [SEP]', tensor(1), tensor(0))
```

The latter sentence was also classified as negative instead of positive. Presumably this is due to the negative connotation of the word "mess," which is not the main issue here. Rather the emphasis is on "Timmons ville is at it again" and "thank God it is Friday," so presumably the support of a sports club or similar, which contains positive sentiment.

```
("[CLS]RT @ USER #Londonriots is trending 3rd worldwide..... This is NOT something to be proud of United Kingdom!!! Sort it out!!!! [SEP]", tensor(0), tensor(1))
```

In the case of this sentence, an obviously negative sentence was classified as positive. One can clearly see that the sentence has a negative meaning, as is particularly evident in "NOT something to be proud of", with the negative "NOT" even written in capital letter, and "sort it out!!!!" with three question marks. However, the model did not classify the sentence as negative, but as positive, which can probably be attributed to "trending 3rd worldwide" in the sentence, since that in itself, without including the context, is a positive message. However, given that it is "Londonriots" that are trending so high, the sentence takes on a negative meaning.

Similarly, the following sentence was classified as positive, while it should have been classified as negative.

```
("[CLS]@ USER Let me guess, Rita was just another one of your fictional callers? Amazing how the call became political... c'mon! [SEP]", tensor(0), tensor(1))
```

In this case, the misclassification is due to the sarcastic nature of the sentence, which was not recognised by the model. For with "just another one of your fictional callers" someone is openly making fun of someone who only receives calls from imaginary people. Furthermore, the "amazing" is also meant ironically, because the sentence was probably classified predominantly as positive, because the politicisation of the conversation is basically criticised with this ironic remark.

- Classification of negative/positive sentiments as neutral

```
("[CLS] @ USER I put'Do not take out or upload a 2nd time ( for example Youtube, Vimeo, Tudou, Youku, etc )'its the same thing. [SEP]", tensor(0), tensor(2))
```

Less problematic but still misclassified are the sentences in which neutral sentiment is classified as negative or positive. For instance, the sentence above was classified as neutral, but it should have been classified as negative due to the negation “Do not”. Nevertheless, this misclassification is understandable, because despite the negation, the sentence is neutral in terms of the overall message.

```
('[CLS] Robbed by George Osborne... while the royals play decoy | Kevin McKenna @ URL [SEP]', tensor(0), tensor(2))
```

```
('[CLS] The Borno state police commissioner, Aderemi Opadokun, on Monday confirmed fresh attacks by the Boko Haram sect... @ URL [SEP]', tensor(0), tensor(2))
```

In other cases, such as the latter two sentences, the misclassification is less understandable, because the negatively connoted words such as "robbed" or "attacks", "Boko Haram sect" stand out directly due to their negative connotations and are also not weakened by other positive components in the sentence.

```
("[CLS] In celebration of National Women's Day on 9th of August, Poise Brands, in association with Body Sense Boutique... @ URL [SEP]", tensor(1), tensor(2))
```

Similarly, positive sentences were often classified as neutral, such as in the example above, which is a neutral description of a fact. However, the word "celebration" stands out because of its positive connotation, so the sentence should be classified as positive. However, again, this is also a misclassification to a weaker extent, since otherwise the content of the statement is very factual and everything with the exception of this word is rather neutral.

```
('[CLS] Sun. is National Ice Cream Day! Celebrate at @ USER with FREE sundae toppings from 12noon - 5pm. @ URL [SEP]', tensor(1), tensor(2))
```

Of course, there are also more extreme cases of this kind of misclassification, such as the sentence above, which was classified as neutral despite not just one, but several clearly positive sentiments such as “Celebrate” und “FREE sundae toppings”. However, the reason for this could be a problem in dealing with the word "Ice Cream Day", which occurred in many sentences and the respective sentence was almost always considered positive even though it was mostly labelled as neutral. Additionally, these kinds of announcement tweets with location and time were mostly labelled as neutral in the data set.

- Classification of neutral sentiments as negative/positive

```
("[CLS] Sunday is just not Dustin Johnson's day @ URL [SEP]",  
tensor(2), tensor(0))
```

