# NLP Final Project Report

Jainam Shroff
jshro046@uottawa.ca

Honey Patel
hpate142@uottawa.ca

Aagaypal Kaur
akaur041@uOttawa.ca

Megha Agarwal
magar098@uottawa.ca

## ABSTRACT

Before understanding this project as a whole we need to first understand the technicalities behind it. Broadly speaking, the project is based on sequential sentence classification, firstly, In NLP, sentence classification is organizing, structuring, and categorizing multiple kinds of text in a sentence, from documents, medical studies and files to all types of content over the web. But a sequential sentence classification is the task of classifying the sentence that has a specific sequence (in other words, the sequence in which the sentence occurs in the dataset matters too). Some of the applications related to sentence classification are sentiment analysis and question answering.

In this project, our focus is on classifying sentences in medical abstracts, more specifically for RCTs (randomized controlled trials), as they are commonly considered to be the best source of medical evidence. Since sentences in an abstract appear in a sequence (and has an impact on the model), we call this task the sequential sentence classification task, in order to distinguish it from general text or sentence classification that does not have any context. The goal of this project in terms of the output is to explore the ability of NLP models to classify sentences which appear in sequential order. By structured abstract it means that it must contain between 3 to 9 sections. The label of each section is given by the authors of the article, typically by following the guidelines given by the journals. As many labels exist, PubMed maps them into a smaller set of standardized labels which includes background, objective, methods, results, conclusions, “None”, “Unassigned”, or “” (empty string).

## KEYWORDS

Natural Language processing, Joint sentence classification, PubMed 200k RCT.

## OBJECTIVE

The main objective of this project is to replicate the architecture of the model that is powering the Neural Networks for Joint Sentence Classification in Medical Paper Abstracts research paper.

## MOTIVATION

When performing any experiment or while working on any project there is always some underlying motivation behind it. For this project, the main motivation was that the number of RCTs publications is increasing year on year basis.

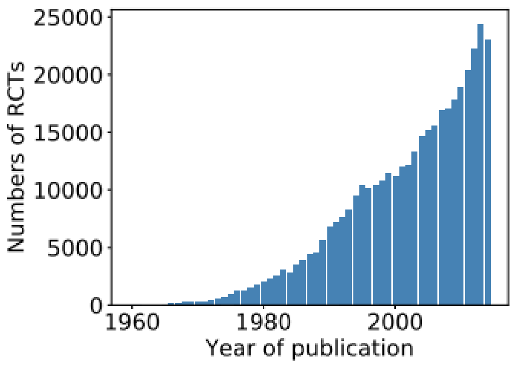


**Figure 1**

As we can see from the graph above, the rate of publication is increasing on a yearly basis.

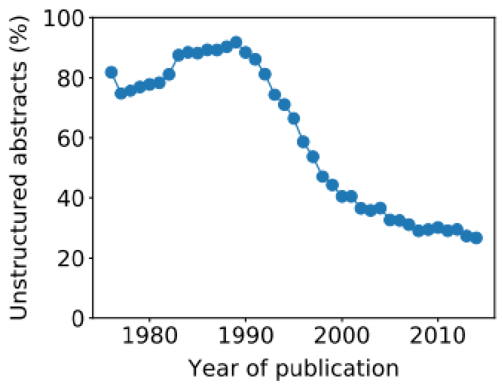


**Figure 2**

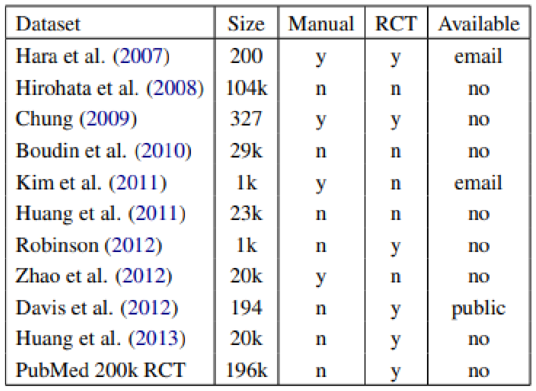
From the graph above we can see that not only publication rate has increased but the amount of research papers that have unstructured abstracts is close to 50% in the year 2014. Since nearly half of all the published papers have unstructured abstracts, this project can solve the problem by allowing people to skim through the abstracts easily.

Other than the rate of publications and the number of unstructured abstracts being published other motivation to pursue this project is:

- Existing models based on artificial neural networks (ANNs) for sentence classification often do not incorporate the context in which sentences appear and classify sentences individually.
- However, traditional sentence classification approaches have been shown to greatly benefit from jointly classifying subsequent sentences, such as with conditional random fields.
- we present an ANN architecture that combines the effectiveness of typical ANN models to classify sentences in isolation, with the strength of structured prediction.

