# Implementing an efficient approach in Sentiment Analysis using BERT and SqueezeBERT

Saurabh Kant (2020IMT-089)

Supervisor: Dr.W Wilfred Godfrey

ABV-Indian Institute of Information Technology and Management Gwalior

विश्वजीवनामृतं ज्ञानम्

## Outline

## Introduction

- Nowadays, Sentiment Analysis is considered one of the most significant sub-areas in Natural Language Processing (NLP) research.
- Sentiment analysis examines people's opinions, attitudes, and emotions computationally through texts or reviews of items. Whether the text is a document, paragraph, or phrase, this sort of analysis may be used to discover sentiment polarity (e.g., positive or negative opinion).
- Sentiment analysis enabled consumers to communicate their ideas and feelings more openly, and understanding people's emotions is now a critical topic for businesses.
- Businesses nowadays are becoming customer-centric, so brands that can actively listen to their consumers and customize products and services to fit their requirements by automatically evaluating client input from survey replies to social media chats or reviews from customers' side are growing. Besides businesses, this technology is also used by social media networks for filtering out offensive content and understanding user behavior.
- The pre-training process is costly and requires super-computers for the training but the fine-tuning of BERT and BERT-like models can be done using entry-level hardware.

## Motivation

- Globally, the quantity of data generated, collected, copied, and consumed is expected to grow quickly, reaching 120 zettabytes by 2023. Global data production is expected to reach more than 180 zettabytes in the following five years, up to 2025.

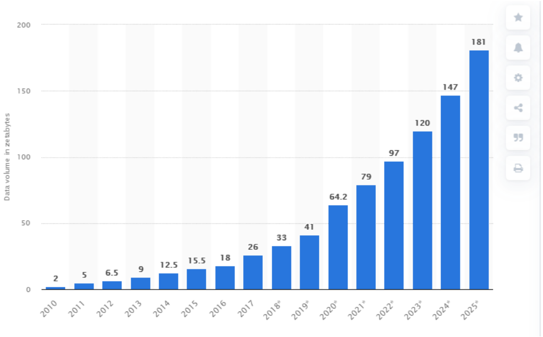


Figure 1: Data created, consumed, and stored 2010-2025

## Motivation

- Social media users tend to consume most of the content on their smartphones and laptops. Because most textual data is created and viewed on mobile devices, deploying the NLP models on the same mobile devices makes sense. So that when a person receives a text message, NLP models can classify those messages based on emotions, priority, context, etc.

- This increasing data requires better machine learning algorithms and methods for extracting useful information because the processing power is not growing at the same rate, and that would mean that we will have to device methods that use our resources efficiently and can get more insights from the same dataset.

- We have recently seen some great metrics scored by various implementations of multi-head self-attention modules Vaswani et al. (2017) in several NLP tasks. The GPT model and the later developed BERT modelDevlin et al.(2019) have given state-of-the-art scores in solving many NLP problems. Since then, other variations of BERT like ROBERTa, DISTILBERT, and SqueezeBERT have been given to improve on shortcomings of the original model. One problem in using BERT for the current tasks and using it to generate valuable results is its slow speed in giving predictions; several newer alternatives have tried to tackle this problem and, in this project, I will be using one of the newer methods called SqueezeBERT for making a model that does Multilabel Sentiment Analysis.[1]

## Research Gap

- BERT requires a large amount of computational resources and memory to fine-tune and run, which makes it difficult to deploy on low-end devices.[2]
- BERT is sensitive to the choice of hyperparameters such as learning rate, batch size, number of epochs, etc. To achieve optimal performance, these hyperparameters must be carefully tuned for each task and dataset.[3]
- BERT may not be able to capture the sentiment of some complex or ambiguous sentences, especially when they involve sarcasm, irony, humor, or negation. For example, BERT may misclassify the sentence "I love this product so much that I want to throw it away" as positive, while it is negative.
- BERT may not be able to generalize well to new domains or languages that are different from the ones it was trained on. For example, BERT may perform poorly on sentiment analysis of social media posts that contain slang, emojis, hashtags, or misspellings. Similarly, BERT may be unable to handle languages with different word order, morphology, or syntax than English.

