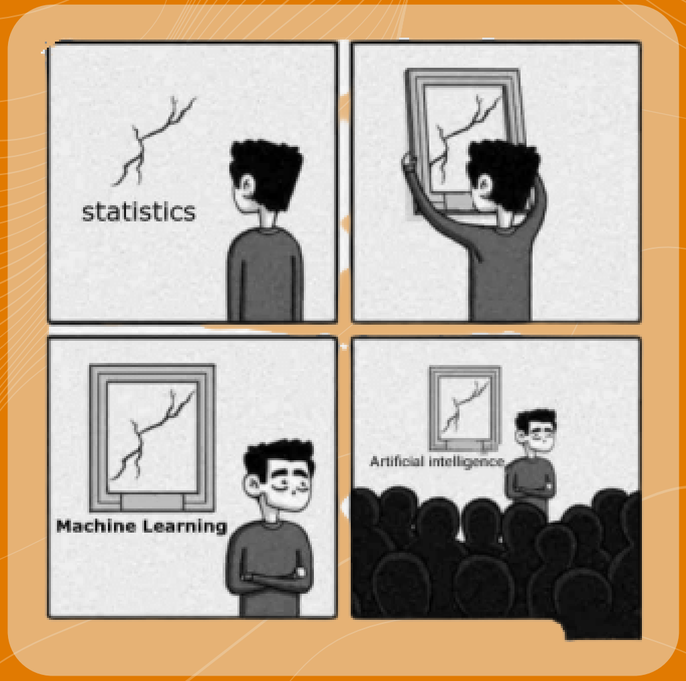
CS771A: Course Project

Detecting and Rating Humor and Offence



Under the guidance of:

Dr. Piyush Rai

CS771 PROJECT PRESENTATION | FALL 2020

Problem Description

Motivation: SemEval-2021, Competition Task-7*

Hahackathon: Linking Humor and Offense Across Different Age Groups

- Humor, like most figurative language, poses interesting linguistic challenges to NLP, due to its emphasis on multiple word senses, cultural knowledge, and pragmatic competence.
- Humor appreciation is also a highly subjective phenomenon, with age, gender and socio-economic status known to have an impact on the perception of a joke.

TASK 1(a)

This is a binary classification task. We need to predict if a particular text is considered humorous of not for an average user

TASK 1(b)

If the text is classed as humorous in the previous , predict how humorous it is (for an average user). The values vary between 0 and 5.

Other than these the competition also has two other tasks, which involve predicting the offence rating and controversy rating. For the domain of project, only tasks 1(a) and 1(b) have been considered

*https://competitions.codalab.org/competitions/27446

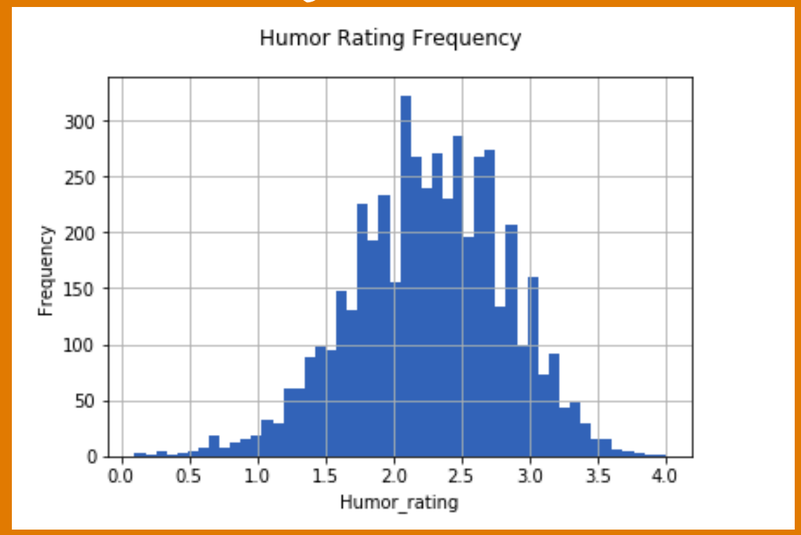
Corpus Description

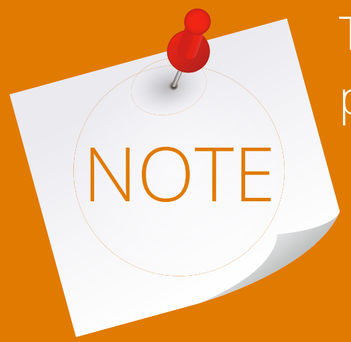
id	text	is_humor	humor_rating	humor_controversy	offense_rating
1	TENNESSEE: We're the best state. Nobody even c	1	2.42	1	0.2
2	A man inserted an advertisement in the classifieds "	1	2.5	1	1.1
3	How many men does it take to open a can of beer?	1	1.95	0	2.4
4	Told my mom I hit 1200 Twitter followers. She pointe	1	2.11	1	0
5	Roses are dead. Love is fake. Weddings are basica	1	2.78	0	0.1
6	Trabajo,' the Spanish word for work, comes from the	0			0
7	I enrolled on some skill training and extra curricula a	0			0.1

Dataset annotated and provided by the organizers of competition

The dataset has 8000 distinct short texts, with the following 4 labels:

- is_humor: binary rating, 1 denoting humor and 0 denoting not humor
- humor_rating: rating (basis average from all annotators), ranging from 0-4
- humor_controversy: binary, 1 for controversy, 0 for not
- offence_rating: 0-5 rating, independent of humor_rating





The data was splitted randomly in the following way preserving the class ratio

- Training: 80% (6400)
- Validation: 10% (800)
- Test: 10% (800)

CS771 PROJECT PRESENTATION | FALL 2020

Subtask 1(a)

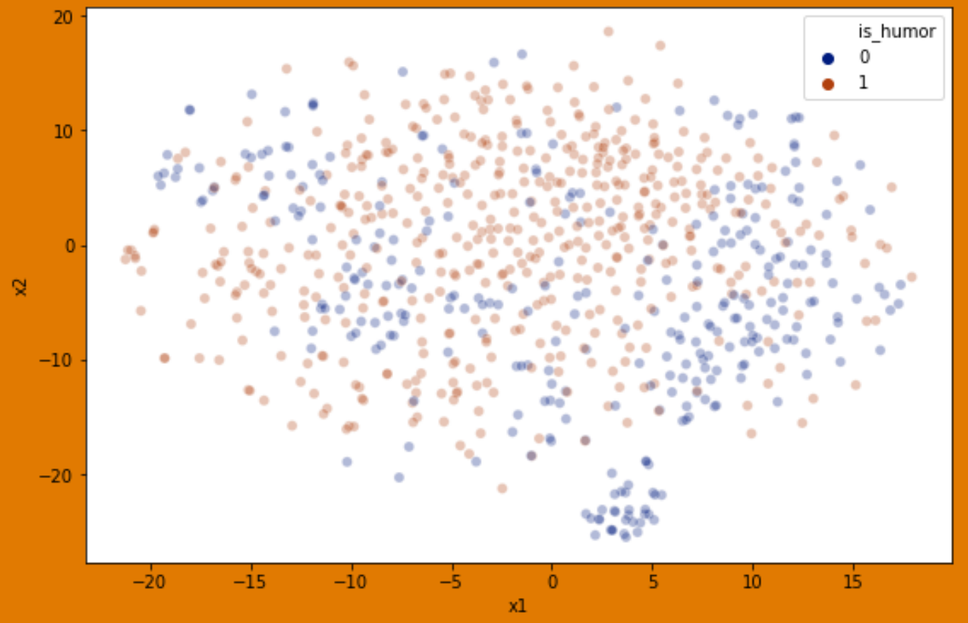
Classical Models

The feature vectors for supervised learning were created in the following way:

- Step1: Tokenization using nltk
- Step2: Removing the stop words using nltk
- Step3: Getting the embedding vectors for tokens using GloVe
  - Common Crawl (840B tokens, 2.2M vocab, cased, 300d vectors) dataset used
- Step4: The final feature vector was taken as sum average of all tokens

The following baseline models were implemented in sklearn

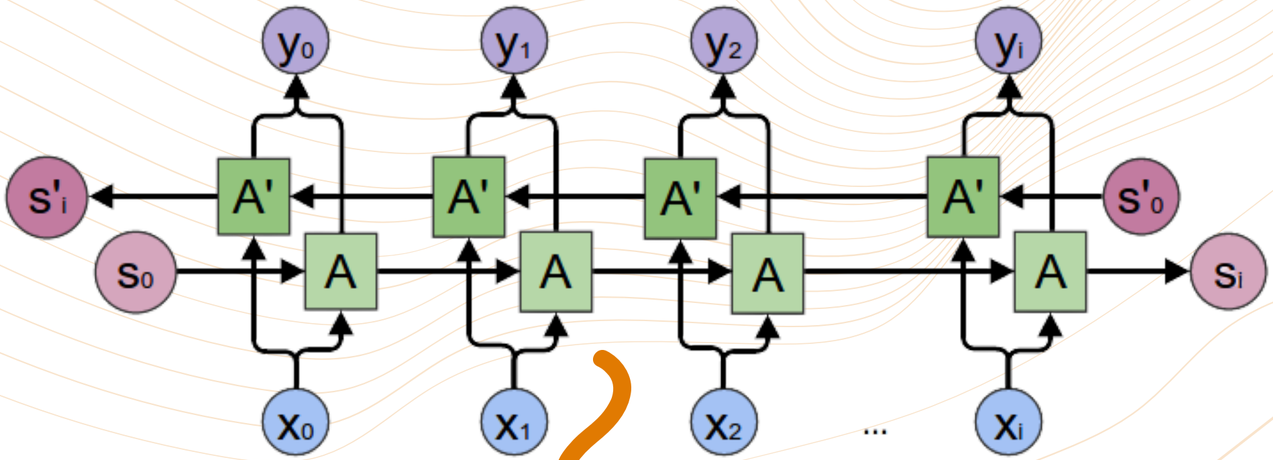
- Logistic Regression
- Decision Tree
- Random Forest
- SVM
- XGBoost
- AdaBoost
- K-Nearest Neighbours