## PREVIOUS WORK

Existing datasets for classifying sentences in medical abstracts are either small, not publicly available, or do not focus on RCTs. Here we can see the existing datasets for sentence classification. The dataset that we are working on currently is the last one in the table shown below called PubMed 200k RCT



| Dataset | Size | Manual | RCT | Available |
|---|---|---|---|---|
| Hara et al. (2007) | 200 | y | y | email |
| Hirohata et al. (2008) | 104k | n | n | no |
| Chung (2009) | 327 | y | y | no |
| Boudin et al. (2010) | 29k | n | n | no |
| Kim et al. (2011) | 1k | y | n | email |
| Huang et al. (2011) | 23k | n | n | no |
| Robinson (2012) | 1k | n | y | no |
| Zhao et al. (2012) | 20k | y | n | no |
| Davis et al. (2012) | 194 | n | y | public |
| Huang et al. (2013) | 20k | n | y | no |
| PubMed 200k RCT | 196k | n | y | no |

Table 1: Overview of existing datasets for sentence classification in medical abstracts. The size is expressed in terms of number of abstracts.

**Figure 3**

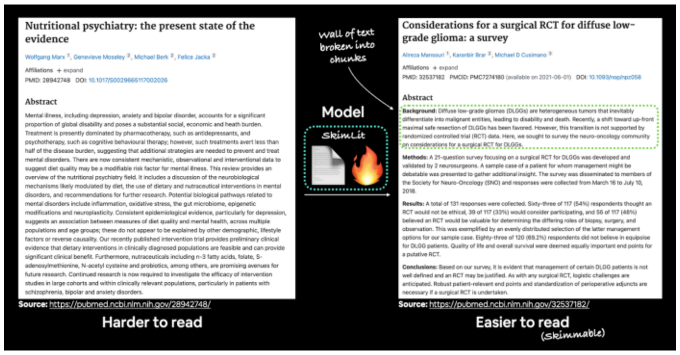
**In Terms of Modelling Experiment.**

Existing systems for sequential sentence classification are mostly based on naive Bayes, support vector machines (SVMs), Hidden Markov models (HMMs), and conditional random fields (CRFs). These often require numerous hand-engineered features based on lexical, semantic (synonyms, hyponyms), structural (part-of-speech tags, headings), and sequential (sentence position, surrounding features) information.

Moreover, recent approaches to natural language processing (NLP) based on artificial neural networks (ANNs) do not require manual features, as they are trained to automatically learn features based on word as well as character embeddings. ANN-based models have achieved state-of-the-art results on various NLP tasks. For short-text classification, many ANN models use word embeddings.

## EXAMPLE

Here is a visual example of how the model should work at the end



**Nutritional psychiatry: the present state of the evidence**

Wolfgang Marx [1], Genevieve Moseley [2], Michael Berk [2], Felice Jacka [2]

Affiliations + expand

PMID: 28942748 DOI: 10.1017/S0029665117002026

**Abstract**

Mental illness, including depression, anxiety and bipolar disorder, accounts for a significant proportion of global disability and poses a substantial social, economic and heath burden. Treatment is presently dominated by pharmacotherapy, such as antidepressants, and psychotherapy, such as cognitive behavioural therapy; however, such treatments avert less than half of the disease burden, suggesting that additional strategies are needed to prevent and treat mental disorders. There are now consistent mechanistic, observational and interventional data to suggest diet quality may be a modifiable risk factor for mental illness. This review provides an overview of the nutritional psychiatry field. It includes a discussion of the neurobiological mechanisms likely modulated by diet, the use of dietary and nutraceutical interventions in mental disorders, and recommendations for further research. Potential biological pathways related to mental disorders include inflammation, oxidative stress, the gut microbiome, epigenetic modifications and neuroplasticity. Consistent epidemiological evidence, particularly for depression, suggests an association between measures of diet quality and mental health, across multiple populations and age groups; these do not appear to be explained by other demographic, lifestyle factors or reverse causality. Our recently published intervention trial provides preliminary clinical evidence that dietary interventions in clinically diagnosed populations are feasible and can provide significant clinical benefit. Furthermore, nutraceuticals including n-3 fatty acids, folate, S-adenosylmethionine, N-acetyl cysteine and probiotics, among others, are promising avenues for future research. Continued research is now required to investigate the efficacy of intervention studies in large cohorts and within clinically relevant populations, particularly in patients with schizophrenia, bipolar and anxiety disorders.

**Source:** https://pubmed.ncbi.nlm.nih.gov/28942748/

**Considerations for a surgical RCT for diffuse low-grade glioma: a survey**

Alireza Mansouri [1], Karanbir Brar [2], Michael D Cusimano [3]

Affiliations + expand

PMID: 32537182 PMCID: PMC7274180 (available on 2021-06-01) DOI: 10.1093/nop/npz058

**Abstract**

**Background:** Diffuse low-grade gliomas (DLGGs) are heterogeneous tumors that inevitably differentiate into malignant entities, leading to disability and death. Recently, a shift toward up-front maximal safe resection of DLGGs has been favored. However, this transition is not supported by randomized controlled trial (RCT) data. Here, we sought to survey the neuro-oncology community on considerations for a surgical RCT for DLGGs.

**Methods:** A 21-question survey focusing on a surgical RCT for DLGGs was developed and validated by 2 neurosurgeons. A sample case of a patient for whom management might be debatable was presented to gather additional insight. The survey was disseminated to members of the Society for Neuro-Oncology (SNO) and responses were collected from March 16 to July 10, 2018.

