Christian Martin, Awais Naeem, Allison Pujol

INF 385T: Natural Language Processing and Applications

Final Project: Segmenting “Tricky Abstracts” as an NLP Intervention

December 4, 2023

I. Introduction

Novel medical research findings – and being able to understand and communicate those findings – are crucial to a healthy and effective public health system. When embarking on the research process, many biomedical experts will look at a published paper's abstract to determine whether something is worth reading. However, many medical research abstracts use highly technical language that slows down overall readability and sentence comprehension [1]. Language barriers (i.e., reading an abstract in English as a non-native or non-fluent English speaker) may slow down the process even more. Furthermore, a researcher may want to compare only one part of the abstract against another abstract (such as seeing different methods for the same biological process, comparing the "Results" sections of two distinct studies about HIV, etc.).

As an NLP intervention, our group has evaluated a structured viewing of paper abstracts through which classifies sentences according to the part of the paper it describes. While this project does focus on medical abstracts, we see the benefits of applications to other academic disciplines such as the humanities or social sciences. Thus, our intended audience (beyond the broader NLP-focused community) would be medical students and researchers in the medical field; with further development, the methods in this paper could be useful for other researchers as well.

To find the best way of classifying the sentences, we implemented & evaluated three different approaches to determine which is most effective. The first method used a traditional ML approach of creating GloVe/Word2Vec sentence embeddings and then using a model to predict classes. The second method used a fine-tuned LLM model, such as T5, in a Sequence Classification task. The third method used a deep learning approach which utilized an RNN

Model and LSTM/GRU for short-term memory. To evaluate the models, we split the dataset and compared the results based on the labels in the dataset. After comparing the classification reports for each model, we concluded that the neural network approach (GRU) was the most effective.

II. **Dataset**

A. About the Data

The dataset for this project is composed of 200,000 medical research paper abstracts [1] from PubMed obtained via the website Kaggle [2]. In the dataset, each abstract is split into sentences with a distinct label/tag assigned to each sentence i.e., “background,” “methods,” “results,” or “conclusion,” which is given as the first word in the line. There are also blank lines and comments (lines starting with a “#”) which were removed:

```
###20497432
BACKGROUND	The aim of this study was to evaluate the efficacy , safety and complications of orbital steroid injection versus oral steroid therapy in the management of thyroid-related ophthalmopathy .
METHODS	A total of 29 patients suffering from thyroid ophthalmopathy were included in this study .
METHODS	Patients were randomized into two groups : group I included 15 patients treated with oral prednisolone and group II included 14 patients treated with peribulbar triamcinolone orbital injection .
METHODS	Only 12 patients in both groups ( 16 female and 8 male ) completed the study .
RESULTS Both groups showed improvement in symptoms and in clinical evidence of inflammation with improvement of eye movement and proptosis in most cases .
RESULTS Mean exophthalmometry value before treatment was 22.6 1.98 mm that decreased to 18.6 0.996 mm in group I , compared with 23 1.86 mm that decreased to 19.08 1.16 mm in group II .
RESULTS Mean initial clinical activity score was 4.75 1.2 and 5 1.3 for group I and group II before treatment , respectively , which dropped to 0.83 1.2 and 0.83 1.02 , 6 months after treatment , respectively .
RESULTS There was no change in the best-corrected visual acuity in both groups .
RESULTS There was an increase in body weight , blood sugar , blood pressure and gastritis in group I in 66.7 % , 33.3 % , 50 % and 75 % , respectively , compared with 0 % , 0 % , 8.3 % and 8.3 % in group II .
RESULTS No adverse local side effects were observed in group II .
CONCLUSIONS	Orbital steroid injection for thyroid-related ophthalmopathy is effective and safe .
CONCLUSIONS	It eliminates the adverse reactions associated with oral corticosteroid use .
```

