SCIATICA: A Query-by-Example for Scientific Article Retrieval

Ashwani Kumar Kamal
20CS10011
ashwanikamal.im421@gmail.com

Hardik Pravin Soni
20CS30023
iamhardikat11@gmail.com

Shiladitya De
20CS30061
shiladityade.bwn2001@gmail.com

Sourabh Soumyakanta Das
20CS30051
dassourabh103@gmail.com

Abstract

Since, it is very unlikely that user provides a properly formatted query, we are using a facet based query search on the documents based on the rhetorical structure of an academic paper. We are using an ensemble learning approach where we are ensembling 6 language models which contribute to the final results based on a weight assigned to them. Apart from that, we also did a grid search on the number of layers in the *DistilBertModel* and presented the results.

1 Description of Problem Statement

Its very common for users to not to come up with perfect queries (following proper syntax or method). Hence, the idea of query by example (QBE) came into being. It allows the user to search about something based on some structure that it follows. In this project, documents are retrieved based on the rhetorical structure of a paper like background, method, results etc.
So, we intended to develop a search engine that will fetch papers based on a given query and a facet. Moreover the query need not be well written, rather it can be any query (even with spelling or grammmar mistakes).

2 Motivation behind the proposed solution

We propose to provide a solution based on ensemble learning approach (Weighted Voting) using 6 models which are *nli-roberta-base-v2*, *paraphrase-TinyBERT-L6-v2*, *all-mpnet-base-v2*, *all-distilroberta-v1*, *paraphrase-albert-small-v2*, *allenai/specter*.
The motivation behind this solution came when we were testing thsese models on the CSFCube (Mysore et al., 2021) dataset and we found that only a few models were having higher values of the metrics (*viz.* NDCG, Precision@20, Recall@20 etc.) than the others. But these models could sometimes overfit on the dataset so choosing only those models which performed very well might not be a wise decision. So we used the ensemble learning approach in which each model has been assigned a weight, which determines the model's contribution to the final result. The weights have been allocated based on the results that the corresponding model has generated on the dataset (i.e. the better the results , the higher is the weight).

3 Related Work

3.1 Document Classification for COVID-19 Literature.

Bernal Jiménez Gutiérrez and Juncheng Zeng (2011) (Bernal Jim´enez Guti´errez, 2020) have undertaken the challenge of facilitating the swift and accurate retrieval of scientific literature pertaining to COVID-19 for the benefit of the scientific and medical community. To achieve this, they have analyzed document classification models on the LitCovid dataset, which currently comprises over 23,000 research papers that focus on the novel coronavirus that emerged in 2019. Their objective was to investigate the generalizability and data efficiency of these models, and to identify important issues that require further attention in future research.

Through their analysis, Bernal Jim´enez Guti´errez, 2020 have discovered that fine-tuning pretrained language models yields the best performance for document classification on the LitCovid dataset. In addition, they have examined 50 errors made by the best performing models on the LitCovid documents, and have identified two main issues that contribute to these errors. Firstly, certain labels are often correlated too closely together, leading to ambiguity and confusion. Secondly, these models may fail to focus on discriminative sections of the articles, thus overlooking key information that could enhance their accuracy.

3.2 Discovering Relevant Scientific Literature on the Web.

Kurt D. Bollacker and Giles, 2000 discuss the development and implementation of a machine learning system that can be used to explore the relationships between scientific articles through their citations. The system uses a combination of neural networks and statistical algorithms to group articles based on their citation patterns, and then analyses the clusters to identify the most influential articles in a given field.

The authors describe their research on identifying influential articles in a given field and exploring relationships between different research topics. They argue that their system can be used to improve user experience by constantly updating user profiles and providing information on research fields and clusters. They also discuss limitations of their approach and suggest further research to improve the system's effectiveness.

Difference between Our approach and the previous approaches

In our research, we explored many machine learning models and proposed a solution based on an ensemble learning approach (Weighted Voting) that uses six models: *nli-roberta-base-v2*, *paraphrase-TinyBERT-L6-v2*, *all-mpnet-base-v2*, *all-distilroberta-v1*, *paraphrase-albert-small-v2*, and *allenai/specter*. We allocated weights to the models based on the results they generated on the dataset; the better the results, the higher the weight.
Bernal Jim´enez Guti´errez, 2020 worked on similar lines but focused more on finding the shortcomings of these models individually, rather than using the ensemble approach, where the shortcomings of one are compensated by the other.
In contrast, Kurt D. Bollacker and Giles, 2000 worked more on clustering articles using a combination of neural networks and statistical algorithms to identify the most influential papers and explore connections between research topics while focusing on improving the user experience.