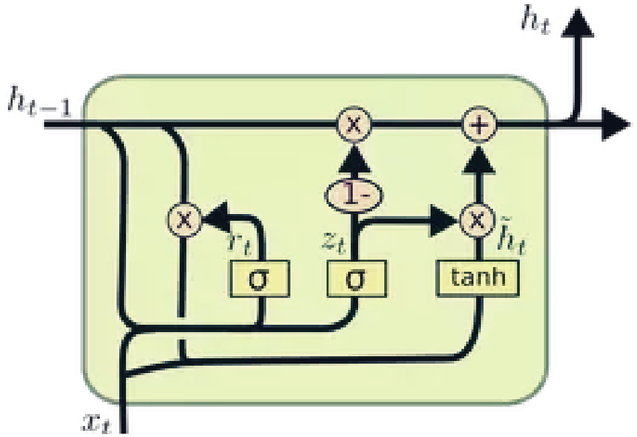


Classical Models: Results (1a)

Model	Hyperparameters	Precision	Recall	FI	Accuracy (%)
LR	solver='saga', Ll_ratio=0.7	0.873	0.868	0.871	84.41
Decision Tree	default_params	0.781	0.751	0.765	71.66
Random Forest	max_depth=20	0.854	0.912	0.882	85.01
SVM	kernel= 'rbf '	0.9	0.88	0.89	86.64
XGBoost	**n_estimators=150**	**0.893**	**0.902**	**0.898**	**87.39**
AdaBoost	n_estimators=200	0.883	0.86	0.871	84.39
KNN	n_neighbors=28	0.926	0.791	0.853	83.27

Bi-directional Gated Recurrent Unit





$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

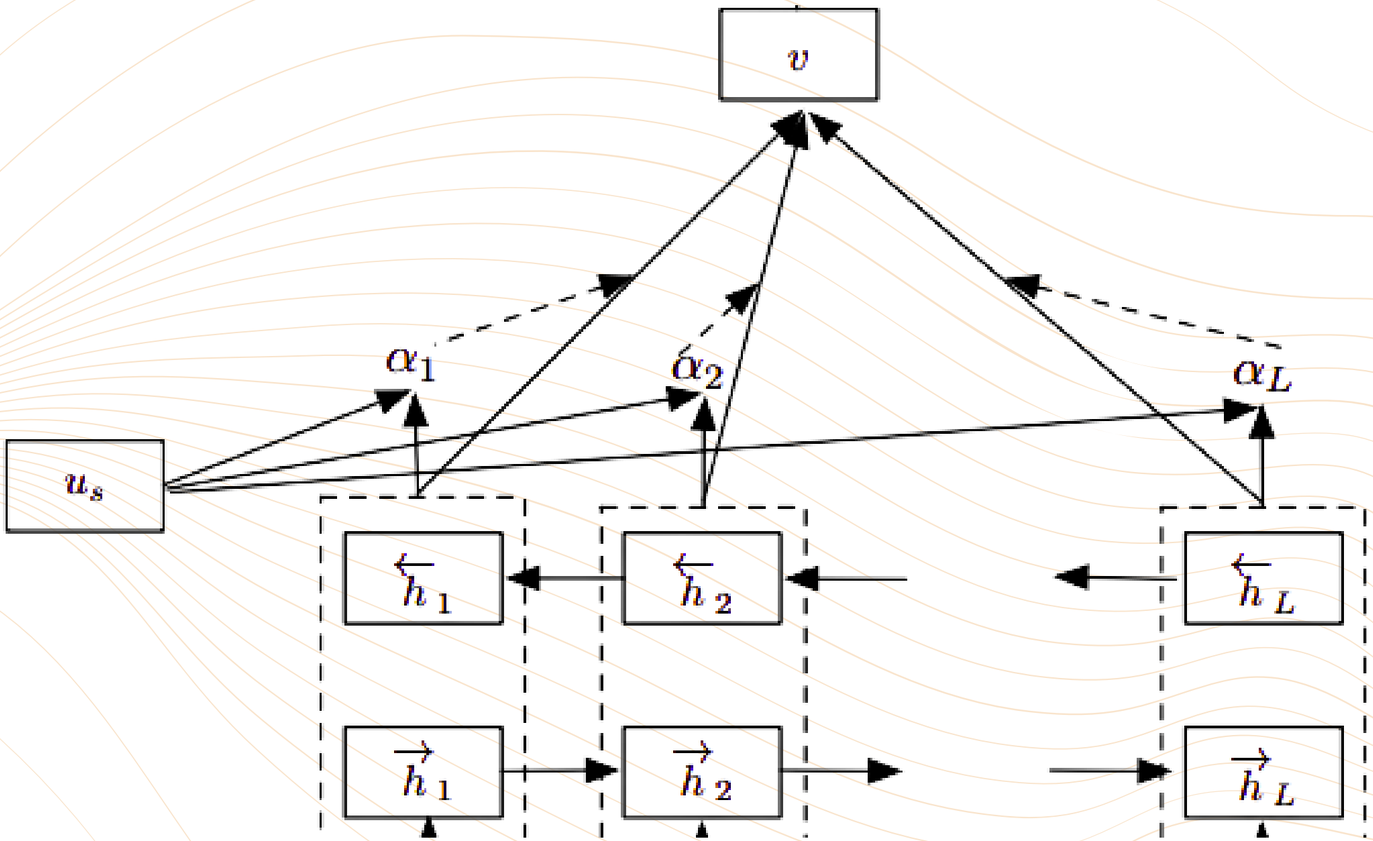
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Image courtesy http://colah.github.io/posts/2015-08-Understanding-LSTMs/

- We use GloVe Embeddings as feature vectors for our input sequence.
- GRU is a variant of the popular RNN network introduced to capture long term dependencies.
- We used 2-layered BiGRU as it can capture both the future as well as the past information.
- Concatenated the final hidden state of the forward and the backward GRU and then passed through a Linear and a Sigmoid layer for predicting label.

BiGRU + Attention

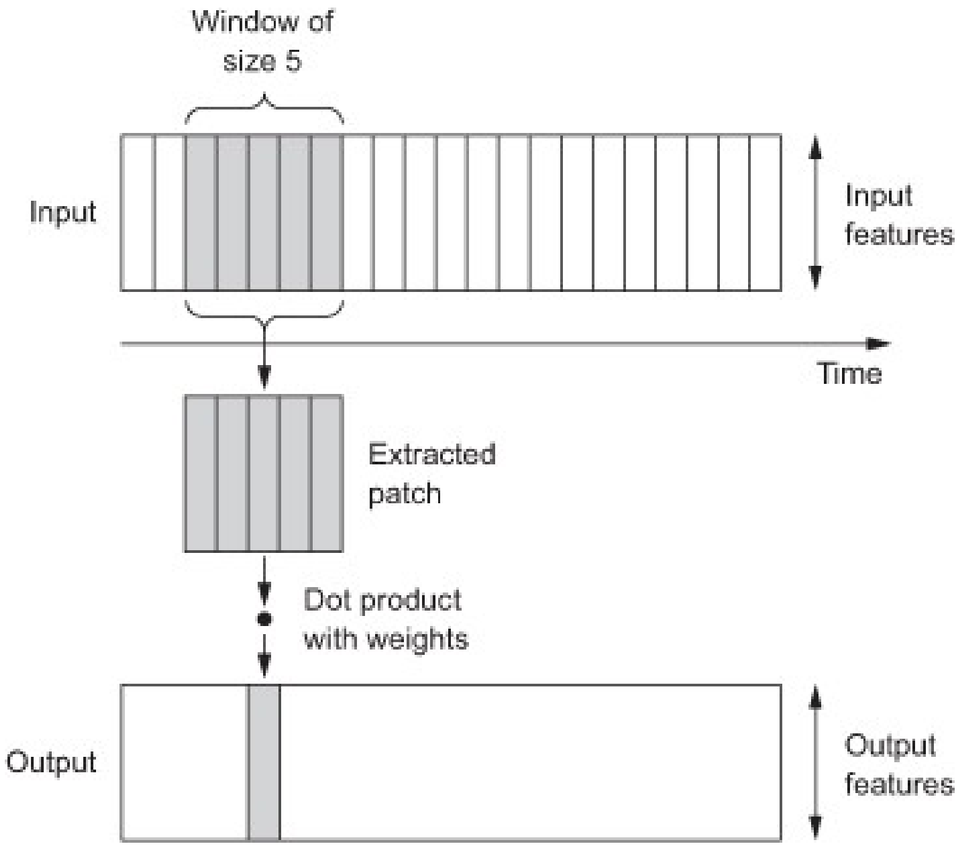


Images courtesy https://www.aclweb.org/anthology/N16-1174.pdf

$$u_i = \tanh(W_s h_i + b_s),$$
$$\alpha_i = \frac{\exp(u_i^\top u_s)}{\sum_i \exp(u_i^\top u_s)},$$
$$v = \sum_i \alpha_i h_i,$$

- Here we have added an extra Attention layer after our BiGRU layer and passed the concatenated hidden states through this layer.
- Then the final output of the attention layer is passed through a Linear layer and then a Sigmoid layer.

1D-CNN for Text Classification



- We use GloVe Embeddings as feature vectors for our input sequence.
- The input vectors were then passed through a sequence of 4 (Conv1D+Maxpool1D) layers.
- The output was then flattened and passed through a Linear and then a sigmoid layer.