B. Applying Preprocessing

Pre-processing was applied to clean-up the document and get it ready for use. First, initial cleanup removed blank lines and unwanted comments. Then, a regular expression-based filter (regex) was used to obtain the abstract name, text, and text label:

	Abstract Name	Text	Label
0	###24491034	The emergence of HIV as a chronic condition me...	BACKGROUND
1	###24491034	This paper describes the design and evaluation...	BACKGROUND
2	###24491034	This study is designed as a randomised control...	METHODS
3	###24491034	The intervention group will participate in the...	METHODS
4	###24491034	The program is based on self-efficacy theory a...	METHODS

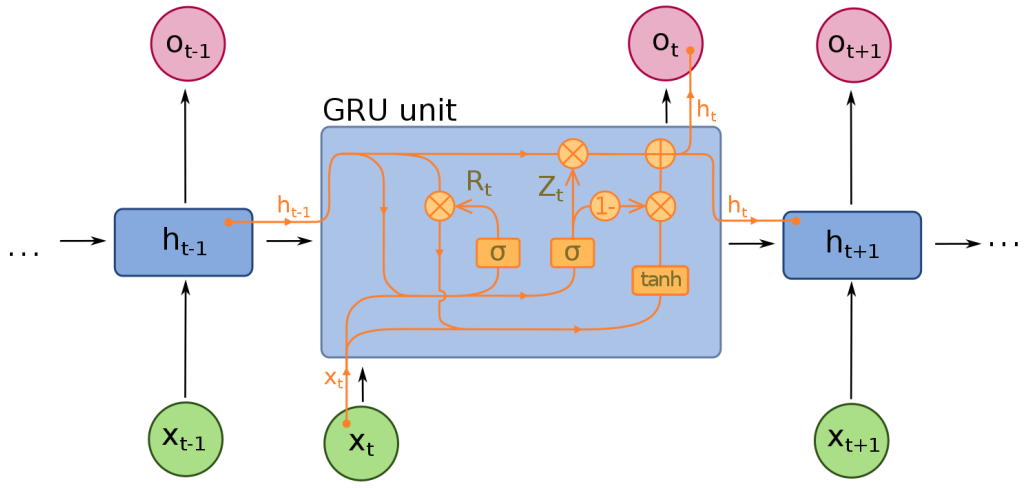
We tokenized the text and filtered to include the abstracts with at least 5 tokens. Originally, the dataset contained 200,000 abstracts; after pre-processing, there were approximately 1 million labeled texts to work with.

III. **Methods**

A. Gated Recurrent Unit (GRU) Neural Network

Gated Recurrent UNIT (GRU) [4] is a gating mechanism in recurrent neural networks just like long short-term memory to remember or forget certain features of the textual information. Its single unit is more or less similar to that of LSTM but it lacks an output gate resulting in less number of parameters compared to LSTM. However, as Chung et al note, both a GRU and an LSTM have comparable effectiveness in tackling sequence labeling [4].

A diagram of the model is depicted below:



To train the GRU recurrent neural network, the first step was to prepare the custom data loaders to feed the data into the network during the training. For this, an indexed vocabulary of unique words was prepared using the training data which came out to have vocabulary size of ~300K words and max sequence length to be of 338. Text data was padded or truncated beyond the max sequence length. Finally, labels/text data was converted into PyTorch tensors and custom data loaders were prepared with training, test and validation batch sizes to be 2048, 64 and 64, respectively.

Then a GRU recurrent neural network architecture was specified with the following architecture:

1. Embedding Layer (Input Size: 200k, Embedding Size: 200)
2. GRU Layer (Hidden Size: 64, Num Layers: 2)
3. Fully Connected Layer 1 (Input Units: 64, Output Units: 32)
4. Fully Connected Layer 1 (Input Units: 32, Output Units: 16)
5. Fully Connected Layer 1 (Input Units: 16, Output Units: 5)

For the training, Adam optimizer was selected along with cross-entropy loss to track the loss of the multi-class classification. For a total training record of ~1 million, the training was run for 5 epochs with a batch size of 2048 which took around 4-5 hours on Google Colab Pro. Initial loss was reported around 0.5212 which was reduced to around 0.2789 after 5 epochs.