4 Techniques or Methods Implemented

The main research question is to fetch the results for a query and a facet given by the user. So as to solve this, the primary task was to generate the encodings of the documents and the query using language models.
As we have gone through various research papers, we found that a greater emphasis was given on language models as compared to other models like probabilistic models (*bm25Okapi* etc.) or classic information retrieval models like *TF-IDF* or Bag of Words (*BoW*). Our approach is described in the following three subsections:

4.1 Base Models

So we began with generating encodings from the pretrained sentence transformer models. We used the following pretrained transformer models:

- BERT-NLI: *nli-roberta-base-v2*
- BERT-PP: *paraphrase-TinyBERT-L6-v2*
- MPNET: *all-mpnet-base-v2*
- DistilBERT: *all-distilroberta-v1*
- ALBERT: *paraphrase-albert-small-v2*
- SPECTER: *allenai/specter*

Each of the above described above described have been used to encode the abstract of each document and saved the encodings in a json file. Then the same task was carried out with the test queries of each facet.
Then testing is carried out using various metrics which are $NDCG$ (Normalised Discounted Cumulative Gain), $Ranked\ Precision$, $Precision$@20, $Recall$@20 etc.

Apart from the transformer models, the tf-idf vectorizer has been used to generate the embeddingsfor the TF-IDF model followed by carrying out the similar procedures as described above.

4.2 Grid Search

After carrying out the previous method, it was observed that there is a need to tune the models to adjust to our dataset. So as to do it the different parameters of the models were considered.
The standard grid search technique is applied to tune the hyper parameters of *DistilBertModel* of AutoTransformers. The various hyperparameters of the model are n_layers, $qa\_dropout$, $dropout$ etc.
A one-dimensional grid search on the n_layers (i.e. Number of layers of the neural network) is performed as it was more close to the given task of document retrieval.
The other parameters like $qa\_dropout$ or $dropout$

did not have much impact when they were changed along with the number of layers.
More details about it are present in the experiments and results sections.

4.3 Ensemble Approach

To ensure robustness of the model used and prevention of model overfitting, ensemble approach is used. This combines the results of all the models that are used, i.e., *nli-roberta-base-v2*, *paraphrase-TinyBERT-L6-v2*, *all-mpnet-base-v2*, *all-distilroberta-v1*, *paraphrase-albert-small-v2*, and *allenai/specter* and generates a final score based on their initial scores and the weights assigned to all the models.
Various metrics parameters that are considered for this purpose are $NDCG$ and $Ranked\ Precision$. The better score a model provides on these parameters, the higher weight it is assigned during ensembling. After combining all the scores for each document, a final sorted list of documents is generated out of which required number of documents are retrieved.
Since this runs all the models, it is supposed to be slower. So, multiprocessing of documents with each model running on each core can be done which would theoretically take the same time as running a single model on the documents and query and therefore give better results in almost the same time.

5 Experiments

Different experiments have been carried out with the various transformer models whose details are given below:

5.1 Base Models

As mentioned in the techniques section, different transformer models as well as word vector models like *TF-IDF* have been used to encode the abstracts of all the documents as well as abstracts of the corresponding facets.
The encodings are saved in the corresponding directories as $json$ files as they will be used to test against the test queries as well as those given by the user.
The encodings generated are tested against the relevance measure given in the git repository of CSFCube (Mysore et al., 2021) which are $Ranked\ Precision$, $Precision@k$, $Recall@k$ and $NDCG\%20$. The results are discussed in the next section.

5.2 Grid Search

The grid search is done on the *DistilBertModel* of Auto transformers. When the standard model is applied on the dataset the results were not that good. Of the various methods of improving the metrics like incorporating gaussian noise (Chuhan Wu, 2022) in the embeddings to improve the prediction, grid search to do hyperparameter tuning or training the model on contrasting set of documents from our dataset using SetFit first followed by generating the encodings etc. the grid search method is implemented.

In the Grid Search from the various hyperparameters like $dropout$, qa_droput, n_layers etc. n_layers has been chosen as the other parameters did not have a significant change in the metrics.
The n_layers is varied in the range $[4, 10]$, each time creating the embedding out of the models and testing against the given test queries to get the metrics and plotting them using python's *matplotlib.pyplot* function. The results and plots of the grid search has been discussed in the next section.