Image Courtesy: https://medium.com/@jon.froiland/convolutional-neural-networks-for-sequence-processing-part-1-420dd9b500

Image Courtesy: https://www.rottentomatoes.com/m/transformers_the_movie

INTRODUCTION TO THE FAMILY OF TRANSFORMERS

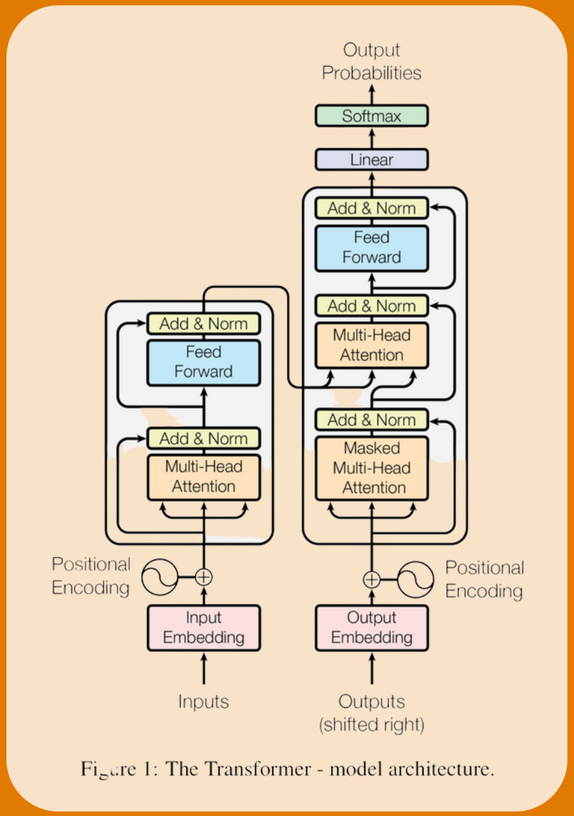


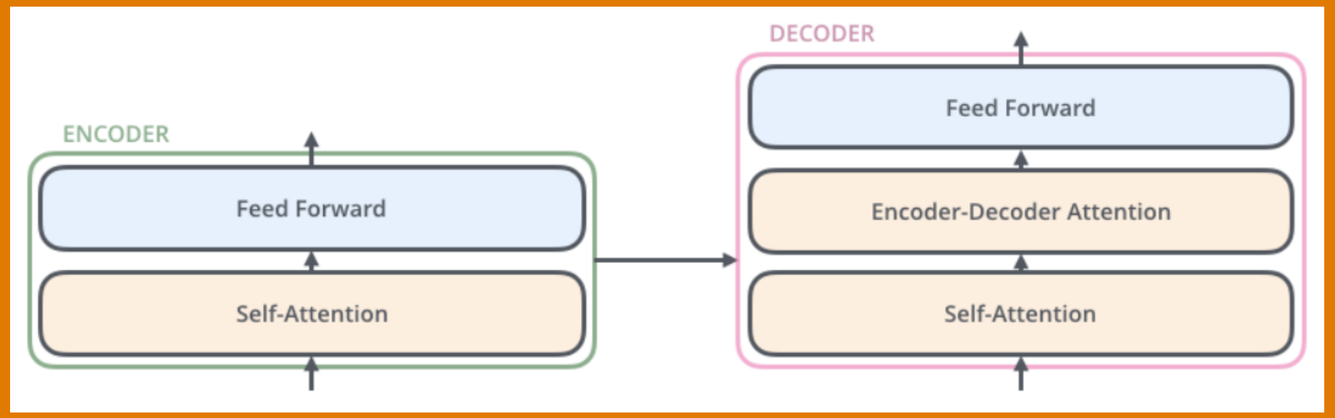
Figure 1: The Transformer - model architecture.

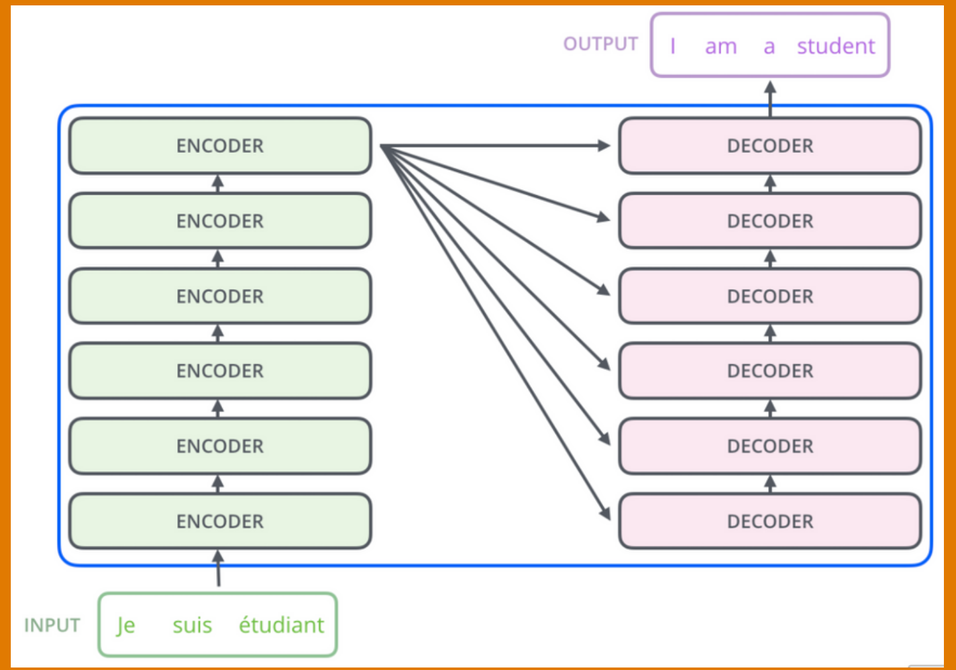
Image Courtesy: https://medium.com/inside-machine-learning/what-is-a-transformer-d07dd1fbec04

CS771 PROJECT PRESENTATION | FALL 2020

Vanilla Transformer

- Attention is All you Need.
- Encoder + Decoder Architecture with Self-Attention.
- Incorporates Positional Embeddings for modeling sequential data.

