Abstract

Learning scientific document representations can be substantially improved through contrastive learning objectives, where the challenge lies in creating positive and negative training samples that encode the desired similarity semantics. Prior work relies on discrete citation relations to generate contrast samples. However, discrete citations enforce a hard cutoff to similarity. This is counter-intuitive to similarity-based learning, and ignores that scientific papers can be very similar despite lacking a direct citation – a core problem of finding related research. Instead, we use controlled nearest neighbor sampling over citation graph embeddings for contrastive learning. This control allows us to learn continuous similarity, to sample hard-to-learn negatives and positives, and also to avoid collisions between negative and positive samples by controlling the sampling margin between them. The resulting method SciNCL outperforms the state-of-the-art on the SciDocs benchmark. Furthermore, we demonstrate that it can train (or tune) language models sample-efficiently, and that it can be combined with recent training-efficient methods. Perhaps surprisingly, even training a general-domain language model this way outperforms baselines pretrained in-domain.

1 Introduction

Large pretrained language models (LLMs) achieve state-of-the-art results through fine-tuning on many NLP tasks (Rogers et al., 2020). However, the sentence or document embeddings derived from LLMs are of lesser quality compared to simple baselines like GloVe (Reimers and Gurevych, 2019), as their embedding space suffers from being anisotropic, i.e. poorly defined in some areas (Li et al., 2020).

One approach that has recently gained attention is the combination of LLMs with contrastive fine-tuning to improve the semantic textual similarity between document representations (Wu et al., 2020; Gao et al., 2021). These contrastive methods learn to distinguish between pairs of similar and dissimilar texts. As part of metric learning, they traditionally focused on defining new loss functions, while Musgrave et al. (2020) showed that newer metric losses lead to insignificant performance gains when compared fairly. Instead, recent works on self and supervised contrastive learning has started to focus on developing techniques that generate better positive and negative data augmentations for efficient contrastive learning (Tian et al., 2020; Rethmeier and Augenstein, 2021a,b; Shorten et al., 2021).

In this paper, we focus on learning scientific document representations (SDRs). The core distinguishing feature of this domain is the presence of citation information that complement the textual information. SDR methods like SciBERT (Beltagy et al., 2019) pretrain a Transformer on domain-specific text but neglect citations. The current state-of-the-art by Cohan et al. (2020) uses discrete cita-
tion information to generate positive and negative samples for contrastive fine-tuning of SciBERT via a triplet loss (Schroff et al., 2015). Papers cited by the query are positive samples, while non-cited papers are negative samples.

This discrete cut-off to similarity is counter-intuitive to (continuous) similarity-based learning. It encourages overfitting to human similarity annotations, i.e., citations, which may reflect politeness and policy rather than semantic similarity (Paster-nack, 1969). Considering only one citation direction may also cause positive and negative samples to collide, since a paper pair could be treated as a positive and negative instance at the same time. Saunshi et al. (2019) have shown that generating more negatives harms contrastive optimization performance if collision are not avoided. Instead, the generation of non-colliding contrastive samples should be based on a continuous similarity function that allows us to find semantically similar papers, even without direct citations.

**Contributions:**

- We propose neighborhood contrastive learning for scientific document representations with citation graph embeddings (SciNCL).
- We sample similar and dissimilar papers from neighboring citation graph embeddings, such that both are hard to learn to avoid long training times and gradient collapse.
- As in recent contrastive learning works, we address sample generation semantics based on contrastive learning theory insights rather than designing new loss functions.
- We compare against the state-of-the-art approach SPECTER (Cohan et al., 2020) and other strong methods on the SciDOCS benchmark and find that SciNCL outperforms SPECTER on average and on 9 of 12 metrics.
- Finally, we demonstrate that with SciNCL, using only 1% of the triplets for training, starting with a general-domain language model, or training only the bias terms of the model is sufficient to outperform the baselines.
- Our code and models are publicly available.\(^1\)

2 Related Work

**Contrastive Learning** pulls representations of similar data points (positives) closer together, while representations of dissimilar documents (negatives) are pushed apart. A common contrastive objective is the triplet loss (Schroff et al., 2015) that Cohan et al. (2020) used for scientific document representation learning, as we describe below. However, as Musgrave et al. (2020) point out, contrastive objectives work best when specific requirements are respected. (Req. 1) Views of the same data should introduce new information, i.e. the mutual information between views should be minimized (Tian et al., 2020). We use citation graph embeddings to generate contrast label information that supplements text-based similarity. (Req. 2) For training time and sample efficiency, negative samples should be hard to classify, but should also not collide with positives (Saunshi et al., 2019).

(Req. 3) Recent works like Musgrave et al. (2020); Khosla et al. (2020) use multiple positives. However, positives need to be consistently close to each other (Wang and Isola, 2020), since positives and negatives may otherwise collide, e.g., Cohan et al. (2020) consider only ‘citations by the query’ as similarity signal and not ‘citations to the query’. Such unidirectional similarity does not guarantee that a negative paper (not cited by the query) may cite the query paper and thus could cause collisions, the more we sample. Our method treats both citing and being cited as positives (Req. 2), while it also generates hard negatives and hard positives (Req. 2+3). Hard negatives are close to but do not overlap positives (red band in Fig. 1). Hard positives are close, but not trivially close to the query document (green band in Fig. 1). Related work on triplet mining is discussed in Appendix A.1.

**Scientific Document Representations** based on Transformers (Vaswani et al., 2017) and pretrained on domain-specific text dominate today’s scientific document processing. There are SciBERT (Beltagy et al., 2019), BioBERT (Lee et al., 2019) and SciGPT2 (Luu et al., 2021), to name a few. Recent works modify these domain LLMs to support cite-worthiness detection (Wright and Augenstein, 2021) or fact checking (Wadden et al., 2020).