```
('[CLS] @ USER @ USER @ USER Factual error : Army chief did not  
spend Eid night there as he was in Pindi for Open House Sat  
evening. [SEP]', tensor(2), tensor(0))
```

Probably the most common case was that neutral sentiments were classified as positive or negative because often parts, even if only single words in the sentence, carry a negative or positive meaning as the example sentences show. The above sentences are neutral in themselves, but the negation “not” seem to be misleading and cause misclassification as negative because, apart from this negation and the word “error”, the sentences are clearly neutral and there is nothing to indicate negative (or positive) sentiment. This source of error can be observed in most of the neutral sentences classified as negative.

```
('[CLS] Hate to say Randy Orton is underrated but too many people  
sleep / shit on him. Had he been in Cenas spot Wed have seen 10  
yrs of great matches [SEP]', tensor(2), tensor(0))
```

In other sentences, several words with negative connotations appear, including not only negatives but also, for example, "hate" and "shit", as the above sentence shows, which is misleading and makes classification difficult and not necessarily clear even to people at first glance. On closer inspection, however, it becomes clear that despite the negative words, the sentence itself has a neutral message, as the person tweeting is merely reflecting other people's bad opinion of Randy Orton and not his own.

```
('[CLS] Be at Ten Thousand Villages this Thurs 4 - 8pm. Choc  
tasting, henna tattoos, and proceeds go to support refugees in the  
Triangle! [SEP]', tensor(2), tensor(1))
```

In other cases, neutral sentences have been classified as positive, the analysis of which seemed less straightforward than in the case of neutral sentiments classified as negative. However, in some examples it was easier to draw conclusions, for example in the case of the sentence above which was rated as positive overall because of “support refugees”, which has a strong positive connotation, and the affirmative nature of the sentence itself.

