Carl von Ossietzky Universität Oldenburg
Faculty II - Department of Informatics
Digitalisierte Energiesysteme

Embedding of Scientific Publications with Machine Learning

Masterarbeit

submitted by

Melchior Schilewa

born on 12.02.1999 in Oldenburg

First Examiner:	Prof. Dr.-Ing. Astrid Nieße
Second Examiner:	M.Sc. Thomas Wolgast
Supervisor:	M.Sc. Thomas Wolgast

Oldenburg, September 3, 2024

Abstract
The number of scientific publications continues to increase, making new tools essential to assist researchers in efficiently working with large amounts of information. Embedding is an approach that can provide a basis to automatically solve various tasks with machine learning and artificial intelligence. It describes a numerical representation that captures the semantic meaning of text. This thesis explores how a novel approach to create document embeddings of scientific publications can be developed and evaluated. By leveraging a continuous co-citation signal, a document embedding model is trained that achieves similar performance to state-of-the-art baselines on downstream tasks. The developed approach uses much less training data thus drastically reduces training time. Moreover, the influence of multiple contrastive learning loss functions on the learned embeddings is investigated, where a varying impact on performance across different task formats is identified. The results present insights into the training of large language models and highlight the need to extend current evaluation approaches.

Contents

Abbreviations

AI	Artificial Intelligence
BOW	Bag-of-Words
CBOW	Continous Bag-of-Words
DRSM	Disease Research State Model
FoS	Fields of Study
k-NN	k-Nearest Neighbors
LLM	Large Language Model
MAG	Microsoft Academic Graph
MAP	Mean Average Precision
MeSH	Medical Subject Headings
ML	Machine Learning
MLM	Masked Language Model
MRR	Mean Reciporal Rank
MTEB	Massive Text Embedding Benchmark
nDCG	Normalized Discounted Cumulative Gain
NLP	Natural Language Processing
NSP	Next Sentence Prediction
OOV	Out-of-Vocabulary
PLM	Pre-trained Language Model
PMI	Pointwise Mutual Information

PV-DM	Paragraph Vector-Distributed Memory
SOTA	State of the Art
STS	Semantic Textual Similarity
TF-IDF	Term Frequency-Inverse Document Frequency

Nomenclature

d^+	Positive document
d^-	Negative document
m	Margin hyperparameter
q	Query document
s	Similarity measure
w	A word
η	Learning rate
θ	Embedding
τ	Temperature hyperparameter

1 Introduction

Researchers face the increasingly difficult challenge of keeping up to date with publications. Every year, the number of scientific publications increases [LI10], as can be seen in Figure 1.1 for the top ten mega-journals [Dom16]. Artificial Intelligence (AI) tools may help researchers to tackle the tasks of identifying and managing relevant literature [Mus+21]. Especially content-based approaches that can infer semantic information without human feedback are needed [Kin+23; Ost23]. This can be achieved with algorithms from the field