## Research Objective

1 To research and analyze previously proposed models and algorithms to better understand the topic.

2 Collect real-world datasets for training and testing.

3 Doing pre-processing on the raw data to prepare it for Natural Language Processing models.

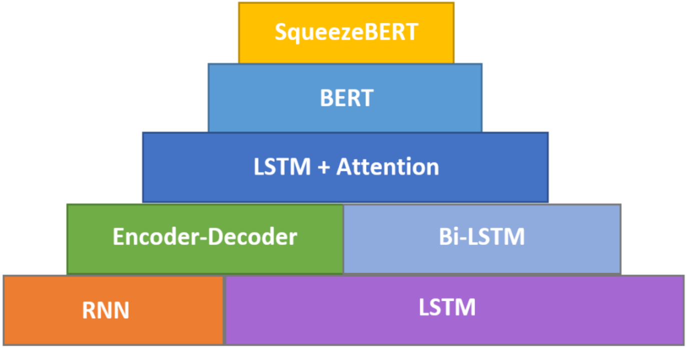
4 Utilizing BERT and SqueezeBERT models for solving multi-label sentiment analysis.

5 Fine-tuning the pre-trained model parameters to get the best performance

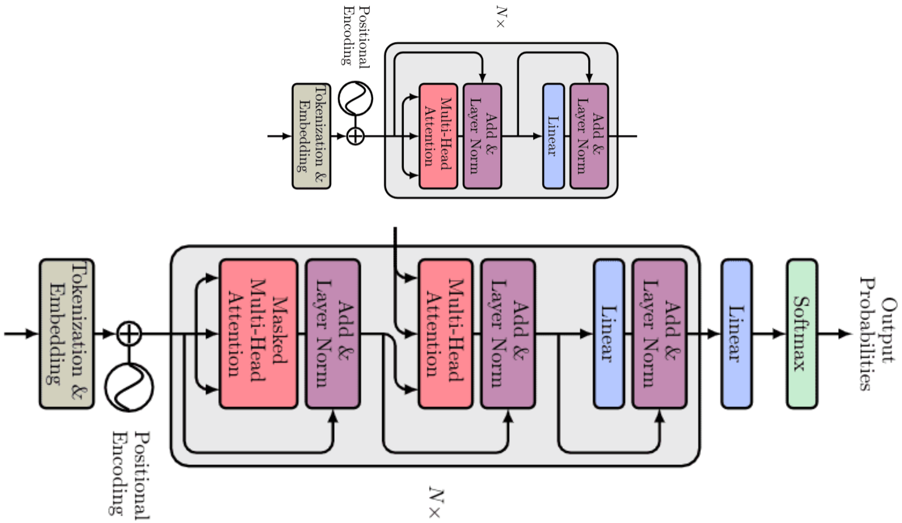
6 Identifying and implementing metrics for measuring the efficiency and accuracy of all models.

7 Generate a report for the complete process with the results of the thesis.

## Models Previously Used



## Transformers



## Working of BERT

- BERT stands for Bidirectional Encoder Representations from Transformers, a state-of-the-art NLP model introduced by Google researchers in 2018.
- The building process starts at transformers, so the transformer architecture consists of two parts: encoder + decoder. In BERT we take the encoder part and stack multiple encoders one after another (12 layers). This encoder stack is then taken and trained heavily on "fake tasks." The innovation part of what Google researchers did was figuring out what tasks to train BERT on, and the second thing they did was train it heavily. These "fake tasks" I mentioned are fake because we don't care if BERT performs well in these tasks. We want to learn how human language works by working on them. The BERT is pre-trained on the Masked Language Model and Next Sentence Prediction (Bert paper).
- So, the BERT layers were first trained on MLM, then the final output layers were changed, and it was again trained on the next sentence prediction. This is similar to fine-tuning; we take the pre-trained BERT and change the last layer to do specific tasks using it. Using a pre-trained model increases the scores on metrics because the previous tasks have taught BERT how to read the English language by training on very large datasets and fine-tuning is like teaching it to simple tasks based on that learned language, like telling what sentiments does a sentence wants to convey.
- BERT encoder block, self-attention is the concept which enables parallelization in transformers.