B. Fine-tuned LLM Model

For our second approach, we created several fine-tuned models. The first of these models is a fine-tuned T5 model[6] trained on only one sentence at a time. A T5 model, which uses the Transformer architecture and can measure context, is also an advanced language model. An advantage to the T5 approach here would be its ability to perform in data-rich fields where a

classification-based encoder-only approach might falter [6]. The second model is also a T5 model which trained on the sentence concatenated with the full text of the abstract, using a pipe token "|" as a separator since T5 does not support special separator tokens. The third model is a DistilBERT[7] model trained on the sentence concatenated with the full text of the abstract using the special "[SEP]" separator token. And the final model is a DistilBERT model with the sentence concatenated after the full abstract, instead of before. Research indicates that a DistilBERT model brings the potential to "reduce the size of a BERT model by 40%, while retaining 97% of its language understanding capabilities and being 60% faster" [8]. This is because of the compression capabilities that DistilBERT has as a streamlined version of the BERT model, which allows DistilBERT to reduce its layers but optimize the Transformer architecture it operates with. [8]

The fine-tuning code follows the same basic approach:

```
from transformers import T5Tokenizer, T5ForSequenceClassification

tokenizer = T5Tokenizer.from_pretrained("t5-small")
model = T5ForSequenceClassification.from_pretrained("t5-small", num_labels=len(labels))
```

```
def prep(df):
  text = [f"{task_prefix}: {text}" for text in df['Text'].to_list()]
  inputs = [tokenizer(f"{task_prefix}: {text}", padding=True, truncation=True, return_tensors="pt").input_ids[0] for text in df['Text'].to_list()]
  y = [labels.index(y) for y in df['Label'].to_list()]
  return Dataset.from_dict({"input_ids": inputs, "text": text, "labels": y})
```

Since the abstracts in the training split are still in the order of the dataset, the full text can be prepared by using groupby() and transform() operations to concatenate the individual strings together[5].

C. Word2Vec and Logistic Regression

For our final approach, we took a tried-and-true approach: we trained a multi-class Logistic Regression using Word2Vec for creating average word vectors. While relatively simple compared to RNNs and LLMs, Word2Vec represents words as dense vectors whose embeddings

can serve as features for logistic regression. The basic code can be seen below:

```
# Build Word2Vec vocabulary
word2vec_model = Word2Vec(sentences=data['Text'].to_list(), vector_size=100, window=5, min_count=1, workers=4)
#word2vec_model.build_vocab(data['Text'])

# Train Word2Vec model
word2vec_model.train(data['Text'], total_examples=word2vec_model.corpus_count, epochs=10)  # Train the model
```

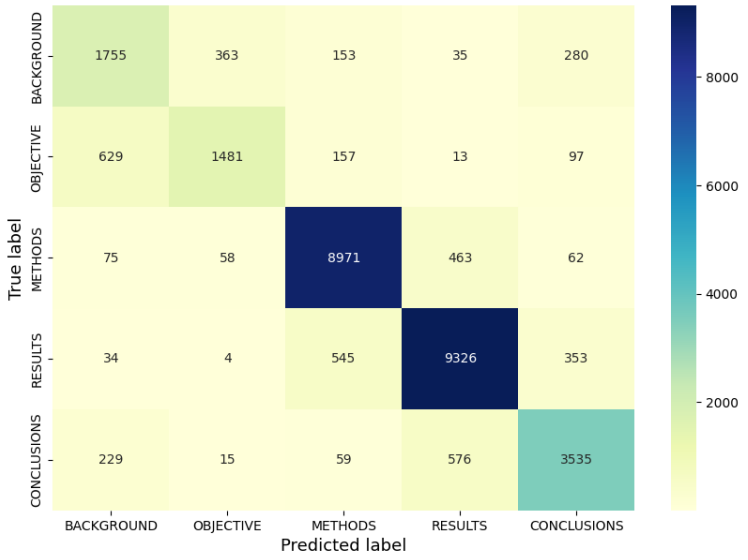
```
X = [average_word_vectors(words, word2vec_model, word2vec_model.wv.index_to_key, 100) for words in data['Text']]
y = data['Label']

# Split data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train Logistic Regression using Word2Vec embeddings
classifier = LogisticRegression(max_iter=1000)
classifier.fit(X_train, y_train)

# Predict on test set
y_pred = classifier.predict(X_test)
```