Aside from text, citations are a valuable signal for the similarity of research papers. Paper (node) representations can be learned using the citation graph (Wu et al., 2019; Perozzi et al., 2014; Grover and Leskovec, 2016). Especially for recommendations of papers or citations, hybrid combinations of text and citation features are often employed (Han et al., 2018; Jeong et al., 2020; Brochier et al.,

\(^1\) [https://anonymous.4open.science/r/scincl-1553/](https://anonymous.4open.science/r/scincl-1553/)
Closest to SciNCL are Citeomatic (Bhagavatula et al., 2018) and SPECTER (Cohan et al., 2020). While Citeomatic relies on bag-of-words for its textual features, SPECTER is based on SciBERT. Both leverage citations to learn a triplet-based document embedding model, whereby positive samples are papers cited in the query. Easy negatives are random papers not cited by the query. Hard negatives are citations of citations—papers referenced in positive citations of the query, but are not cited directly by it. Citeomatic also uses a second type of hard negatives, which are the nearest neighbors of a query that are not cited by it.

Unlike our approach, Citeomatic does not use the neighborhood of citation embeddings, but instead relies on the actual document embeddings from the previous epoch. Despite being related to SciNCL, the sampling approaches employed in Citeomatic and SPECTER do not account for the pitfalls of using discrete citations as a signal for paper similarity. Our work addresses this issue.

## 3 Methodology

Our goal is to learn citation-informed representations for scientific documents. To do so we sample three document representation vectors and learn their similarity. For a given query paper vector \( d^Q \), we sample a positive (similar) paper vector \( d^+ \) and a negative (dissimilar) paper vector \( d^- \). This produces a ‘query, positive, negative’ triplet \( (d^Q, d^+, d^-) \) — represented by (\( \bullet \), \( \circ \), \( \times \)) in Fig. 1. To learn paper similarity, we need to define three components: (§3.1) how to calculate document vectors \( d \) for the loss over triplets \( \mathcal{L} \); (§3.2) how citations provide similarity between papers; and (§3.3) how negative and positive papers \( (d^-, d^+) \) are sampled as (dis-)similar documents from the neighborhood of a query paper \( d^Q \).

### 3.1 Contrastive Learning Objective

Given the textual content of a document \( d \) (paper), the goal is to derive a dense vector representation \( d \) that best encodes the document information and can be used in downstream tasks. A Transformer language model \( f \) (SciBERT; Beltagy et al. (2019)) encodes documents \( d \) into vector representations \( f(d) = d \). The input to the language model is the title and abstract separated by the \([\text{SEP}]\) token.\(^2\)

The final layer hidden state of the \([\text{CLS}]\) token is then used as a document representation \( f(d) = d \).

Training with a masked language modeling objectives alone has been shown to produce sub-optimal document representations (Li et al., 2020; Gao et al., 2021). Thus, similar to the SDR state-of-the-art method SPECTER (Cohan et al., 2020), we continue training the SciBERT model (Beltagy et al., 2019) using a self-supervised triplet margin loss (Schroff et al., 2015):

\[
\mathcal{L} = \max \left\{ \| d^Q - d^+ \|_2 - \| d^Q - d^- \|_2 + \xi, 0 \right\}
\]

Here, \( \xi \) is a slack term (\( \xi = 1 \) as in SPECTER) and \( \| d \|_2 \) is the \( L^2 \) norm, used as a distance function. However, the SPECTER sampling method has significant drawbacks. We will describe these issues and our contrastive learning theory guided improvements in detail below in §3.2.

### 3.2 Citation Neighborhood Sampling

Compared to the textual content of a paper, citations provide an outside view on a paper and its relation to the scientific literature (Elkiss et al., 2008), which is why citations are traditionally used as a similarity measure in library science (Kessler, 1963; Small, 1973). However, using citations as a discrete similarity signal, as done in Cohan et al. (2020), has its pitfalls. Their method defines papers cited by the query as positives, while paper citing the query could be treated as negatives. This means that positive and negative learning information collides between citation directions, which Saunshi et al. (2019) have shown to deteriorate performance. Furthermore, a cited paper can have a low similarity with the citing paper given the many motivations a citation can have (Teufel et al., 2006).

Likewise, a similar paper might not be cited.

To overcome these limitations, we learn citation embeddings first and then use the citation neighborhood around a given query paper \( d^Q \) to construct similar (positive) and dissimilar (negative) samples for contrast by using the \( k \) nearest neighbors. This builds on the intuition that nodes connected by edges should be close to each other in the embedding space (Perozzi et al., 2014). Using citation embeddings allows us to: (1) sample paper similarity on a continuous scale, which makes it possible to: (2) define hard to learn positives, as well as (3) hard or easy to learn negatives. Points (2-3) are important in making contrastive learning efficient as will describe below in §3.3.

\(^2\)Cohan et al. (2019) evaluated other inputs (venue or author) but found the title and abstract to perform best.
3.3 Positives and Negatives Sampling

**Positive samples:** $d^+$ should be semantically similar to the query paper $d^Q$, i.e. sampled close to the query embedding $d^Q$. Additionally, as Wang and Isola (2020) find, positives should be sampled from comparable locations (distances from the query) in embedding space and be dissimilar enough from the query embedding, to avoid gradient collapse (zero gradients). Therefore, we sample $c^+$ positive (similar) papers from a close neighborhood around query embedding $d^Q (k^+ - c^+, k^+)$, i.e. the green band in Fig. 1. When sampling with KNN search, we use a small $k^+$ to find positives and later analyze the impact of $k^+$ in Fig. 2.

**Negative samples:** can be divided into easy ☐ and hard ☐ negative samples (light and dark red in Fig. 1). Sampling more hard negatives is known to improve contrastive learning (Bucher et al., 2016; Wu et al., 2017). However, we make sure to sample hard negatives (red band in Fig. 1) such that they are close to potential positives but do not collide with positives (green band), by using a tunable ‘sampling induced margin’ We do so, since Saunshi et al. (2019) showed that sampling a larger number of hard negatives only improves performance if the negatives do not collide with positive samples, since collisions make the learning signal noisy. That is, in the margin between hard negatives and positives we expect positives and negatives to collide, thus we avoid sampling from this region. To generate a diverse self-supervised citation similarity signal for contrastive SDR learning, we also sample easy negatives that are farther from the query than hard negatives. For negatives, the $k^-$ should be large when sampling via KNN to ensure samples are dissimilar from the query paper.

