UNIVERSITY OF
GOTHENBURG

DEPARTMENT OF PHILOSOPHY,
LINGUISTICS AND THEORY OF SCIENCE

LAUGHTER PREDICTION IN TEXT BASED DIALOGUES

Predicting Laughter using Transformer-Based Models

Hemanth Kumar Battula

Master's Thesis:	15 credits (select what applies in your case)
Programme:	Master's Programme in Language Technology
Level:	Advanced level
Semester and year:	Spring, 2022
Supervisor:	Vladislav Maraev
Examiner:	Christine Howes
Report number:	(number will be provided by the administrators)
Keywords:	Transformer, BERT, Laughter, Sliding Window

Abstract

In this paper we will attempt to predict and assess the performance of predicting laughter using a BERT model (*Devlin et al., 2019*), and a BERT model finetuned on the **Open subtitles** dataset with and without considering dialogue-acts classes as well as sliding window of dialogues. We hypothesize that fine tuning a BERT on the open subtitles might increase the performance. Our results will be compared with those of *Maraev et al., 2021a* paper which show predicting actual laughs in dialogue and address it with various deep learning models, namely recurrent neural network (RNN), convolution neural network (CNN) and combinations of these. The Switchboard dialogue Act Corpus (SWDA)*, Jurafsky et al., 1997a*) (US English, phone conversations where two participants that are not familiar with each other discuss a potentially controversial subject, such as gun control or the school system) is processed first in the project to make it appropriate for the BERT model. We then analyze dialogue acts within the Switchboard Dialogue Act Corpus with their collocation with laughter and supply some qualitative insights. SWDA is tagged with a collection of 220 dialogue act tags which, following *Jurafsky et al. (1997b)*, we cluster into a smaller set of 42 tags. The major purpose of this research is to show that a BERT model would outperform the Convolution Neural Network (CNN) and Recurrent Neural Network (RNN) models presented in the IWSDS publication.

Contents

1. Introduction

The most prevalent nonverbal vocalization is laughter. Sounds that are not words are called nonverbal vocalizations. Happiness, enjoyment, or relief can all be expressed through laughter. It can also be used to express agreement or to indicate that someone is having fun. When integrating laughter and discourse, it's vital to consider when and why people laugh. People may laugh in response to a joke, while they are having a good time, or when they are trying to make a tense or embarrassing situation less stressful. Laughter can be evoked in response to something that isn't funny but is meant to show support for the speaker. The social and pragmatic functions of laughter should also be considered. Laughter can be used to form and strengthen bonds, convey support for others, and indicate that someone is having a good time. By keeping the conversation going, laughter can help to alleviate tense or uncomfortable situations. (Mazzocconi, C., 2020)

The goal is to predict and evaluate the placement of laughter in dialogues. We will examine the performance of various approaches for predicting laughter from dialogues in this task. The SWDA will be used. (Jurafsky et al., 1997a) *"The Switchboard Dialogue Act Corpus (SWDA) (US English, phone conversations where two participants that are not familiar with each other discuss a potentially controversial subject, such as gun control or the school system) Non-verbally vocalized dialogue acts make up 1.7 percent of all dialogue acts (full utterances identified as non-verbal, 66 percent of which contain laughter). Laughter tokens account for 0.5 percent of all tokens in the corpus." (*Maraev et al., 2021a*)* Laughter relates to the discourse structure of dialogue and can refer to a laughable, which can be a perceived event or an entity in the discourse *(*Maraev et al., 2021 b*).* Laughable is considered any such as to cause laughter or of a kind to provoke laughter.

The Open Subtitles are downloaded from OPUS. OPUS is a growing collection of web-based translations. OPUS convert and align free internet data, add linguistic annotation, and provide the community with a publicly available parallel corpus as part of the OPUS project. The corpus is also supplied as an open content package. OPUS is based on open-source goods. To put together the current collection, we used a variety of techniques. The entire pre-processing is carried out automatically. There have been no manual corrections made. Using this huge corpus of open subtitles data to fine tune a BERT model, we hypothesize that the performance of the model would significantly increase.

The dialogue act (DA) is based on the speech act (Austin, 1975). Speech Act Theory, in contrast to traditional semantic theory, includes not just the propositional content of an utterance, but also the actions it performs, such as promising or apologizing. Dialogue acts are a variation on the speech act that emphasizes the interactional element of most speech. There are over 200 DA tags in the SWDA corpus. The Coders' Manual outlines a method for reducing them to 44 tags. (They claim 42; and their table has 43 rows, so it could just be a tiny counting error). Due to the established link between laughter and particular Dialogue Act tags, we predicted that adding Dialogue Act tags to our data would improve the performance of the model in predicting where laughter occurs.

The research questions below will assist us in our work.

i) Can dialogue data be used to predict laughter using transformer models like a BERT and provide a better performance compared to models like RNN and CNN?

ii) Can a BERT model fine-tuned on Open subtitles can increase the performance of the model? How well can laughter be predicted using a BERT model fine-tuned on open-subtitles dataset?

iii) We hypothesize that adding Dialogue Act tags to our data will improve the model's performance in predicting where laughing occurs because of the proven association between laughter and specific Dialogue Act tags. What is the effect of dialogue acts on laughter prediction using the BERT model?

iv) Does the speaker have any effect in predicting laughter using transformer models like a BERT?

The Thesis is organized as follows:

The prior research on laughter detection on the SWDA dataset is described in Section 2. The SWDA dataset and how it was processed to be used with the BERT model are explained in Section 3. The BERT model architecture is explained in Section 4. We explain the experiments in Section 5 and present the results table. The results are discussed in section 6.

1. Previous Research

Several research had been done using the Switchboard Dialogue Act corpus dataset, on how to predict laughing in dialogues and how laughter affects classification of dialogue acts.

Maraev et al., 2021 a formed a portion of dialogue text using sliding windows of tokens (50/100 tokens) and then checked for laughter tokens. If a laughter token is present at the end of the sliding window, it is marked as 1; otherwise, it is marked as 0. They sought to predict laughter in these sliding windows of texts using this training data in CNN, RNN, and combination of CNN and RNN models utilizing fusion and hybrid methods. It used an Amazon Mechanical Turk (AMT) experiment to assess human performance on the activity of predicting laughter. The key finding of this study is that deep learning models outperforms untrained humans.

Maraev et al., 2021 b, uses a hierarchical data representation to show how the presence of a laughter token affects Dialogue Acts Recognition (total 43 distinct dialogue acts identified). It has utterance-level representation (BERT/CNN) and discourse-level representation (RNN). This study also assesses performance using the BERT model, which has been fine-tuned using the Open subtitles dataset. This study of adult phone conversations reveals how specific laughter patterns, both from the speaker and from the partner, characterize different dialogue acts. It also shows that laughter has a good impact on Transformer-based models' performance in a dialogue act recognition challenge.

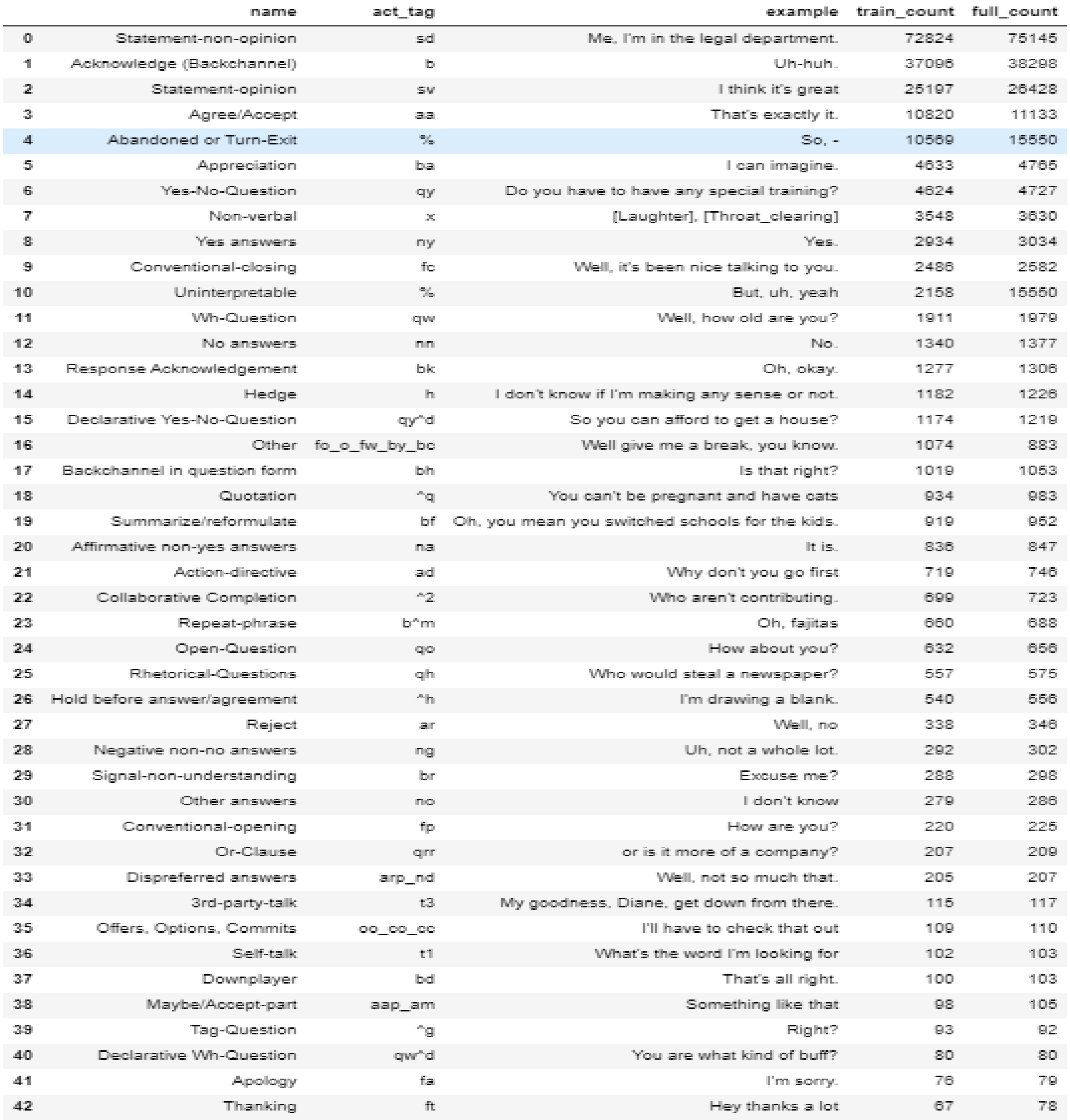
Lala et al., 2017 created a model of shared laughter generation for conversational robots. As part of that system, they train models which predict if shared laughter will occur, given that the user has laughed. Models trained using combinations of acoustic, prosodic features and laughter type and were compared with online versions considered to better quantify their performance in a real system. It was found that these models perform better than chance, with the multimodal combination of acoustic and prosodic features performing the best.