Word2Vec can be powerful due to its popularity, efficiency, and ability to address semantics; however, its biggest downfall is its inability to address context. Notably, we believe that comparing Word2Vec's performance in this project with that of the other two models will yield insight as to the degree to which context matters in labeling parts of an abstract.

IV. **Results**

A. Gated Recurrent Unit (GRU) Neural Network

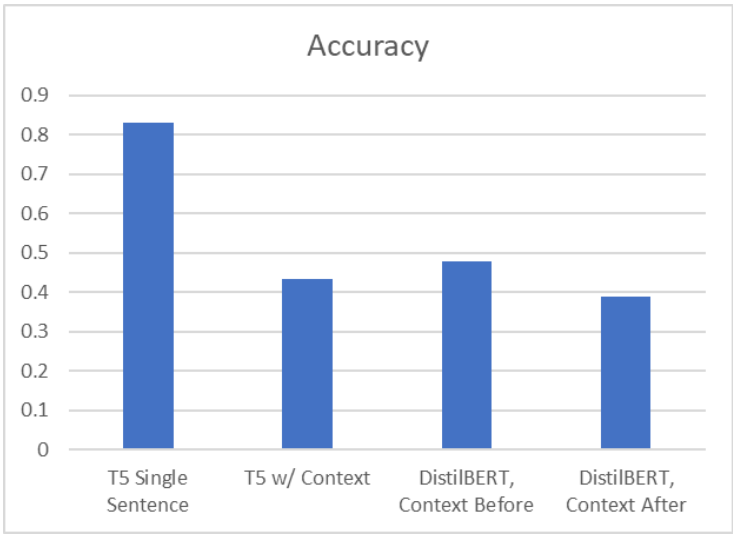
In order of the methodology section, here are the results of the models. The custom GRU model performed extremely well, with an accuracy of 0.86 and with a strong showing in the confusion matrix.



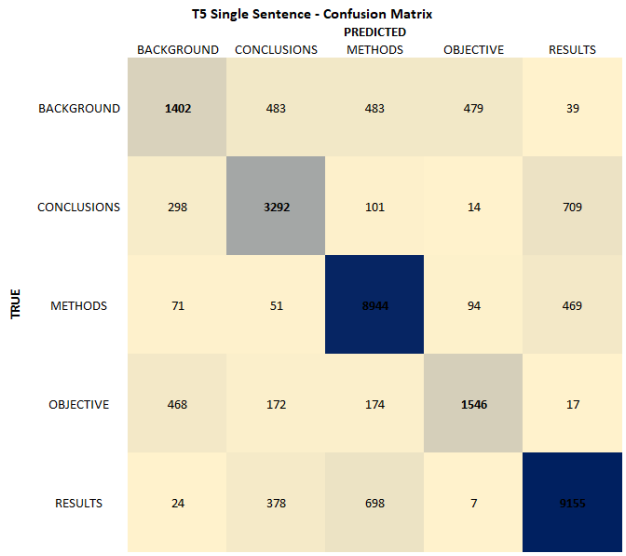
For the areas where there is some confusion, there is a plausible reasoning. For example, the confusion between background and objective makes some sense as they are both providing the context for the problem being researched, and the confusion between results and conclusions makes sense as they both explain what can be taken away from the study.