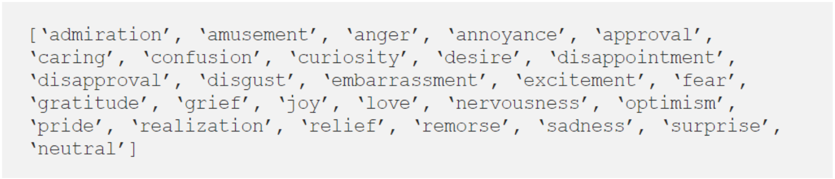
## Working of SqueezeBERT

- The SqueezeBERT model works almost the same as the BERT model
- SqueezeBERT brings the concept of convolution from computer vision to natural language processing.The SqueezeBERT model has implemented the idea of grouped convolutions in place of position-wise fully connected (PFC) layers used for generating Q, K, and V vectors.
- By implementing input vectors in form of group convolutions the SqueezeBERT model is able to change the Q, K, V and two other feed-forward layers to groups - 4 instead of groups - 1, which in turn increases the efficiency with which the model can run the self-attention process.
- The SqueezeBERT model is trained using SqueezeBERT uncased pretrained algorithm, which is a modified version of the BERT algorithm that is designed to be more efficient and faster. It does this by replacing the pointwise fully-connected layers in the BERT model with grouped convolutions. Grouped convolutions allow the model to learn more efficiently by reducing the number of parameters that need to be trained.

## Dataset

- The dataset I have chosen for this project is "GoEmotions" from huggingface.co. The reason for choosing this dataset is that it contains 58,000 comments, which are carefully labeled into 28 categories. The dataset is in English language and because it is not a ratings and review dataset, it contains a wide range of human emotions required for training a multi-label classification model. The various emotions it contains are :
- The dataset have been split into the following segments -

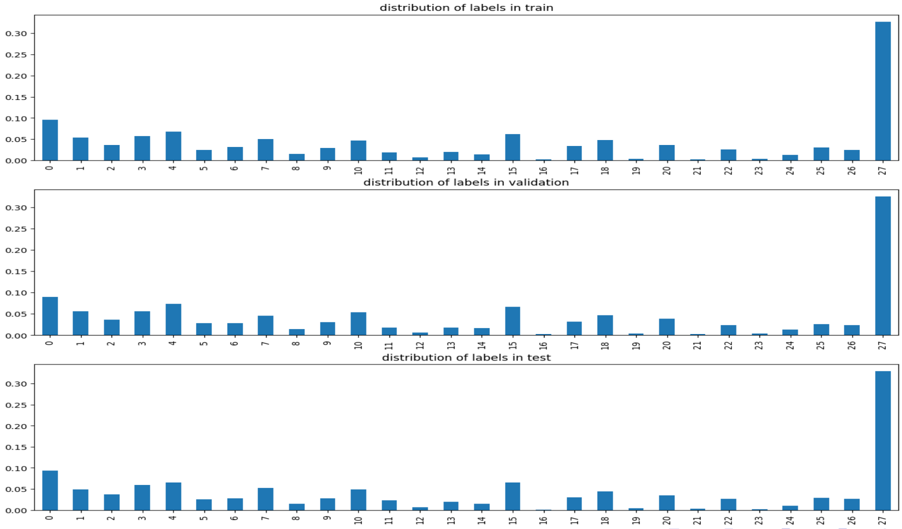
1. Training - 43410 entries.
2. Validation - 5426 entries.
3. Test - 5427 entries



```
['admiration', 'amusement', 'anger', 'annoyance', 'approval',
'caring', 'confusion', 'curiosity', 'desire', 'disappointment',
'disapproval', 'disgust', 'embarrassment', 'excitement', 'fear',
'gratitude', 'grief', 'joy', 'love', 'nervousness', 'optimism',
'pride', 'realization', 'relief', 'remorse', 'sadness', 'surprise',
'neutral']
```