Chen et al., 2017 evaluated speakers’ humor usage, and built a presentation corpus containing humorous utterances based on TED talks. Compared to previous data resources supporting humor recognition research, this study has several advantages, including both positive and negative instances coming from a homogeneous data set, containing many speakers, and being open. Focusing on using lexical cues for humor recognition, this study systematically compares a newly emerging text classification method based on Convolutional Neural Networks (CNNs) with a well-established conventional method using linguistic knowledge. The advantages of the CNN method are both getting higher detection accuracies and being able to learn essential features automatically.

3. Data

The annotated transcripts of 1,155 five-minute phone conversations between two participants make up the SWDA dataset. During these conversations, callers discuss about topics such as childcare, recycling, and the news media. 440 persons speak in these 1,155 talks, totaling 221,616 utterances. It has tags for utterance-level dialogue acts.

Hugging face datasets software was used to download the SWDA dataset. The relevant features to investigate are "text" and "act tag". The "text" field contains the actual speech content, whereas the "act tag" column contains act tag shorthand notation. We also confined our coverage to only 43 DAs listed on the dataset website, in line with Maraev et al, 2001 b. **(See Figure Below)**



	name	act_tag	example	train_count	full_count
0	Statement-non-opinion	sd	Me, I'm in the legal department.	72824	75145
1	Acknowledge (Backchannel)	b	Uh-huh.	37096	38298
2	Statement-opinion	sv	I think it's great	25197	26428
3	Agree/Accept	aa	That's exactly it.	10820	11133
4	Abandoned or Turn-Exit	%	So, -	10569	15550
5	Appreciation	ba	I can imagine.	4633	4765
6	Yes-No-Question	qy	Do you have to have any special training?	4624	4727
7	Non-verbal	x	[Laughter], [Throat_clearing]	3548	3630
8	Yes answers	ny	Yes.	2934	3034
9	Conventional-closing	fc	Well, it's been nice talking to you.	2486	2582
10	Uninterpretable	%	But, uh, yeah	2158	15550
11	Wh-Question	qw	Well, how old are you?	1911	1979
12	No answers	nn	No.	1340	1377
13	Response Acknowledgement	bk	Oh, okay.	1277	1306
14	Hedge	h	I don't know if I'm making any sense or not.	1182	1226
15	Declarative Yes-No-Question	qy^d	So you can afford to get a house?	1174	1219
16	Other	fo_o_fw_by_bc	Well give me a break, you know.	1074	883
17	Backchannel in question form	bh	Is that right?	1019	1053
18	Quotation	^q	You can't be pregnant and have cats	934	983
19	Summarize/reformulate	bf	Oh, you mean you switched schools for the kids.	919	952
20	Affirmative non-yes answers	na	It is.	836	847
21	Action-directive	ad	Why don't you go first	719	746
22	Collaborative Completion	^2	Who aren't contributing.	699	723
23	Repeat-phrase	b^m	Oh, fajitas	660	688
24	Open-Question	qo	How about you?	632	656
25	Rhetorical-Questions	qh	Who would steal a newspaper?	557	575
26	Hold before answer/agreement	^h	I'm drawing a blank.	540	556
27	Reject	ar	Well, no	338	346
28	Negative non-no answers	ng	Uh, not a whole lot.	292	302
29	Signal-non-understanding	br	Excuse me?	288	298
30	Other answers	no	I don't know	279	286
31	Conventional-opening	fp	How are you?	220	225
32	Or-Clause	qrr	or is it more of a company?	207	209
33	Dispreferred answers	arp_nd	Well, not so much that.	205	207
34	3rd-party-talk	t3	My goodness, Diane, get down from there.	115	117
35	Offers, Options, Commits	oo_co_cc	I'll have to check that out	109	110
36	Self-talk	t1	What's the word I'm looking for	102	103
37	Downplayer	bd	That's all right.	100	103
38	Maybe/Accept-part	aap_am	Something like that	98	105
39	Tag-Question	^g	Right?	93	92
40	Declarative Wh-Question	qw^d	You are what kind of buff?	80	80
41	Apology	fa	I'm sorry.	76	79
42	Thanking	ft	Hey thanks a lot	67	78

Using Pandas (Mckinney et al., 2011*)*, we obtained the 43 SWDA act tag table. It has characteristics such as "name" and "act tag," as well as "example" etc. The "name" feature contains the complete DA name, whereas the "act tag" column contains the DA shorthand notation. Then we replaced act tag with its entire name in the SWDA dataset. Only these 43 DAs were kept from the main SWDA dataset.

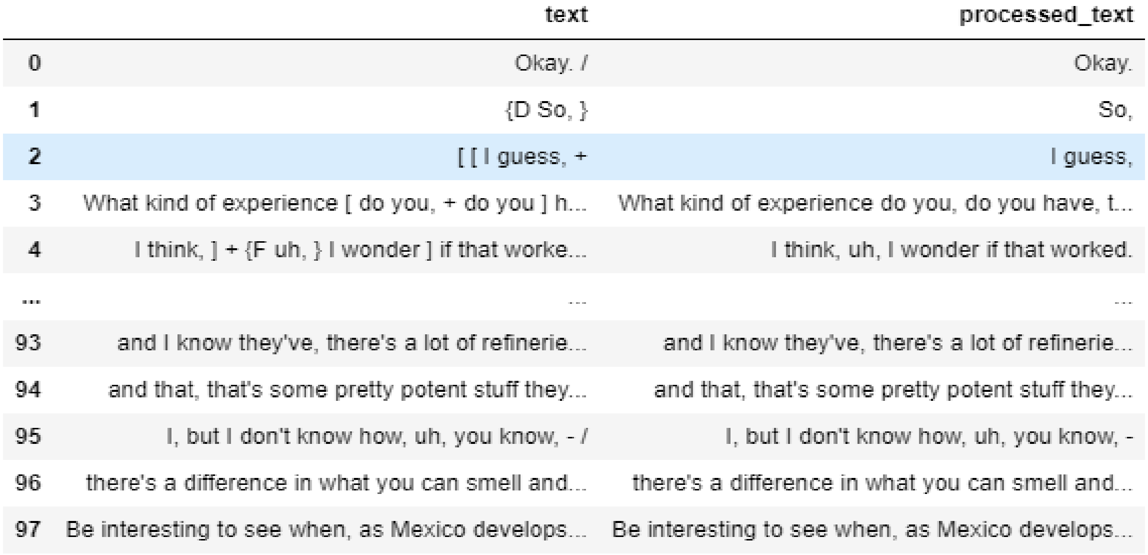
3.1 Data Preprocessing

In the project, first we processed the SWDA to make it suitable for the BERT model. Special characters like different brackets, /, +, <<, >> symbols are removed. Consecutive duplicate *laughter* words are removed and only a single occurrence of the word is retained. This would help us to avoid duplication while labelling the sliding window of tokens as laughter and non-laughter.

BERT constitutes of two different models.

bert-base-uncased - This model is uncased: it does not make a difference between english and English.
bert-base-cased - This model is case-sensitive: it makes a difference between english and English.

All text converted to lower text to maintain uniformity for training the data with *bert-base-uncased* model.

