Deep Natural Language Processing Project Classification of Scientific Articles using SPECTER Embedding Model

Thomas Avare
Master of Data Science, PoliTo
exchange student at Politecnico di Torino from Grenbole INP - ENSIMAG
Torino, Italy/Grenoble, France
thomas.avare@grenoble-inp.fr

***Abstract*—SPECTER, standing for *Scientific Paper Embeddings using Citation-informed TransformERs* is a process to generate document-level embedding of scientific documents. SPECTER is based on pretrained Transformer language model and incorporating the citation graph to improve documents relatedness. SPECTER was benchmarked using SciDocs, an additional contribution to SPECTER, along previous scientific embedding models. SciDocs is an evaluation benchmark consisting of 7 document-level tasks such as citation prediction, document classification and recommendation.**

The goal of this project is to use SPECTER as a classification tool. The first part consists of understanding the principle of document-level embedding with SPECTER and its specificity, using SPECTER and SciDocs to reproduce the results presented in the original paper, focus on the text classification part being the subject of this project and some extension(s).

In the approach of the development of SPECTER, The main idea was to use the citation-graph to introduce document relatedness into a triplet loss with the query paper, a cited paper and a non cited paper as inputs and thus in the embedding. An idea for and extension for this project was to modify the loss to also include the date difference between the query paper and the positive paper to add/modify the information regarding the document relatedness.

I. Introduction to SPECTER

SPECTER (Cohan et al., 2020 [1]) is an attempt to improve the embedding of sentences or whole documents by the mean of pre-trained language models to learn embeddings for scientific documents.The main idea was to incorporate inter-document relatedness into the Transformer language models such that the embedding could be more effective various tasks without any task-specific training for the embedding model. Transformer language models already exists, such as BERT, none of them used any inter-document context/information during their training. SPECTER uses the citation-graph to add inter-document context/information and these informations are used during the training of the model by the introduction of a triplet margin loss with a query paper $\mathcal{P}^Q$, a positive paper $\mathcal{P}^+$ which corresponds to a paper cited by the query paper and a negative paper $\mathcal{P}^-$ which corresponds to a non-cited paper by the query paper. So the triplet margin loss looks like:

$$\mathcal{L} = \max\left\{d\left(\mathcal{P}^Q, \mathcal{P}^+\right) - d\left(\mathcal{P}^Q, \mathcal{P}^-\right) + m, 0\right\}$$

where d is a distance between two embeddings, m a loss margin hyperparameter.

There are also different type of negative cases to a query paper. The ”easy negatives” which are random papers not cited by the query paper and ”hard negatives” which are cited paper by a cited paper but not by the query paper ($\mathcal{P}^1 \xrightarrow{cite} \mathcal{P}^2 \xrightarrow{cite} \mathcal{P}^3$ but $\mathcal{P}^1 \not\xrightarrow{cite} \mathcal{P}^3$). This distinction was made provide more nuance in the training.

They also used pretrained transformers network as a foundation for SPECTER, specifically the SciBERT (Beltagy et al., 2019 [3]). After the training, it ended up becoming the state of the art model in embedding model used for various task (see next part for an introduction to SciDocs evaluation suite.

It’s interesting to note that the idea of the citation graph was reused, the main idea was to transform the citation graph into a discrete space and pick the positives, easy negatives and hard negatives according to their distance in that space to the query paper and this model has become the new state of the art embedding model outperforming SPECTER (SciNCL, Ostendorff et al. 2022 [2]).

II. Introduction to Scidocs

SciDocs (Cohan et al., 2020 [1]) is an evaluation framework introduce to compare different embedding models on different tasks. These tasks are Document classification, Citation prediction, user activity and Recommendation. The document classification is divided in two different type: MeSh classification, standing fro *Medical Subject Headings*, and the paper topic classification known as MAG, standing for *Microsoft Academic Graph*. The MeSH dataset consists of 23K academic medical papers distributed in 11 classes and the MAG consists of 25K papers distributed in 19 classes. Since we’re not interested in other tasks, we’re not going to talk more about them.

III. Goal of this Project

The goal of this project is to use SPECTER as a tool for scientific document classification, so for MAG and MeSH classification. Another goal of this project was to introduce an extension to the initial project, we will discuss about my idea in a forthcoming section, unfortunately we weren't able to complete it but we will still briefly talk about it.

IV. The Classification Task

As said before, SPECTER is an embedding model made to be versatile so it doesn't need any task-specific training.

A. Importing SPECTER

The first step to our classification journey was to import SPECTER. To do so, we simply cloned the official Github repository. After following the installation instructions, we had some versioning issues and one of my library wasn't downloading in the right place so we had to solve these issues. Afterwards, we were finally able to compute the first embeddings. Unfortunately, my computer doesn't have any GPU when we first started working and we decided to switch to a Google Colab notebook to pursue the experiments.

Before training my first models, we imported the SPECTER model accessible through the huggingFace transformer's library, then we imported the data provided by SiDocs. This contains all the data used for the SciDocs benchmarking, including the information of all papers (paper id, author, title, abstract, date of publication, citations and cited by), their classification (MAG and MeSH) and their embeddings. To import and manage all these informations, we used Pandas DataFrames which is really easy and convenient for these kind of tasks.

