SPECTROID: A Centroid-Based Citation-Aware Approach to Paper Classification

Capuano Francesco
s295366
s295366@studenti.polito.it

Matteotti Matteo
s294552
s294552@studenti.polito.it

Porcelli Enrico
s296649
s296649@studenti.polito.it

***Abstract*—The quality of text embedding has a profound impact on several document-level tasks such as paper classification, citation prediction and paper recommendation. In previous works such as SPECTER inter-document signals have been incorporated into the training phase of SciBERT, achieving SOTA performances on a multitude of tasks. In this work, we introduce *SPECTROID*, a variation of SPECTER that uses a different sampling method for the selection of *positive* papers, which we extrinsically evaluate on the MeSH and MAG paper classification tasks. We then show how to improve SPECTER classification performance on the MeSH and MAG tasks by mounting a MLP classification head on top of SPECTER. This simple yet powerful intuition yielded a significant increase in Macro F1 Score for both tasks. For the sake of full reproducibility, we published our code at: github.com/fracapuano/Spectroid.**

I. Problem Statement

NLP tools have become an essential component of those platforms dealing with huge amounts of scientific publications.

Key tasks in managing scientific publications are *paper classification*, *citation prediction* and *paper recommendation*. While various approaches have been tried to carry out such tasks, the best performing ones do all heavily rely on the quality of the *embeddings* (*i.e.*, numerical representations) that can be obtained from the raw textual data. Here, we focus on evaluating embedders for two paper classification (PC) tasks, namely MAG [11] and MeSH [8]. In particular, we here compare the results obtained using a standard SPECTER embedder [4] and the ones obtained with a variant of SPECTER itself, which from now on we will refer to as *SPECTROID*. This variant relies on using the number of citations received by each document to compute its probability of being picked as a positive paper, thus aiming at introducing a centroid-like structure in the embedding latent space, for the sake of exploiting the citation network structure to actually perform classification.

We fed to both versions of the SPECTER model a triplet of papers in the form of a query ($\mathcal{P}^Q$), a positive ($\mathcal{P}^+$) and a negative paper ($\mathcal{P}^-$). All the embedder would receive for all $\mathcal{P}$ is a concatenation of $\mathcal{P}$'s title and abstract. By optimizing a loss that enforces a citation-aware structure in the whole collection of papers, SPECTER's authors were able to obtain embeddings achieving SOTA performances in PC.

Together with this triplet, the model will also need the identification code of the documents cited by the query paper. Moreover, SPECTROID would also require the total number of citations received by each paper.

It must be noted that once the trained model is deployed, only the metadata related to the query paper will be required for obtaining the embedding, as both SPECTER and SPECTROID do aim at defining a citation-aware *embedder* for textual content.

In their work, SPECTER's authors classify the embeddings produced through their embedding using an SVM classifier, which will output the predicted label. In the second variant we produced on top of SPECTER we modified the way with which MeSH and MAG classification is performed and, in particular, used various classification heads to properly classify the embeddings. During our experiments (all of which were conducted using Pytorch [9]), we made great use of Wandb [2].

II. Methodology

A. NLP pipeline

The main novelties of this work are two:

- *SPECTROID extension*: We first created the training triplets (composed of tokenized metadata from query, negative and positive papers) from a subset of the Scidocs dataset introduced in [4]. Differently from Specter, the positive paper is not chosen uniformly among the query's citation network but rather it is chosen proportionally to the number of citations it has globally. This is justified by the intuition that it might exist a group of *centroid* papers, *i.e.* papers that are highly cited inside of one category only as *Attention is all you need* is very cited in Deep Learning papers while it is very likely very seldom cited in Astrophysics-related works.
  We then compared the quality of our embeddings with the ones obtained by SPECTER's authors on the MeSH and MAG classification tasks. Lastly, we repeated the classification procedure on 50k papers subset of the much larger (and more hetereogenous) arXiv Dataset [3].
- *Classification Heads extension*. To assign to each paper a label, we first concatenate title and abstract of said paper and then tokenize the text using the SPECTER's tokenizer. The result of this clearly is a numerical representation (yet not contextualized and in a sparse high dimensional representation) of the paper itself, which

we then feed to SPECTER to otbain the corresponding embedding. Unlike SPECTER's authors, however, we did not produce task-agnostic embeddings to be classified using standard ML algorithms (such as SVM or KNN), but rather we mounted a Classification Head on top of SPECTER to fine-tune the embedder for the two tasks considered, namely MeSH and MAG. To guarantee a fair comparison between our approach and the reference one, we evaluated the performance of our classification model using the same metric of SPECTER, namely a *Macro-Avg F1 Score.*