5.3 Ensemble Approach

It was observed that the models *all-mpnet-base-v2*, *all-distilroberta-v1* showed comparatively better results and hence were assigned higher weights as compared to the models *nli-roberta-base-v2*, *paraphrase-albert-small-v2* which showed comparatively worse results and therefore were assigned lower weights. The final weights assigned are as follows:

- *nli-roberta-base-v2:* 0.5
- *paraphrase-TinyBERT-L6-v2:* 1.0
- *all-mpnet-base-v2:* 5.0
- *all-distilroberta-v1:* 5.0
- *paraphrase-albert-small-v2:* 0.5
- *allenai/specter:* 3.0

After assignment of weights to the models, the final score of the similarity of a document with the given query was calculated using the product of their cosine similarity and the weight of the model used. Then the documents were sorted based on the final

score which is the summation of their scores over all models and the required number of high scoring documents are retrieved.
The multiprocessing approach was also tried. The documents and query were passed into all the 6 running processes which would be running individual models at the same time.

6 Analysis of Results

After performing all the experiments on various models as discussed above the results that came out of them are discussed in this section. The results obtained are as follows:

6.1 Base Model

6 models gave been used in this section to generate the embeddings of both facet based queries as well as the entire datset. This is followed by testing them on the given metrics as given by Mysore et al., 2021. The results generated matches with those obtained by Mysore et al., 2021. Additionally, in this section we trained 3 additional models which are *all-distilroberta-v1*, *all-mpnet-base-v2*, *paraphrase-albert-small-v2*. Their results are better than the previous models.

The *all-distilroberta-v1* models has 40% less parameters than RoBERTa model due to which generates the encodings faster than RoBERTa while keeping the metrics better than the former.
The *all-mpnet-base-v2* is a larger model which took around thrice the time taken by *all-distilroberta-v1* to generate the embeddings but the metrics are higher than the others.
The *paraphrase-albert-small-v2* is smaller model which creates the embeddings faster (almost in half of the time taken by *all-distilroberta-v1*) while keeping the results similar as RoBERTa.
All the results has been included in the appendix section.

6.2 Grid Search

The grid search is mainly done to improvise over the base models on our dataset. On applying the grid search method we get that number of layers(n_layers) for which best results were coming were when $n\_layers = 7$. This fact can be easily seen in the plots generated. Even in some cases $n\_layers = 8, 9$ preforms better than the model w the default value of 6.
Thus, performing grid_search on n_layers improves the *DistilBertModel* to some extent.
The plots of different metrics are given in the appendix section.

6.3 Ensemble Approach

It was observed that the ensemble approach showed better results than almost all models. Since ensemble approach reduces overfitting of the model on the query, and makes the model more robust, it is more preferred. Also, during testing for queries, it returns the most relevant docs in slightly less than 2 seconds per query which is considered the standard benchmark for search time. The metrics generated by this approach are mentioned in the appendix section.

7 Future work

7.1 Scraping the H-index

Retrieved results can be scraped for H-Index and citations of the authors to provide an additional feature in the search engine. Features such as results sorted by H-Index or citations and also by year of publishing can be added.

7.2 Expanding to Full S2ORC Corpus

Dataset can be further expanded to the full S2ORC corpus. Embeddings can be efficiently precalculated to get a better candidate pool and make a general purpose research paper query tool.

7.3 Clean and Augment Dataset

The data contains special unicode characters and latex snippets. These can be removed or modified into natural language.

7.4 Applying Semantic Similarity and Intent

By incorporating additional parameters to comprehend the semantic context of a user's queries, a more profound understanding of their intent can be made. Techniques such as word embedding or Latent Semantic Analysis (LSA) can be utilized to generate vector representations of the user's queries.

7.5 Improving Results through User Feedback

A user feedback system can be incorporated into Query By Example (QBE) using machine learning techniques to improve the relevance of search results. The system collects user feedback on search results and uses this feedback to continuously refine the search algorithm.

Acknowledgements

We would like to express our gratitude to all those who contributed to the completion of this research project. First and foremost, we would like to thank our guide, Dr. Somak Aditya, for his guidance, support, and invaluable feedback throughout this project. We are also grateful to Mr. Sachin Vashishtha for providing us with the necessary resources and support throughout the project.
We would like to thank Dr. Mainack Mondal for his ideas and suggestions, all of which had a great impact on our research work.

References

Dongdong Zhang Ping Zhang Yu Su Bernal Jim´enez Guti´errez, Juncheng Zeng. 2020. Document classification for covid-19 literature.

Tao Qi Yongfeng Huang Xing Xie Chuhan Wu, Fangzhao Wu. 2022. Noisytune: A little noise can help you finetune pretrained language models better.