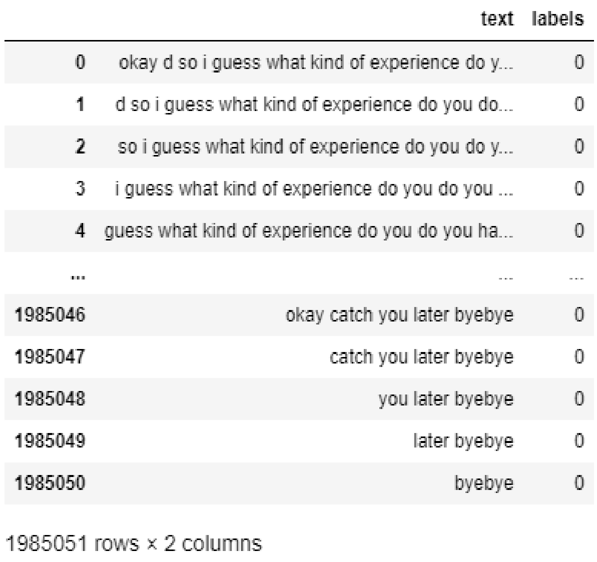


	text	processed_text
0	Okay. /	Okay.
1	{D So, }	So,
2	[[I guess, +	I guess,
3	What kind of experience [do you, + do you] h...	What kind of experience do you, do you have, t...
4	I think,] + {F uh, } I wonder] if that worke...	I think, uh, I wonder if that worked.
...	...	...
93	and I know they've, there's a lot of refinerie...	and I know they've, there's a lot of refinerie...
94	and that, that's some pretty potent stuff they...	and that, that's some pretty potent stuff they...
95	I, but I don't know how, uh, you know, - /	I, but I don't know how, uh, you know, -
96	there's a difference in what you can smell and...	there's a difference in what you can smell and...
97	Be interesting to see when, as Mexico develops...	Be interesting to see when, as Mexico develops...

Fragment of a processed conversation (From the switchboard Corpus)

Due to the scarcity of laughter, we explain this challenge to the corpus being biased towards negative predictions. In fact, the proportion of true laughter tokens in the corpus is roughly 0.5 percent. It's also a difficult and unrealistic undertaking for humans because annotating each token is time-consuming. Instead, we put the point of emphasis on certain locations and try to anticipate the occurrence of laughter at those locations. We choose these points so that the frequency of laughter at these points is equal to the frequency of non-laughs.

We made sliding windows of different sizes, say 50 or 100 with overlap, and verified whether the laughter token was present at the end or not. If the laughter token appears at the end of the sliding window, the section of sliding window is labelled as 1, otherwise 0. The final laughter token at the end of sliding window is also eliminated after labelling. The sliding window dataset that results is severely unbalanced. So, the dataset was balanced by keeping the frequency of laughter and non-laughter the same. We reduced the frequency of non-laughter by randomly selecting the sliding windows which are labelled as 0 from the dataset to balance with laughter frequency which are labelled as 1.



	text	labels
0	okay d so i guess what kind of experience do y...	0
1	d so i guess what kind of experience do you do...	0
2	so i guess what kind of experience do you do y...	0
3	i guess what kind of experience do you do you ...	0
4	guess what kind of experience do you do you ha...	0
...	...	...
1985046	okay catch you later byebye	0
1985047	catch you later byebye	0
1985048	you later byebye	0
1985049	later byebye	0
1985050	byebye	0

1985051 rows × 2 columns

UnBalanced
(without DA tags):

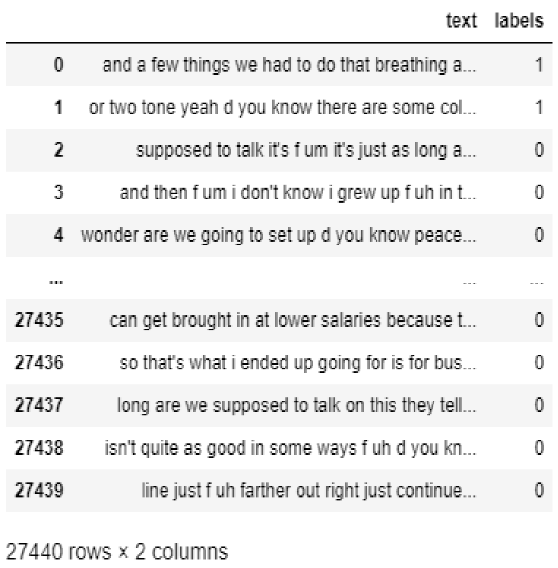
Example for a 50 Sliding window datasets:

Total labels: 1985051

The count of labels 0: 1971331

The count of labels 1: 13720

Fragment of labelled conversation on the sliding window of tokens-unbalanced



	text	labels
0	and a few things we had to do that breathing a...	1
1	or two tone yeah d you know there are some col...	1
2	supposed to talk it's f um it's just as long a...	0
3	and then f um i don't know i grew up f uh in t...	0
4	wonder are we going to set up d you know peace...	0
...	...	...
27435	can get brought in at lower salaries because t...	0
27436	so that's what i ended up going for is for bus...	0
27437	long are we supposed to talk on this they tell...	0
27438	isn't quite as good in some ways f uh d you kn...	0
27439	line just f uh farther out right just continue...	0

27440 rows × 2 columns

Balanced (without DA tags):

Example for a 50 Sliding window datasets:

0 Labels: 13720

1 Labels: 13720

Train labels count: 1 10996

Train labels count: 0 10956

Validation labels count: 1 1381

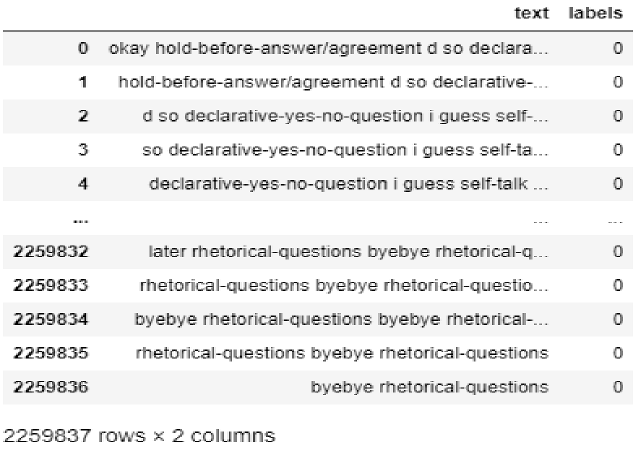
Validation labels count: 0 1363

Test labels count: 0 1383

Test labels count: 1 1361

Fragment of labelled conversation on the sliding window of tokens-balanced

We concatenated the Dialogue Act with its respective speech and then made sliding windows of different sizes, say 50 or 100, with overlap and verified whether the laughter token was last or not. If the laughter token appears at the end of the sliding window, the section of sliding window is labelled as 1, otherwise 0. The final laughter token is also eliminated in the sliding window labelled as 1. The sliding window dataset that results is severely unbalanced. So, the dataset was balanced by keeping the frequency of laughter and non-laughter the same. We reduce the frequency of non-laughter by randomly selecting the sliding windows which are labelled as 0 from the dataset, to balance with laughter frequency which are labelled as 1. Because the *act_tag* column only contained the index from the 43 DA table, the index was replaced with its full DA name. Text pre-processing was used, and the *act_tag* column was lowered.



	text	labels
0	okay hold-before-answer/agreement d so declara...	0
1	hold-before-answer/agreement d so declarative-...	0
2	d so declarative-yes-no-question i guess self-...	0
3	so declarative-yes-no-question i guess self-ta...	0
4	declarative-yes-no-question i guess self-talk ...	0
...	...	...
2259832	later rhetorical-questions byebye rhetorical-q...	0
2259833	rhetorical-questions byebye rhetorical-questio...	0
2259834	byebye rhetorical-questions byebye rhetorical-...	0
2259835	rhetorical-questions byebye rhetorical-questions	0
2259836	byebye rhetorical-questions	0

2259837 rows × 2 columns

UnBalanced
(with DA tags):

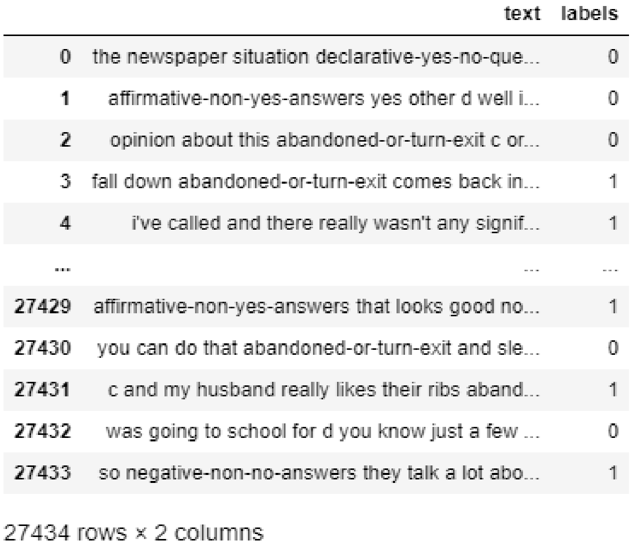
Example for a 50 Sliding window datasets:

Total: 2259837

The count of labels 0:   2246120

The count of labels 1:   13717

Fragment of labelled conversation on the sliding window of tokens-unbalanced with Dialogue Acts



	text	labels
0	the newspaper situation declarative-yes-no-que...	0
1	affirmative-non-yes-answers yes other d well i...	0
2	opinion about this abandoned-or-turn-exit c or...	0
3	fall down abandoned-or-turn-exit comes back in...	1
4	i've called and there really wasn't any signif...	1
...	...	...
27429	affirmative-non-yes-answers that looks good no...	1
27430	you can do that abandoned-or-turn-exit and sle...	0
27431	c and my husband really likes their ribs aband...	1
27432	was going to school for d you know just a few ...	0
27433	so negative-non-no-answers they talk a lot abo...	1

27434 rows × 2 columns

Balanced
(With DA Tags):

Example for a 50 Sliding window datasets:

Labels 0:   13717

Labels 1:   13717

Train labels count 1: 11001

Train labels count 0: 10946

Validation labels count 1: 1404

Validation labels count 0: 1339

Test labels count 0: 1377

Test labels count 1: 1367

Fragment of labelled conversation on the sliding window of tokens-balanced with Dialogue Acts

Fine Tuning BERT:

- The Open Subtitles are downloaded from OPUS. OPUS is a growing collection of web-based translations. OPUS convert and align free internet data, add linguistic annotation, and provide the community with a publicly available parallel corpus as part of the OPUS project.
- The dataset size was around 13GB after extraction.
- Since, the dataset was huge a sample of around 2 million lines are taken for finetuning BERT model on task called "masked language modelling".