**Results:** A total of 131 responses were collected. Sixty-three of 117 (54%) respondents thought an RCT would not be ethical, 39 of 117 (33%) would consider participating, and 56 of 117 (48%) believed an RCT would be valuable for determining the differing roles of biopsy, surgery, and observation. This was exemplified by an evenly distributed selection of the latter management options for our sample case. Eighty-three of 120 (69.2%) respondents did not believe in equipoise for DLGG patients. Quality of life and overall survival were deemed equally important end points for a putative RCT.

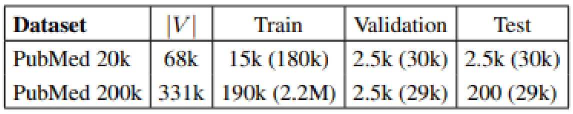
**Conclusions:** Based on our survey, it is evident that management of certain DLGG patients is not well defined and an RCT may be justified. As with any surgical RCT, logistic challenges are anticipated. Robust patient-relevant end points and standardization of perioperative adjuncts are necessary if a surgical RCT is undertaken.

**Source:** https://pubmed.ncbi.nlm.nih.gov/32537182/

**Figure 4**

## DATASET ANALYSIS

The dataset has 195,654 abstracts and is randomly split into three sets training, validation and testing. The validation set contains 2.5k abstracts and the same for the testing set the remaining 190,654 abstracts are in the training set. Dataset with 10% of total abstracts is also available for reducing time gaps between experimentation (model training). From the table below we can see the dataset split information along with the information on the vocabulary size used.

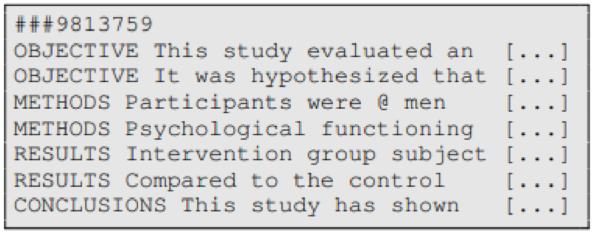


| Dataset | $\lvert V\rvert$ | Train | Validation | Test |
|---|---|---|---|---|
| PubMed 20k | 68k | 15k (180k) | 2.5k (30k) | 2.5k (30k) |
| PubMed 200k | 331k | 190k (2.2M) | 2.5k (29k) | 200 (29k) |

Table 2: Dataset overview. $|V|$ denotes the vocabulary size. For the train, validation and test sets, we indicate the number of abstracts followed by the number of sentences in parentheses.

**Figure 5**

The dataset is provided as three text files: one for the training set, one for the validation set, and one for the test set. Each file has the same format: each line corresponds to either a PMID or a sentence with its capitalized label at the beginning. Each token is separated by a space. Listing 1 shows an excerpt from these files. Below is an example of the same.



```
###9813759
OBJECTIVE This study evaluated an    [...]
OBJECTIVE It was hypothesized that [...]
METHODS Participants were @ men      [...]
METHODS Psychological functioning    [...]
RESULTS Intervention group subject [...]
RESULTS Compared to the control      [...]
CONCLUSIONS This study has shown     [...]
```

Listing 1: Example of one abstract as formatted in the PubMed 200k RCT dataset set. The PMID of the corresponding article is 9813759; the article can be found that https://www.ncbi.nlm.nih.gov/pubmed/9813759.

**Figure 6**

The dataset is provided as three text files for training, validation and test set. Each file has the same format: each line corresponds to either a PMID or a sentence with its capitalized label at the beginning. Each token is separated by a space.

For each abstract, sentence and token boundaries are detected using the Stanford CoreNLP toolkit. Stanford CoreNLP toolkit, an extensible pipeline that provides core natural language analysis. This toolkit is quite widely used, both in the research NLP community and also among commercial and government users of open-source NLP technology

Two versions of dataset are provided: one with the original text and the other where digits are replaced by the @ (at the rate) sign.

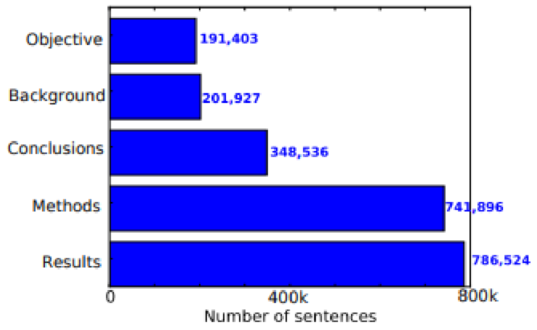


**Figure 7**

The above shown figure counts the number of sentences per label: the least common label (objective) is at about 200k which is approximately four times less frequent than the most common label (results) at around 800k, this indicates that the dataset is not excessively unbalanced.

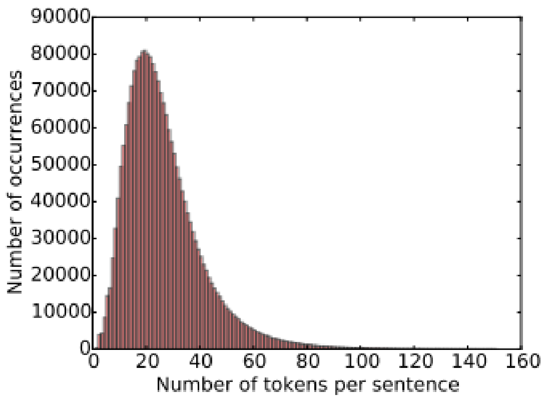


**Figure 8**

The above shown figure illustrates the distribution of the number of tokens per sentence. The graph follows normal distribution and thus we can say that data near the mean with a value of 26.2 is more frequent in occurrence than data far from mean.