3.4 Sampling Strategies

As described in §3.2 and §3.3, our approach improves upon the method by Cohan et al. (2020). Therefore, we reuse their sampling parameters (5 triplets per query paper) and then further optimize our methods’ hyperparameters. For example, to train the triplet loss, we generate the same amount of $(d^Q, d^+, d^-)$ triplets per query paper as SPECTER (Cohan et al., 2020). To be precise, this means we generate $c^+ = 5$ positives (as explained in §3.3). We also generate 5 negatives, three easy negatives $c^-_{easy} = 3$ and two hard negatives $c^-_{hard} = 2$, as described in §3.3.

Below, we describe three strategies (I-III) for sampling triplets. These either sample neighboring papers from citation embeddings (I), by random sampling (II), or using both strategies (III). For each strategy, let $c^+$ be the number of samples for either positives $c^+$, easy negatives $c^-_{easy}$, or hard negatives $c^-_{hard}$.

**Citation Graph Embeddings:** We train a graph embedding model $f_c$ on citations extracted from the Semantic Scholar Open Research Corpus (S2ORC; Lo et al., 2020) to get citation embeddings $C$. We utilize PyTorch BigGraph (Lerer et al., 2019), which allows for training on large graphs with modest hardware requirements. The resulting graph embeddings perform well using the default training settings from Lerer et al. (2019), but given more computational resources, careful tuning may produce even better-performing embeddings. Nonetheless, we conducted a narrow parameter search based on link prediction – see Appendix A.5.

(I) **K-nearest neighbors (KNN):** Assuming a given citation embedding model $f_c$ and a search index (e.g., FAISS §4.3), we run $KNN(f_c(d^Q), C)$ and take $c^+$ samples from a range of the $(k - c^+, k)$ nearest neighbors around the query paper $d^Q$ with its neighbors $N = \{n_1, n_2, n_3, \ldots \}$, whereby neighbor $n_i$ is the $i$-th nearest neighbor in the citation embedding space. For instance, for $c^- = 3$ and $k = 10$ the corresponding samples would be the three neighbors descending from the tenth neighbor: $n_8$, $n_9$, and $n_{10}$. To reduce computing effort, we sample the neighbors $N$ only once via $\{0; \text{max}(k^+ + k^-_{hard})\}$, and then generate triplets by range-selection in $N$; i.e. positives = $(k^+ - c^+; k^+)$, and hard negatives = $(k^-_{hard} - c_{hard}; k^-_{hard})$.

(II) **Random sampling:** Sample any $c^+$ papers without replacement from the corpus.

(III) **Filtered random:** Like (II) but excluding the papers that are retrieved by KNN, i.e., all neighbors within the largest $k$ are excluded.

The KNN sampling strategy introduces the hyperparameter $k$ that allow for the controlled sampling of positives or negatives with different difficulty (from easy to hard depending on the hyperparameter). Specifically, in Fig. 1 the hyperparameter $k$ define the tunable sample induced margin between positives and negatives, as well as the width and position of the positive sample band (green) and negative sample band (red) around the query.
sample. Besides the strategies above, we experiment with similarity threshold, k-means clustering and sorted random sampling, neither of which performs well (see negative results in Appendix A.7).

4 Experiments

In the following, we introduce our experiments including the data sets and implementation details.

4.1 Evaluation Dataset

We evaluate on the SciDOCS benchmark (Cohan et al., 2020). A key difference to other benchmarks is that embeddings are the input to the individual tasks without explicit fine-tuning. The SciDOCS benchmark consists of the following four tasks:

- **Document classification** (CLS) with Medical Subject Headings (MeSH) (Lipscomb, 2000) and Microsoft Academic Graph labels (MAG) (Sinha et al., 2015).
- **Co-views and co-reads** (USR) prediction based on the L2 distance between embeddings.
- **Direct and co-citation** (CITE) prediction based on the L2 distance between the embeddings.
- **Recommendations** (REC) generation based on embeddings and paper metadata.

4.2 Training Datasets

The experiments mainly compare SciNCL against SPECTER on the SciDOCS benchmark. However, we found 40.5% of SciDOCS’s papers leaking into SPECTER’s training data (the leakage affects only the unsupervised paper data but not the gold labels – see Appendix A.3). To be transparent about this leakage, we train SciNCL on two datasets:

- **SPECTER replication (w/ leakage)**: We replicate SPECTER’s training data including its leakage. Unfortunately, SPECTER provides neither citation data nor a mapping to S2ORC, which our citation embeddings are based on. We successfully map 96.2% of SPECTER’s query papers and 83.3% of the corpus from which positives and negatives are sampled to S2ORC. To account for the missing papers, we randomly sample papers from S2ORC (without the SciDOCS papers) such that the absolute number of papers is identical with SPECTER.

- **S2ORC subset (w/o leakage)**: We select a random subset from S2ORC that does not contain any of the mapped SciDOCS papers. This avoids SPECTER’s leakage, but also makes the scores reported in Cohan et al. (2020) less comparable. We successfully map 98.6% of the SciDOCS papers to S2ORC. Thus, only the remaining 1.4% of the SciDOCS papers could leak into this training set.

The details of the dataset creation are described in Appendix A.2 and A.4. Both training sets yield 684K triplets (same count as SPECTER). Also, the ratio of training triplets per query remains the same (§3.4). Our citation embedding model is trained on the S2ORC citation graph. In w/ leakage, we include all SPECTER papers even if they are part of SciDOCS, the remaining SciDOCS papers are excluded (52.5 nodes and 463M edges). In w/o leakage, all mapped SciDOCS papers are excluded (52.4M nodes and 447M edges) such that we avoid leakage also for the citation embedding model.