B. SciBERT module

The SciBERT [1] encoder is SPECTER's (and, therefore, SPECTROID's too) backbone. SciBERT is a BERT model [6] trained on scientific documents. It is trained using a corpus obtained from semanticscholar.org which contains more than 1M full-text papers. An ad-hoc vocabulary called *scivocab* has been built by the authors to better suit the training corpus. SciBERT reached SOTA performances in sentence classification tasks on the SciCite, PubMed and ACL-ARC datasets. Albeit very promising, due to its training procedure (which is designed to resemble BERT's) SciBERT is not able to capture inter-document relatedness and for this reason its performances on document-level tasks such as scientific paper classification are not as promising as the ones on sentence-level tasks that we just mentioned.

C. SPECTER

SPECTER overcomes the limitations of SciBERT by incorporating a signal of inter-document relatedness into the model, which is retrieved from the information contained in the citation network of each paper $\mathcal{P}$. This signal is captured by forming triplets composed of the metadata of a query ($\mathcal{P}^Q$), a positive ($\mathcal{P}^+$) and a negative ($\mathcal{P}^-$) paper. Positive papers are papers cited by the query one while negative papers are papers that are not cited by the query paper but that may or may not be cited by the positive one (this information gives us the distinction between hard and easy negatives, respectively).

These training triplets are then fed into a SciBERT embedder which is trained to minimize the following loss function:

$$\mathcal{L} = \max\big\{\big(d(\mathcal{P}^Q, \mathcal{P}^+) - d(\mathcal{P}^Q, \mathcal{P}^-) + m\big), 0\big\} \tag{1}$$

where $d(\mathcal{P}^A, \mathcal{P}^B) = \|v_A - v_B\|_2$ is a measure of the difference between two different embeddings, with v being the pooled output of the SciBERT's encoder. Clearly enough, this loss incentivizes the model to embed the query and the positive paper similarly while making the embedding of query and negative paper different.

D. SPECTROID

What distinguishes our implementation from SPECTER's, is the criteria with which the positive paper is selected. In the following section we will indicate with $(\mathcal{P})^{outC}$ the set of documents *cited* by $\mathcal{P}$ and with $(\mathcal{P})^{inC}$ the set of documents *that do cite* paper $\mathcal{P}$.

In [4], positive papers for $\mathcal{P}^Q$ are selected by drawing a paper at random from a uniform probability distribution defined over $(\mathcal{P}^Q)^{outC}$. In our implementation, this probability distribution is instead expressed as:

$$\mathbb{P}(\mathcal{P} = \mathcal{P}^+) = \begin{cases} \frac{|\mathcal{P}^{inC}|}{Z} & \text{if } \mathcal{P} \in (\mathcal{P}^Q)^{outC} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where:

$$Z = \sum_{\mathcal{P} \in (\mathcal{P}^Q)^{outC}} |\mathcal{P}^{inC}| \tag{3}$$

This formulation of $\mathbb{P}(\mathcal{P} = \mathcal{P}^+)$ will give to papers that have received a high number of citations a very high chance of being picked as the positive one. This tweak to the way triplets are formed stems from the observation that there are some very popular papers that receive a high number of citations in specific fields (*e.g.*, "Attention is All You Need" [12] is a very popular paper with 65k+ citations, the vast majority of which is very likely to be from Computer Science papers). In this sense, positive papers could offer a point of reference for the embedding of other papers in the same field.

E. Classification Head(s)

This section refers to the classification tools utilized in our second extension of SPECTER. In order to allow comparability with the results of the original paper, the procedure has been tested on the standard SPECTER embedder rather than to our re-trained version.