Khalid El-Arini and Carlos Guestrin. 2011. Beyond keyword search: Discovering relevant scientific literature.

Steve Lawrence Kurt D. Bollacker and C. Lee Giles. 2000. Discovering relevant scientific literature on the web.

Mark Neumann Rodney Kinney Kyle Lo, Lucy Lu Wang and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. in proceedings of the 58th annual meeting of the association for computational linguistics.

Stephen Cranefield M.A. Angrosh and Nigel Stanger. Contextual information retrieval in research articles: Semantic publishing tools for the research community.

Sheshera Mysore, Tim O'Gorman, Andrew McCallum, and Hamed Zamani. 2021. CSFCube - a test collection of computer science research articles for faceted query by example. in proceedings of the thirty-fifth conference on neural information processing systems datasets and benchmarks track. 2.

Chong Wang and David M. Blei. 2011. Collaborative topic modeling for recommending scientific articles.

A WebApp Interface (GUI Interface)

A WebApp has been created to better visualise our work in the form of a Graphical User Interface using python streamlit. The WebApp has three components through which user gives a query which are a dropdown to select the facet (all, background, method, results), a slider to choose the number of documents to retrieve and a search bar to input the query. When search button is entered after filling all the details it gives the documents retrieved along with its ensemble score.

B Results

B.1 Example Queries and Results (for Ensemble Learning)

Query: NLP
Facet: all
Number of Docs: 2
Results Returned:

- **Title:** The Stanford CoreNLP Natural Language Processing Toolkit
  Score: 9.677622374919379
  Authors: Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, David McClosky
  Year: 2014
  DOI: 10.3115/v1/P14-5010
  Venue: ACL
  Abstract: We describe the design and use of the 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.
  We suggest that this follows from a simple, approachable design, straightforward interfaces, the inclusion of robust and good quality analysis components, and not requiring use of a large amount of associated baggage.

- **Title:** Natural language processing future
  Score: 9.279081539093376
  Authors: M. Chandhana Surabhi
  Year: 2013
  DOI: 10.1109/ICOISS.2013.6678407
  Venue: 2013 International Conference on Optical Imaging Sensor and Security (ICOSS)
  Abstract:
  Natural Language Processing is a technique where machine can become more human and thereby reducing the distance between human being and the machine can be reduced. Therefore in simple sense NLP makes human to communicate with the machine easily. There are many applications developed in past few

decades in NLP. Most of these are very useful in everyday life for example a machine that takes instructions by voice. There are lots of research groups working on this topic to develop more practical are useful systems. Natural Language Processing holds great promise for making computer interfaces that are easier to use for people, since people will hopefully be able to talk to the computer in their own language, rather than learn a specialized language of computer commands. For programming, however, the necessity of a formal programming language for communicating with a computer has always been taken for granted. We would like to challenge this assumption. We believe that modern Natural Language Processing techniques can make possible the use of natural language to express programming ideas, thus drastically increasing the accessibility of programming to non-expert users. To demonstrate the feasibility of Natural Language Programming, this paper tackles what are perceived to be some of the hardest cases: steps and loops.

Query: Neural Network
Facet: background
Number of Docs: 4
Results Returned:

- **Title:** A Sequential Model for Multi-Class Classification
  Score: 7.521788252177167
  Authors: Yair Even-Zohar, Dan Roth
  Year: 2001
  DOI: None
  Venue: EMNLP
  Abstract: Many classification problems require decisions among a large number of competing classes. These tasks, however, are not handled well by general purpose learning methods and are usually addressed in an ad-hoc fashion.

- **Title:** Sentiment analysis is not solved! Assessing and probing sentiment classification
  Score: 6.971706352075436
  Authors: Jeremy Barnes, Lilja Ovrelid, Erik Velldal
  Year: 2019
  DOI: 10.18653/v1/w19-4802
  Venue: ArXiv
  Abstract: Neural methods for SA have led to quantitative improvements over previous approaches, but these advances are not always accompanied with a thorough analysis of the qualitative differences. Therefore, it is not clear what outstanding conceptual challenges for sentiment analysis remain. In this work, we attempt to discover what challenges still prove a problem for sentiment classifiers for English and to provide a challenging dataset.

- **Title:** ABL: Alignment-Based Learning
  Score: 6.544283949675442
  Authors: Menno van Zaanen
  Year: 2001
  DOI: 10.3115/992730.992785
  Venue: Proceedings of the 18th International Conference on Computational Linguistics (COLING); Saarbrucken, Germany. pages 961-967
  Abstract: This paper introduces a new type of grammar learning algorithm, inspired by string edit distance (Wagner and Fischer, 1974).