4.3 Model Training and Implementation

We replicate the training setup from SPECTER as closely as possible. We implement SciNCL using Huggingface Transformers (Wolf et al., 2020), initialize the model with SciBERT’s weights (Beltagy et al., 2019), and train via the triplet loss (Equation 3.1). The optimizer is Adam with weight decay (Kingma and Ba, 2015; Loshchilov and Hutter, 2019) and learning rate $\lambda=2^{-5}$. To explore the effect of computing efficient fine-tuning we also train a BitFit model (Zaken et al., 2021) with $\lambda=1^{-4}$ (§7.2). We train SciNCL on two NVIDIA GeForce RTX 6000 (24G) for 2 epochs (approx. 24 hours of training time) with batch size 8 and gradient accumulation for an effective batch size of 32 (same as SPECTER). The graph embedding training is performed on an Intel Xeon Gold 6230 CPU with 60 cores and takes approx. 6 hours. The KNN strategy is implemented with FAISS (Johnson et al., 2021) using a flat index (exhaustive search) and takes less than 30min for indexing and retrieval of the triplets.

4.4 Baseline Methods

We compare against the following baselines (details in Appendix A.6): USE (Cer et al., 2018), BERT (Devlin et al., 2019), BioBERT (Lee et al., 2019), SciBERT (Beltagy et al., 2019), CiteBERT (Wright and Augenstein, 2021), DeCLUTR (Giorgi et al., 2021), the graph-convolution approach SGC (Wu et al., 2019), Citeomatic (Bhagavatula et al., 2018), and SPECTER (Cohan et al., 2020).

Also, we compare against Oracle SciDOCS which is identical to SciNCL except that its triplets are generated based on SciDOCS’s validation and test set using their gold labels. For example, papers with the same MAG labels are positives and papers with different labels are negatives. In total, this
Tab. 1 shows the results, comparing SciNCL with CI Table 1: Results on the SciDocs Oracle SciDocs et al. (2020). S2ORC, SciNCL outperforms SPECTER in terms of avg. score with 1.7 points. The scores with * are from Cohan σ reported as mean and standard deviation previous best avg. score by 1.8 points and also outperforms the baselines in 9 of 12 task metrics. Our scores are dominated sampling. As random sampling accounts for a large fraction of the triplets (in the form of easy negatives), we report the mean scores and standard deviation based on ten random seeds (seed ∈ [0, 9]).

For MAG classification, SPECTER achieves the best result with 82.0 F1 followed by SciNCL with 81.4 F1 (-0.6 points). For MeSH classification, SciNCL yields the highest score with 88.7 F1 (+2.3 compared to SPECTER). Both classification tasks have in common that the chosen training settings lead to over-fitting. Changing the training by using only 1% training data, SciNCL yields 82.2 F1@MAG (Tab. 2). In all user activity and citation tasks, SciNCL yields higher scores than all baselines. Moreover, SciNCL outperforms SGC on direct citation prediction, where SGC outperforms SPECTER in terms of ndCG. On the recommender task, SPECTER yields the best P@1 with 20.0, whereas SciNCL achieves 19.3 P@1 (in terms of ndCG SciNCL and SPECTER are on par).

When training SPECTER and SciNCL without leakage, SciNCL outperforms SPECTER even in 11 of 12 metrics and is on par in the other metric. This suggests that SciNCL’s hyperparameters have a low corpus dependency since they were only optimized on the corpus with leakage.

<table>
<thead>
<tr>
<th>Task →</th>
<th>Classification</th>
<th>User activity prediction</th>
<th>Citation prediction</th>
<th>Recomm. Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model / Metric →</td>
<td>MAG</td>
<td>MeSH</td>
<td>Co-View</td>
<td>Co-Read</td>
</tr>
<tr>
<td>Oracle SciDocs †</td>
<td>87.1</td>
<td>94.8</td>
<td>87.2</td>
<td>93.5</td>
</tr>
<tr>
<td>USE (2018)</td>
<td>80.0</td>
<td>83.9</td>
<td>77.2</td>
<td>88.1</td>
</tr>
<tr>
<td>Citeomatic* (2018)</td>
<td>67.1</td>
<td>75.7</td>
<td>81.1</td>
<td>90.2</td>
</tr>
<tr>
<td>SGC* (2019)</td>
<td>78.6</td>
<td>72.7</td>
<td>87.2</td>
<td>88.0</td>
</tr>
<tr>
<td>BERT (2019)</td>
<td>79.9</td>
<td>74.3</td>
<td>59.9</td>
<td>78.3</td>
</tr>
<tr>
<td>SciBERT* (2019)</td>
<td>79.7</td>
<td>70.7</td>
<td>50.7</td>
<td>73.1</td>
</tr>
<tr>
<td>BioBERT (2019)</td>
<td>77.7</td>
<td>73.0</td>
<td>53.3</td>
<td>74.0</td>
</tr>
<tr>
<td>CiteBERT (2021)</td>
<td>78.8</td>
<td>74.8</td>
<td>53.2</td>
<td>73.6</td>
</tr>
<tr>
<td>DeCLUTR (2021)</td>
<td>81.2</td>
<td>88.0</td>
<td>63.4</td>
<td>80.6</td>
</tr>
<tr>
<td>SPECTER* (2020)</td>
<td>82.0</td>
<td>86.4</td>
<td>83.6</td>
<td>91.5</td>
</tr>
</tbody>
</table>

Replicated SPECTER training data (w/ leakage):

SciNCL (ours) | 81.4 | 82.7 | 85.3 | 92.3 | 87.5 | 93.9 | 93.6 | 97.3 | 91.6 | 96.4 | 53.9 | 19.3 | 81.8 |

Random S2ORC training data (w/o leakage):

SciNCL (ours) | 81.3 | 84.4 | 83.1 | 91.3 | 84.0 | 92.1 | 86.2 | 93.9 | 87.8 | 94.7 | 52.2 | 17.5 | 79.4 |

Table 1: Results on the SciDocs benchmark. With replicated SPECTER training data, SciNCL surpasses the previous best avg. score by 1.8 points and also outperforms the baselines in 9 of 12 task metrics. Our scores are reported as mean and standard deviation σ over ten random seeds. With training data randomly sampled from S2ORC, SciNCL outperforms SPECTER in terms of avg. score with 1.7 points. The scores with * are from Cohan et al. (2020). Oracle SciDocs † is the upper bound of the performance with triplets from SciDoc’s data.