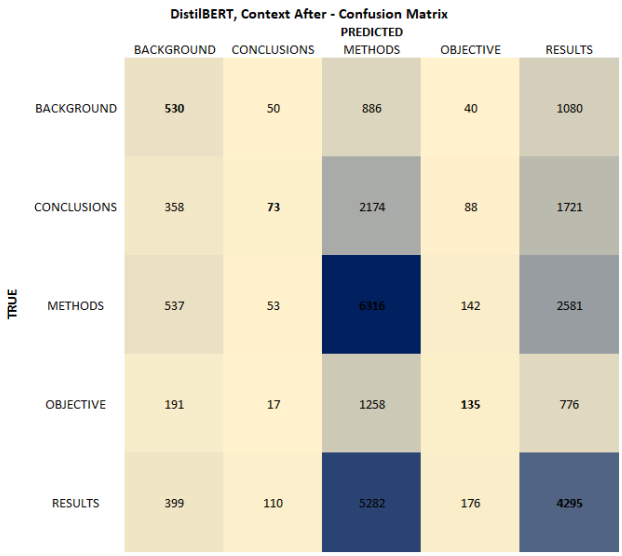
B. Fine-tuned LLM Model

With the fine-tuned models, we can see that the single sentence T5 model performed about twice as well as any of the models with the full abstract:



Comparing the confusion matrices of the fine-tuned models, using the best (single sentence T5 at 0.83 accuracy) and worst (DistilBERT with context after at 0.39), we can see that the latter model is significantly over-predicting the methods and results labels.





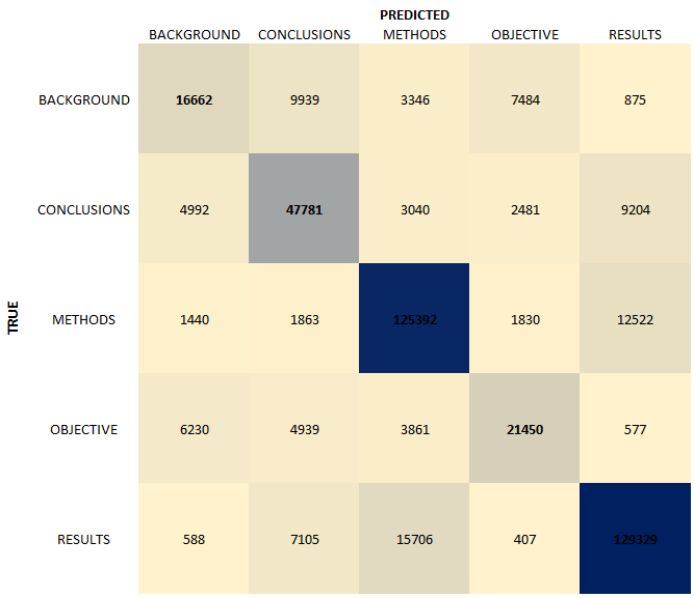
This is likely due to there being an outsized number of methods- and results-labeled sentences in the training dataset and the model having a large number of those words appearing in each input from the full abstract.

C. Word2Vec+Logistic Regression

The Word2Vec+Logistic Regression model performed well, at an accuracy of 0.78.

	precision	recall	f1-score	support
BACKGROUND	0.56	0.43	0.49	38306
CONCLUSIONS	0.67	0.71	0.69	67498
METHODS	0.83	0.88	0.85	143047
OBJECTIVE	0.64	0.58	0.61	37057
RESULTS	0.85	0.84	0.85	153135
accuracy			0.78	439043
macro avg	0.71	0.69	0.70	439043
weighted avg	0.77	0.78	0.77	439043

However, we can see in the confusion matrix that the confusions are a lot more spread out, particularly when compared to the custom GRU model. For example, there isn't an obvious answer for the confusions between conclusions and objective, or conclusions and methods.