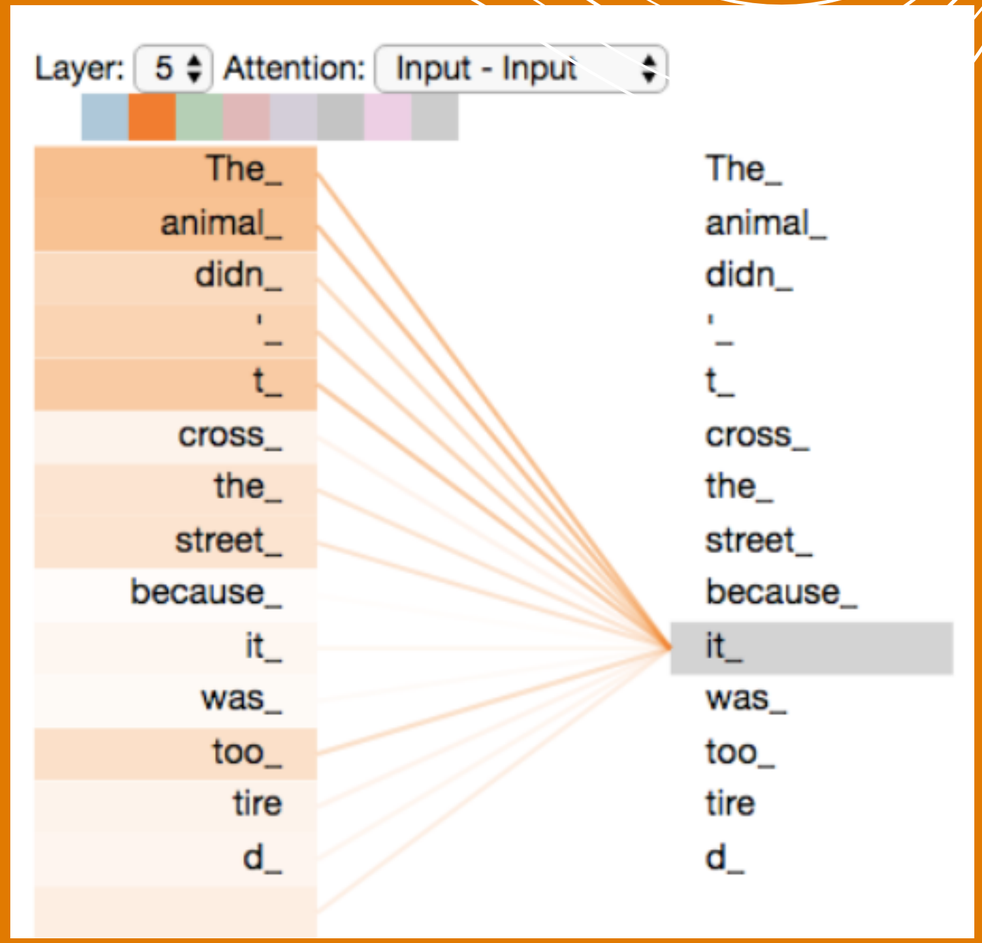




Images Courtesy: JayAlammar's Illustrated BERT and Transformer

Self-Attention

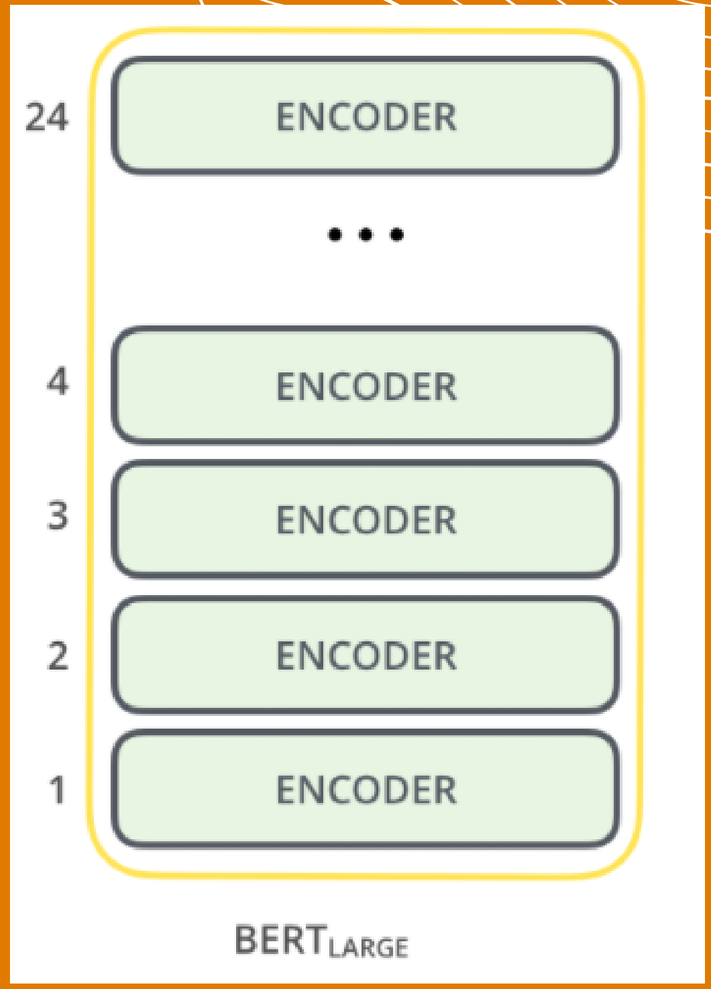
- Computes O(n^2) attention in length n sequential input
- When the model is processing the word “it”, self-attention allows it to associate “it” with “animal”.
- For each word 'w', self attention allows it to look at other positions in the input sequence for clues that can help lead to a better encoding for this word 'w'.



Images Courtesy: JayAlammar's Illustrated BERT and Transformer

BERT-large

- Stack of 24 encoders on top of each other.
- Motivated from:
  - OpenAI Transformer's removing one stack.
  - ULMFiT's Transfer Learning.
  - ELMo's bidirectional nature.
- Introduces MLM (Masked Language Modelling) and NSP (Next Sentence Prediction) tasks for pre-training.



Images Courtesy: JayAlammar's Illustrated BERT and Transformer

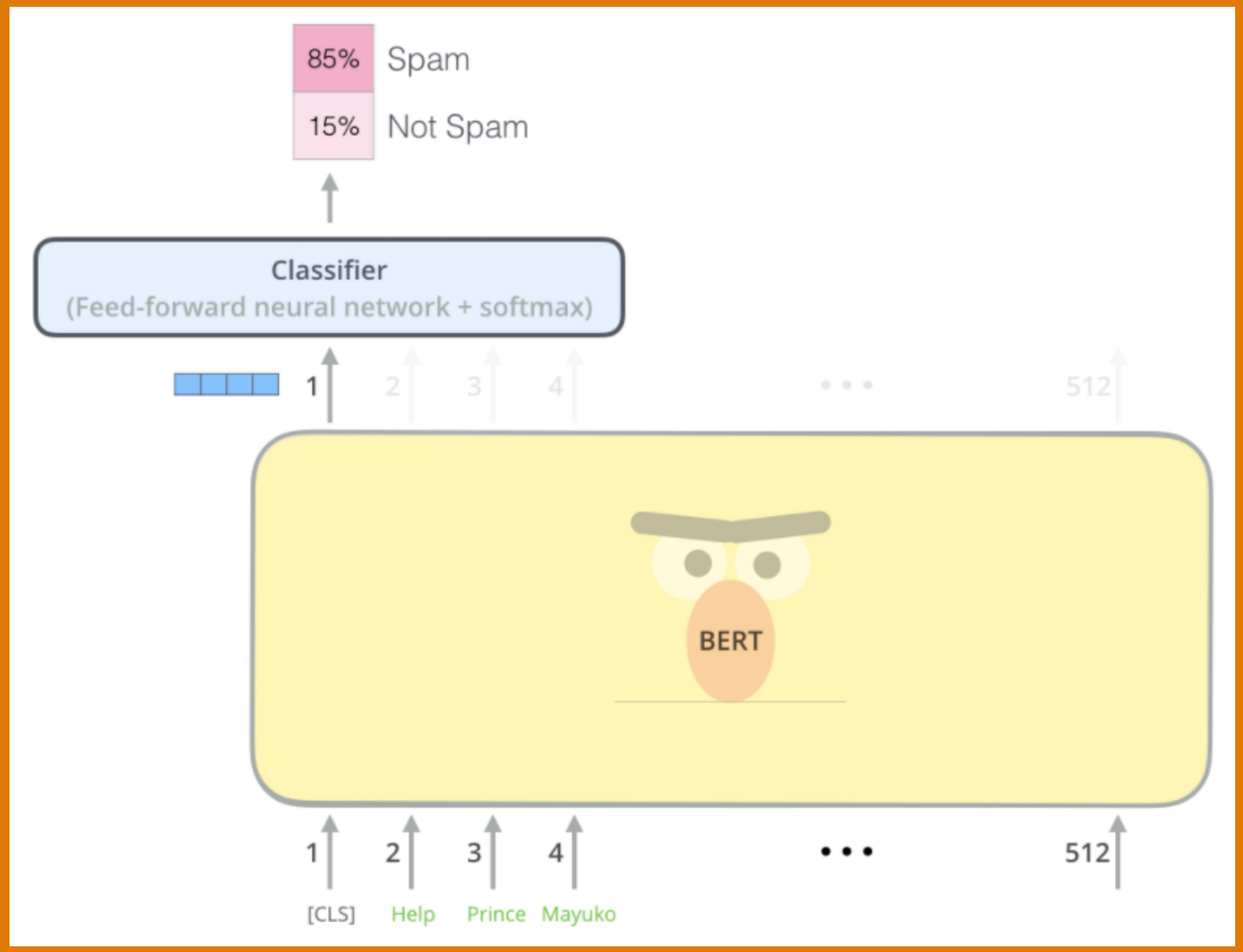
Mutations

	BERT	RoBERT	DistilBERT	XLNet
Size (millions)	Base:110 Large: 340	Base:110 Large: 340	Base: 66	Base: 110 Large: 340
Training Time	Base:8 x V100 x 12 days* Large: 64 TPU Chips x 4 days (or 280 x V100 x 1 days*)	Large: 1024 x V100 x 1 day; 4-5 times more than BERT.	Base: 8 x V100 x 3.5 days;4 times less than BERT.	Large: 512 TPU Chips x 2.5 days;5 times more than BERT.
Performance	Outperforms state-of-the-art in Oct 2018	2-20% improvement over BERT	5% degradation from BERT	2-15% improvement over BERT
Data	16 GB BERT data (Books Corpus + W ikipedia). 3.3 Billion words.	160 GB (16 GB BERT data + 144 GB additional)	16 GB BERT data. 3.3 Billion words.	Base: 16 GB BERT data Large: 113 GB (16 GB BERT data + 97 GB additiona l). 33 Billion words.
Method	BERT (Bidirectional Transfo rmer w ith MLM and NSP)	BERT without NSP* *	BERT Distillation	Bidirectional Transformer with Permutation based modeling

Image courtesy: https://www.kdnuggets.com/2019/09/bert-roberta-distilbert-xlnet-one-use.html