5 Overall Results

Tab. 1 shows the results, comparing SciNCL with the best validation performance against the baselines. With replicated SPECTER training data (w/ leakage), SciNCL achieves an average performance of 81.8 across all metrics, which is a 1.8 point absolute improvement over SPECTER (the next-best baseline). When trained without leakage, the improvement of SciNCL over SPECTER is consistent with 1.7 points but generally lower (79.4 avg. score). In the following, we refer to the results obtained through training on the replicated SPECTER data (w/ leakage) if not otherwise mentioned.

We find the best validation performance based on SPECTER’s data when positives and hard negatives are sampled with KNN, whereby positives are \(k^+ = 25\), and hard negatives are \(k^-_{hard} = 1000\) (§6). Easy negatives are generated through filtered random sampling. As random sampling accounts for a large fraction of the triplets (in the form of easy negatives), we report the mean scores and standard deviation based on ten random seeds (seed ∈ [0, 9]).
Regarding the LLM baselines, we observe that the general-domain BERT, with a score of 63.4, outperforms the domain-specific BERT variants, namely SciBERT (59.6) and BioBERT (58.8). LLMs without citations or contrastive objectives yield generally poor results. This emphasizes the anisotropy problem of embeddings directly extracted from current LLMs and highlights the advantage of combining text and citation information.

In summary, we show that SciNCL’s triplet selection leads on average to a performance improvement on SciDOCS, with most gains being observed for user activity and citation tasks. The gain from 80.0 to 81.8 is particularly notable given that even Oracle SciDocs yields with 83.0 an only marginally higher avg. score despite using test and validation data from SciDOCS for the triplet selection.

6 Impact of Sample Difficulty

The benefit of SciNCL is that the sampling strategies can be tuned (§3.3). In this section, we present the results of this tuning procedure. We optimize the sampling strategies for positives and hard or easy negatives with partial grid search on a random sample of 10% of the replicated SPECTER training data (sampling based on queries). Our experiments show that optimizations on this subset correlate with the entire dataset. The scores in Fig. 2 and 3 are reported as the mean over three random seeds including standard deviations.

6.1 Positive Samples

Fig. 2 shows the avg. scores on the SciDOCS validation set depending on the selection of positives with the kNN strategy. We only change $k^+$, while negative sampling remains fixed to its best setting (§6.2). The performance is relatively stable for $k^+<100$ with peak at $k^+=25$, for $k^+>100$ the performance declines as $k^+$ increases. Wang and Isola (2020) state that positive samples should be semantically similar to each other, but not too similar to the query. For example, at $k^+=5$, positives may be a bit “too easy” to learn, such that they produce less informative gradients than the optimal setting $k^+=25$. Similarly, making $k^+$ too large leads to the sampling induced margin being too small, such that positives collide with negative samples, which creates contrastive label noise that degrades performance (Saunshi et al., 2019).

6.2 Hard Negative Samples

Fig. 3 presents the validation results with kNN using 10% training data. This plateau can also be observed in the test performance, where $k_{\text{hard}}=2000$ and $k_{\text{hard}}=3000$ yield a marginally lower score of 81.6 and 81.7 (Tab. 2).

In summary, we show that SciNCL’s triplet selection leads on average to a performance improvement on SciDOCS, with most gains being observed for user activity and citation tasks. The gain from 80.0 to 81.8 is particularly notable given that even Oracle SciDocs yields with 83.0 an only marginally higher avg. score despite using test and validation data from SciDOCS for the triplet selection.

6.3 Easy Negative Samples

Fig. 3 shows the avg. scores on the SciDOCS validation set depending on the selection of positives with the kNN strategy. We only change $k^+$, while negative sampling remains fixed to its best setting (§6.2). The performance is relatively stable for $k^+<100$ with peak at $k^+=25$, for $k^+>100$ the performance declines as $k^+$ increases. Wang and Isola (2020) state that positive samples should be semantically similar to each other, but not too similar to the query. For example, at $k^+=5$, positives may be a bit “too easy” to learn, such that they produce less informative gradients than the optimal setting $k^+=25$. Similarly, making $k^+$ too large leads to the sampling induced margin being too small, such that positives collide with negative samples, which creates contrastive label noise that degrades performance (Saunshi et al., 2019).

6.3 Easy Negative Samples

Fig. 3 presents the validation results with kNN using 10% training data. This plateau can also be observed in the test performance, where $k_{\text{hard}}=2000$ and $k_{\text{hard}}=3000$ yield a marginally lower score of 81.6 and 81.7 (Tab. 2).

For $k_{\text{hard}}>4000$, the performance starts to decline again. This suggests that for large $k_{\text{hard}}$ the samples are not “hard enough”. The need for hard negatives confirms the findings of Cohan et al. (2020).

6.3 Easy Negative Samples

Filtered random sampling of easy negatives yields the best validation performance compared pure random sampling (Tab. 2). However, the performance difference is marginal. When rounded to one decimal, their average test scores are identical. The marginal difference is caused by the large corpus size and the resulting small probability of randomly
The last three rows of Tab. 2 show the results regarding data and computing efficiency. When keeping the citation graph unchanged but training the language model with only 10% of the original triplets, SciNCL still yields a score of 81.1 (-0.6). Even with only 1% (6840 triplets), SciNCL achieves a score of 80.8 that is 1.0 points less than with 100% but still 0.8 points more than the SPECTER baseline. With this textual sample efficiency, one could manually create triplets or use existing supervised datasets as in Gao et al. (2021).

Lastly, we evaluate BitFit training (Zaken et al., 2021), which only trains the bias terms of the model while freezing all other parameters. This corresponds to training only 0.1% of the original parameters. With BitFit, SciNCL yields a considerable score of 81.2 (-0.5 points). As a result, SciNCL could be trained on the same hardware with even larger (general-domain) language models (§7.1).

### 8 Conclusion

We present a novel approach for contrastive learning of scientific document embeddings that addresses the challenge of selecting informative positive and negative samples. By leveraging citation graph embeddings for sample generation, SciNCL achieves a score of 81.8 on the SciDOCS benchmark, a 1.8 point improvement over the previous best method SPECTER. This is purely achieved by introducing tunable sample difficulty and avoiding collisions between positive and negative samples, while existing LLM and data setups can be reused. This improvement over SPECTER can be observed even when excluding the SciDOCS papers during training (see w/o leakage in Tab. 1). Furthermore, SciNCL’s improvement from 80.0 to 81.8 is particularly notable given that even oracle triplets, which are generated with SciDOCS’s test and validation data, yield with 83.0 only a marginally higher score.