- **Title:** Stacking classifiers for anti-spam filtering of e-mail
  Score: 6.4259512215625705
  Authors: G. Sakkis, I. Androutsopoulos, G. Paliouras, V. Karkaletsis, C. D. Spyropoulos, P. Stamatopoulos
  Year: 2001
  DOI: None
  Venue: Proceedings of"Empirical Methods in Natural Language Processing"(EMNLP 2001), L. Lee and D. Harman (Eds.), pp. 44-50, Carnegie Mellon University, Pittsburgh, PA, 2001
  Abstract: We evaluate empirically a scheme for combining classifiers, known as stacked generalization, in the context of anti-spam filtering, a novel cost-sensitive application of text categorization. Unsolicited commercial e-mail, or"spam", floods mailboxes, causing frustration, wasting bandwidth, and exposing minors to unsuitable content.

B.2 Results of Grid Search

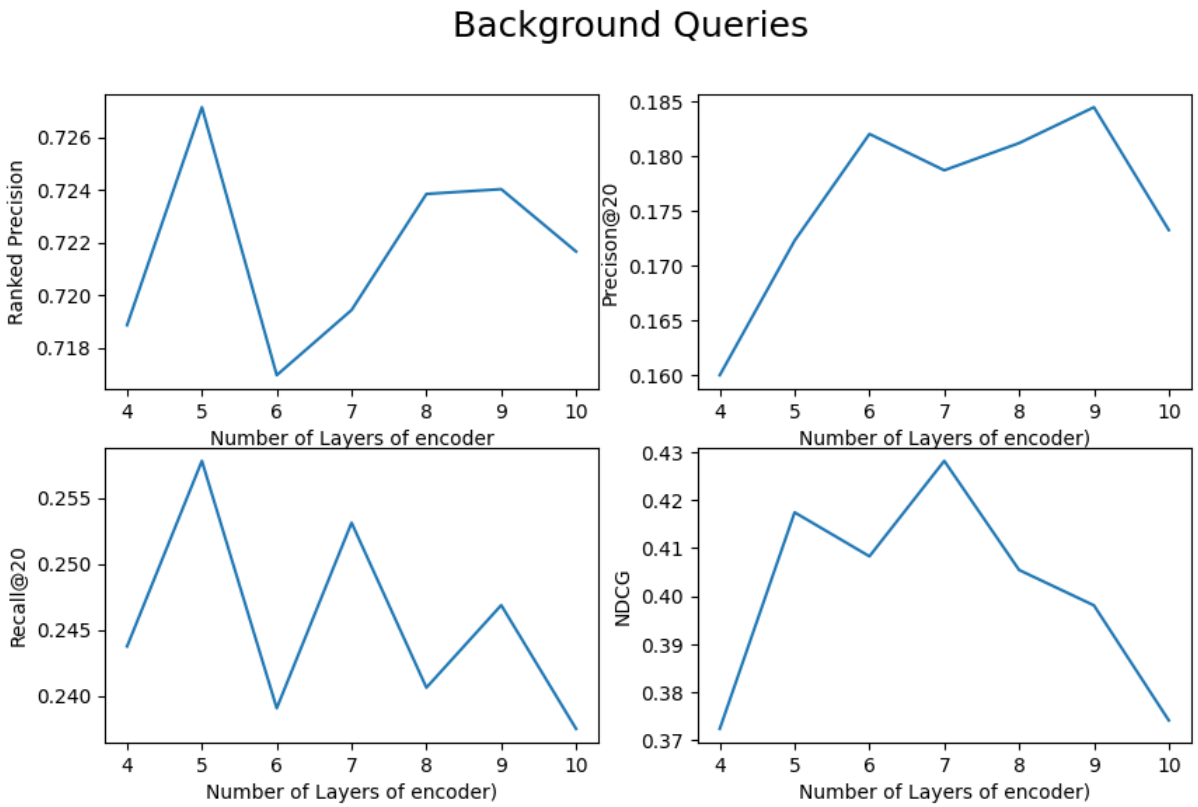

Figure 1: For Background Queries

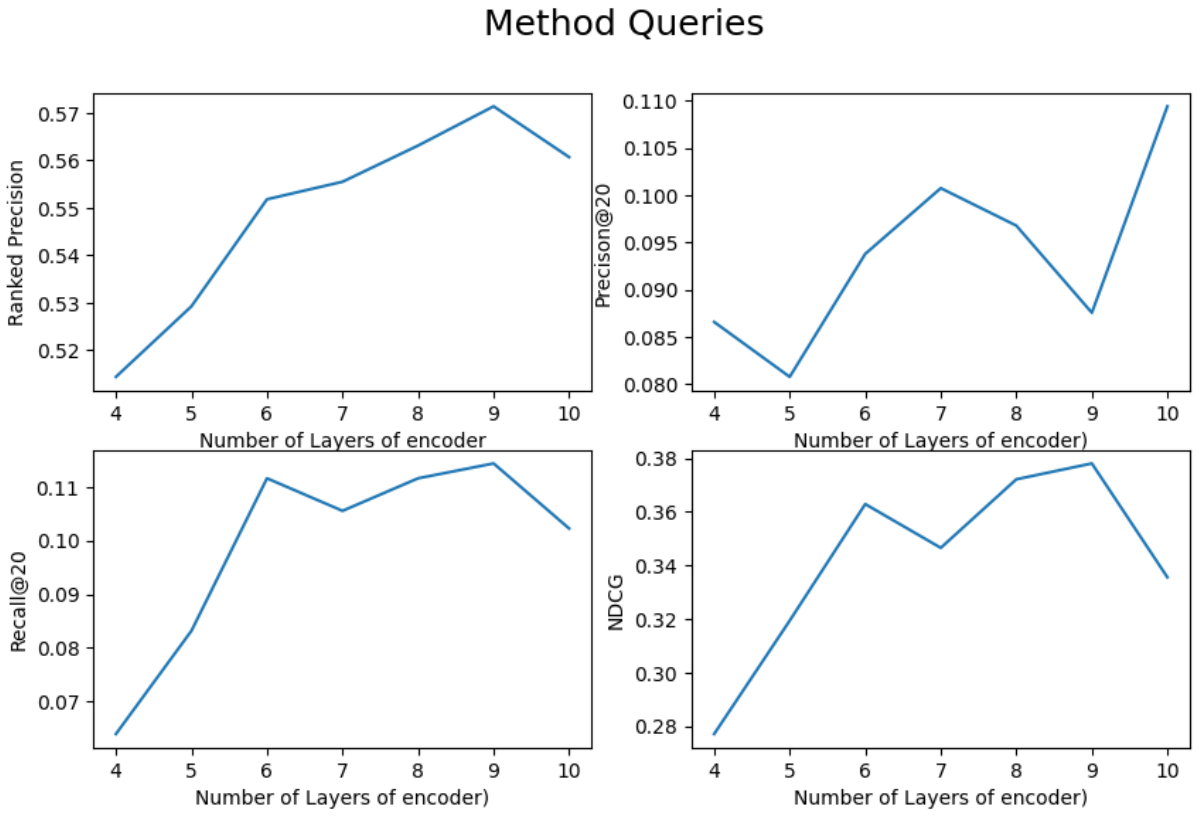

Figure 2: For Method Queries

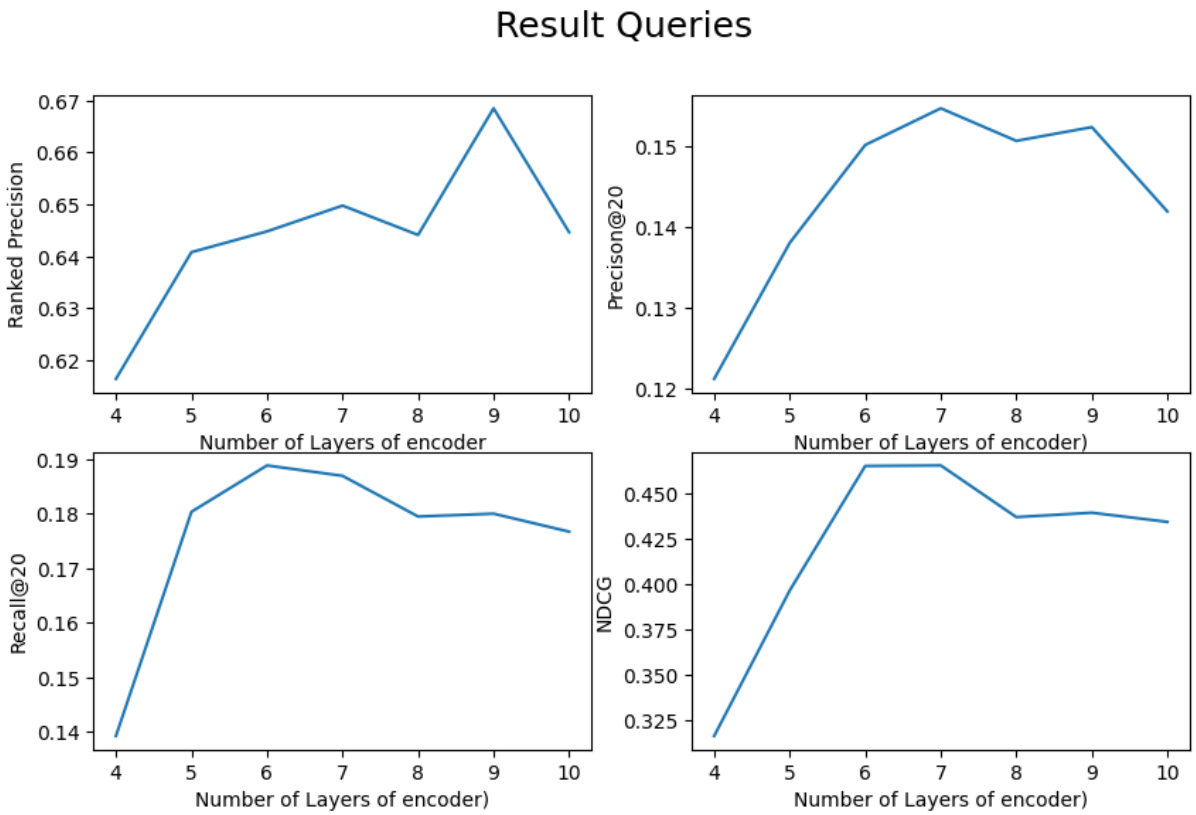

Figure 3: For Results Queries