BERT For Sequence Classification

BERT Model + Dropout + Linear Layer + Softmax

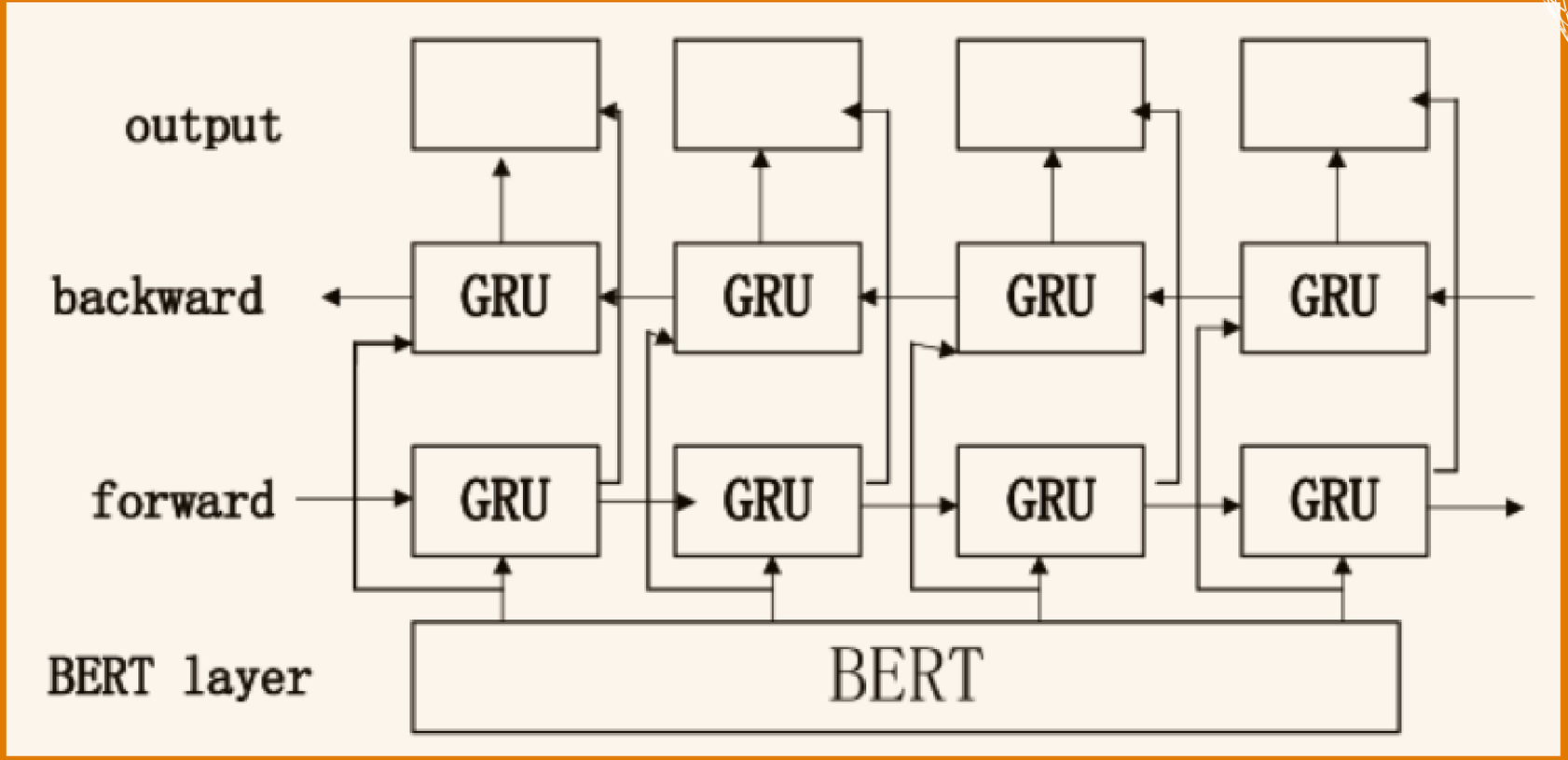


A HUGGINGFACE IMPLEMENTATION

CS771 PROJECT PRESENTATION | FALL 2020

Images Courtesy: JayAlammar's IllustratArchitechture of BERT with BiGRU on top.This image was taken from (Huang et al., 2019)ed BERT and Transformer

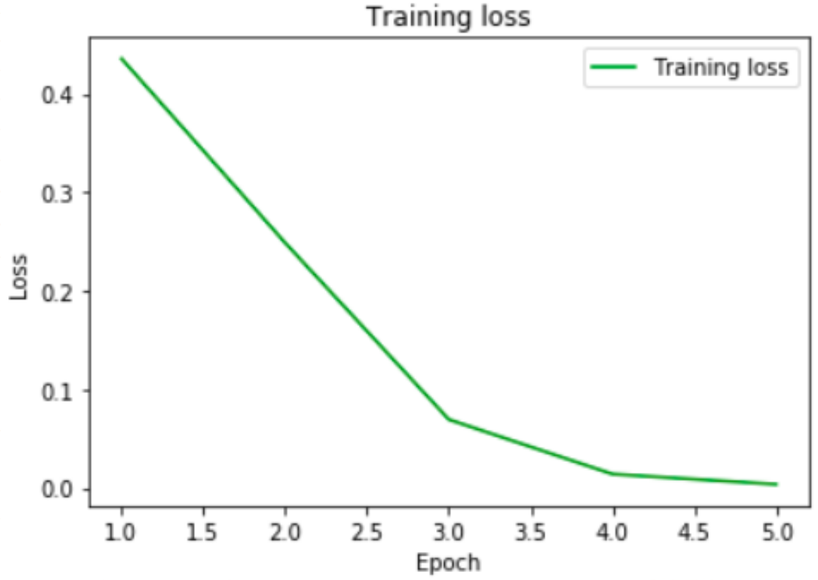
Yet another Bi-GRU layer instead of Linear?

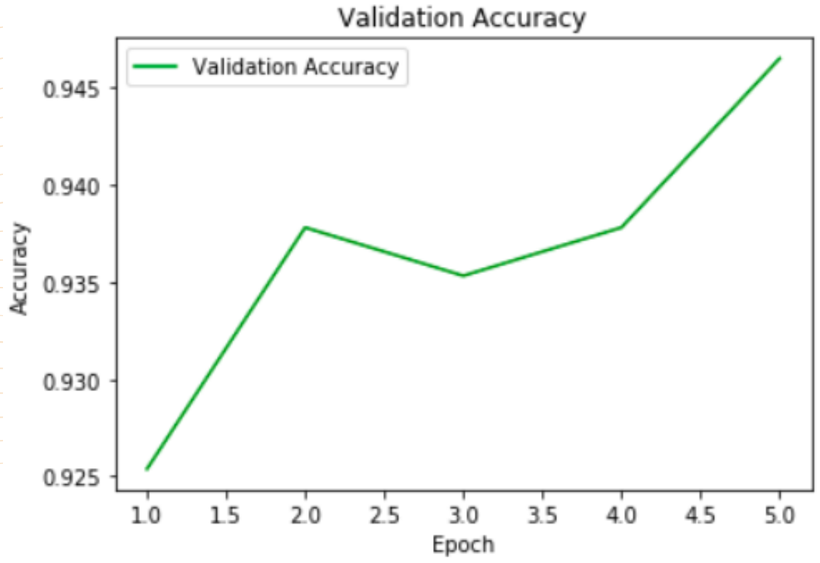


Architechture of BERT with BiGRU on top.This image was taken from (Huang et al., 2019)

Fine-tuning BERT-large on Subtask 1(a)

- Batch size was set to 6! (due to colab restrictions).
- Number of epochs set to 5. (we experimented from 3 to 6 as recommended by the authors of BERT paper).
- Maximum sequence length as 128 covered all input examples easily.





Results On Subtask 1(a)

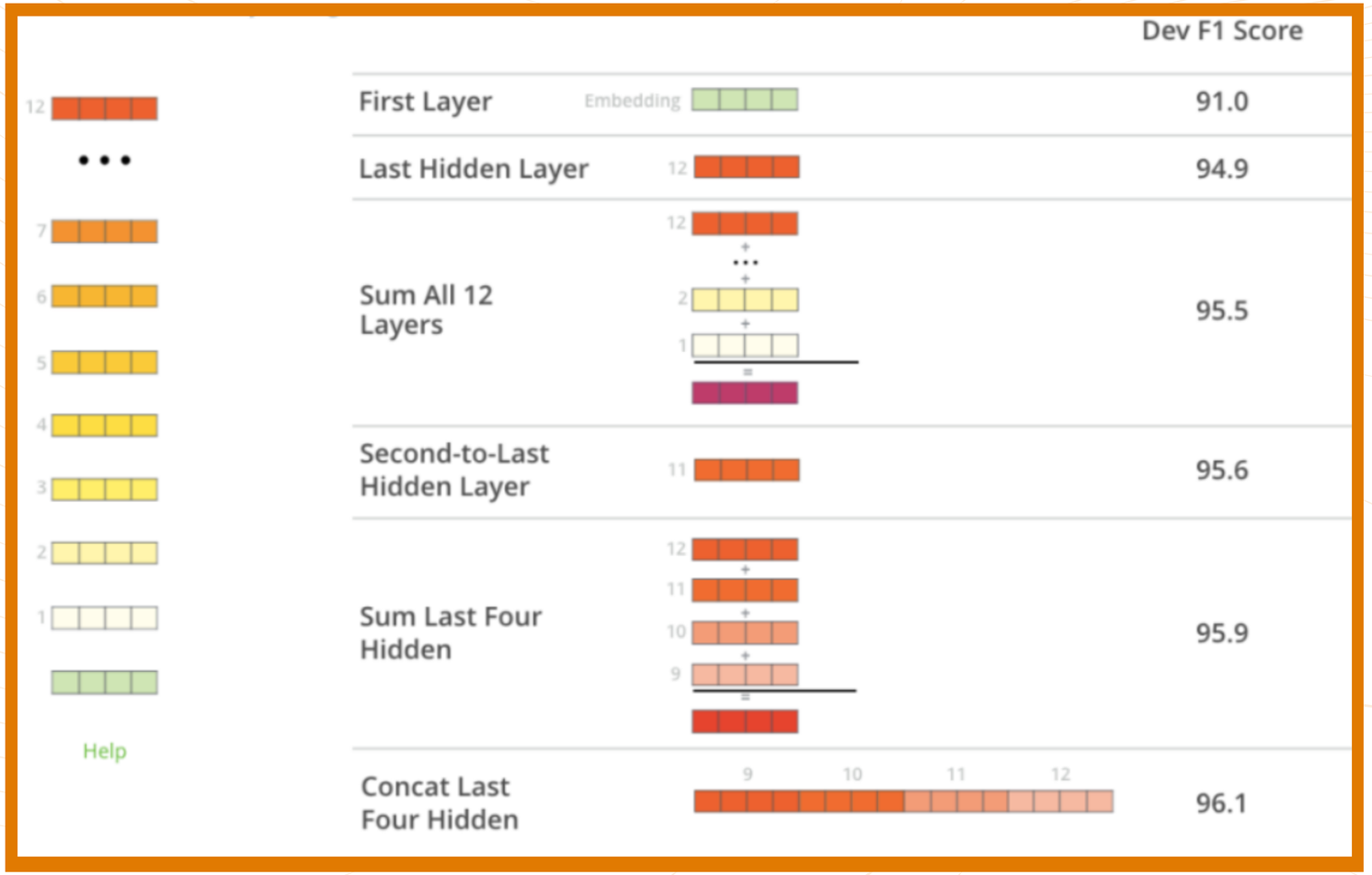
Model	Hyperparameters	Precision	Recall	F1	Accuracy (%)
ANN (MLP)	lr = 5e-2, solver= 'lbfgs',max_itr=10000	0.914	0.882	0.898	87.62
CNN	lr = 5e-4, bs = 32, e = 6, k = 2, s = 1	0.869	0.851	0.859	86.89
BiGRU	lr = 5e-4, bs = 32, e = 5, h_dim= 64	0.881	0.869	0.875	88.26
BiGRU + Att.	lr = 5e-4, bs = 32, e = 5, h_dim = 64	0.853	0.863	0.858	86.27
BERT-large	lr = 2e-5, bs = 6, e = 5, seq_len = 128	0.926	0.932	0.929	93.26
RoBERTa-large	lr = le-6, bs = 6, e = 6, seq_len = 128	0.912	0.89	0.901	89.91
XLNet-large	lr = 2e-6, bs = 6, e = 6, seq_len = 128	0.905	0.927	0.916	92.01
BERT + BiGRU	**lr = 2e-5, bs = 6, e = 5, n_layers = 1**	**0.928**	**0.936**	**0.932**	**93.89**

lr = Learning Rent
bs = batch size
e = epochs
k = kernal size

s = stride
h_dimension = hidden layer dimension
max_itr = maximum iterations (sklearn)
solver = optimizations algorithm

BERT for feature extraction?