We used SimpleTransformers (Rajapakse, T. C., 2019*)* task based *LanguageModelingModel* class for fine-tuning 'BERT' model on opensubtitles. The resulted language model is used for training, evaluation and testing of the SWDA.

4. Methods

4.1 Transformer

The original transformer (Figure-1 Below), proposed in the paper **Attention is all you need** (2017), is an encoder-decoder-based neural network that is mainly characterized by the use of the so-called attention (i.e. a mechanism that determines the importance of words to other words in a sentence or which words are more likely to come together) and the non-use of recurrent connections (or recurrent neural networks) to solve tasks that involve sequences (or sentences)

The encoder-decoder architecture, as well as the attention mechanism, are not new ideas. Previous neural network architectures had employed these techniques to accomplish numerous NLP tasks, such as machine translation. The transformer's originality is that it demonstrates that we can handle tasks involving sequences (such as machine translation) with attention alone, without the usage of recurrent connections, which is a benefit given that recurrent connections might obstruct the parallelization of the training process.

Both the encoder and decoder are composed of attention modules, feed-forward (or fully connected) layers, residual (or skip) connections, normalization layers, dropout, label smoothing, embedding layers, positional encoding. The decoder part is also composed of a linear layer followed by a softmax to solve the specific NLP task (for example, predict the next word in a sentence).

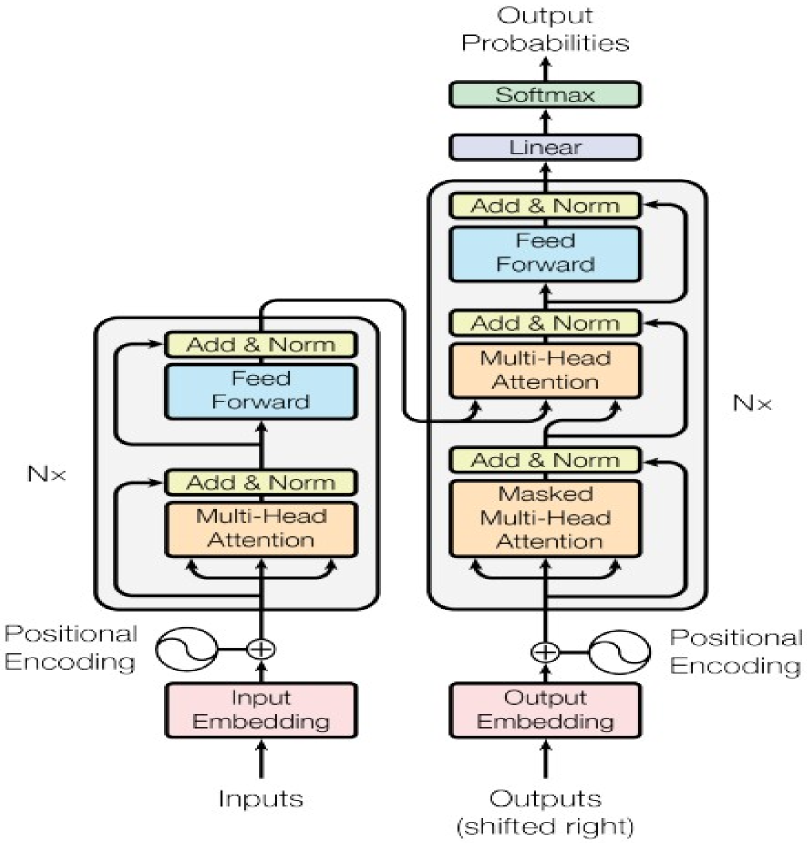


Figure 1: The Transformer - model architecture.

4.2 BERT

BERT (Devlin et al., 2019) is a machine learning framework for natural language processing (NLP) that is open source. BERT is a program that uses surrounding text to help computers grasp the meaning of ambiguous words in text. BERT is a transformer-based language model. which stands as Bidirectional Encoder Representations from Transformers.

So, as the name suggests, it is a way of learning representations of a language that uses a transformer, specifically, the encoder part of the transformer. BERT is only an encoder, while the original transformer is composed of an encoder and decoder. Given that BERT uses an encoder that is very similar to the original encoder of the transformer, we can say that BERT is a transformer-based model. So, BERT does not use recurrent connections, but only attention and feed-forward layers.

Language models could previously only interpret text input in one of two ways: left-to-right or right-to-left, but not at the same time. BERT is unique in that it can read in both directions at the same time. This capability, enabled by the introduction of Transformers, is known as bidirectionality.

BERT is aimed to pre-train deep bidirectional representations from unlabeled text by conditioning on both left and right context in all layers. As a result, the pre-trained BERT model may be fine-tuned with just one additional output layer to provide state-of-the-art models for a variety of tasks, such as question answering and language inference, without requiring significant task-specific architecture changes.

How Bert Works:

Google trained BERT on BooksCorpus (800 million words) and English Wikipedia (2.5 billion words). Masked Language Model (LM) and Next sentence prediction were the two training procedures (or ways in which BERT was trained). (Jacob Devlin, 2019)

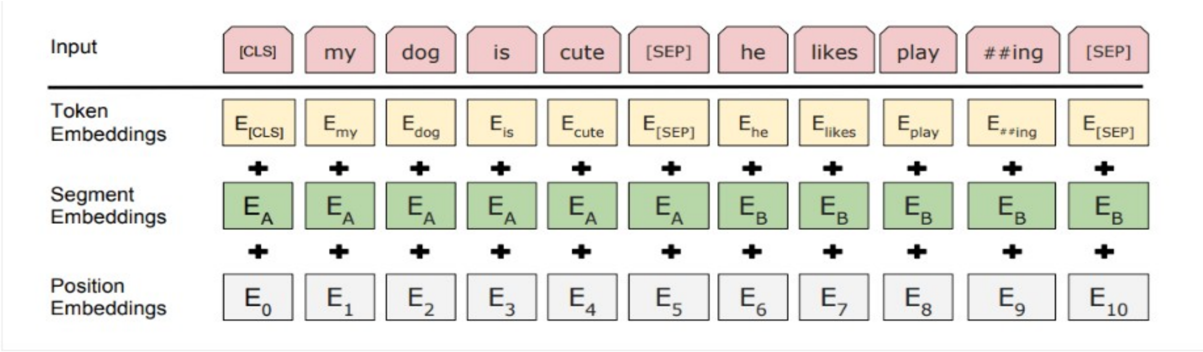
BERT is powered by a Transformer (the attention mechanism that learns contextual relationships between words in a text). A sequence of tokens is fed into the BERT encoder, which is subsequently transformed into vectors and processed by the neural network. However, before BERT can begin processing, the input must be altered and enhanced with additional metadata:

Token embeddings: A classification [CLS] token is added to the input word tokens at the beginning of the first sentence and a [SEP] token is inserted at the end of each sentence.

Segment embeddings: A marker indicating Sentence A or Sentence B is added to each token. This allows the encoder to distinguish between sentences.