```
('[CLS] im trying the sydney dalton diet starting monday : ) ) ) )  
) ) [SEP]', tensor(2), tensor(1))
```

A completely different phenomenon that has not yet been taken into account are emojis and other acronyms that are typical for netspeak. In the example above, a neutral sentence is classified as 'positive'. The sentence is affirmative but its content is neutral and what may have caused the model to give the sentence a positive sentiment could be the smiley at the end.

5 Conclusion (short)

Working with google colab provided access to additional resources (GPU), though the online functions limited the efficiency: uploading an existing model for testing took about 30 minutes (433MB) and if the connection to the server was interrupted, the whole process was corrupted. Additionally, a GPU was not always available, resulting in much longer run times. The advantages outweigh the curtailments, though adapting this gigantic model with 110 million parameters to this task was quite a tedious and frustrating process. The results we obtained might seem comparable to the ones of the first assignment (qualitatively), though this classification was much more difficult than the binary one. Additionally, the tweet messages are very challenging, and we could not always follow some specific gold label decisions.

The model had problems classifying irony, sarcasm and not literal speech. With its lack of "understanding" negation, it is disputable whether this can be accounted as semantic knowledge. Nonetheless, the model showcases good results with its powerful mechanism and word representations, which are applicable to completely new datasets. It can extract the most important points out of the tweets without any feature definition and it is only occasionally tricked by its classification approaches to wrongly assign a sentiment.

6 Resources *(Please list all the external resources that you have used, together with a link (sentiment lexicons, models, code snippets, ...)).*

Code: <https://towardsdatascience.com/text-classification-with-bert-in-pytorch-887965e5820f>

Models:

- pre-trained BERT model (base) from Hugging Face
- BERTTokenizer from bert-base-cased model

Data:

- SemEval2013 train/testset <https://www.kaggle.com/azzouza2018/semevaldatadets>
- SemEval2016 test set: [https://www.dropbox.com/s/byzr8yoda6bua1b/2017_English_final.zip?](https://www.dropbox.com/s/byzr8yoda6bua1b/2017_English_final.zip?Path:/2017_English_final/GOLD/Subtask_A/twitter-2016test-A.txt)
Path: /2017_English_final/GOLD/Subtask_A/twitter-2016test-A.txt

Papers:

- [Devlin et al. \(2019\)](#): BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
- [Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020](#). A primer in bertology: What we know about how bert works.
- [Alsaeedi, Abdullah. \(2019\)](#). EFTSA: Evaluation Framework for Twitter Sentiment Analysis. Journal of Software. 24-35. 10.17706/jsw.14.1.24-35.

7 Appendix

We report some first model architectures, developments and results obtained in the following. Only looking at the accuracy/F1 scores was not efficient enough to decide on the final model and moreover when to stop training. Due to the unbalanced data set, the negative labels posed a challenge in this task. Our [Notebook](#)

Train data set, Drop: 0.1, batch 7 lr 1e-5

100% ██████████ | 1099/1099 [14:07<00:00, 1.30it/s]
 Epochs: 1 | Train Loss: 0.114 | Train Accuracy: 0.642 | Val Loss: 0.087 | Val Accuracy: 0.759
 100% ██████████ | 1099/1099 [14:07<00:00, 1.30it/s]
 Epochs: 2 | Train Loss: 0.068 | Train Accuracy: 0.829 | Val Loss: 0.106 | Val Accuracy: 0.736
 100% ██████████ | 1099/1099 [14:07<00:00, 1.30it/s]
 Epochs: 3 | Train Loss: 0.042 | Train Accuracy: 0.921 | Val Loss: 0.145 | Val Accuracy: 0.761
 100% ██████████ | 1099/1099 [14:11<00:00, 1.29it/s]
 Epochs: 4 | Train Loss: 0.025 | Train Accuracy: 0.961 | Val Loss: 0.176 | Val Accuracy: 0.758

E0	E1	E2	E3
Test Accuracy: 0.778 Recall: 0.778 Precision: 0.786 F1_Score: 0.778 F2_Score: 0.779 Confusion Matrix: [[537 49] [107 269]] [[393 123] [65 381]] [[780 42] [42 98]]]	Test Accuracy: 0.745 Recall: 0.745 Precision: 0.760 F1_Score: 0.746 F2_Score: 0.748 Confusion Matrix: [[456 130] [56 320]] [[453 63] [149 297]] [[770 52] [40 100]]]	Test Accuracy: 0.781 Recall: 0.781 Precision: 0.783 F1_Score: 0.781 F2_Score: 0.781 Confusion Matrix: [[517 69] [80 296]] [[411 105] [81 365]] [[785 37] [50 90]]]	Test Accuracy: 0.777 Recall: 0.777 Precision: 0.780 F1_Score: 0.778 F2_Score: 0.777 Confusion Matrix: [[516 70] [82 294]] [[407 109] [81 365]] [[786 36] [52 88]]]
LABEL: 2 as neutral Class Accuracy: 0.854 ***assigned as class: 0 %: 6.502 ***assigned as class: 1 %: 8.072	LABEL: 2 as neutral Class Accuracy: 0.666 ***assigned as class: 1 %: 25.112 ***assigned as class: 0 %: 8.296	LABEL: 2 as neutral Class Accuracy: 0.818 ***assigned as class: 0 %: 6.054 ***assigned as class: 1 %: 12.108	LABEL: 2 as neutral Class Accuracy: 0.818 ***assigned as class: 0 %: 5.830 ***assigned as class: 1 %: 12.332
LABEL: 1 as positive Class Accuracy: 0.715 ***assigned as class: 2 %: 25.000 ***assigned as class: 0 %: 3.457	LABEL: 1 as positive Class Accuracy: 0.851 ***assigned as class: 0 %: 3.989 ***assigned as class: 2 %: 10.904	LABEL: 1 as positive Class Accuracy: 0.787 ***assigned as class: 2 %: 18.617 ***assigned as class: 0 %: 2.660	LABEL: 1 as positive Class Accuracy: 0.782 ***assigned as class: 2 %: 19.149 ***assigned as class: 0 %: 2.660
LABEL: 0 as negative Class Accuracy: 0.700 ***assigned as class: 2 %: 20.714 ***assigned as class: 1 %: 9.286	LABEL: 0 as negative Class Accuracy: 0.714 ***assigned as class: 2 %: 15.714 ***assigned as class: 1 %: 12.857	LABEL: 0 as negative Class Accuracy: 0.643 ***assigned as class: 2 %: 25.000 ***assigned as class: 1 %: 10.714	LABEL: 0 as negative Class Accuracy: 0.629 ***assigned as class: 2 %: 26.429 ***assigned as class: 1 %: 10.714

Note: Although a great accuracy is achieved, the results are shallow for the negative label.

Train data set, Drop: 0.5, batch 7 lr 1e-5