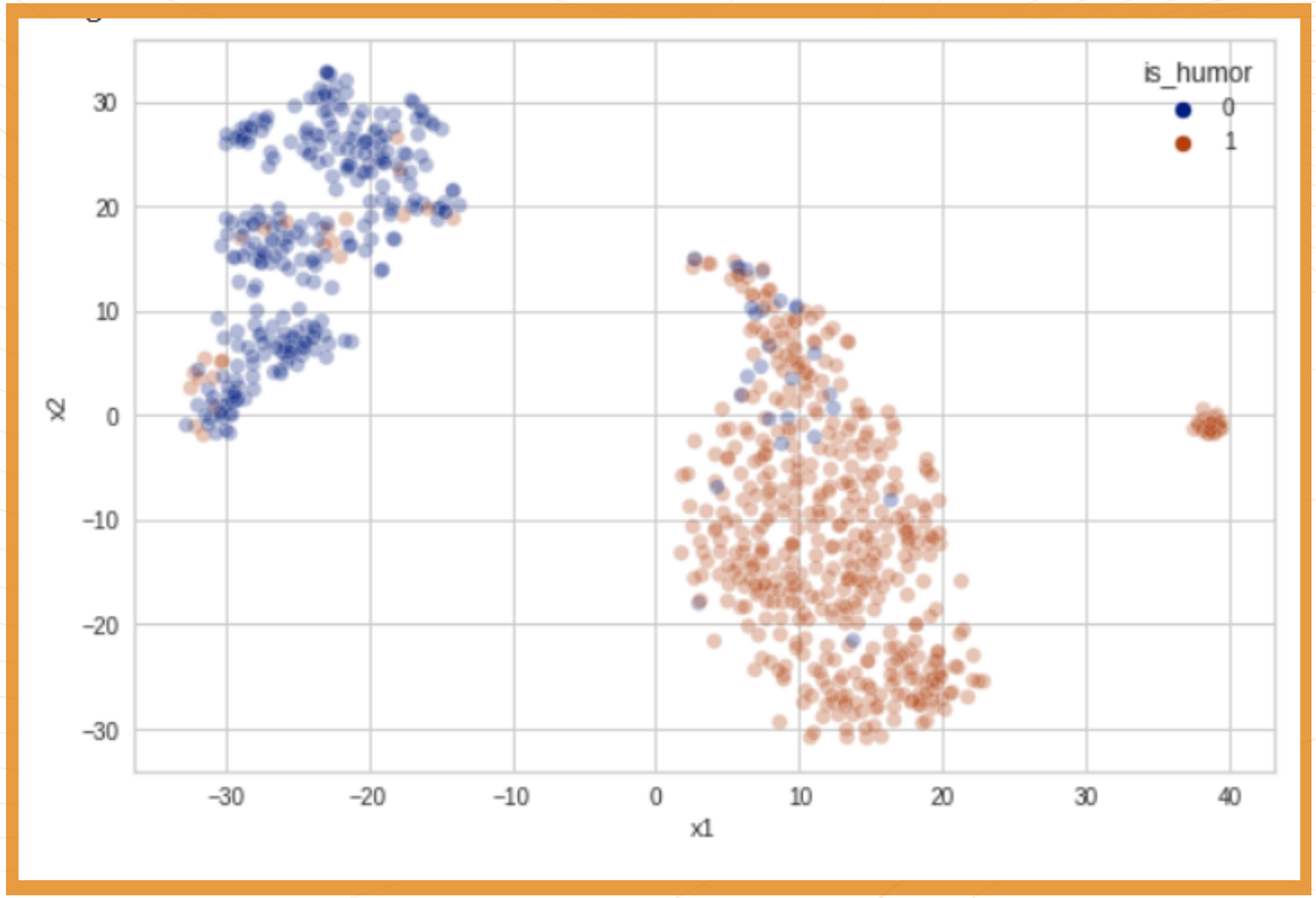
- Performance is experimental and depends on the problem.
- Can be used for more sophisticated downstream tasks.



Images Courtesy: JayAlammar's Illustrated BERT and Transformer

tSNE on [CLS] embeddings

- CLS embeddings are 1024 dimensional feature vectors for the whole input text.
- A fine-tuned BERT on humor detection data encodes information related to the 'humor-ness' of the input.



What makes the text humorous for the model?

[CLS] my aggressive driving made at least a dozen people angry this morning but it was worth it because i got to work 15 seconds earlier [SEP]

[CLS] my boss is going to fire the employee with the worst posture i have a hu ##nch it might be me [SEP]

CLS] i met a girl last night at a bar she said she wanted the night to be magical so i fucked her and disappeared [SEP]

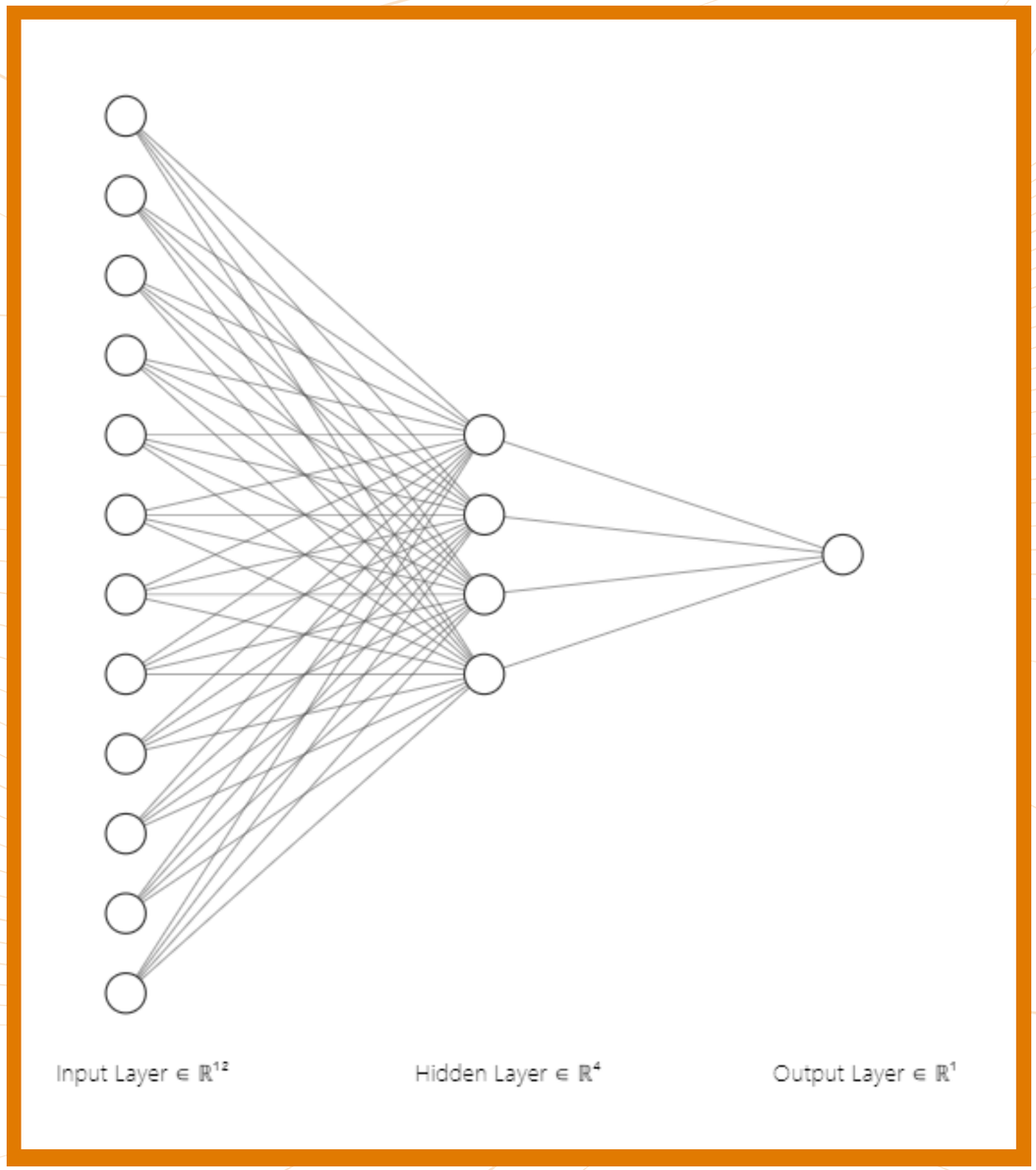
[CLS] waitress can i ask you something about the menu please waitress slap ##s me a good one across the face the men i please are none of your damn business [SEP]

Subtask 1(b)