In order to assign to each input paper a label, we tested various architectures for the classification head (CH). The different CHs mainly differ in the order of non-linearity that they introduce with their decision boundaries. The intuition of choosing the appropriate CH based on the non-linearity introduced derives from the fact, also presented in [10], that a single-layer no-activation architecture network resembles a linear decision boundary and that, therefore, by stacking up more and more complex layers one makes, from an empirical standpoint, the classification boundary more *non-linear.*

These are shown in Figure 1. The key difference with respect to the original SPECTER classification is that our implementation back-propagates the error generated by the MLP classifier to both the MLP and the embedder's weights. Although we do not expect to observe major changes in extensively trained parameters of SPECTER, the fine-tuning should yield a significant improvement in the downstream task.

III. EXPERIMENTS

First Extension: SPECTROID

To get comparable results, we decided to use the same dataset and training procedure (and, ultimately, the same code) for the training of both for the original SPECTER embedder (*i.e.*, the one not concerned with our extension) and for *SPECTROID*. In other words, we retrained SPECTER from scratch and compared its results against the ones yielded by SPECTROID.

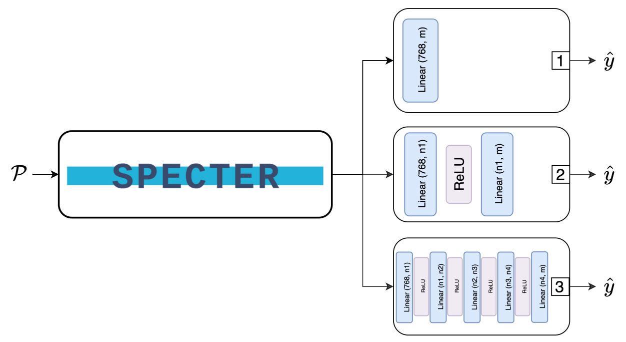

Fig. 1. Classification head architectures. Presented, from top to bottom, in increasing amount of non-linearity that they are able to convey to the decision boundary.

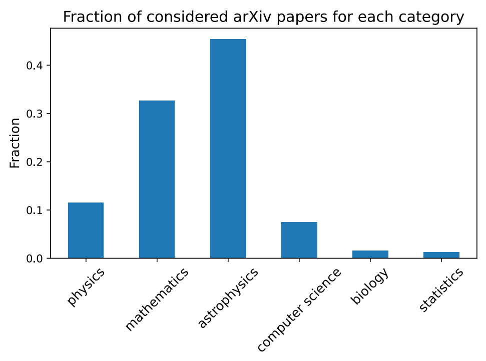

Fig. 2. Number of arXiv papers for each of the considered categories.

1) *Datasets:* Among the papers in Scidocs, we isolated all English papers where both title and abstract are present. We then restricted the citation network to the said papers and we formed 5 triplets for each of them. Due to limited computational resources, we restricted the total number of training triplets to 150k (while in [4] these triplets are 680+K), whilst we keep unchanged the number of validation triplets and test triplets (both set to 80k).

2) *Experimental setting:* Following the procedure implemented by the authors of the model, we set the margin parameter $m = 1$ and used the Adam Optimizer [7] with choice of hyperparameters as suggested by [5].

We trained our model on an AWS EC2 Instance equipped with a single NVIDIA T4 Tensor Core GPU for 2 epochs with a batch size equal to 4. Each training process lasted approximately 16 hours.

3) *Evaluation:* To compare the performance of our embedder against SPECTER, we ran three different classification tasks on three different test sets, embedded with both methods. The first test set is composed of $title + abstract$ for 25k papers coming from the MAG dataset. The second test set is composed of title+abstract for 25k papers coming from the MeSH dataset. The third test set is composed of $title + abstract$ of 50k papers coming from the arXiv dataset [3].

In particular, we decided to create our own subset of labels for the arXiv dataset. Specifically, we associated with each paper the topic it belongs to, according to the *categories* field of the arXiv dataset. We only considered papers belonging to one category only. The categories we have chosen and the label distribution is presented in Figure 2.

4) *Results:* For the 6 different classifications, we trained a Linear SVM classifier using 70/15/15 train/val/test split holdout and the code presented in [4].