Our work highlights the importance of sample generation in a contrastive learning setting. We show that language model training with 1% of triplets is already sufficient to outperform SPECTER, whereas the remaining 99% provide only 1.0 additional points (80.8 to 81.8). This sample efficiency is achieved by adding reasonable effort for sample generation, i.e., graph embedding training and kNN search. We also demonstrate that in-domain LLM pretraining (like SciBERT) is beneficial, while general-domain LLMs can achieve a comparable performance and even outperform SPECTER. This indicates that controlling sample difficulty and avoiding collisions is more effective than in-domain pretraining, especially in scenarios where training a LLM from scratch is infeasible.
References


Andreas Nugaard Holm, Barbara Plank, Dustin Wright, and Isabelle Augenstein. 2022. Longitudinal Citation Prediction using Temporal Graph Neural Networks. In AAAI 2022 Workshop on Scientific Document Understanding (SDU 2022).


A Appendix

A.1 Extended Related Work

Triplet Mining remains a challenge in NLP due to the discrete nature of language which makes data augmentation less trivial as compared to computer vision (Gao et al., 2021). Examples for augmentation strategies are: translation (Fang et al., 2020), or word deletion and reordering (Wu et al., 2020). Positives and negatives can be sampled based on the sentence position within a document (Giorgi et al., 2021). Gao et al. (2021) utilize supervised entailment datasets for the triplet generation.

Language- and text-independent approaches are also applied: Kim et al. (2021) use intermediate BERT hidden state for positive sampling and Wu et al. (2021) add noise to representations to obtain negative samples. Xiong et al. (2020) present an approach similar to SciNCL where they sample hard negatives from the k nearest neighbors in the embedding space derived from the previous model checkpoint. While Xiong et al. rely only on textual data, SciNCL integrates also citation information which are especially valuable in the scientific context as Cohan et al. (2020) have shown.

A.2 Mapping to S2ORC

Table 3: Mapping to S2ORC citation graph

<table>
<thead>
<tr>
<th>S2ORC mapping</th>
<th>Success rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>SciDocs papers</td>
<td></td>
</tr>
<tr>
<td>- with S2ORC IDs</td>
<td>220,815 / 223,932 (98.6%)</td>
</tr>
<tr>
<td>- in S2ORC graph</td>
<td>197,811 / 223,932 (88.3%)</td>
</tr>
<tr>
<td>SPECTER papers</td>
<td></td>
</tr>
<tr>
<td>- with S2ORC IDs</td>
<td>311,094 / 311,860 (99.7%)</td>
</tr>
<tr>
<td>- in S2ORC graph</td>
<td>260,014 / 311,860 (83.3%)</td>
</tr>
</tbody>
</table>

Neither the SPECTER training data nor the SciDocs test data comes with a mapping to the S2ORC dataset, which we use for the training of the citation embedding model. However, to replicate SPECTER’s training data and to avoid leakage of SciDocs test data such a mapping is needed. Therefore, we try to map the papers to S2ORC based on PDF hashes and exact title matches. The remaining paper metadata is collected through the Semantic Scholar API. Tab. 3 summarizes the outcome of mapping procedure. Failed mappings can be attributed to papers being unavailable through the Semantic Scholar API (e.g., retracted papers) or papers not being part of S2ORC citation graph.

A.3 SPECTER-SciDocs Leakage

When replicating SPECTER (Cohan et al., 2020), we found a substantial overlap between the papers used during the model training and the papers from their SciDocs benchmark. In both datasets, papers are associated with Semantic Scholar IDs. Thus, no custom ID mapping as in App. A.2 is required to identify papers that leak from training to test data. From the 311,860 unique papers used in SPECTER’s training data, we find 79,201 papers (25.4%) in the test set of SciDocs and 79,609 papers (25.5%) in its validation set. When combining test and validation set, there is a total overlap of 126,176 papers (40.5%). However, this overlap affects only the ‘unsupervised’ paper metadata (title, abstract, citations, etc.) and not the gold labels used in SciDocs (e.g., MAG labels or clicked recommendations).

A.4 Dataset Creation

As describe in §4.2, we conduct our experiments on two datasets. Both datasets rely on the citation graph of S2ORC (Lo et al., 2020). More specifically, S2ORC with the version identifier 20200705v1 is used. The full citation graph consists of 52.6M nodes (papers) and 467M edges (citations). Tab. 4 presents statistics on the datasets and their overlap with SPECTER and SciDocs. The steps to reproduce both datasets are:

Replicated SPECTER (w/ leakage) In order to replicate SPECTER’s training data and do not increase the leakage, we exclude all SciDocs papers which are not used by SPECTER from the S2ORC citation graph. This means that apart from the 110,538 SPECTER papers not a single other SciDocs paper is included. The resulting citation graph has 52.5M nodes and 463M edges and is used for training the citation graph embeddings.

For the SciNCL triplet selection, we also replicate SPECTER’s query papers and its corpus from which positive and negatives are sampled. Our mapping and the underlying citation graph allows us to use 227,869 of 248,007 SPECTER’s papers for training. Regarding query papers, we use 131,644 of 136,820 SPECTER’s query papers. To align the number training triplets with the one from SPECTER, additional papers are randomly sampled from the filtered citation graph.

1https://github.com/allenai/specter/issues/2
2https://github.com/allenai/scidocs
Random S2ORC subset (w/o leakage) To avoid leakage, we exclude all successfully mapped SciDOCS papers from the S2ORC citation graph. After filtering the graph has 52,3 nodes and 447M edges. The citation graph embedding model is trained on this graph.