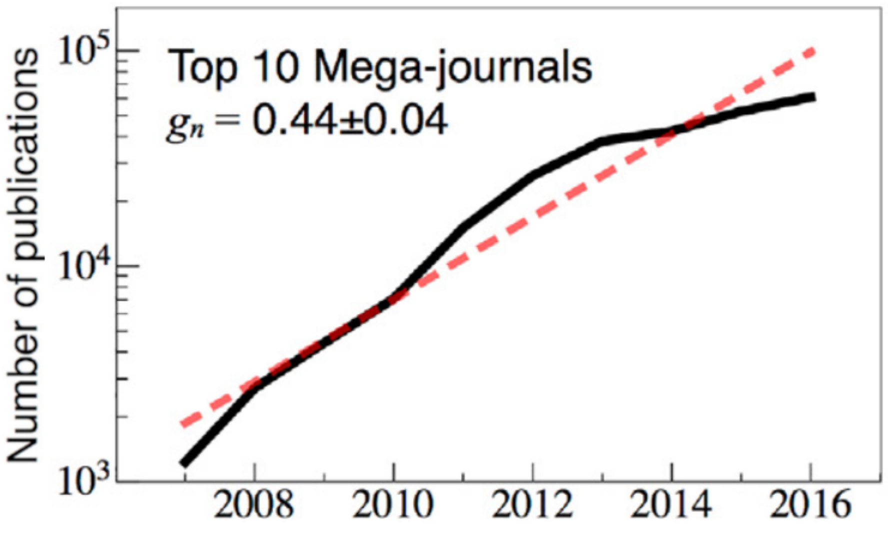


Figure 1.1: Number of publications from top 10 mega-journals, g_n = growth rate [Pet+19]

of AI and Machine Learning (ML), that analyze and evaluate publications on their own. These methods can work with textual semantics, compared to graph-based approaches [Ost23] such as citation analysis, which measures the interconnections between publications [Nic+21]. In addition to citation-based metrics [Nic+21], they can support literature searches and automate tasks that previously had to be performed by humans. This includes tasks such as the classification of the research field, the search for relevant publications, and the summarization of long documents. Online platforms such as "Semantic Scholar"

[Kin+23] or "scite.ai" [Nic+21] already offer similar services, including the recommendation of similar publications, the creation of abstractive text summarizations, citation intent classification, and more.
To automatically perform these Natural Language Processing (NLP) tasks on documents, a feature representation of the text is needed as an input for those algorithms. These representations affect the information that can be extracted from the text, thus they play an essential role as the basis of NLP algorithms [WZJ20; LLS20]. Creating appropriate feature representations is still a main challenge in NLP because lexical semantics such as polysemy, synonyms, or negations need to be considered and the meaning of words, sentences, or whole documents can change in a different context [CP18].
Embedding is a concept that maps language entries such as words or text into an n-dimensional space creating a numerical representation that inherits the semantic meaning of the text enabling efficient processing for downstream tasks [WZJ20; COF23]. In recent years deep learning algorithms enabled the implementation of better embeddings [Mue+23], particularly using context information with methods such as ELMo [Pet+18] or BERT [Dev+19] and its derivatives [RG19], instead of creating static embeddings with methods like word2vec [Mik+13a]. Approaches to improve the learned representations use fine-tuning for specific domains and tasks [BLC19; Gur+20], additional input such as metadata [Zha+21] or citations as inter-document signals and source of semantic information [Ost23] for the generation of whole-document embeddings [Coh+20], and contrastive learning [RA23].
Embeddings can be used for many NLP tasks that are relevant to support researchers but methods, especially concerning documents, are still facing many challenges. These include Out-of-Vocabulary (OOV) words [WZJ20], the semantics of longer documents and expert domains [Ost23] as well as computational efficiency and performance overall [Mue+23]. Also, there is a lack of studies comparing different contrastive learning loss functions and despite the success of using citation to improve embeddings of scientific publications [Coh+20; Ost+22] there is no study exploring a continuous co-citation signal, where co-citations are an alternative method that indicates closely related publications [Pet20; MCH22]. To address these research gaps and explore solutions to improve the embedding of scientific publications, the following research question is formulated:

> *How can a novel approach using machine learning to create document embeddings of scientific publications that are applicable to diverse downstream NLP tasks be developed and evaluated compared to state-of-the-art solutions?*

2 Background

Nowadays more scholarly data than ever before is accessible and can be processed automatically. This is due to the emergence of mega-journals [Dom16], a new type of open-access journal, and refined citation indices that store contextual information [Nic+21]. This data includes full-text publications, citations, authors, publication dates, and more [Kin+23]. Algorithms can use different features from this data to create science maps [Pet20; Chi+19; SL20] and knowledge graphs [EW16; Kin+23] that directly depict the structure and dynamics of scientific domains. This can be used to support literature searches, but to automate tasks of exploring scientific publications that previously had to be performed by humans, it is necessary to find feature representations of this data that can be used as input to downstream tasks [LLS20]. This chapter introduces citation-based (section 2.1) and text-based representations (section 2.2) as they are two complementary approaches to achieve this. Citations can be used as a metric for relatedness between publications [Coh+20] whereas text-based representations can entail semantic meaning and can benefit from external knowledge to improve their representations and understanding of documents [LLS20].