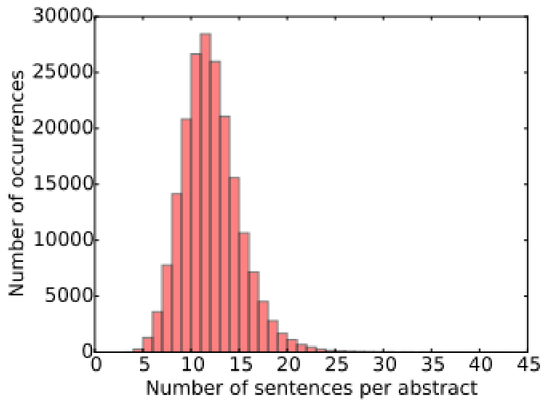


**Figure 9**

The above shown figure illustrates the distribution of the number of sentences per abstract. Again, the graph follows normal distribution. The min and max value are 3 and 51 respectively

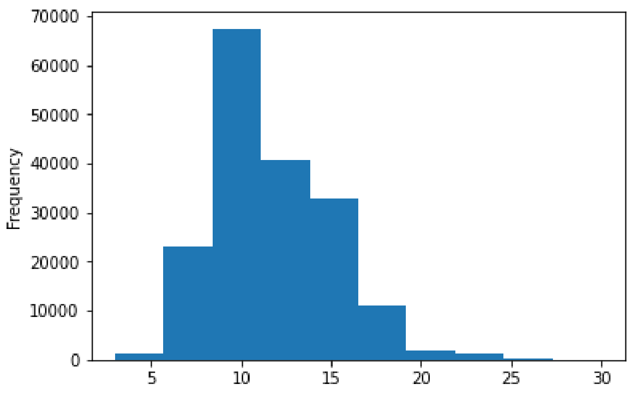


**Figure 10**

From the above shown graph we can see that most of the abstracts are around 7 to 15 sentences in length.

## PROBLEM STATEMENT

The number of RCT papers released is continuing to increase, those without structured abstracts can be hard to read and in turn, slow down researchers moving through the literature, Our NLP model will solve this problem and thus save time and improve overall convenience.

## METHODOLOGY

In total 5 modelling experiments are performed as mentioned:

Model 0: Multinomial Naïve Bayes (Baseline)
Model 1: Conv1D with token embedding.
Model 2: Feature extraction with pre-trained token embedding.
Model 3: Conv1D with character embedding.
Model 4: Combining Pretrained token embeddings + character embedding (hybrid).
Model 5: Transfer Learning with pre-trained token embeddings + character embeddings + positional embeddings

These models' performance is tested and compared based on the 10% dataset.

### Model 0: Multinomial Naïve Bayes (Baseline)

The first model is a TF-IDF Multinomial Naive Bayes as recommended by Scikit-Learn's machine learning map.

To build it, we'll create a Scikit-Learn Pipeline which uses the TfidfVectorizer class to convert our abstract sentences to numbers using the TF-IDF (term frequency-inverse document frequency) algorithm and then learns to classify our sentences using the Multinomial Naïve Bayes algorithm. Due to the speed of the Multinomial Naive Bayes algorithm, it trains very quickly.

### Preprocessing steps for deep learning experiments

For the deep learning models, we've got to create vectorization and embedding layers. The vectorization layer converts the text to numbers and the embedding layer captures the relationships between those numbers. The token vectorization layer maps the words in the text directly to numbers. However, this doesn't necessarily capture the relationships between those numbers.

To create a richer numerical representation of our text, we have used an embedding. As our model learns (by going through many different examples of abstract sentences and their labels), it'll update its embedding to better represent the relationships between tokens in the corpus.

However, there are still a few steps we implemented to make them work faster with our models. The main step used with our data is to turn it into a Prefetch Dataset of batches. In doing so we ensured that TensorFlow loads our data onto the GPU as fast as possible, which in turn, lead to faster training time. To create a batched Prefetch Dataset we used the methods batch () and prefetch (), the parameter tf.data.AUTOTUNE will also allow TensorFlow to determine the optimal amount of computing to use to prepare datasets.

All our deep models follow a similar structure:
Input (text) -> Tokenize -> Embedding -> Layers -> Output (label probability)

**Model 1: Conv1D with token embeddings**

The first model we're going to build is a 1-dimensional Convolutional Neural Network. Below is the pictorial representation of the architecture.