```

100%|██████████| 1099/1099 [14:12<00:00, 1.29it/s]
Epochs: 1 | Train Loss: 0.107 | Train Accuracy: 0.681 | Val Loss: 0.089
| Val Accuracy: 0.764
100%|██████████| 1099/1099 [14:14<00:00, 1.29it/s]
Epochs: 2 | Train Loss: 0.068 | Train Accuracy: 0.840 | Val Loss: 0.103
| Val Accuracy: 0.758
100%|██████████| 1099/1099 [14:17<00:00, 1.28it/s]
Epochs: 3 | Train Loss: 0.042 | Train Accuracy: 0.924 | Val Loss: 0.150
| Val Accuracy: 0.759
100%|██████████| 1099/1099 [14:16<00:00, 1.28it/s]
Epochs: 4 | Train Loss: 0.024 | Train Accuracy: 0.961 | Val Loss: 0.181
| Val Accuracy: 0.735

```

E0	E1	E2	E3
Test Accuracy: 0.772 Recall: 0.772 Precision: 0.776 F1_Score: 0.771 F2_Score: 0.773 Confusion Matrix: [[[538 48] [104 272]] [[[404 112] [81 365]] [[[763 59] [34 106]]]] LABEL: 2 as neutral Class Accuracy: 0.818 ***assigned as class: 1 %: 8.744 ***assigned as class: 0 %: 9.417 LABEL: 1 as positive Class Accuracy: 0.723 ***assigned as class: 2 %: 23.138 ***assigned as class: 0 %: 4.521 LABEL: 0 as negative Class Accuracy: 0.757 ***assigned as class: 2 %: 17.857 ***assigned as class: 1 %: 6.429	Test Accuracy: 0.772 Recall: 0.772 Precision: 0.773 F1_Score: 0.773 F2_Score: 0.773 Confusion Matrix: [[[501 85] [74 302]] [[[418 98] [101 345]] [[[786 36] [44 96]]]] LABEL: 2 as neutral Class Accuracy: 0.774 ***assigned as class: 1 %: 16.816 ***assigned as class: 0 %: 5.830 LABEL: 1 as positive Class Accuracy: 0.803 ***assigned as class: 2 %: 17.021 ***assigned as class: 0 %: 2.660 LABEL: 0 as negative Class Accuracy: 0.686 ***assigned as class: 2 %: 24.286 ***assigned as class: 1 %: 7.143	Test Accuracy: 0.769 Recall: 0.769 Precision: 0.769 F1_Score: 0.769 F2_Score: 0.769 Confusion Matrix: [[[513 73] [85 291]] [[[418 98] [100 346]] [[[771 51] [37 103]]]] LABEL: 2 as neutral Class Accuracy: 0.776 ***assigned as class: 1 %: 14.126 ***assigned as class: 0 %: 8.296 LABEL: 1 as positive Class Accuracy: 0.774 ***assigned as class: 2 %: 18.883 ***assigned as class: 0 %: 3.723 LABEL: 0 as negative Class Accuracy: 0.736 ***assigned as class: 2 %: 19.286 ***assigned as class: 1 %: 7.143	Test Accuracy: 0.742 Recall: 0.742 Precision: 0.745 F1_Score: 0.742 F2_Score: 0.743 Confusion Matrix: [[[487 99] [76 300]] [[[427 89] [132 314]] [[[762 60] [40 100]]]] LABEL: 2 as neutral Class Accuracy: 0.704 ***assigned as class: 1 %: 19.955 ***assigned as class: 0 %: 9.641 LABEL: 1 as positive Class Accuracy: 0.798 ***assigned as class: 2 %: 15.691 ***assigned as class: 0 %: 4.521 LABEL: 0 as negative Class Accuracy: 0.714 ***assigned as class: 2 %: 21.429 ***assigned as class: 1 %: 7.143

Note: The standard drop out rate (0.5) prevented overfitting of the model on the training set. Nevertheless: class accuracy of the negative label was below 75%, one of our demands.