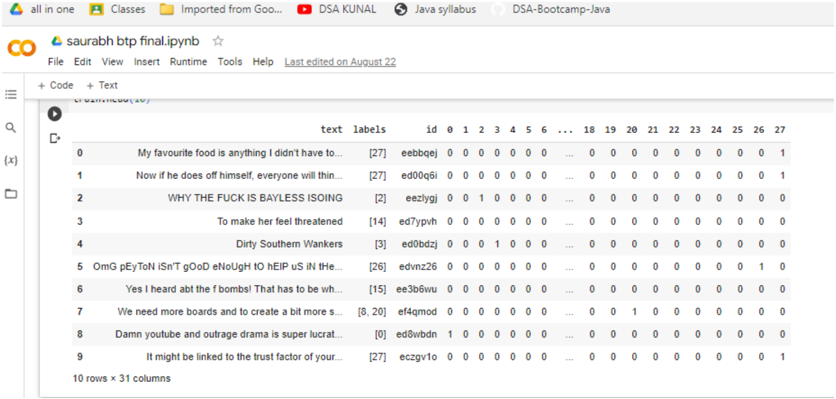
Figure 3: 28 emotions from the GoEmotions dataset

## Consistent label distribution in training, test and validation splits



## Results

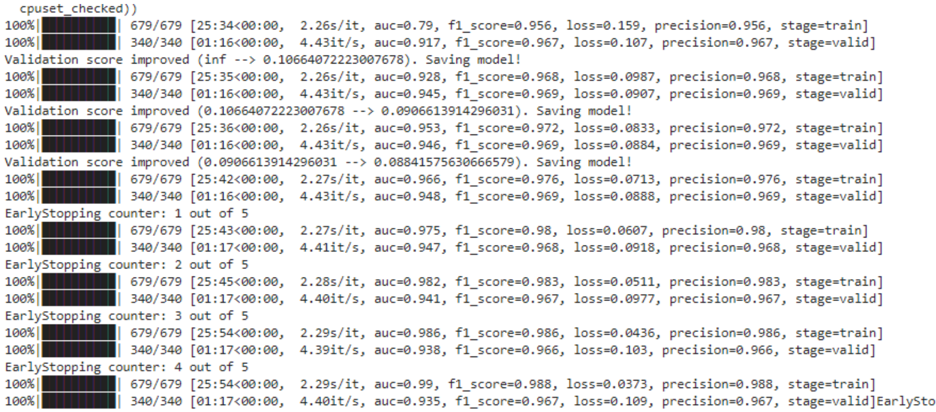
- Comment sample we have taken from the dataset



| | text | labels | id | 0 | 1 | 2 | 3 | 4 | 5 | 6 | ... | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | My favourite food is anything I didn't have to... | [27] | eebbqej | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | Now if he does off himself, everyone will thin... | [27] | ed00q6i | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 2 | WHY THE FUCK IS BAYLESS ISOING | [2] | eezlygj | 0 | 0 | 1 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | To make her feel threatened | [14] | ed7ypvh | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | Dirty Southern Wankers | [3] | ed0bdzj | 0 | 0 | 0 | 1 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | OmG pEyToN iSn'T gOoD eNoUgH tO hElP uS iN tHe... | [26] | edvnz26 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 6 | Yes I heard abt the f bombs! That has to be wh... | [15] | ee3b6wu | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | We need more boards and to create a bit more s... | [8, 20] | ef4qmod | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | Damn youtube and outrage drama is super lucrat... | [0] | ed8wbdn | 1 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | It might be linked to the trust factor of your... | [27] | eczgv1o | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