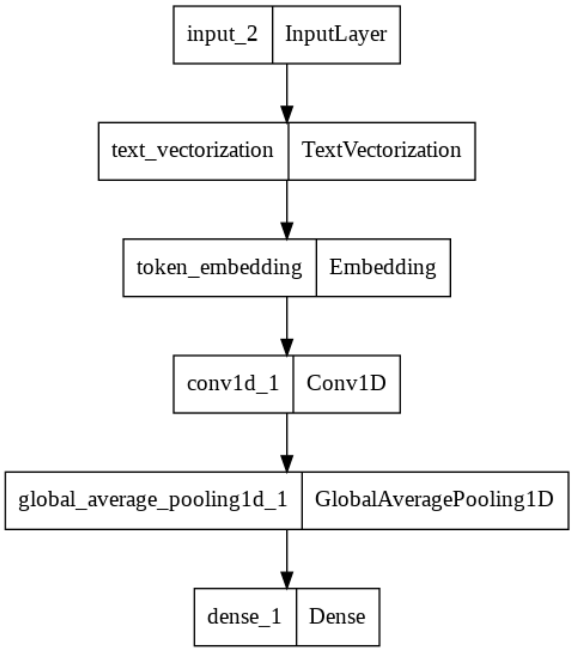

input_2 | InputLayer
text_vectorization | TextVectorization
token_embedding | Embedding
conv1d_1 | Conv1D
global_average_pooling1d_1 | GlobalAveragePooling1D
dense_1 | Dense

**Figure 11**

**Model 2: Feature extraction with pre-trained token embeddings**

We used the pre-trained Universal Sentence Encoder embeddings from TensorFlow Hub. We can notice the lack of tokenization layer we've used in a previous model. This is because the Universal Sentence Encoder (USE) already has a tokenization layer built into it. This type of model is called transfer learning, or more specifically, feature extraction transfer learning. In other words, taking the patterns a model has learned elsewhere and applying them to our own problem. Below is the pictorial representation of the architecture.

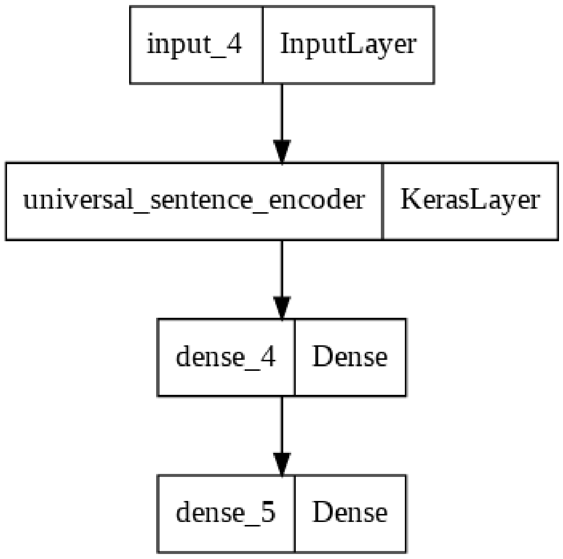


**Figure 12**

**Model 3: Conv1D with character embeddings**

We've already built models with a custom token embedding and a pre-trained token embedding, how about we build one using a character embedding? The difference between a character and token embedding is that the character embedding is created using sequences split into characters (e.g., hello -> [h, e, l, l, o]) whereas a token embedding is created on sequences split into tokens. We created a character-level embedding by first vectorizing our sequences (after they've been split into characters) using the Text Vectorization class and then passed those vectorized sequences through an Embedding layer.

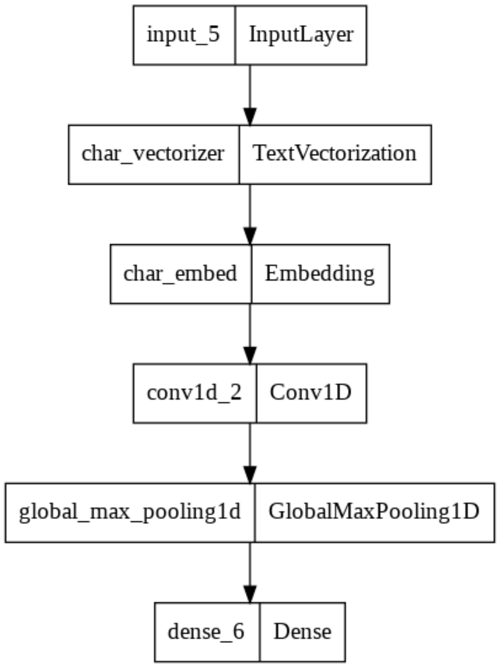

input_5 | InputLayer
char_vectorizer | TextVectorization
char_embed | Embedding
conv1d_2 | Conv1D
global_max_pooling1d | GlobalMaxPooling1D
dense_6 | Dense

**Figure 13**

**Model 4: Combining pre-trained token embeddings + character embeddings (hybrid embedding layer)**

This hybrid token embedding layer is a combination of token embeddings and character embeddings. In other words, they create a stacked embedding to represent sequences before passing them to the sequence label prediction layer.

So far, we have built two models which have used token and character-level embeddings, however, these two models have used each of these embeddings exclusively.