Train data set CLEANED @USER, @LINK, Drop: 0.5, batch 7 lr 1e-5

100% ██████████ | 1099/1099 [13:31<00:00, 1.35it/s]
 Epochs: 1 | Train Loss: 0.106 | Train Accuracy: 0.675 | Val Loss: 0.091 | Val Accuracy: 0.751
 100% ██████████ | 1099/1099 [13:32<00:00, 1.35it/s]
 Epochs: 2 | Train Loss: 0.067 | Train Accuracy: 0.845 | Val Loss: 0.112 | Val Accuracy: 0.744
 100% ██████████ | 1099/1099 [13:32<00:00, 1.35it/s]
 Epochs: 3 | Train Loss: 0.041 | Train Accuracy: 0.927 | Val Loss: 0.155 | Val Accuracy: 0.767
 100% ██████████ | 1099/1099 [13:33<00:00, 1.35it/s]
 Epochs: 4 | Train Loss: 0.024

E0	E1	E2	E3
Test Accuracy: 0.757 Recall: 0.757 Precision: 0.763 F1_Score: 0.753 F2_Score: 0.758 Confusion Matrix: [[[539 47] [106 270]] [[417 99] [102 344]] [[734 88] [26 114]]]	Test Accuracy: 0.757 Recall: 0.757 Precision: 0.775 F1_Score: 0.759 F2_Score: 0.760 Confusion Matrix: [[[455 131] [45 331]] [[453 63] [145 301]] [[782 40] [44 96]]]	Test Accuracy: 0.774 Recall: 0.774 Precision: 0.782 F1_Score: 0.775 F2_Score: 0.776 Confusion Matrix: [[[543 43] [109 267]] [[392 124] [68 378]] [[772 50] [40 100]]]	Test Accuracy: 0.756 Recall: 0.756 Precision: 0.759 F1_Score: 0.756 F2_Score: 0.756 Confusion Matrix: [[[481 105] [67 309]] [[438 78] [124 322]] [[770 52] [44 96]]]
LABEL: 2 as neutral Class Accuracy: 0.771 ***assigned as class: 1 %: 8.520 ***assigned as class: 0 %: 14.350	LABEL: 2 as neutral Class Accuracy: 0.675 ***assigned as class: 1 %: 25.561 ***assigned as class: 0 %: 6.951	LABEL: 2 as neutral Class Accuracy: 0.848 ***assigned as class: 1 %: 7.623 ***assigned as class: 0 %: 7.623	LABEL: 2 as neutral Class Accuracy: 0.722 ***assigned as class: 0 %: 8.072 ***assigned as class: 1 %: 19.731
LABEL: 1 as positive Class Accuracy: 0.718 ***assigned as class: 2 %: 21.809 ***assigned as class: 0 %: 6.383	LABEL: 1 as positive Class Accuracy: 0.880 ***assigned as class: 0 %: 2.394 ***assigned as class: 2 %: 9.574	LABEL: 1 as positive Class Accuracy: 0.710 ***assigned as class: 2 %: 24.734 ***assigned as class: 0 %: 4.255	LABEL: 1 as positive Class Accuracy: 0.822 ***assigned as class: 0 %: 4.255 ***assigned as class: 2 %: 13.564
LABEL: 0 as negative Class Accuracy: 0.814 ***assigned as class: 2 %: 12.143 ***assigned as class: 1 %: 6.429	LABEL: 0 as negative Class Accuracy: 0.686 ***assigned as class: 2 %: 19.286 ***assigned as class: 1 %: 12.143	LABEL: 0 as negative Class Accuracy: 0.714 ***assigned as class: 2 %: 22.143 ***assigned as class: 1 %: 6.4293	LABEL: 0 as negative Class Accuracy: 0.686 ***assigned as class: 2 %: 19.286 ***assigned as class: 1 %: 12.143