The results are presented in table I. We observed a slight increase in performance for MAG (75.45 $\rightarrow$ 74.90), whereas slightly worse performances have been obtained on MeSH (82.68 $\rightarrow$ 82.31) and arXiv subset (76.24 $\rightarrow$ 75.50). We now give a plausible justification for our results. Our fundamental assumption is that the most popular papers would act as a central reference point for their category, resulting in a skewed selection of positive papers inside of different categories. In light of Table I, we analyzed the citation distribution on the Scidocs subset we created. Clearly, only a small portion of the papers in the said dataset is given a label, namely the ones related to MeSH and MAG. We isolated the top-5 popular papers amongst the ones cited by at least two papers with a known MeSH label to then analyze the number of unique MeSH labels (*i.e.*, MeSH categories) of the papers citing them.

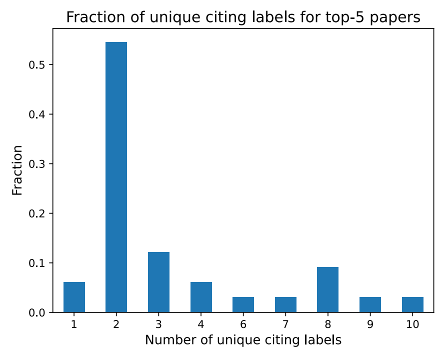

Fig. 3. Distribution of the unique number of MeSH labels of the papers citing the top-5 popular papers in Scidocs, only considering the ones cited by at least two papers with a known MeSH label

Differently from our expectations, we found that for the MeSH dataset highly popular papers are cited by papers belonging to more than one category, as presented in Figure 3. This clearly hinders performance increases for *SPECTROID* as the latent space geometry would clearly be driven towards class super-position for highly cited papers.

Second Extension: MLP-based Classification

5) *Datasets:* Due to computational limitations, we resorted to making use of a portion only of the MeSH and MAG datasets. Namely, we filtered out those papers that do not

	MAG	*MeSH*	*arXiv*
SPECTER	75.45	82.68	76.24
SPECTROID (ours)	**74.90**	**82.31**	**75.50**

TABLE I
RESULTS OF THE FIRST EXTENSION ON MAG, MESH, AND ARXIV CLASSIFICATION

present both title *and* abstract, as well as filtering out the papers not in English. Filtering reduced the dataset by ~24%. After this reduction, the title and abstract of the remaining papers have been concatenated (with a [SEP] token in between) and tokenized to create the input feature for the SPECTER embedder. A column containing the labels is retrieved from auxiliary files and added to the dataset so that the data are now ready to be used for fine-tuning. In particular, we fine-tuned SPECTER for both MeSH and MAG classification. We will refer to the two fine-tuned versions of SPECTER as fSPECTER-MeSH and fSPECTER-MAG, respectively. The MeSH subset we used consists of 23154 labeled examples whereas the MAG subset consists of 14083.

6) Experimental design: Considering the sizes of the datasets considered, papers have been divided into train and test sets following using 90% - 10% holdout sets.

The papers have then been fed to the standard SPECTER model so that the embeddings corresponding to the usual `[CLS]` token would then be fed to an MLP classifier. The fine-tuning stage consisted of 5 epochs only on both datasets with two separate models. We experimented with Google Colab Tesla T4 GPUs with a batch size of 8 (with which the training time of fSPECTER-MeSH and fSPECTER-MAG would take approximately 2 hours and 40 minutes and 1 hour and 40 minutes, respectively) and A100 GPUs with a batch size of 32 (with which the training time of fSPECTER-MeSH and fSPECTER-MAG would take approximately 1 hour and 30 minutes respectively). We used the AdamW optimizer with a constant learning rate of 5e-5 and used it as classification loss the Cross-Entropy Loss.

We then tested various architectures for the MLP block:

- *CH1: No-hidden layers*, i.e. a classification head using a linear layer only to project SPECTER's output into the label set.
- *CH2: One hidden layer only*
- *CH3: Five hidden layers*

Each hidden layer is formed by 64 units and all CHs do use the ReLU activation function. Lastly, to better test our hypotheses on the classification head structure, we also tested another architecture for the classification head, one with three hidden layers, that we will refer to *CH2.2*. Differently from all other CHs, CH2.2 has been trained for classification with a much larger batch size (32 versus 8) and using A100 GPUs.