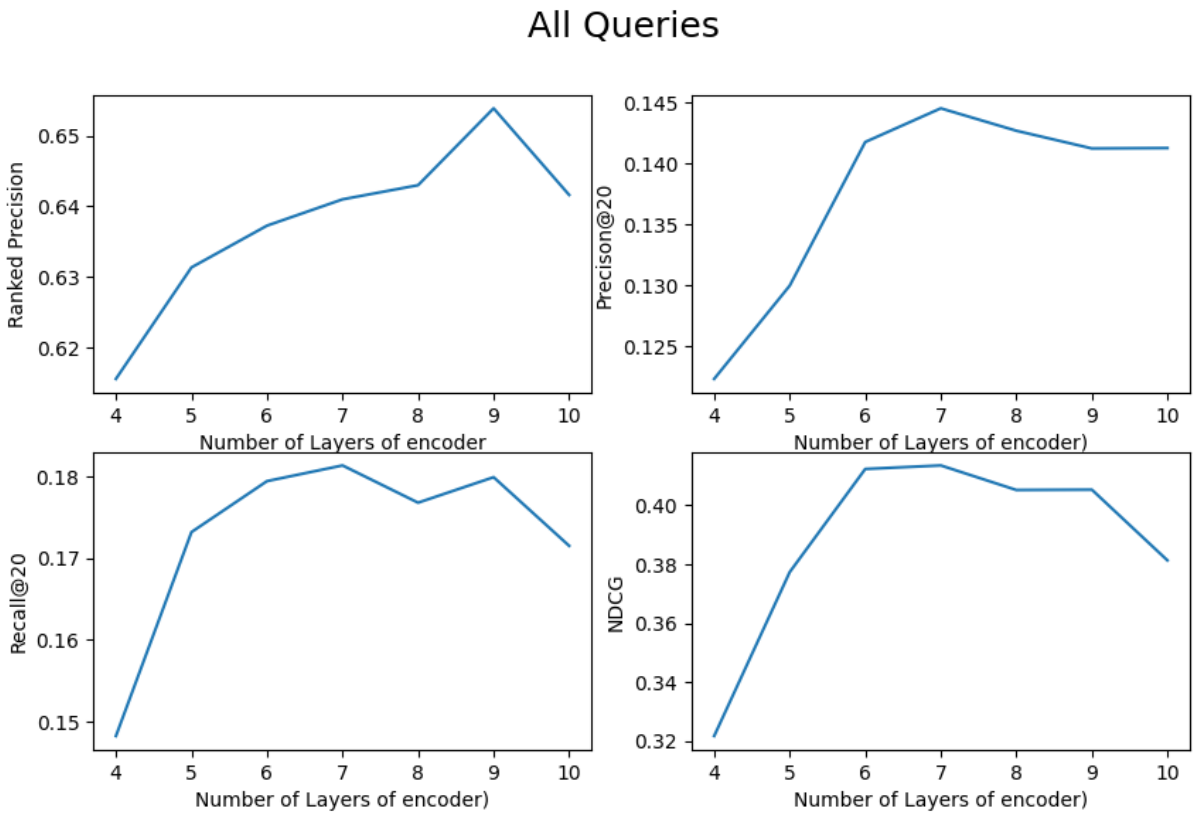

Figure 4: For Aggregate Queries

B.3 Results of Base Model

	Background					*Method*				
	RP	**P@20**	**R@20**	**NDCG$_{\%100}$**	**NDCG$_{\%20}$**	**RP**	**P@20**	**R@20**	**NDCG$_{\%100}$**	**NDCG$_{\%20}$**
BERT_NLI	0.2004	0.2750	0.4328	0.7735	0.5781	0.1656	0.1028	0.3265	0.6056	0.3393
BERT_PP	0.2332	0.3109	0.5024	0.7760	0.5974	0.1826	0.0998	0.3388	0.6350	0.3865
SPECTER	0.2353	0.3125	0.4936	0.7994	0.6407	0.1843	0.1097	0.4107	0.6269	0.3744
DISTILBERT (pretrained)	0.3249	0.3781	0.6224	0.8544	0.7264	0.1416	0.1490	0.4753	0.6731	0.4518
ALL_MPNET	0.2797	0.3469	0.5750	0.8536	0.7166	0.2005	0.1641	0.4734	0.6633	0.4544
TF-IDF	0.1777	0.2266	0.3789	0.7262	0.4795	0.0892	0.0748	0.2434	0.5439	0.2440
ALBERT	0.2510	0.2828	0.4119	0.7809	0.5951	0.1285	0.1045	0.3568	0.5994	0.3346
ENSEMBLED MODEL	0.2959	0.3594	0.5829	0.8583	0.7187	0.1981	0.1630	0.4940	0.6769	0.4656

	Result					*Aggregated*				
	RP	**P@20**	**R@20**	**NDCG$_{\%100}$**	**NDCG$_{\%20}$**	**RP**	**P@20**	**R@20**	**NDCG$_{\%100}$**	**NDCG$_{\%20}$**