10 rows × 31 columns

## BERT MODEL OUTPUT

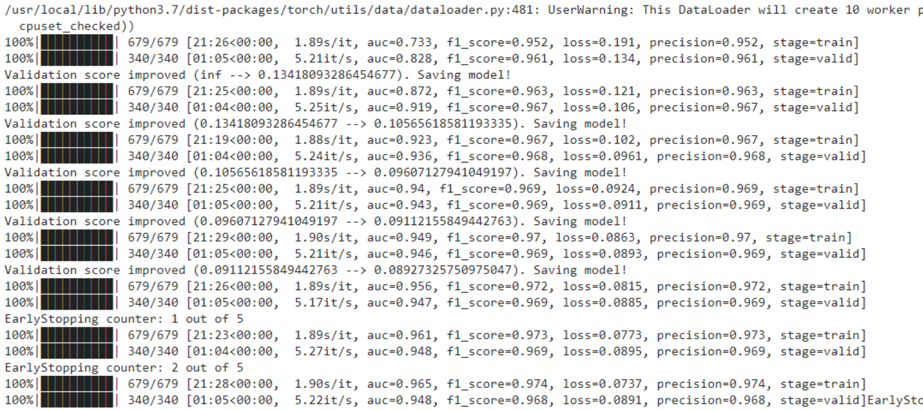


```
  cpuset_checked))
100%|██████████| 679/679 [25:34<00:00,  2.26s/it, auc=0.79, f1_score=0.956, loss=0.159, precision=0.956, stage=train]
100%|██████████| 340/340 [01:16<00:00,  4.43it/s, auc=0.917, f1_score=0.967, loss=0.107, precision=0.967, stage=valid]
Validation score improved (inf --> 0.10664072223007678). Saving model!
100%|██████████| 679/679 [25:35<00:00,  2.26s/it, auc=0.928, f1_score=0.968, loss=0.0987, precision=0.968, stage=train]
100%|██████████| 340/340 [01:16<00:00,  4.43it/s, auc=0.945, f1_score=0.969, loss=0.0907, precision=0.969, stage=valid]
Validation score improved (0.10664072223007678 --> 0.0906613914296031). Saving model!
100%|██████████| 679/679 [25:36<00:00,  2.26s/it, auc=0.953, f1_score=0.972, loss=0.0833, precision=0.972, stage=train]
100%|██████████| 340/340 [01:16<00:00,  4.43it/s, auc=0.946, f1_score=0.969, loss=0.0884, precision=0.969, stage=valid]
Validation score improved (0.0906613914296031 --> 0.08841575630666579). Saving model!
100%|██████████| 679/679 [25:42<00:00,  2.27s/it, auc=0.966, f1_score=0.976, loss=0.0713, precision=0.976, stage=train]
100%|██████████| 340/340 [01:16<00:00,  4.43it/s, auc=0.948, f1_score=0.969, loss=0.0888, precision=0.969, stage=valid]
EarlyStopping counter: 1 out of 5
100%|██████████| 679/679 [25:43<00:00,  2.27s/it, auc=0.975, f1_score=0.98, loss=0.0607, precision=0.98, stage=train]
100%|██████████| 340/340 [01:17<00:00,  4.41it/s, auc=0.947, f1_score=0.968, loss=0.0918, precision=0.968, stage=valid]
EarlyStopping counter: 2 out of 5
100%|██████████| 679/679 [25:45<00:00,  2.28s/it, auc=0.982, f1_score=0.983, loss=0.0511, precision=0.983, stage=train]
100%|██████████| 340/340 [01:17<00:00,  4.40it/s, auc=0.941, f1_score=0.967, loss=0.0977, precision=0.967, stage=valid]
EarlyStopping counter: 3 out of 5
100%|██████████| 679/679 [25:54<00:00,  2.29s/it, auc=0.986, f1_score=0.986, loss=0.0436, precision=0.986, stage=train]
100%|██████████| 340/340 [01:17<00:00,  4.39it/s, auc=0.938, f1_score=0.966, loss=0.103, precision=0.966, stage=valid]
EarlyStopping counter: 4 out of 5
100%|██████████| 679/679 [25:54<00:00,  2.29s/it, auc=0.99, f1_score=0.988, loss=0.0373, precision=0.988, stage=train]
100%|██████████| 340/340 [01:17<00:00,  4.40it/s, auc=0.935, f1_score=0.967, loss=0.109, precision=0.967, stage=valid]EarlySto
```

## BERT MODEL OUTPUT

- Trained Model Size: 1220+ MB
- Total Training Time: 216 mins
- Average Training Time per Epoch : 27 mins
- ROC Area Under Curve Score : 0.935
- Micro F1 Score : 0.967