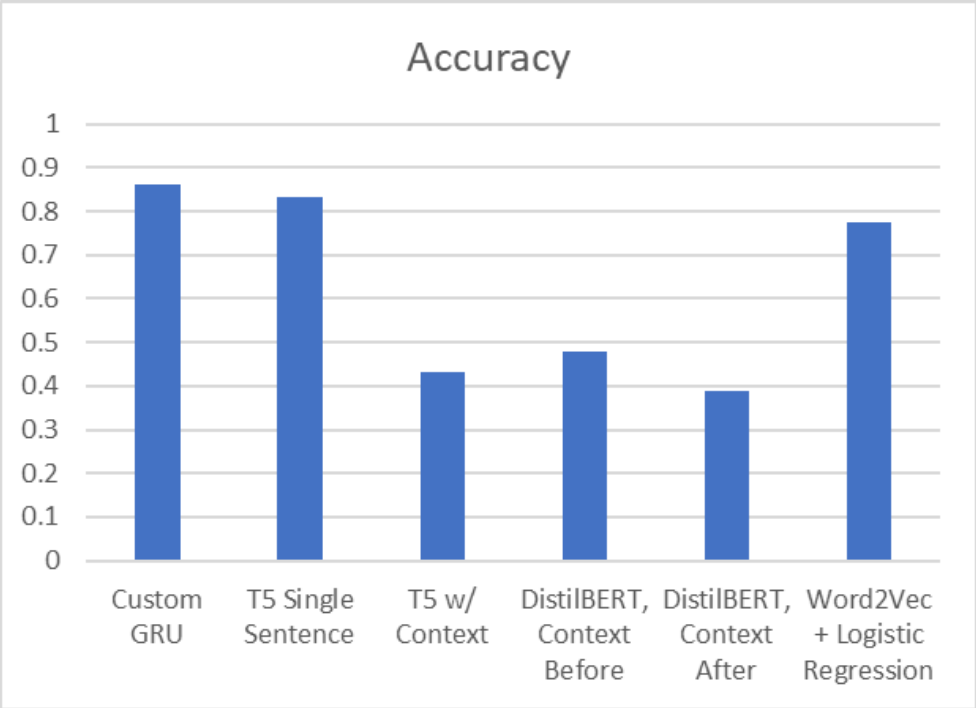


As can be inferred from the above figure, Word2Vec can study things like word choice –

but it can't analyze syntax, really. This is because Word2Vec considers context, but not grammatical structure. Given that one would expect abstract sentences to share syntactic similarities rather than express particular diction patterns, it was surprising that fairly accurate results were obtained by this relatively simple model.

D. Overall

Overall, the custom GRU model performed best, with the single sentence T5 in close second. All of the models with the full abstract context performed the worst, at about half the accuracy of the custom GRU & single sentence T5 models.



When comparing the Custom GRU and single sentence T5 models, the GRU model performs slightly better on almost all of the classification scores:

Custom GRU model:

	precision	recall	f1-score	support
BACKGROUND	0.64	0.68	0.66	2586
OBJECTIVE	0.77	0.62	0.69	2377
METHODS	0.91	0.93	0.92	9629
RESULTS	0.90	0.91	0.90	10262
CONCLUSION	0.82	0.80	0.81	4414
accuracy			0.86	29268
macro avg	0.81	0.79	0.80	29268
weighted avg	0.86	0.86	0.86	29268

Fine-tuned T5 single sentence model:

	precision	recall	f1-score	support
BACKGROUND	0.62	0.54	0.58	2586
OBJECTIVE	0.72	0.65	0.68	2377
METHODS	0.89	0.93	0.91	9629
RESULTS	0.88	0.89	0.89	10262
CONCLUSIONS	0.75	0.75	0.75	4414
accuracy			0.83	29268
macro avg	0.77	0.75	0.76	29268
weighted avg	0.83	0.83	0.83	29268

V. **Conclusion**

We were able to achieve good performance using relatively simple methods; using a custom GRU model allowed us to get a little more performance (+3% compared to single sentence T5 & +8% compared to Word2Vec + Logistic Regression).