BERT_NLI	0.1278	0.1826	0.4023	0.6538	0.4072	0.1643	0.1859	0.3866	0.6768	0.4404
BERT_PP	0.1548	0.2273	0.5484	0.7048	0.5183	0.1898	0.2119	0.4631	0.7043	0.4995
SPECTER	0.1904	0.2856	0.6814	0.7649	0.6022	0.2030	0.2353	0.5286	0.7296	0.5379
DISTILBERT (pretrained)	0.2357	0.2818	0.6258	0.7817	0.6246	0.2336	0.2688	0.5745	0.7688	0.5996
ALL_MPNET	0.2498	0.3113	0.7276	0.7899	0.6497	0.2429	0.2733	0.5919	0.7680	0.6055
TF-IDF	0.1083	0.1333	0.3067	0.6425	0.3851	0.1247	0.1437	0.3084	0.6361	0.3676
ALBERT	0.1603	0.2109	0.4837	0.6909	0.4804	0.1795	0.1986	0.4174	0.6895	0.4687
ENSEMBLED MODEL	0.2564	0.3085	0.7178	0.7916	0.6610	0.2496	0.2762	0.5984	0.7748	0.6139

Fig 5: Results of Base Model

Labour Division among the members

Name	Roll Number	Work Division
Ashwani Kumar Kamal	20CS10011	1. Writing the code to generate the embeddings of TF-IDF, BERT PP etc. 2. Reading research papers related to BERT models. 3. Creating the Web application of SCIATICA in streamlit as well as doing the grid search. 4. Creating the Presentation.
Hardik Pravin Soni	20CS30023	1. Writing the code to generate the embeddings of BERT NLI and BERT PP etc. 2. Reading about S2ORC dataset and reading the Papers related to other approaches to solve the document retrieval problem. 3. Reading about ensembling models and writing the code for ensemble model along with Sourabh. 4. Creating the Presentation.
Shiladitya De	20CS30061	1. Writing the code for computing the metrics for different models. 2. Analyzing the results of the Base and ensemble models. 3. Reading about ensemble learning and writing the code of the same along with Hardik. 4. Preparing the report.
Sourabh Soumyakanta Das	20CS30051	1. Writing the code for computing the metrics and analyzing the results of the base models. 2. Reading papers related to evaluation metrics like NDCG (Chong Wang et. al.) 3. Researching about grid search, implementing it and analyzing the results. 4. Preparing the report.