## SqueezeBERT MODEL OUTPUT



```
/usr/local/lib/python3.7/dist-packages/torch/utils/data/dataloader.py:481: UserWarning: This DataLoader will create 10 worker p
  cpuset_checked))
100%|██████████| 679/679 [21:26<00:00,  1.89s/it, auc=0.733, f1_score=0.952, loss=0.191, precision=0.952, stage=train]
100%|██████████| 340/340 [01:05<00:00,  5.21it/s, auc=0.828, f1_score=0.961, loss=0.134, precision=0.961, stage=valid]
Validation score improved (inf --> 0.13418093286454677). Saving model!
100%|██████████| 679/679 [21:25<00:00,  1.89s/it, auc=0.872, f1_score=0.963, loss=0.121, precision=0.963, stage=train]
100%|██████████| 340/340 [01:04<00:00,  5.25it/s, auc=0.919, f1_score=0.967, loss=0.106, precision=0.967, stage=valid]
Validation score improved (0.13418093286454677 --> 0.10565618581193335). Saving model!
100%|██████████| 679/679 [21:19<00:00,  1.88s/it, auc=0.923, f1_score=0.967, loss=0.102, precision=0.967, stage=train]
100%|██████████| 340/340 [01:04<00:00,  5.24it/s, auc=0.936, f1_score=0.968, loss=0.0961, precision=0.968, stage=valid]
Validation score improved (0.10565618581193335 --> 0.09607127941049197). Saving model!
100%|██████████| 679/679 [21:25<00:00,  1.89s/it, auc=0.94, f1_score=0.969, loss=0.0924, precision=0.969, stage=train]
100%|██████████| 340/340 [01:05<00:00,  5.21it/s, auc=0.943, f1_score=0.969, loss=0.0911, precision=0.969, stage=valid]
Validation score improved (0.09607127941049197 --> 0.09112155849442763). Saving model!
100%|██████████| 679/679 [21:29<00:00,  1.90s/it, auc=0.949, f1_score=0.97, loss=0.0863, precision=0.97, stage=train]
100%|██████████| 340/340 [01:05<00:00,  5.21it/s, auc=0.946, f1_score=0.969, loss=0.0893, precision=0.969, stage=valid]
Validation score improved (0.09112155849442763 --> 0.08927325750975047). Saving model!
100%|██████████| 679/679 [21:26<00:00,  1.89s/it, auc=0.956, f1_score=0.972, loss=0.0815, precision=0.972, stage=train]
100%|██████████| 340/340 [01:05<00:00,  5.17it/s, auc=0.947, f1_score=0.969, loss=0.0885, precision=0.969, stage=valid]
EarlyStopping counter: 1 out of 5
100%|██████████| 679/679 [21:23<00:00,  1.89s/it, auc=0.961, f1_score=0.973, loss=0.0773, precision=0.973, stage=train]
100%|██████████| 340/340 [01:04<00:00,  5.27it/s, auc=0.948, f1_score=0.969, loss=0.0895, precision=0.969, stage=valid]
EarlyStopping counter: 2 out of 5
100%|██████████| 679/679 [21:28<00:00,  1.90s/it, auc=0.965, f1_score=0.974, loss=0.0737, precision=0.974, stage=train]
100%|██████████| 340/340 [01:05<00:00,  5.22it/s, auc=0.948, f1_score=0.968, loss=0.0891, precision=0.968, stage=valid]EarlySto
```

## SqueezeBERT MODEL OUTPUT

- Trained Model Size: 585 MB
- Total Training Time: 180 mins
- Average Training Time per Epoch : 22.5 mins
- ROC Area Under Curve Score : 0.948
- Micro F1 Score : 0.968