Steps that were taken to create the model:

- Create a token-level model (like model_1)
- Create a character-level model
- Combine (using layers.Concatenate) the outputs of 1 and 2
- Build a series of output layers on top of 3
- Construct a model which takes token and character-level sequences as input and produces sequence label probabilities as output

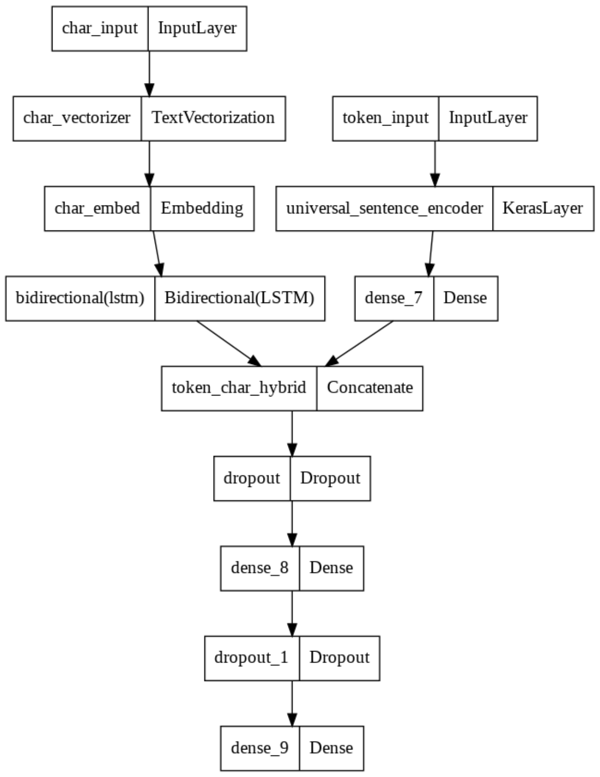


**Figure 14**

Before we move on with model 5, we performed some preprocessing steps.

**Create positional embeddings**

The "line_number" and "total_lines" columns are features which didn't necessarily come with the training data but can be passed to our model as a positional embedding. In other words, the positional embedding is where the sentence appears in an abstract.

Since our "line_number" and "total_line" columns are already numerical, we could pass them as they are to our model. But to avoid our model thinking a line with "line_number"=5 is five times greater than a line with "line_number"=1, we'll use one-hot-encoding to encode our "line_number" and "total_lines" features. To do this, we can use the tf.one_hot utility.

NLP Final Project Report.

**Model 5:** If you were to look at an abstract, would you expect the sentences to appear in order? Or does it make sense if they were to appear sequentially? For example, sequences labelled CONCLUSIONS at the beginning and sequences labelled OBJECTIVE at the end?

This time our model requires four feature inputs

1. Train line numbers one-hot tensor (train_line_numbers_one_hot)
2. Train total lines one-hot tensor (train_total_lines_one_hot)
3. Token-level sequences tensor (train_sentences)
4. Char-level sequences tensor (train_chars)

We can pass these as tuples.

Model 5 is the most complex among the batch, it is the one which incorporates token embeddings, character embeddings and positional embeddings.

More specifically we're have gone through the following steps

1. Create a token-level model (similar to model_1)
2. Create a character-level model (similar to model_3 with a slight modification to reflect the paper)
3. Create a "line_number" model (takes in one-hot-encoded "line_number" tensor and passes it through a non-linear layer)
4. Create a "total_lines" model (takes in one-hot-encoded "total_lines" tensor and passes it through a non-linear layer)
5. Combine (using layers.Concatenate) the outputs of 1 and 2 into a token-character-hybrid embedding and pass it series of outputs to Figure 1 and section 4.2 of *Neural Networks for Joint Sentence Classification in Medical Paper Abstracts*
6. Combine (using layers.Concatenate) the outputs of 3, 4 and 5 into a token-character-positional tribrid embedding
7. Create an output layer to accept the tribrid embedding and output predicted label probabilities
8. Combine the inputs of 1, 2, 3, 4 and outputs of 7 into a tf.keras.Model

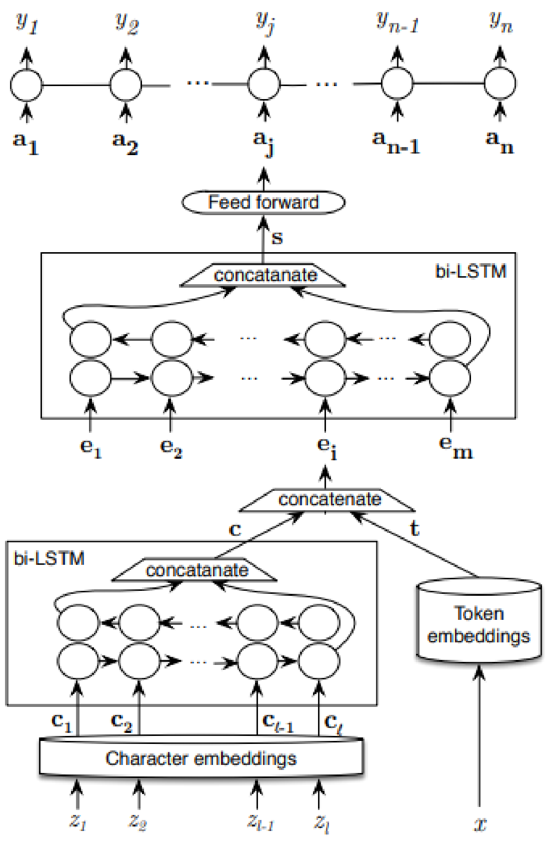
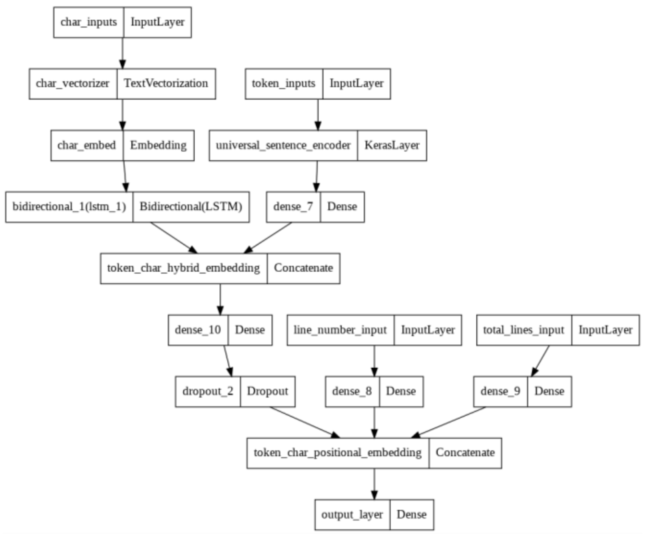