Regression using BERT features

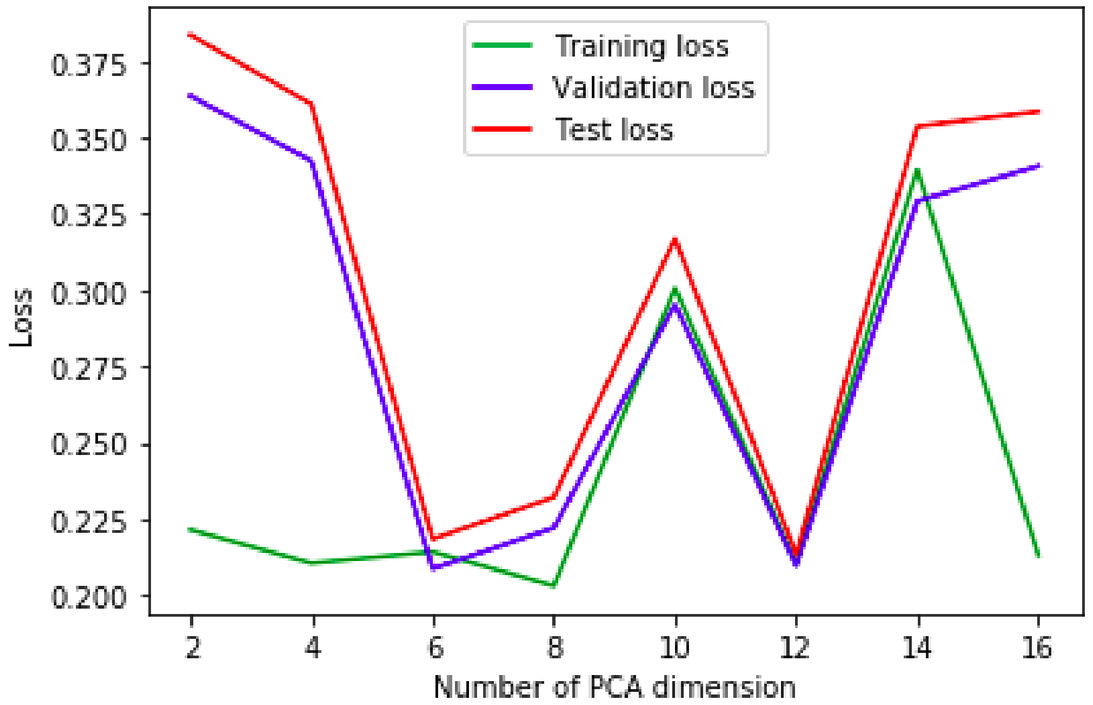
Approaches used:

- Train a Ridge Regression model on BERT features.
- Train a Multi-Layer Perceptron (MLP) on BERT features.



PCA on BERT features?

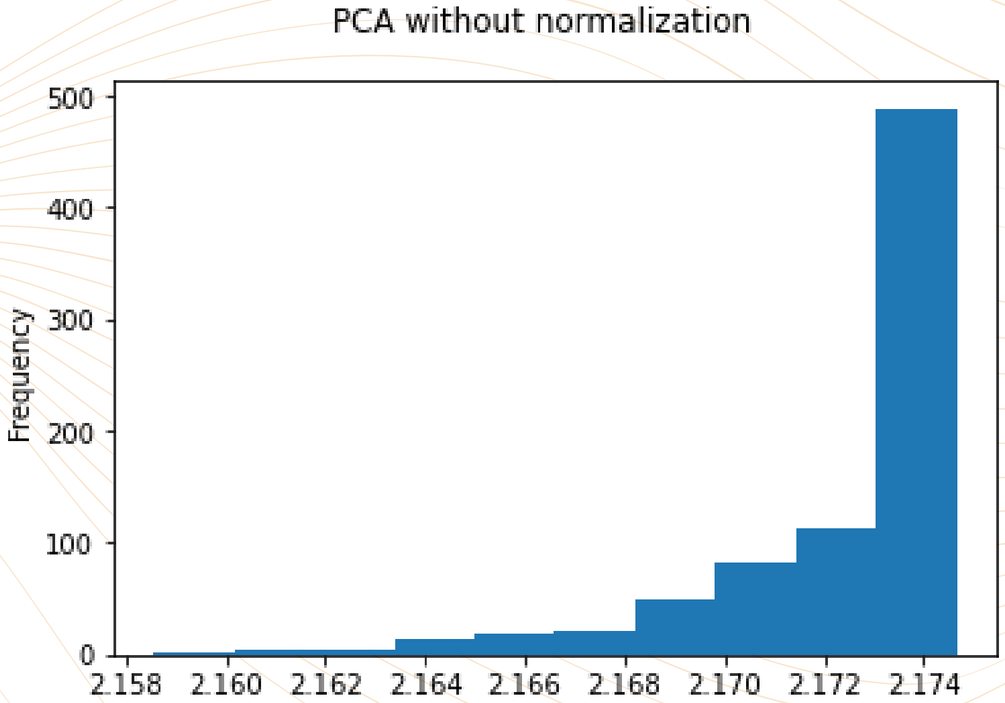
- 1024 dimensional BERT features… HUGE!
- Need to reduce dimensionality PCA!
- Tried out varying the number of Principal Components and used 6 (Gives us the least RMSE on dev set)

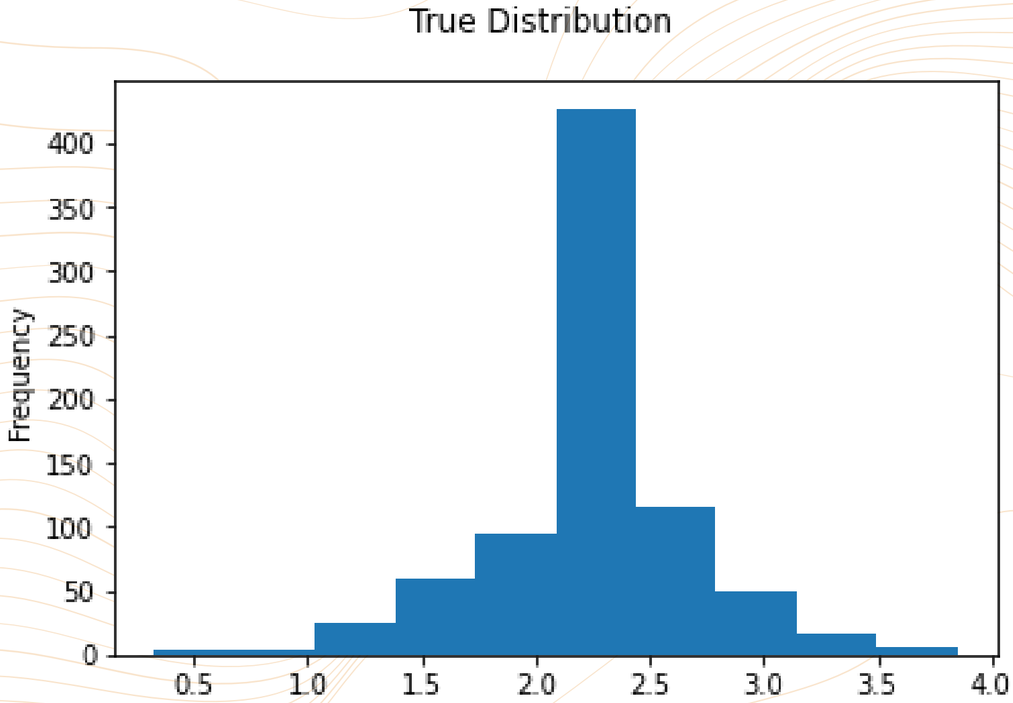


Results on Subtask 1(b)

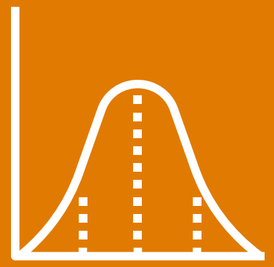
Model	Hyperparameters	RMSE
BERT features + Ridge Regression	α = 100	0.655
BERT features + MLP	4 units hidden layer + ReLU	0.479
BERT features + PCA + MLP	**6 eigenvectors, 2 units hidden layer + ReLU**	**0.446**

However…


PCA without normalization
Frequency
500
400
300
200
100
0
2.158
2.160
2.162
2.164
2.166
2.168
2.170
2.172
2.174


True Distribution
Frequency
400
350
300
250
200
150
100
50
0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

Normalization?