We saw some limitations present within the approach we followed. Because of the nature of dataset labeling, we were limited to predicting one label per sentence. We do recognize, however, that there could be value in predicting multiple labels per sentence – what if, for example, a sentence is describing both a method and its result? This problem is likely to be exacerbated in the humanities or other fields where abstracts follow a less strict outline.

In the results, we can see that a surprisingly large portion of the performance can be explained by word choice alone. The most obvious indicator of this is the 0.78 accuracy of the Word2Vec's averaged word vectors, which has no consideration of grammar or structure, and has not been explicitly trained on the semantic meanings of the words used. Additionally, we can see

this in the significant drop in accuracy when adding the full abstract text to the model inputs when fine-tuning.

However, the results also indicate there is likely an improvement from being able to utilize the structure and semantic meanings of the sentences, as evidenced by the single sentence T5 and custom GRU models performing 5-8% better than Word2Vec+Logistic Regression.

As we have noted previously, the segmentation performed in this project has value for disciplines besides medical research. Being able to determine whether a model is able to actually interpret the context of an abstract's sentences and categorize them effectively – as opposed to learning patterns like sentence order or word choice – would be helpful. Furthermore, we see this project as a stepping stone for broader work on sequence labeling as a tool for both native and non-native speakers alike who may seek an effective way to compare a variety of abstracts.

VI. Sources Consulted

*****All the code used in this project can be found here:**

https://github.com/embedded-robotics/natural-language-processing-final-project

[1] Tal August, Lucy Lu Wang, Jonathan Bragg, Marti A. Hearst, Andrew Head, and Kyle Lo. 2023. Paper Plain: Making Medical Research Papers Approachable to Healthcare Consumers with Natural Language Processing. ACM Trans. Comput.-Hum. Interact. 30, 5, Article 74 (September 2023), 38 pages. https://doi.org/10.1145/3589955

[2] 200000 Medical Research Paper Abstracts https://www.kaggle.com/datasets/anshulmehtakaggl/200000-abstracts-for-seq-sentence-classification

[3] PubMed https://pubmed.ncbi.nlm.nih.gov/

[4] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, Yoshua Bengio. 2014. Paper Plain: Empirical evaluation of Gated Recurrent Neural Networks on Sequence Modelling. NIPS 2014 Deep Learning and Representation Learning Workshop. https://doi.org/10.48550/arXiv.1412.3555

[5] Concatenate strings from several rows using Pandas groupby https://stackoverflow.com/questions/27298178/concatenate-strings-from-several-rows-using-pandas-groupby

[6] Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. arXiv:2003.06713. Document Ranking with a Pretrained Sequence-to-Sequence Model. https://arxiv.org/pdf/2003.06713.pdf

[7] T5 https://huggingface.co/docs/transformers/main/model_doc/t5

[8] Victor Sanh, Lysandre Ddebut, Julien Chaumond, Thomas Wolf. arXiv:1910.01108.

DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.

https://arxiv.org/pdf/1910.01108.pdf

[9] DistilBERT https://huggingface.co/docs/transformers/main/model_doc/distilbert

Appendix: Q and A

Q: What level of meaning within language (semantics, lexical analysis, etc) does this project operate within?

A: Our initial assumption was Discourse, assuming context within multiple sentences/the abstract as a whole is relevant here (which our project seems to perhaps disprove). It may be possible that we are operating within a level of language where lexical analysis (word meaning without context) is more relevant

Q: Which matters more for accuracy – syntax or word choice?

A: The fairly high accuracy levels that we observed for the Word2Vec/Logistic Regression model would seem to indicate that word choice (potentially, word patterns) is more relevant than sentence structure. However, adding context and syntactic relationships into the mix does allow for 5-8% more accuracy in this particular project.