Positional embeddings: A positional embedding is added to each token to indicate its position in the sentence. (Devlin et al., 2019)

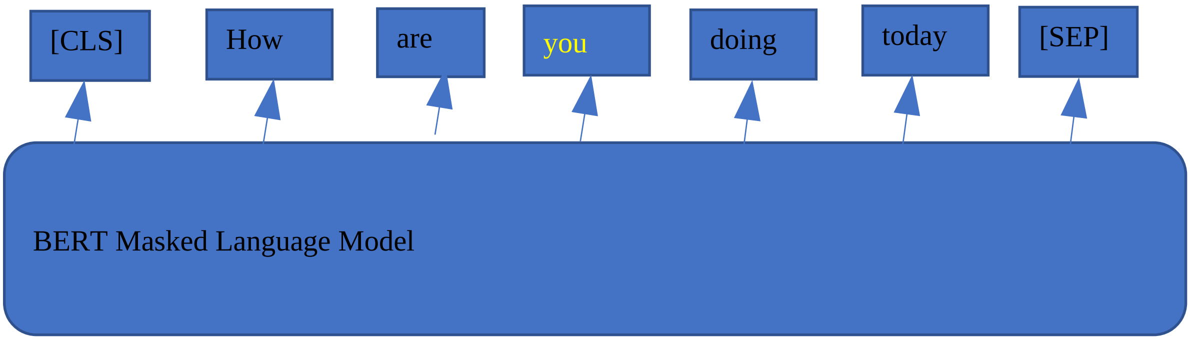
Output

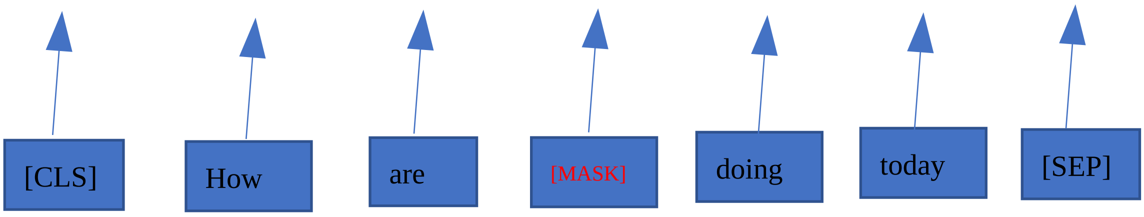

Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

BERT processing input with additional metadata

Masked ML:

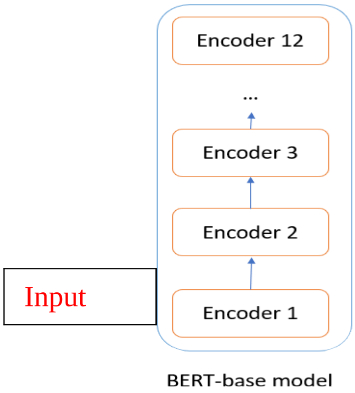
15 percent of the words in each sentence are substituted with a masked before being fed into BERT. This means that it is transformed into a "masked token." Then BERT's role is to guess the hidden or masked word in the phrase by looking at the surrounding words (non-masked words). Based on the context provided by the other, non-masked words in the sequence, the model then attempts to predict the original value of the masked words.





From above diagram, BERT calculates the probability of each word in the vocabulary and predicts the word that has the highest probability.

The BERT model has 12 transformer encoders block which are stacked on top of another. (Figure 1)



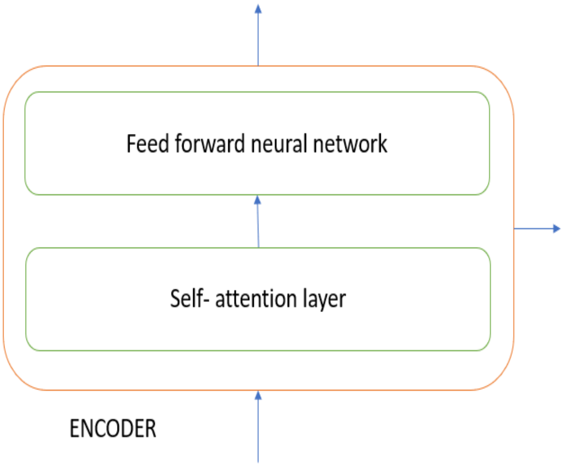


Figure 1 BERT model

Figure 2 Encoder architecture

Every encoder has the same structure and they do not share weights. Each encoder has two components: “Self-attention” and “feed-forward network”. The encoder's inputs pass through a self-attention layer, which allows the encoder to look at other words in the input sentence while encoding a single word. A feed-forward neural network receives the outputs of the self-attention layer. Each position uses the exact same feed-forward network. (See Figure 2)

BERT receives a series of words as input, which continue to flow up the stack. Each layer performs self-attention, transfers the results through a feed-forward network, and then passes the information on to the next encoder. (See Figure 3)

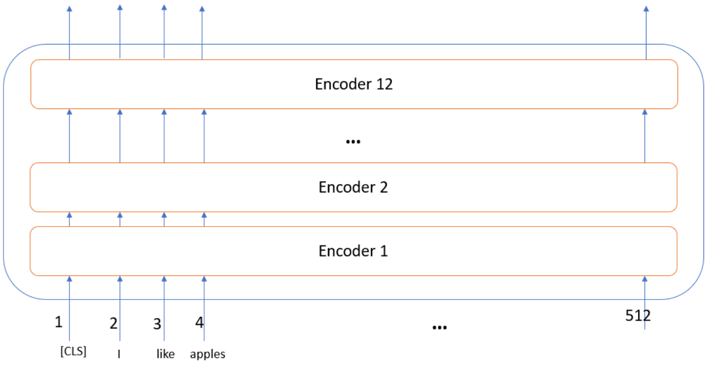


Figure 3 BERT architecture with inputs

Every position generates a hidden vector of length 768 (for *bert-base* model). Only the output of the first position is used to classify sentences, which is the output of the [CLS] token. That vector can now be used as input for any classifier we want. The BERT article uses a single-layer neural network as the classifier and obtains results. Figure 4 shows the BERT architecture with outputs.

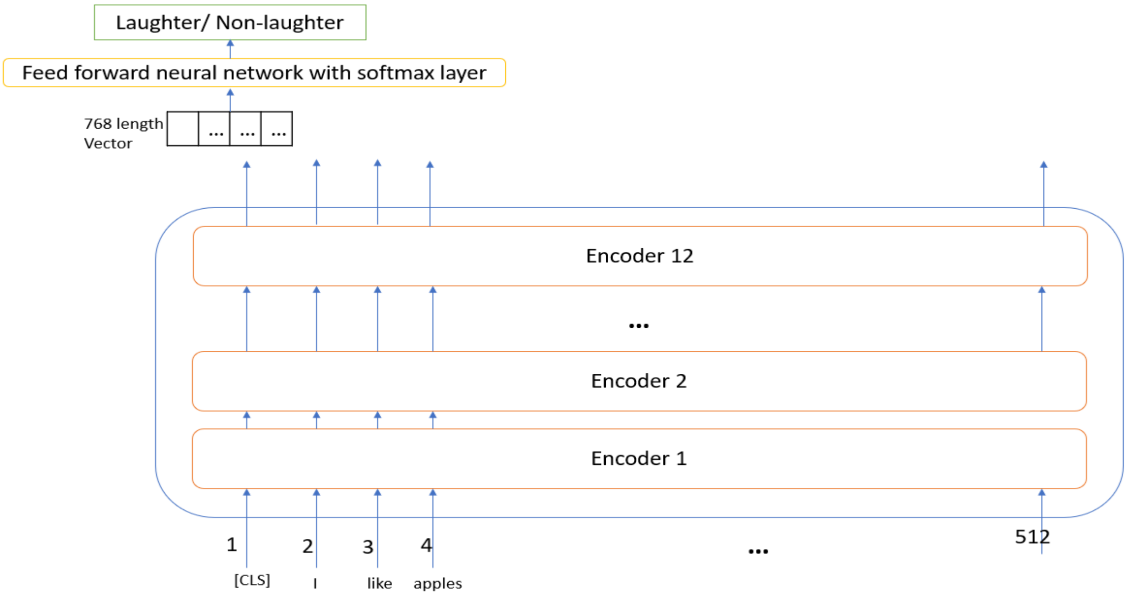
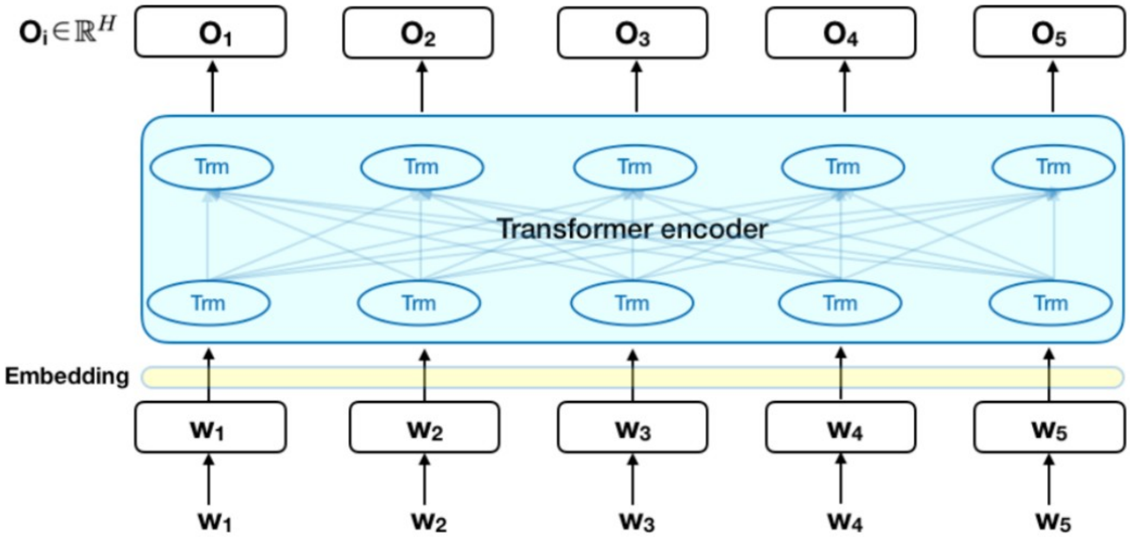


Figure 4 BERT architecture with output

The following visualization shows one layer of BERT. (Jacob Devlin, 2019)



w1, w2, w3, w4 and w4 are tokens (words in a sentence). These are converted into vectors which are taken from BERT vocab. Now it is passed through first layer of BERT which gives 5 output vectors. Notice, that inside BERT layer, how all words relate to each other. This means that if we consider O1 output vector, it not only has an influence of W1, rather all the W's has the influence on O1. All these influences are calculated by weights of the layer. This helps the BERT to learn the context or helps it to learn about the words that are around it. Not only O1, but all the O's has the influence of all the W's according to the weights. Now, if we do this 12 times, then BERTBASE is formed and if we do this 24 times BERTLARGE is formed.