Note: Bad gradients..

Train data set CLEANED @USER, @LINK, Drop: 0.1, batch 7 lr 1e-5

100% ██████████ | 1099/1099 [14:29<00:00, 1.26it/s]
 Epochs: 1 | Train Loss: 0.100 | Train Accuracy: 0.724 | Val Loss: 0.093 | Val Accuracy: 0.760
 100% ██████████ | 1099/1099 [14:27<00:00, 1.27it/s]
 Epochs: 2 | Train Loss: 0.073 | Train Accuracy: 0.853 | Val Loss: 0.113 | Val Accuracy: 0.746
 100% ██████████ | 1099/1099 [14:27<00:00, 1.27it/s]
 Epochs: 3 | Train Loss: 0.052 | Train Accuracy: 0.916 | Val Loss: 0.145 | Val Accuracy: 0.754
 100% ██████████ | 1099/1099 [14:24<00:00, 1.27it/s]
 Epochs: 4 | Train Loss: 0.040 | Train Accuracy: 0.942 | Val Loss: 0.194 | Val Accuracy: 0.751

E0	E1	E2	E3
Test Accuracy: 0.766 Recall: 0.766 Precision: 0.788 F1_Score: 0.770 F2_Score: 0.770 Confusion Matrix: [[[542 44] [104 272]] [[368 148] [55 391]] [[789 33] [66 74]]]	Test Accuracy: 0.774 Recall: 0.774 Precision: 0.775 F1_Score: 0.774 F2_Score: 0.775 Confusion Matrix: [[[507 79] [71 305]] [[436 80] [112 334]] [[764 58] [34 106]]]	Test Accuracy: 0.777 Recall: 0.777 Precision: 0.782 F1_Score: 0.777 F2_Score: 0.778 Confusion Matrix: [[[536 50] [100 276]] [[398 118] [71 375]] [[775 47] [44 96]]]	Test Accuracy: 0.754 Recall: 0.754 Precision: 0.762 F1_Score: 0.756 F2_Score: 0.755 Confusion Matrix: [[[489 97] [71 305]] [[406 110] [103 343]] [[792 30] [63 77]]]
LABEL: 2 as neutral Class Accuracy: 0.877 ***assigned as class: 1 %: 7.175 ***assigned as class: 0 %: 5.157	LABEL: 2 as neutral Class Accuracy: 0.749 ***assigned as class: 1 %: 15.247 ***assigned as class: 0 %: 9.865	LABEL: 2 as neutral Class Accuracy: 0.841 ***assigned as class: 1 %: 8.744 ***assigned as class: 0 %: 7.175	LABEL: 2 as neutral Class Accuracy: 0.769 ***assigned as class: 1 %: 17.713 ***assigned as class: 0 %: 5.381
LABEL: 1 as positive Class Accuracy: 0.723 ***assigned as class: 2 %: 25.000 ***assigned as class: 0 %: 2.660	LABEL: 1 as positive Class Accuracy: 0.811 ***assigned as class: 2 %: 15.160 ***assigned as class: 0 %: 3.723	LABEL: 1 as positive Class Accuracy: 0.734 ***assigned as class: 2 %: 22.606 ***assigned as class: 0 %: 3.989	LABEL: 1 as positive Class Accuracy: 0.811 ***assigned as class: 2 %: 17.287 ***assigned as class: 0 %: 1.596
LABEL: 0 as negative Class Accuracy: 0.529 ***assigned as class: 2 %: 38.571 ***assigned as class: 1 %: 8.571	LABEL: 0 as negative Class Accuracy: 0.757 ***assigned as class: 2 %: 16.429 ***assigned as class: 1 %: 7.857	LABEL: 0 as negative Class Accuracy: 0.686 ***assigned as class: 2 %: 23.571 ***assigned as class: 1 %: 7.857	LABEL: 0 as negative Class Accuracy: 0.550 ***assigned as class: 2 %: 32.143 ***assigned as class: 1 %: 12.857

Note: our final model..! :)