B. Importing the Data

After the first experiments to train models, we realized that it was not working for a very simple reason. The embeddings provided with SciDocs are those computed with the SPECTER model from the SPECTER GitHub model and the SPECTER model from the HuggingFace Transformer library computes different embeddings for the same title and abstract. So we re-computed the embeddings used for the MAG and MeSH classifications and saved it in a json file, for every paper, we have the paper id, The paper id, the title, the abstract and its embedding. To prepare the data for the classification, we simply have to import these information into a data frame and import the classification and merge it according to their paper id. The classification is already split into a train/test dataset for reproducibility purposes for the SciDocs benchmarking.

C. Training Models

After correctly importing the data and solving our embedding issues. Now we can train models for our classification task. We trained multiple types of models and will compare their performances according to their f1 score on the test dataset, just like they did with SciDocs, so we can compare our models with the ones from the SciDocs benchmarking. Unfortunately, even by mimicking the training routine used for the SciDocs, we were never able to level or top it.

To try to top the models for the SciDocs, we tried to mimic the routine, at least on linearSVC and SVC, which is a parameter search over the C parameter, a regularization parameter.

	MAG	MeSH
Linear SVC	74.72	84.87
SVC	74.34	82.89
Gaussian naive bayes	69.79	68.19
K-neighbors	69.25	77.78
SGD	69.93	77.78
CNN	72.05	81.00
SPECTER SciDocs benchmark	81.95	86.44

For the other models, we also did this parameter search search, mostly experimenting with various loss functions.

Unfortunately, the model with the best results is the linear SVC didn't always converge during the parameter search and it took too long when we increased the maximum number of iterations. Also when training a model with the same parameters as the one found with the parameter search, the results weren't improved.

For the CNN network, we empirically tried various architectures, and also "known" architectures. The best one was simply one dense layer with 768 units, so the size of the vector provided by SPECTER after embedding.

D. Using the Models

To simplify the use of the models, we did a simple implementation of the models (except for the Cnn now) and also personal model if someone ever use the implementation. It uses a simple pipeline, It embeds the title and abstract of the paper using the SPECTER model from the hugging face transformer library and then the model is used to classify the embedding. The implementation was made so we can easily switch between models and type of classification, it can also provide the scores per class instead of the classification.

We also did a little benchmark implementation similar to the SciDocs benchmarking task

V. Extension Idea: Modifying the Triplet Loss

For the extension, we had the idea to modify the triplet loss. The actual triplet loss, as said earlier, uses the citation graph and looks like:

$$\mathcal{L} = \max\left\{d\left(\mathcal{P}^Q, \mathcal{P}^+\right) - d\left(\mathcal{P}^Q, \mathcal{P}^-\right) + m, 0\right\}$$

The idea here is to use more information of the paper to modify the citation graph to have more information, to add another dimensionality to the citation graph. To do so, we thought of modifying the weight of the link of the citation graph using the years of the different papers. After having this idea, we had to think of a way to implement it and give it some sense and something that "naturally" came was to use a law similar to the radioactive decay and the half-life. We will not discuss about the radioactive decay, but the idea is that, if two cited papers are spaced out in time by a certain value, the link is twice as low as the original link. With a difference of 10 years (arbitrary) for the half-life, formalizing the idea, we have:

$$e^{10\alpha} = \frac{1}{2} \Longleftrightarrow \alpha = -\frac{\ln 2}{10}$$

This function could be used to both the distance from the query paper to the positive paper and the (hard) negative paper, but we thought it could be more interesting on only the distance from the query paper to the positive paper. So our new triplet loss would look like:

$$\mathcal{L}' = \max\left\{d\left(\mathcal{P}^Q, \mathcal{P}^+\right) \times e^{-\frac{\ln 2 |y_{\mathcal{P}^Q} - y_{\mathcal{P}^+}|}{10}} - d\left(\mathcal{P}^Q, \mathcal{P}^-\right) + m, 0\right\}$$

Unfortunately we weren't able to retrain the model with this loss, but we will probably try in the future when we have the time and means.

VI. Conclusion

This project has actually not only been a project but a whole journey throughout the whole semester. Being introduced and discovering the field has been really interesting and being able to better understand how some tools that we use everyday work was gratifying. Despite not being able to fully complete my project and having to work alone on that project, it was really interesting working on concrete data and problems and with recent (almost state-of-the-art) tools and having to deal and solve different issues with typical DNLP pipelines and more general coding issues.

References

[1] Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld.
2020. SPECTER: Document-level Representation Learning using Citation-informed Transformers

[2] Malte Ostendorff, Nils Rethmeier, Isabelle Augenstein, Bela Gipp, Georg Rehm.
2022. Neighborhood Contrastive Learning for Scientific Document Representations with Citation Embeddings

[3] Iz Beltagy, Kyle Lo, and Arman Cohan.
2019. SciB- ERT: A Pretrained Language Model for Scientific Text. In EMNLP.