5. Experiments

Huggingface transformer framework (Wolf et al., 2020) Datasets package automatically downloaded the SWDA dataset. The downloaded SWDA dataset has "train," "validation," and "test" which are provided by the Huggingface framework. These three datasets are combined to form a single dataset to create sliding window of tokens. These sliding windows of tokens are then labelled as laughter and non-laughter. Then we balance the frequency of laughter and non-laughter labels of the sliding window dataset. Using sklearn (Pedregosa F, et al., 2011), this balanced sliding window dataset was divided into train, validation, and test sets with a ratio of 80%, 10%, and 10%, respectively. Furthermore, this dataset has already been filtered to retain only the 43 required DAs. To make preprocessing easier we used pandas dataframe (Mckinney et al., 2011*)*.

Simple Transformers is a Natural Language Processing (NLP) library designed to simplify the usage of Transformer models. It is built on the amazing work of Hugging Face and their Transformers library. The language model class of the Simpletransformers (Rajapakse, T. C., 2019) python package was used to finetune the *bert-base-uncased* model on the open-subtitles dataset.

BERT is pre-trained on two independent but related NLP tasks using the bidirectional capability: Masked Language Modeling and Next Sentence Prediction.

The goal of Masked Language Model (MLM) training is to hide a word in a sentence and then have the program guess which word was hidden (masked) based on the context of the hidden word. The goal of Next Sentence Prediction training is to have the software predict whether two provided sentences have a logical, sequential connection or are merely random.

All the experiments were conducted with both 50 and 100 tokens sliding window with *bert-base-uncased* model.

The high-level architecture and low-level details for all four experiments done are as below:

i. In the first experiment we made the SWDA conversations into sliding window of tokens, without considering any dialogue act tags using the "*bert-base-uncased*" model. The learning rate was 2e-5, the train and eval batch sizes were 64, the weight decay was 0.01, the evaluation strategy was "epoch," and the warmup steps were 500. The outcomes of the test set with sliding window of size 50 are 83.6 percent f1 score and 84.4 percent accuracy. The accuracy was 79.3 percent and the f1 score was 80.5 percent when the model was trained for sliding window size of 100 tokens. (See Figure 5 – Experiment 1 below)

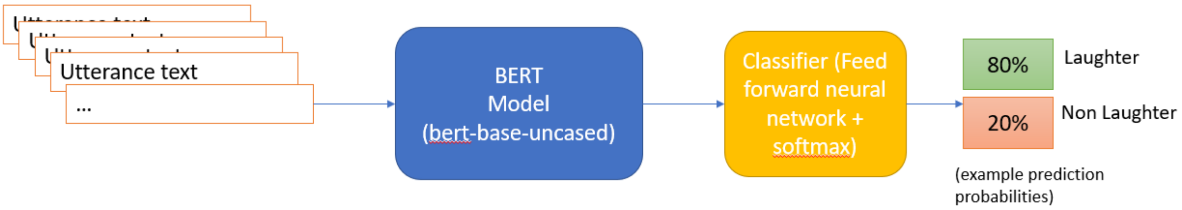

Utterance text
...
BERT Model (bert-base-uncased)
Classifier (Feed forward neural network + softmax)
80%
Laughter
20%
Non Laughter
(example prediction probabilities)

Figure 5 - Experiment 1: Bert without DA

ii. In the second experiment, sliding window dialogues are concatenated with dialogue act tags and then classified using *bert-base-uncased* model. In this approach, there is a slight change in data preprocessing step. Here *processed_text = processed_text + act_tag* taken before creating sliding window of dialogues. This approach gave test set evaluation accuracy = 79.9% and f1_score = 80.8% for sliding window of size 50. The test set results for sliding window size 100 are 79.7% and f1_score=80.6% (See Figure 6 – Experiment 2 below)

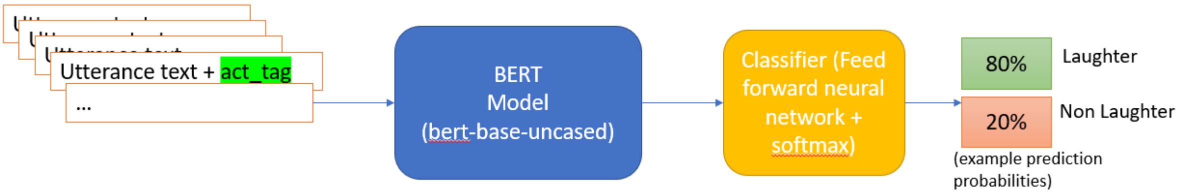

Utterance text + act_tag
...
BERT
Model
(bert-base-uncased)
Classifier (Feed forward neural network + softmax)
80%
Laughter
20%
Non Laughter
(example prediction probabilities)

Figure 6 -Experiment -2 : Bert with DA

iii. In the third experiment, sliding window of dialogues data are classified using *bert-base-uncased* model finetuned on “Open-subtitles” dataset and without any dialogue act tag information. The test evaluation results are f1_score = 83.5% and accuracy = 83% for sliding window size of 50. For sliding window of size 100, the accuracy = 81.7% and f1_score = 81.4%. (See Figure 7 – Experiment 3 below)

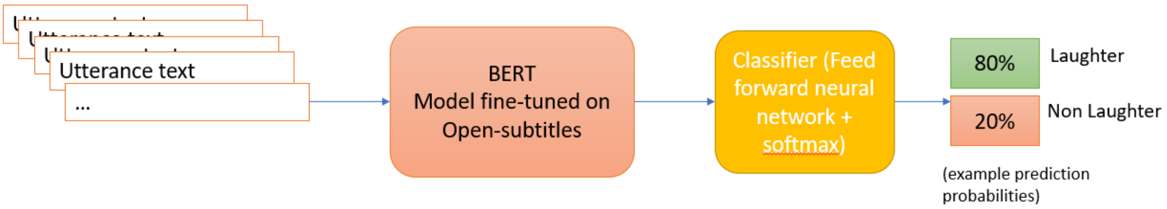

Utterance text
...
BERT
Model fine-tuned on
Open-subtitles
Classifier (Feed forward neural network + softmax)
80%
Laughter
20%
Non Laughter
(example prediction probabilities)

Figure 7 Experiment-3 : Bert finetuned on Opensubtitles and without DA

iv. Sliding window data concatenated with dialogue act tags and then classified using *bert-base-uncased* model finetuned on “Open-subtitles” dataset. The test set evaluation results are f1_score = 77.6 and accuracy = 76.9 for sliding window datasets of size 50. For size 100 of sliding window datasets, accuracy = 77.8 and f1_score = 79.1%. (See Figure 8– Experiment 4 below)

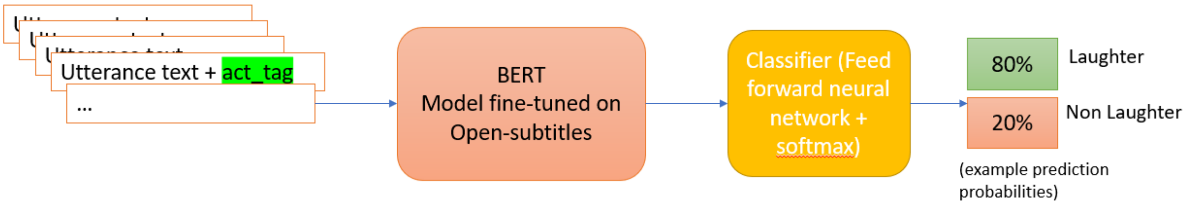
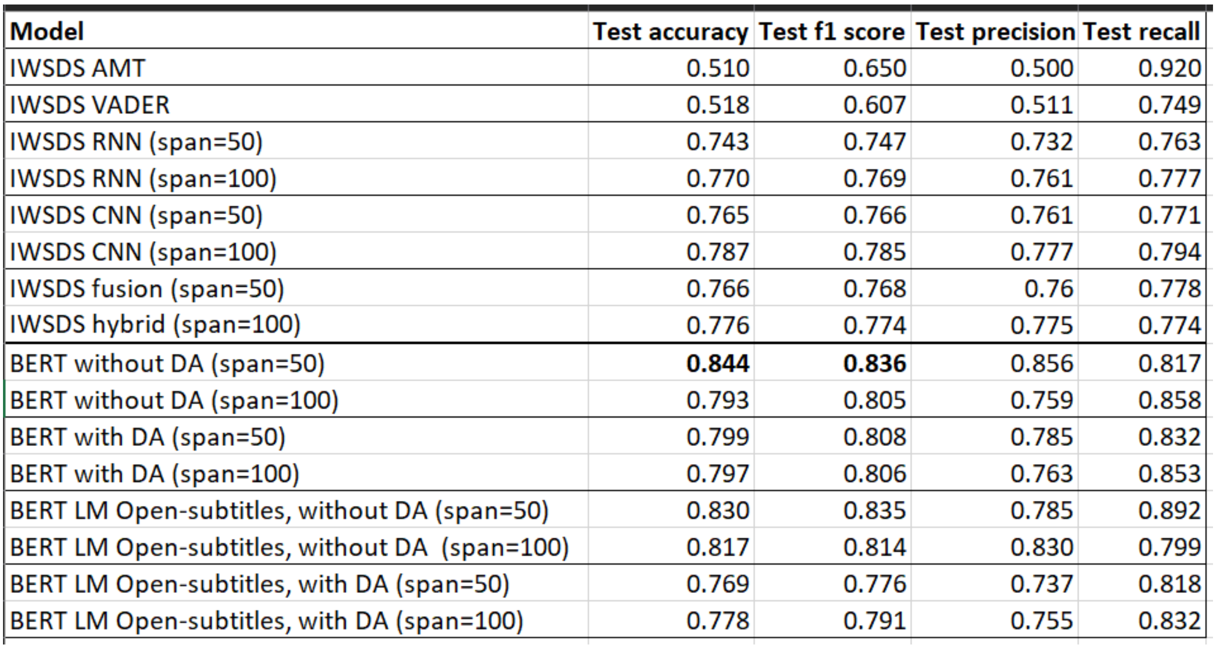