2.1 Citation-Based Representations

Citations are an important component in scientific publications as they reference prior research findings [Nic+21]. Hence, they can expedite their exploration and evaluation, as well as the impact of publications, authors, and journals can be analyzed with them [MK21]. Other prominent use cases are the mapping of research fields [Pet20] and the analysis of the exchange of knowledge within and between domains [SL20]. The creation of knowledge graphs [EW16], which can entail additional information such as automatically generated summaries or structurally parsed text [Kin+23], can help to organize and access this information. Citations also provide a tool to measure the connections between publications [Coh+20] and can serve as a source of semantic information and an indicator of similarity [Ost23]. It has been shown that their combination with ML algorithms that process scientific documents can improve the performance in downstream NLP tasks [Coh+20].
Citations are used for different purposes by the authors of scientific publications [TB18]. For instance, the normative citation theory [Mer73] describes how citations are used to

highlight publications that influenced the author and to give credit to prior work or to contrast research results. This citation intent and other contextual information surrounding citations, such as their location within a document, are not covered by traditional citation indices [Nic+21] thus leading to misinterpretations when looking at raw citation counts. Other factors that need to be considered are their relevance to the topic of the publication and the typical number of citations used within the specific research field [TB18].
Frequently used metrics are *direct citations, bibliographic coupling*, and *co-citations*. *Direct citation* is the most intuitive way to measure the connections between publications. It assumes that documents are more similar to each other if one cites the other, leading to a discrete signal of semantic similarity [Ost23]. *Bibliographic coupling* [Kes63] is a different technique measuring the strength of similarity for publications by counting how many shared bibliographic items they have. Conversely, *co-citations* [Sma73] measures similarity by counting how often a document pair is referenced together by other documents. Normalizing these metrics can be useful as the raw frequencies do not represent the degree of similarity [Pet20] since publications from prominent journals might be found and referenced more often [TB18] and because of citation inflation [Pet+19]. This leads to different strengths of co-citations and thus different measurements of similarity. Other approaches to analyze citations are graph-embeddings [PC20], which are not discussed in this work.

2.2 Text-Based Representations

To work with natural language and solve tasks such as summarization, semantic textual similarity, classification, and more, the text has to be transferred into a numerical representation, which is typically achieved through fixed length vectors [LLS20]. As a first step towards this, a tokenizer is used in a preprocessing task to extract language entries such as characters, sub-words, or words, whereby the different types of language entries extracted have different advantages and drawbacks [Mie+21]. Subsequently, different methods can be used to represent the text separated into tokens. Many of these text-based approaches aim to capture the semantic meaning of the text such that the vectors form an n-dimensional semantic space in which similar vectors are closer to each other than unrelated ones. These methods are based on the *distributional hypothesis* [Har70], which states that linguistic entries with similar distribution tend to have similar meanings, thus, these distributed representations are constrained by the context distribution in the training data [WZJ20]. This section introduces the linguistic fundamentals (section 2.2.1) that need to be considered when designing document embedding algorithms followed by count-based approaches (section 2.2.2), which are the basis for embeddings [PC20]. Afterwards, embeddings (section 2.2.3) and specific methods to create them are explained.