The MLP classifier mounted on top of SPECTER in order to fine-tune it was trained using a cross-entropy loss. As in the original paper, the performance has been tested on the MeSH and MAG classification tasks by means of the corresponding Macro-Avg F1 Score.

7) Results: The results we obtained can be found in Table II and Table III. We did obtain an increase in performance for both tasks, amounting to a +7.5 increase on the Macro-Avg F1 Score for MeSH and a +17 increase on the same metric for MAG PC. These results can be justified with an argument deriving from an observation of SPECTER's results on the same tasks. As reported in Table III, SPECTER's authors did indeed obtain a Macro-F1 of 87.7 and 79.4,

	fSPECTER-MeSH	***fSPECTER-MAG***
Best Performing CH	CH2	CH1
Worst Performing CH	CH3	CH3
MacroAvg F1 Score	95.20	97.27

TABLE II
RESULTS OF THE SECOND EXTENSION ON MESH AND MAG CLASSIFICATION

	MacroAvg F1 - MeSH	***MacroAvg F1 - MAG***
SPECTER	87.7	79.4
CH1 (ours)	93.70	**97.27**
CH2 (ours)	**95.20**	95.31
CH3	4.09	0.97
CH2.2	*88.16*	*91.18*

TABLE III
MACRO-AVG F1 SCORE FOR THE CLASSIFICATION HEADS TESTED OUT

for MeSH and MAG PC respectively, using a soft-margin Linear Kernel SVM. This result clearly indicates a significant linear separability of classes in the space produced through SPECTER's embeddings, which is a key fact that explains why our one-layer MLP systematically outperforms a 3 layers one such as the one presented in CH2.2. Once more, this directly derives from the high linear separability in the embedding latent space.

In the few epochs considered for fine-tuning, indeed, our one-layer architecture draws a linear decision boundary on data points whose collocation in the considered latent space adjusts as per the effect of loss back-propagation.

In plain words, the quality of (an already very well-performing) linear decision boundary improves during training, as paper embeddings are very likely more and more driven to be linearly separable and a linear decision boundary is fit to moving data points.

Conversely, different (and more complex) CHs do not capture this inherent linear separability of SPECTER's embeddings and, in the 5 epochs considered, are not able to yield the same performances of CH1, even with very similar experimental settings.

IV. CONCLUSIONS

Although SPECTROID turned out to yield sub-optimal embeddings, the analysis of the citation distribution can partly justify its behavior. Citations of very popular papers may be not concentrated in a single category, as we initially thought, but they are shared among two or three of these. This can affect the centroid-like behavior that we expected to observe for these papers because it means that the embeddings of papers from different MAG or MeSH categories are being aligned, which is not ideal for paper classification.

Our second extension proved much more successful considering that an MLP-based classification yielded SOTA results for both MeSH and MAG tasks.

REFERENCES

[1] Iz Beltagy, Arman Cohan, and Kyle Lo. "SciBERT: Pretrained Contextualized Embeddings for Scientific Text". In: *CoRR* abs/1903.10676 (2019).

[2] Lukas Biewald. *Experiment Tracking with Weights and Biases*. Software available from wandb.com. 2020. URL: https://www.wandb.com/.

[3] Colin B. Clement et al. *On the Use of ArXiv as a Dataset*. 2019.

[4] Arman Cohan et al. “SPECTER: Document-level Representation Learning using Citation-informed Transformers”. In: (2020).

[5] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2018.

[6] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *CoRR* abs/1810.04805 (2018).

[7] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014.

[8] Carolyn E. Lipscomb. “Medical Subject Headings (MeSH).” In: *Bulletin of the Medical Library Association* 88 3 (2000), pp. 265–6.

[9] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[10] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[11] Arnab Sinha et al. “An Overview of Microsoft Academic Service (MAS) and Applications.” In: (2015), pp. 243–246.

[12] Ashish Vaswani et al. “Attention Is All You Need”. In: (2017).