Figure 8 Experiment-4 : Bert finetuned on Opensubtitles and with DA

The results are compared with results from Maraev et al, 2021a which used CNN, RNN and combination of hybrid and fusion model for the same task.



Model	Test accuracy	Test f1 score	Test precision	Test recall
IWSDS AMT	0.510	0.650	0.500	0.920
IWSDS VADER	0.518	0.607	0.511	0.749
IWSDS RNN (span=50)	0.743	0.747	0.732	0.763
IWSDS RNN (span=100)	0.770	0.769	0.761	0.777
IWSDS CNN (span=50)	0.765	0.766	0.761	0.771
IWSDS CNN (span=100)	0.787	0.785	0.777	0.794
IWSDS fusion (span=50)	0.766	0.768	0.76	0.778
IWSDS hybrid (span=100)	0.776	0.774	0.775	0.774
BERT without DA (span=50)	**0.844**	**0.836**	0.856	0.817
BERT without DA (span=100)	0.793	0.805	0.759	0.858
BERT with DA (span=50)	0.799	0.808	0.785	0.832
BERT with DA (span=100)	0.797	0.806	0.763	0.853
BERT LM Open-subtitles, without DA (span=50)	0.830	0.835	0.785	0.892
BERT LM Open-subtitles, without DA (span=100)	0.817	0.814	0.830	0.799
BERT LM Open-subtitles, with DA (span=50)	0.769	0.776	0.737	0.818
BERT LM Open-subtitles, with DA (span=100)	0.778	0.791	0.755	0.832

Figure 9 Test data evaluation results

6. Discussion

Effect of Sliding Window size:

The BERT model was not created with conversational data in mind. While the dataset we used for training comprises two persons conversing and taking turns in the same chat. As a result, this model frequently switches points of view. The SWDA also has disfluency such as 'oh, um, pause, silence', and so on. Furthermore, the sentence forms in SWDA differs from what BERT was originally trained on. SWDA has conversational data with some disfluency terms while BERT was trained on Wikipedia and BooksCorpus.

For words that are not in the original Bert model vocabulary, the "*bert-base-uncased*" model performs sub-word tokenization. Common words should not be broken down into smaller sub words, but rare words should be broken down into meaningful sub words, according to sub word tokenization procedures. For example, "provoking" is a rare word that may be broken down into "pro" and "###voking", Both "pro" and "###voking" would appear more frequently as standalone sub words, while the meaning of "provoking" would be preserved by the composite meaning of "pro" and "###voking".

The line "Recently I'm reading the great Gatsby, its really thought provoking" is broken down into ['recently', 'im','reading', 'the', 'great', 'gas', '##t', '##by', 'its', 'really', 'thought', 'pro', '##voking'] by the "*bert-base-uncased*" model. As you can see, the terms "Gatsby" and "provoking" have been broken down into sub-words. BERT employs a word piece tokenizer, which can handle up to 512 tokens per observation. When we utilize lengthier text, such as 100 token sliding window, the BERT model truncates all tokens after 512 tokens. As a result, substantial sections of text are lost in each sliding window, and therefore context is lost.

Many words in our sliding window dataset are not included in the BERT model's vocabulary. It's possible that BERT model sub-word tokenization of a 100-word text, the list of sub tokens exceeds 512, causing us to lose context. As a result, the performance of the 100 tokens sliding window was inferior to that of 50 tokens sliding window.

To confirm this, we have performed additional experiments using 25 tokens of sliding window size and 150 tokens of sliding window size. The results are according to our hypothesis. The BERT model performs better with a lesser token sliding window size

Model	**Test Accuracy**	**Test f1 Score**	**Test Precision**	**Test Recall**
BERT without DA (span=25)	0.915	0.917	0.910	0.925
BERT without DA (span=150)	0.790	0.801	0.779	0.844
BERT with DA (span=25)	0.905	0.901	0.923	0.916
BERT with DA (span=150)	0.789	0.779	0.757	0.771

Effect of Dialogue Act tags:

We hypothesized that combining Dialogue Acts with actual dialogues would significantly increase the performance of BERT. When we combine the names of dialogue acts with the text of a speaker from SWDA, the context moves slightly, and the position of laughter tokens shifts. For example, if a sliding window text ends with the *laughter* token and the dialogue acts are concatenated after that, the laughter token is suppressed in this sample and it is not tagged as laughter, even if the sample was initially laughter. Similarly, if the dialogue acts are concatenated before the actual text of speaker, the laughter token which is at the end of sliding window is shifted on to the right and this sample is not tagged as laughter, even if the sample was initially laughter. So, this affects the performance of BERT while using the dialogue acts.

Also Maraev et al., 2021 b shown that laughter is a valuable cue for Dialogue Act Recognition (DAR) task. Laughter is particularly useful in distinguishing between literal and non-literal meaning. Mazzocconi et al. 2020 coined the term pragmatic incongruity to describe the presence of an incongruity between what is said and what is intended. Laughter can aid in pragmatic disambiguation and help a computer model assign meaning to an utterance. According to the findings, vocalizations such as laughter are more pronounced in SWDA, making them more useful in disambiguating discourse activities. So, the vice-versa that Dialogue Acts add more disambiguation in detecting laughter in SWDA is true. Hence our hypothesis that Dialogue Acts would increase the performance of BERT in detecting laughter in SWDA was false.

Fine-Tuning BERT:

On 2 million lines of text from Open subtitles, the BERT model was finetuned. We used simple transformers task based *LanguageModelingModel* class for fine-tuning 'BERT' model on open subtitles. The resulted language model is used for training, evaluation and testing of the SWDA. The finetuned language model on Open subtitles creates a Masked Language Model fine-tuned on BERT model and does not follow the format we are utilizing, such as producing sliding windows and labelling the produced sliding window datasets. As a result, the Bert model struggles to adapt to this type of conversational data. This is the reason we see the performance of BERT fine tuned on Open subtitles is inferior to actual BERT model.

Effect of Speaker:

Sometimes some characters are more humorous than others. They are being one of the speakers in the dialogue could potentially affect the dialogue's chance of becoming humorous/non-humorous and generate laughter. B. N. Patro, 2021 study proves Combining the textual dialogues with speakers increased the accuracy to 67.26%.

Thus, to study the effect of this modality using the SWDA dataset other than the dialogues alone, we chose to experiment while providing the model with both the textual dialogues and the speakers. There are total 1115 transcripts in SWDA dataset with each transcript constituting of two speakers, always named as 'A' and 'B' in each transcript. The speaker's name is appended to their respective dialogue, before converting the combined text to sliding windows of tokens. The BERT performance had significant increase when the SWDA textual dialogues are combined with a speaker and without Dialogue Acts. In SWDA the speaker has very little effect on the laughter prediction when Dialogue Acts are used in addition.