We experimented with the following normalization approaches:

Min-Max

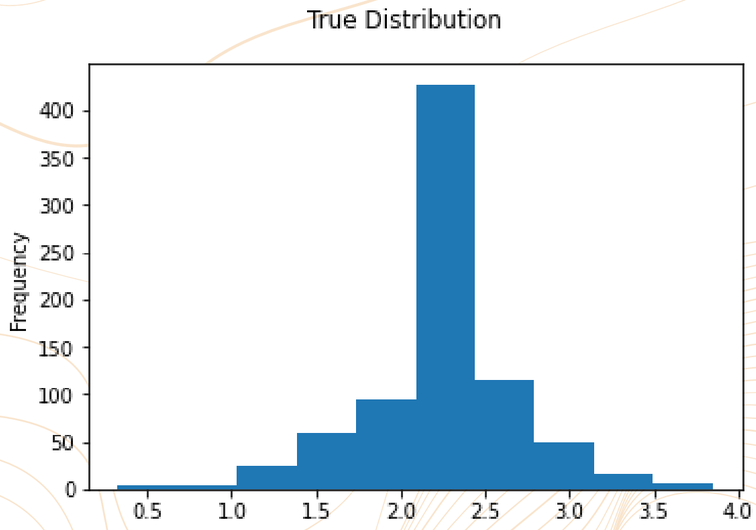
$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

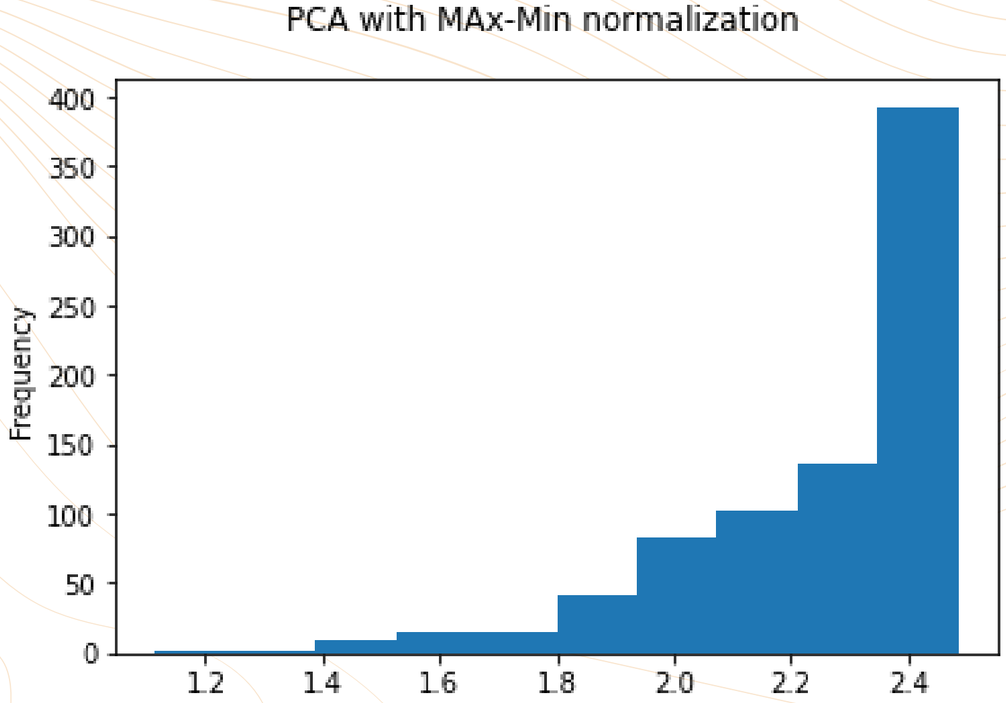
Z-Score

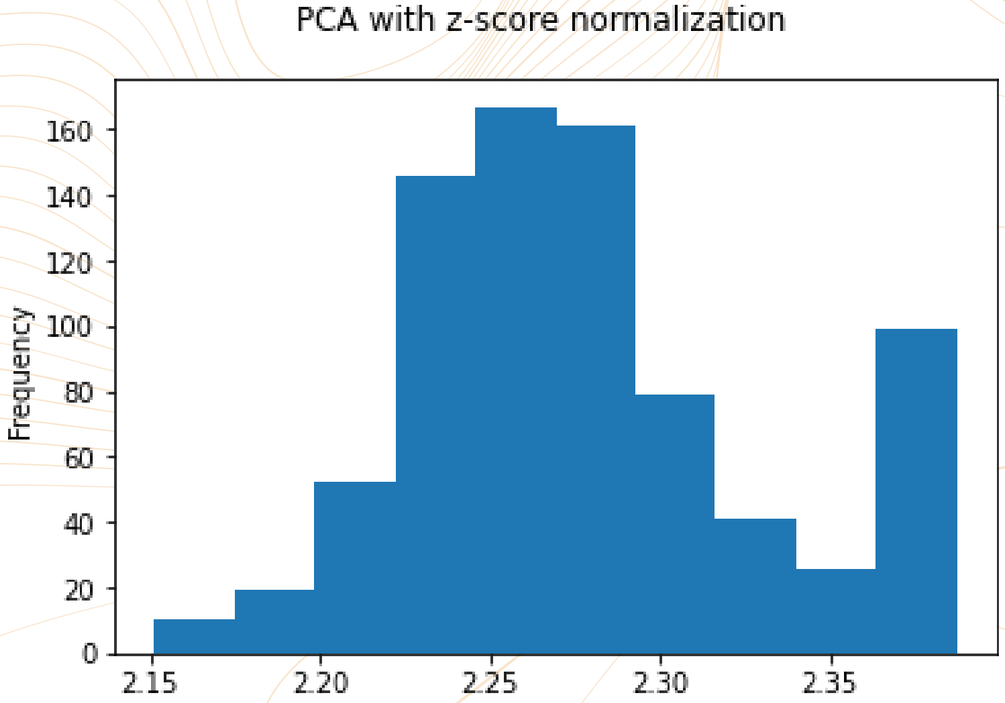
$$z = \frac{x - \mu}{\sigma}$$

μ = Mean
σ = Standard Deviation

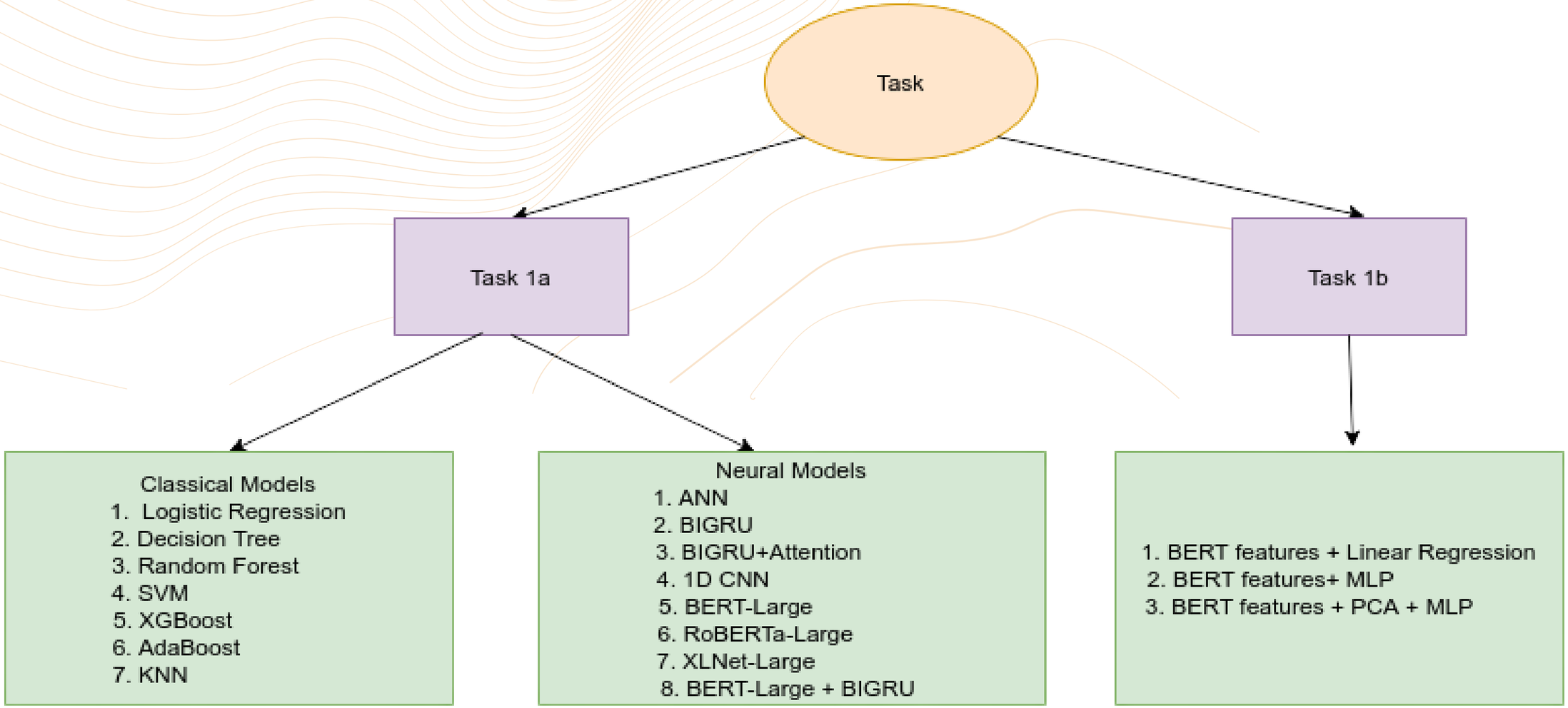
Normalization Results







Overview of Models Used:



Conclusion

In summary:

- We experimented with a variety of models ranging from Decision Trees to BERT large models.
- We visualized what was considered relevant by the model while making predictions.
- We attempted to solve subtask 1(b) using the extracted BERT features.

Future Work

- Inspecting the odd cluster in tSNE.
- Experimenting with more normalization techniques for a better score of subtask 1(b).
- Approach for subtask 1(c) and subtask 2?
- Ensembling to the rescue!
- Data Augmentation!
- More and better features! (Twitter's sentiment based embeddings and PAD values)

SemEval-2021 is the target!

Thank you

Vijit Malik

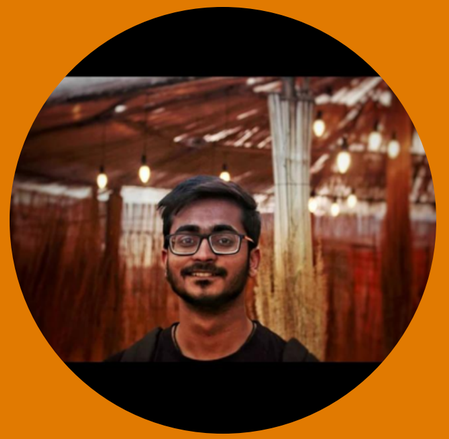
170791

Rishabh Sanjay

170573

Nitik Jain

170448



Aakash Khandelwal

170004