**Figure 15**

Here we can see the neural network model shown in the paper, and below we can see what we have created.



char_inputs | InputLayer
char_vectorizer | TextVectorization
token_inputs | InputLayer
char_embed | Embedding
universal_sentence_encoder | KerasLayer
bidirectional_1(lstm_1) | Bidirectional(LSTM)
dense_7 | Dense
token_char_hybrid_embedding | Concatenate
dense_10 | Dense
line_number_input | InputLayer
total_lines_input | InputLayer
dropout_2 | Dropout
dense_8 | Dense
dense_9 | Dense
token_char_positional_embedding | Concatenate
output_layer | Dense

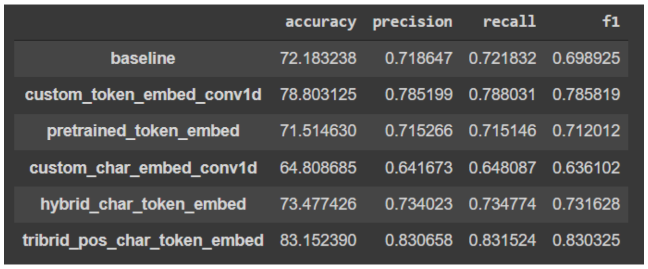
**Figure 16**

You'll notice our model is looking very similar to the model shown in Figure 1 of *Neural Networks for Joint Sentence*

*Classification in Medical Paper Abstracts*. However, a few differences remain:

- We're using pretrained TensorFlow Hub token embeddings instead of Glove embeddings.
- We're using a Dense layer on top of our token-character hybrid embeddings instead of a bi-LSTM layer.
- Section 3.1.3 of the paper mentions a label sequence optimization layer (which helps to make sure sequence labels come out in a respectable order) but it isn't shown in Figure 1. To makeup for the lack of this layer in our model, we've created the positional embeddings layers.
- Section 4.2 of the paper mentions the token and character embeddings are updated during training, our pretrained TensorFlow Hub embeddings remain frozen.
- The paper uses the SGD optimizer, we're going to stick with Adam.

## PERFORMANCE EVALUATION



| | accuracy | precision | recall | f1 |
|---|---|---|---|---|
| baseline | 72.183238 | 0.718647 | 0.721832 | 0.698925 |
| custom_token_embed_conv1d | 78.803125 | 0.785199 | 0.788031 | 0.785819 |
| pretrained_token_embed | 71.514630 | 0.715266 | 0.715146 | 0.712012 |
| custom_char_embed_conv1d | 64.808685 | 0.641673 | 0.648087 | 0.636102 |
| hybrid_char_token_embed | 73.477426 | 0.734023 | 0.734774 | 0.731628 |
| tribrid_pos_char_token_embed | 83.152390 | 0.830658 | 0.831524 | 0.830325 |

**Figure 17**

Comparison of the performance metrics for all the modelling experiments, we can see here that the tribrid model is performing the best.

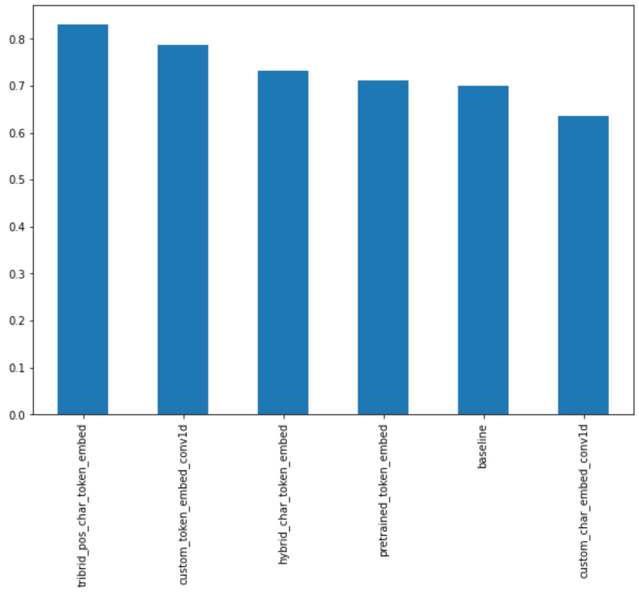


**Figure 18**

From the graph shown above we can see the f1 score comparison of all the models pictorially sorted from high to low.