Next, we reproduce triplet selection from SPECTER. Any random 136,820 query papers are selected from the filtered graph. For each query, we generate five positives (cited by the query), two hard negatives (citation of citation), and three random nodes from the filtered S2ORC citation graphs. This sampling produces 684,100 training triplets with 680,967 unique papers IDs (more compared to the replicated SPECTER dataset). Based on these triplets the SPECTER model for this dataset is trained with the same model settings and hyperparameters as SciNCL (second last row in Tab. 1).

Lastly, the SciNCL triplets are generated based on the citation graph embeddings of the same 680,967 unique papers IDs, i.e., the FAISS index contains only these papers and not the remaining S2ORC papers. Also, the same 136,820 query papers are used.

Table 4: Statistics for our two datasets and their overlap with SPECTER and SciDOCS respectively.

<table>
<thead>
<tr>
<th>Training triplets</th>
<th>Replicated SPECTER (w/ leakage)</th>
<th>Random S2ORC subset (w/o leakage)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unique paper IDs</td>
<td>248,007</td>
<td>680,967</td>
</tr>
<tr>
<td>- in SPECTER</td>
<td>227,869</td>
<td>9,182</td>
</tr>
<tr>
<td>- in SciDOCS</td>
<td>110,538</td>
<td>0</td>
</tr>
<tr>
<td>- in SciDOCS and in SPECTER</td>
<td>110,538</td>
<td>0</td>
</tr>
<tr>
<td>Query paper IDs</td>
<td>136,820</td>
<td>136,820</td>
</tr>
<tr>
<td>- in SciDOCS</td>
<td>69,306</td>
<td>0</td>
</tr>
<tr>
<td>- in SPECTER queries</td>
<td>131,644</td>
<td>463</td>
</tr>
<tr>
<td>Citation graph</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Nodes</td>
<td>52,526,134</td>
<td>52,373,977</td>
</tr>
<tr>
<td>- Edges</td>
<td>463,697,639</td>
<td>447,697,727</td>
</tr>
</tbody>
</table>

A.6 Baseline Details
If not otherwise mentioned, all BERT variations are used in their base-uncased versions.

The weights for BERT (bert-base-uncased), BioBERT (biobert-base-cased-v1.2), CiteBERT (citebert), DeCLUTR (declutr-sci-base) are taken from Huggingface Hub\(^5\). We use Universal Sentence Encoder (USE) from Tensorflow Hub\(^6\). For Oracle SciDOCS, we use the SciNCL implementation and under-sample the triplets from the classification tasks to ensure a balanced triplet distribution over the tasks. The SPECTER version for the random S2ORC training data (w/o leakage) is also trained with the SciNCL implementation. Please see Cohan et al. (2020) for additional baseline methods and their implementation details.

A.7 Negative Results
We investigated additional sampling strategies and model modification of which none led to a significant performance improvement.

Table 5: Link prediction performance of BigGraph embeddings trained on S2ORC citation graph with different dimensions and distance measures.

<table>
<thead>
<tr>
<th>Dim.</th>
<th>Dist.</th>
<th>MRR</th>
<th>Hits@1</th>
<th>Hits@10</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>Cos.</td>
<td>54.09</td>
<td>43.39</td>
<td>75.21</td>
<td>85.75</td>
</tr>
<tr>
<td>128</td>
<td>Dot</td>
<td>89.75</td>
<td>85.84</td>
<td>96.13</td>
<td>97.70</td>
</tr>
<tr>
<td>512</td>
<td>Dot</td>
<td>94.60</td>
<td>92.47</td>
<td>97.64</td>
<td>98.64</td>
</tr>
<tr>
<td>768</td>
<td>Dot</td>
<td>95.12</td>
<td>93.22</td>
<td>97.77</td>
<td>98.74</td>
</tr>
</tbody>
</table>

Tab. 5 shows the link prediction performance measured in MRR, Hits@1, Hits@10, and AUC. Dot product is substantially better than cosine similarity as distance measure. Also, there is a positive correlation between the performance and the size of the embeddings. The larger the embedding size the better link prediction performance. Graph embeddings with \(d=768\) were the largest possible size given our compute resources (available disk space was the limiting factor).

A.5 Graph Embedding Evaluation
To evaluate the underlying citation graph embeddings, we experiment with a few of BigGraph’s hyperparameters. We trained embeddings with different dimensions \(d\in\{128, 512, 768\}\) and different distance measures (cosine similarity and dot product) on 99% of the data and test the remaining 1% on the link prediction task. An evaluation of the graph embeddings with SciDOCS is not possible since we could not map the papers used in SciDOCS to the S2ORC corpus. All variations are trained for 20 epochs, margin \(m=0.15\), and learning rate \(\lambda=0.1\) (based on the recommended settings by Lerer et al. (2019)).
**Undirected Citations** Our graph embedding model considers citations as directed edges by default. We also train a SciNCL model with undirected citations by first converting a single edge \((a, b)\) into the two edges \((a, b)\) and \((b, a)\). This approach yields a slightly worse performance (81.7 avg. score; -0.1 points) and, therefore, was discarded for the final experiments.

**KNN with interval large than c** Our best results are achieved with kNN where the size of the neighbor interval \((k - c'; k]\) is equal to the number of samples \(c'\) that the strategy should generate. In addition to this, we also experimented with large intervals, e.g., \((1000; 2000]\), from which \(c'\) papers are randomly sampled. This approach yields comparable results but suffers from a larger effect of randomness and is therefore more difficult to optimize.

**K-Means Cluster for Easy Negatives** Easy negatives are supposed to be far away from the query. Random sampling from a large corpus ensures this as our results show. As an alternative approach, we tried k-means clustering whereby we selected easy negatives from the centroid that has a given distance to the query’s centroid. However, this decreased the performance.

**Sampling With Similarity Threshold** As an alternative to kNN, we select samples based on cosine similarity in the citation embedding space. Take \(c'\) papers that are within the similarity threshold \(t\) of a query paper \(d^Q\) such that \(s(f_c(d^Q), f_c(d_i)) < t\), where \(s\) is the cosine similarity function.