Model	Test Accuracy	Test f1 Score	Test Precision	Test Recall
BERT with speaker and without DA (span=50)	0.947	0.947	0.943	0.951
BERT with speaker and with DA (span=50)	0.798	0.808	0.763	0.858

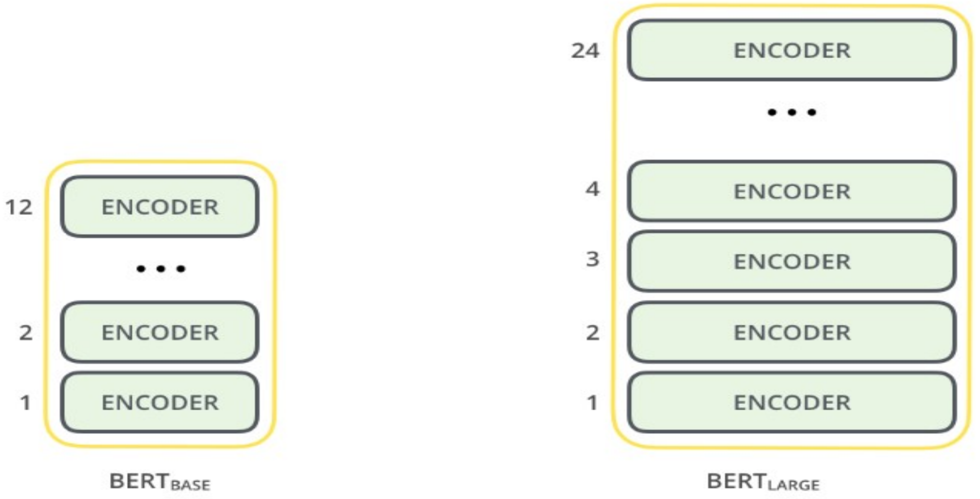
We may conclude that the BERT model for the task of classifying laughter detection on the SWDA dataset delivers better performance. Several experiments with a BERT like 50 token sliding window without DA, 25 token sliding window without DA, 50 token sliding window with speaker and without DA, outperforms the top performing model in the IWSDS publication, which is a CNN model with 100 token sliding window. This is a significant increase in efficiency and proves our hypothesis that a BERT model would outperform the Convolution Neural Network (CNN) and Recurrent Neural Network (RNN) models presented in the IWSDS publication.

6.1 Future Work

BERT base vs BERT large:

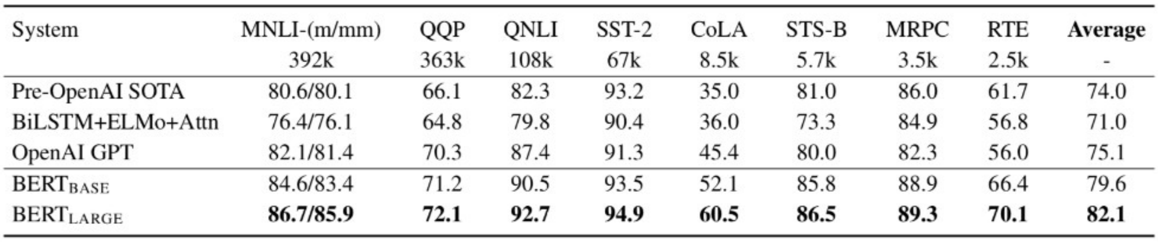
BERT is based on encoder layers that are stacked one on top of the other. The number of encoder layers is the difference between BERT base and BERT large. The BERT base model has 12 encoder layers layered on top of one another, but the BERT large model has 24 encoder layers placed on top of one another.

The image below shows standard two different BERT: BERT base and BERT large.



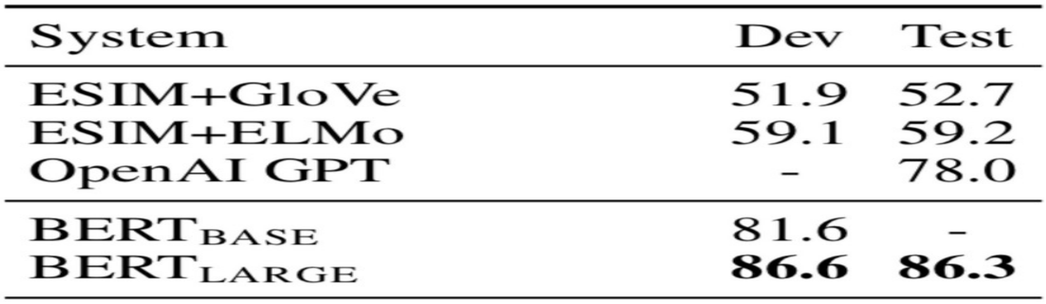
The number of parameters (weights) and attention heads rise as the number of layers in the BERT large increases. The BERT base has 110 million parameters and 12 attention heads (which allow each token in the input to focus on other tokens). BERT large, on the other hand, contains 16 attention heads and 340 million parameters. The BERT base contains 768 hidden layers, whereas the BERT large has 1024.

As we can see in the image below, the results of GLUE benchmarks show us that the BERT performs better than the other models. And BERT large increases the performance of BERT base further. So, using BERT large instead of BERT base could have significantly increased our performance in the laughter prediction using SWDA dataset.



System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
BERT$_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT$_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

The image below shows SWAG Dev and test accuracy. BERT beats other models where BERT large performs better than BERT base.



System	Dev	Test
ESIM+GloVe	51.9	52.7
ESIM+ELMo	59.1	59.2
OpenAI GPT	-	78.0
BERT$_{\text{BASE}}$	81.6	-
BERT$_{\text{LARGE}}$	**86.6**	**86.3**

These results also cement the claim that increasing the model size would lead to the improvement in results. Although the large model performs better, fine tuning and training such a model is difficult and requires a lot of horsepower.

Masked Language Modelling VS Binary Classification Model

We used simple transformers task based *LanguageModelingModel* class for fine-tuning 'BERT' model on open subtitles. The resulted language model is used for training, evaluation and testing of the SWDA. The finetuned language model on Open subtitles creates a Masked Language Model fine-tuned on BERT model and does not follow the format we are utilizing, such as producing sliding windows. We hypothesize that following the same pattern and creating a Classification model instead of a Masked Language model might increase the performance of BERT fine-tuned on open subtitles. To, accomplish so, we had to first identify all the laughter's and synonyms from the Open Subtitles dataset, and then build sliding window of tokens data and label them like SWDA. We can then use the simple transformers task based *ClassificationModel* class for fine-tuning BERT model on open subtitles and create a binary classification model with two classes instead of a Masked Language Model. For making predictions on other text, *TransformerModel* comes with a *predict(to_predict)* method which given a list of sliding windows of tokens, returns the model predictions and the raw model outputs. Using a Binary Classification Model rather than a Masked LM might increase the performnace of BERT fine-tuned on open sibtitles.

7. References

Stolcke, A., Ries, K., Coccaro, N., Shriberg, E., Bates, R., Jurafsky, D., ... & Meteer, M. (2000). Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, *26*(3), 339-373.

John Langshaw Austin. 1975. How to do things with words, volume 88. Oxford university press.

Maraev, V., Howes, C., & Bernardy, J. P. (2021). Predicting Laughter Relevance Spaces in Dialogue. In *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction* (pp. 41-51). Springer, Singapore.

Maraev, V., Noble, B., Mazzocconi, C., & Howes, C. (2021, September). Dialogue act classification is a laughing matter. In *Proceedings of the 25th Workshop on the Semantics and Pragmatics of Dialogue*.

Mazzocconi, C., Tian, Y., & Ginzburg, J. (2020). What's your laughter doing there? A taxonomy of the pragmatic functions of laughter. IEEE Transactions on Affective Computing.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding.

Chen, L., & Lee, C. (2017, September). Predicting audience's laughter during presentations using convolutional neural network. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications* (pp. 86-90).

Lala, Divesh, Koji Inoue and Tatsuya Kawahara (2020). "Prediction of Shared Laughter for Human-Robot Dialogue". In: *Companion Publication of the 2020 International Conference on Multimodal Interaction*. ICMI '20 Companion. Virtual Event, Netherlands: Association for Computing Machinery, pp. 62–66.

D Jurafsky, E Shriberg, and D Biasca. 1997a. Switchboard dialog act corpus. International Computer Science Inst. Berkeley CA, Tech. Rep.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, 2019*. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding"* on ArXiv: https://arxiv.org/abs/1810.04805

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018*. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* arXiv:1810.04805 [cs].

Daniel Jurafsky, Liz Shriberg, and Debra Biasca. 1997b. *Switchboard SWBD-DAMSL Shallow Discourse-Function Annotation Coders Manual.*

Merlin Teodosia Suarez, Jocelynn Cu, and Madelene Sta. Maria. 2012. *Building a Multimodal Laughter Database for Emotion Recognition. In Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12), pages 2347–2350, Istanbul, Turkey. European Language Resources Association (ELRA).*

Wolf, Thomas, Debut, Lysandre, Sanh, Victor, Chaumond, Julien, Delangue, Clement, Moi, Anthony, Cistac, Pierric, Rault, Tim, Louf, Rémi, Funtowicz, Morgan, Davison, Joe, Shleifer, Sam, Platen, Patrick von, Ma, Clara, Jernite, Yacine, Plu, Julien, Xu, Canwen, Scao, Teven Le, Gugger, Sylvain, Drame, Mariama, Lhoest, Quentin, and Rush, Alexander M. "Transformers: State-of-theArt Natural Language Processing". In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. url: https://www.aclweb.org/anthology/2020.emnlpdemos.6.

Mckinney, Wes. “Pandas: A Foundational Python Library for Data Analysis and Statistics”. Python for High-Performance and Scientific Computing, Volume 14, 2011.

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine Learning in Python. vol. 12. Microtome Publishing; 2011. p. 2825–2830.

Rajapakse, T. C., 2019; Simple Transformers: https://github.com/ThilinaRajapakse/simpletransformers

B. N. Patro, M. Lunayach, D. Srivastava, S. Sarvesh, H. Singh and V. P. Namboodiri, "Multimodal Humor Dataset: Predicting Laughter tracks for Sitcoms," *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2021, pp. 576-585, doi: 10.1109/WACV48630.2021.00062.