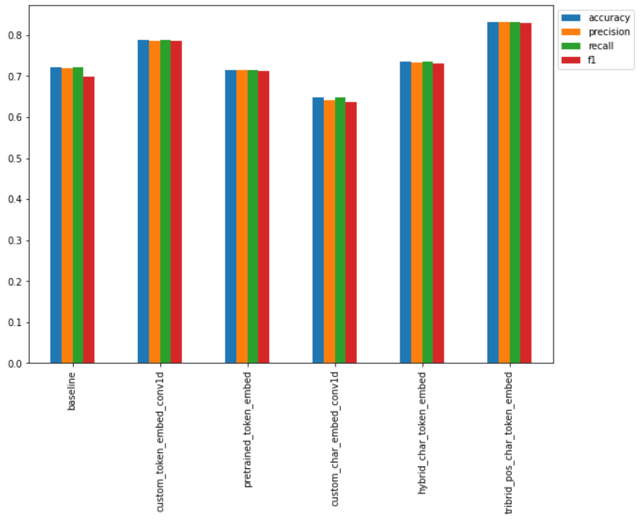
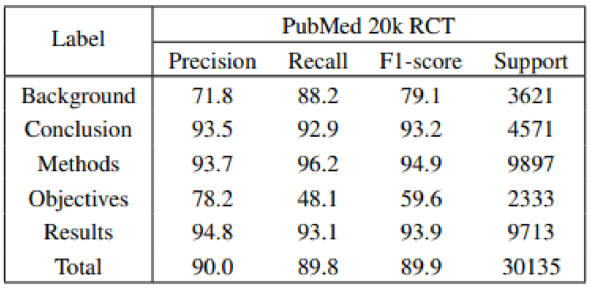


**Figure 19**

From the graph above we can see the performance metrics for all the modelling experiments graphically. We can see here that the tribrid model is performing the best, overall. Followed by the custom token embedding layer.



| Label | PubMed 20k RCT | | | |
|---|---|---|---|---|
| | Precision | Recall | F1-score | Support |
| Background | 71.8 | 88.2 | 79.1 | 3621 |
| Conclusion | 93.5 | 92.9 | 93.2 | 4571 |
| Methods | 93.7 | 96.2 | 94.9 | 9897 |
| Objectives | 78.2 | 48.1 | 59.6 | 2333 |
| Results | 94.8 | 93.1 | 93.9 | 9713 |
| Total | 90.0 | 89.8 | 89.9 | 30135 |

**Table 3:** Detailed results of our model on the PubMed 20k RCT dataset.

**Figure 20**

The above shown table denoted the results obtained in the original paper on the test data

It seems our best model still has some ways to go to match the performance of the results in the paper (the original paper gets 90.0 F1-score on the test dataset, whereas ours gets ~82.1 F1-score).

However, as we discussed before our model has only been trained on 20,000 out of the total ~180,000 sequences in the RCT 20k dataset. We also haven't fine-tuned our pretrained embeddings. So, there's a couple of extensions we could try to improve our results as mentioned in the future work section of this report.

## CONCLUSION

In this project, we have presented an ANN architecture to classify sentences that appear in sequence. We demonstrate that jointly predicting the classes of all sentences in each text improves the quality of the predictions and yields better performance than a CRF. Our model achieves state-of-the-art results for sentence classification in medical abstracts. Moreover, we discussed the various ways in which the experimentation with different model architecture will take place. We hope that this project will accelerate the development of algorithms for sequential sentence classification and increase the interest of the text mining community in the study of RCTs. The model architecture can be tweaked further to improve the performance metrics.

## FUTURE WORK.

- Train `model_5` on all of the data in the training dataset for as many epochs until it stops improving. Since this might take a while, you can use an early stopping callback.

- Check out the Keras guide on using pre-trained GloVe embeddings. Try integrating this with one of the modelling experiments.

- Try replacing the TensorFlow Hub Universal Sentence Encoder pretrained embedding for the TensorFlow Hub BERT PubMed expert (a language model pretrained on PubMed texts) pretrained embedding.

- What happens if you were to merge our line_number and total_lines feature for each sequence? For example, created an X_of_Y feature instead?

## ACKNOWLEDGMENT

We would like to express our gratitude to our primary mentor and professor, Diana Inkpen, who guided us throughout this project, and the knowledge obtained from her classes is implemented in this project.
.

## REFERENCES

[1] Franck Dernoncourt, Ji Young Lee, and Peter Szolovits. 2017. Neural Networks for Joint Sentence Classification in Medical Paper Abstracts. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 694–700, Valencia, Spain. Association for Computational Linguistics.

[2] Dernoncourt, Franck & Lee, Ji. (2017). PubMed 200k RCT: a Dataset for Sequential Sentence Classification in Medical Abstracts.

[3] [Xiao and Cho2016] Yijun Xiao and Kyunghyun Cho. 2016. Efficient character-level document classification by combining convolution and recurrent layers. arXiv preprint arXiv:1602.00367

[4] [Jinha2010] Arif E Jinha. 2010. Article 50 million: an estimate of the number of scholarly articles in existence. Learned Publishing, 23(3):258–263.