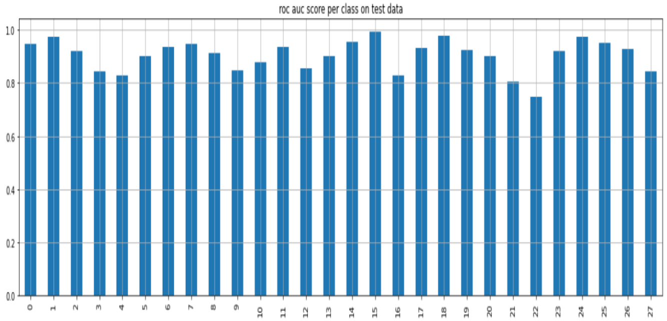
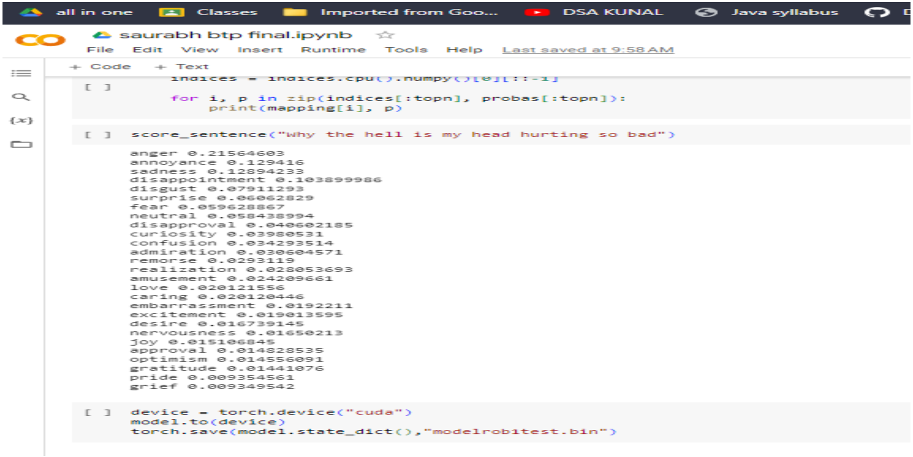


Figure 5: SqueezeBERT ROC Area Under Curve prediction score for all the labels

## SqueezeBERT MODEL OUTPUT



```
        indices = indices.cpu().numpy()[0][::-1]

        for i, p in zip(indices[:topn], probas[:topn]):
            print(mapping[i], p)
```

```
score_sentence("Why the hell is my head hurting so bad")
```

```
anger 0.21564603
annoyance 0.129416
sadness 0.12894233
disappointment 0.103899986
disgust 0.07911293
surprise 0.06062829
fear 0.059628867
neutral 0.058438994
disapproval 0.040602185
curiosity 0.03980531
confusion 0.034293514
admiration 0.030604571
remorse 0.0293119
realization 0.028053693
amusement 0.024209661
love 0.020121556
caring 0.020120446
embarrassment 0.0192211
excitement 0.019013595
desire 0.016739145
nervousness 0.01650213
joy 0.015106845
approval 0.014828535
optimism 0.014556091
gratitude 0.01441076
pride 0.009354561
grief 0.009349542
```

```
device = torch.device("cuda")
model.to(device)
torch.save(model.state_dict(),"modelrob1test.bin")
```

Figure 6: Result analysis

## Conclusion

- During our testing, it became clear that implementing group convolutions instead of fully connected layers would make the model efficient.
- The model trained by SqueezeBERT is 50% smaller in size than the base BERT model, which makes easier to deploy and takes 17% less time to fully train while delivering the same or better overall accuracy and F1 score.
- In addition, the SqueezeBERT model takes almost half the time to analyze new statements. Further improvements can still be made in overall training time and prediction accuracy; future approaches could try and implement aspect-based sentiments, which will be a very interesting use case for this approach.

## Future Scope

1 **Efficiency Improvements:** Continued research can focus on optimizing the model's architecture, compression techniques, and inference speed.

2 **Task-Specific Fine-Tuning:** Investigate how SqueezeBERT performs on a wide range of downstream NLP tasks when fine-tuned.

3 **Transfer Learning to Other Domains:** Explore the applicability of SqueezeBERT in other domains beyond traditional NLP. Investigate how transfer learning can be leveraged to adapt the model for tasks like medical text analysis, legal documents, scientific literature, and more.

4 **Self-Supervised Learning:** Experiment with self-supervised learning techniques to pretrain SqueezeBERT on large amounts of unlabeled data. Self-supervised learning could enhance the model's ability to capture useful features from raw text data, leading to better downstream task performance.

## References I

[1] F. N. Iandola, A. E. Shaw, R. Krishna, and K. W. Keutzer, “Squeezebert: What can computer vision teach nlp about efficient neural networks?,” *arXiv preprint arXiv:2006.11316*.

[2] J. D. M.-W. C. Kenton and L. K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of naacL-HLT*, vol. 1, p. 2.

[3] S. Abdel-Salam and A. Rafea, “Performance study on extractive text summarization using bert models,” *Information*, vol. 13, no. 2, p. 67, 2022.

Thank you!