For example, given the similarity scores \(S = \{0.9, 0.8, 0.7, 0.1\}\) (ascending order, the higher the similarity is the closer the candidate embedding to the query embedding is) with \(c' = 2\) and \(t = 0.5\), the two candidates with the largest similarity scores and larger than the threshold would be 0.8 and 0.7. The corresponding papers would be selected as samples. While the positive threshold \(t^+\) should close to 1, the negative threshold \(t^-\) should be small to ensure samples are dissimilar from \(d^Q\). However, the empirical results suggest that this strategy is inferior compared to kNN.

**Hard Negatives With Similarity Threshold**

Selecting hard negatives based on the similarity threshold yields a test score of 81.7 (-0.1 points). Fig. 4 show the validation results for different similarity thresholds. A similar pattern as in Fig. 3 can be seen. When the negatives are closer to the query paper (larger similarity threshold \(t\)), the validation score decreases.

**Positives with Similarity Threshold** Positive sampling with SIM performs poorly since even for small \(t^+ < 0.5\) many query papers do not have any neighbors within this similarity threshold (more than 40%). Solving this issue would require changing the set of query papers which we omit for comparability to SPECTER.

**Sorted Random** Simple random sampling does not ensure if a sample is far or close to the query. To integrate a distance measure in the random sampling, we first sample \(n\) candidates, then order the candidates according to their distance to the query, and lastly select the \(c'\) candidates that are the closest or furthest to the query as samples.

**Mask Language Modeling** Giorgi et al. (2021) show that combining a contrastive loss with a mask language modeling loss can improve text representation learning. However, in our experiments a combined function decreases the performance on SciDOCS, probably due to the effects found by (Li et al., 2020).

**Graph Embedding Prediction Loss** We combine the triplet loss (Equation 3.1) with a MSE loss of the predicted embedding and the graph embeddings. This approach yields a comparable performance but adds additional computational complexity and was therefore discarded for the final experiments.

**A.8 Task-specific Results**

Fig. 5 and 6 present the validation performance like in §6 but on a task-level and not as an average over

---

**Figure 4**: Results on the validation set w.r.t. hard negative sampling with SIM using 10% training data.
Figure 5: Task-level validation performance w.r.t. $k^+$ with KNN strategy using 10% training data.

Figure 6: Task-level validation performance w.r.t. $k^-_{hard}$ with KNN strategy using 10% training data.
all tasks. The plots show that the optimal $k^+$ and $k^-_{\text{hard}}$ values are partially task dependent.

A.9 Examples
Tab. 6 lists three examples of query papers with their corresponding positive and negative samples. The complete set of triplets that we use during training are available in our code repository\(^1\).
Table 6: Example query papers with their positive and negative samples.

<table>
<thead>
<tr>
<th>Query: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positives:</td>
</tr>
<tr>
<td>• A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference</td>
</tr>
<tr>
<td>• Looking for ELMo’s Friends: Sentence-Level Pretraining Beyond Language Modeling</td>
</tr>
<tr>
<td>Positives:</td>
</tr>
<tr>
<td>• GLUE : A MultiTask Benchmark and Analysis Platform for Natural Language Understanding</td>
</tr>
<tr>
<td>• Dissecting Contextual Word Embeddings: Architecture and Representation</td>
</tr>
<tr>
<td>• Universal Transformers</td>
</tr>
<tr>
<td>• Planning for decentralized control of multiple robots under uncertainty</td>
</tr>
<tr>
<td>• Graph-Based Relational Data Visualization</td>
</tr>
<tr>
<td>Negatives:</td>
</tr>
<tr>
<td>• Linked Stream Data Processing</td>
</tr>
<tr>
<td>• Topic Modeling Using Distributed Word Embeddings</td>
</tr>
<tr>
<td>• Adversarially-Trained Normalized Noisy-Feature Auto-Encoder for Text Generation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query: BioBERT: a pre-trained biomedical language representation model for biomedical text mining</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positives:</td>
</tr>
<tr>
<td>• Exploring Word Embedding for Drug Name Recognition</td>
</tr>
<tr>
<td>• A neural joint model for entity and relation extraction from biomedical text</td>
</tr>
<tr>
<td>Positives:</td>
</tr>
<tr>
<td>• Event Detection with Hybrid Neural Architecture</td>
</tr>
<tr>
<td>• Improving chemical disease relation extraction with rich features and weakly labeled data</td>
</tr>
<tr>
<td>• GLUE : A MultiTask Benchmark and Analysis Platform for Natural Language Understanding</td>
</tr>
<tr>
<td>• Weakly Supervised Facial Attribute Manipulation via Deep Adversarial Network</td>
</tr>
<tr>
<td>• Applying the Clique Percolation Method to analyzing cross-market branch banking ...</td>
</tr>
<tr>
<td>Negatives:</td>
</tr>
<tr>
<td>• Perpetual environmentally powered sensor networks</td>
</tr>
<tr>
<td>• Labelling strategies for hierarchical multi-label classification techniques</td>
</tr>
<tr>
<td>• Domain Aware Neural Dialog System</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query: A Context-Aware Citation Recommendation Model with BERT and Graph Convolutional Networks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positives:</td>
</tr>
<tr>
<td>• Content-based citation analysis: The next generation of citation analysis</td>
</tr>
<tr>
<td>• ScisummNet: A Large Annotated Dataset and Content-Impact Models for Scientific Paper ...</td>
</tr>
<tr>
<td>Positives:</td>
</tr>
<tr>
<td>• Citation Block Determination Using Textual Coherence</td>
</tr>
<tr>
<td>• Discourse Segmentation Of Multi-Party Conversation</td>
</tr>
<tr>
<td>• Argumentative Zoning for Improved Citation Indexing</td>
</tr>
<tr>
<td>• Adaptive Quantization for Hashing: An Information-Based Approach to Learning ...</td>
</tr>
<tr>
<td>• Trap Design for Vibratory Bowl Feeders</td>
</tr>
<tr>
<td>Negatives:</td>
</tr>
<tr>
<td>• Software system for the Mars 2020 mission sampling and caching testbeds</td>
</tr>
<tr>
<td>• Applications of Rhetorical Structure Theory</td>
</tr>
<tr>
<td>• Text summarization for Malayalam documents — An experience</td>
</tr>
</tbody>
</table>