2.2.1 Linguistic Fundamentals

Natural language text comprises unstructured data that can come from sources such as books or articles, spanning diverse domains like law or science [LLS20]. Complete semantic information is usually expressed within sentences composed of individual words according to certain rules, whereas each word can also have multiple meanings depending on the context [LLS20]. To create embeddings of scientific publications that can be used for diverse NLP tasks, these different granularities of language entries need to be related to each other and linguistic building blocks need to be captured by those embeddings.
Morphology describes how words are formed, what their constituent parts are, and how they relate to each other [PC20]. In morphology, words can be segmented into morphemes that are defined as the smallest meaningful constituents of a linguistic expression [SH10]. This is similar to tokenizers [RB21] that convert text into a numerical representation and especially to sub-word tokenization techniques that can deal with OOV words using sub-word units [Wu+16; Mie+21] where words and their meaning are a sequence of constituent parts. **Syntax** defines the rules for the structure of sentences [PC20]. Understanding this structure is important for algorithms because they need to solve issues of syntactic ambiguity where it is unclear which words belong together. This ambiguity can alter the meaning of the sentence. **Semantics** is concerned with the meaning of linguistic terms, which is desired to be achieved with embedding algorithms [PC20]. One major challenge is lexical ambiguity, which describes that a word can belong to multiple syntactic classes, such as a noun or verb, thus having a different meaning, also called polysemy. Other relations between word senses are, for instance, synonyms and antonyms [YM20]. WordNet [Mil95] is a lexical database that was developed to catalog the different senses of words and can be used as a knowledge base within NLP algorithms [LLS20]. Whilst words can have different literal meanings depending on their function in a sentence as their context is described through semantics, **pragmatics** describes the meaning depending on the way the sentences are used, or in other words, the intended meaning [YM20]. This mismatch between the literal and intended meaning can also be found in figurative language such as idioms or sarcasm. Capturing this in embeddings is a major challenge, as it is sometimes not possible to derive the correct meaning without external knowledge [PC20].

2.2.2 Count-Based Models

By analyzing word frequencies and co-occurrences of words, count-based models are capable of creating representations of natural language text that capture the semantic meaning of the text [PC20] based on the assumption made with the distributional hypothesis [Har70]. Simpler methods, such as one-hot representations of words over a fixed vocabulary, do not contain this semantic or syntactic information [LLS20].

The **Bag-of-Words (BOW)** model [Har70] evaluates if a word from a predefined vocabulary is present in the text. It creates a vector with the length of the vocabulary, where each entry represents a single word as, for instance, the number of its occurrences, thus it creates a symbol-based representation [LLS20]. However, BOW does not encode word order, which renders sentences such as "Alice called Bob" and "Bob called Alice" indistinguishable. A similar approach is the **Vector Space Model** [SWY75] that was developed for document retrieval. It creates a term-document matrix with distributed vectors of words. In this matrix, the rows are used to represent the terms and columns for documents. The terms can be either weighted using word frequencies or unweighted using binary values.

However, the raw frequencies are not a reliable measure as they can be biased towards frequent words such as "the", "and", "or". Therefore, the **Term Frequency-Inverse Document Frequency (TF-IDF)** algorithm [SPA72] determines the relevancy of a term by comparing the in-document frequency with the number of occurrences in other documents. The TF-IDF score is computed by scaling the term frequency with the inverse document frequency which describes the specificity of the term in the corpus of documents. There are different ways of defining the term frequency $f_{t,d}$ [MRS08]. In Equation 2.1, it is calculated using the raw frequency of a word in a single document.

$$tfidf(t,d,D) = f_{t,d} * log(\frac{|D|}{1 + \sum_{d \in D} 1}) + 1 \tag{2.1}$$

Where t=term, d=document, D=corpus. **Pointwise Mutual Information (PMI)** [CH89] is a similar method that can be used to assess whether the co-occurrence of words is meaningful. As shown in Equation 2.2, PMI achieves that by normalizing the probability of co-occurring words (w_1 and w_2) by their individual probability to occur, which can be calculated by the raw frequency in the corpus.

$$PMI(w_1, w_2) = log \frac{P(w_1, w_2)}{P(w_1)P(w_2)} \tag{2.2}$$

Other count-based methods analyze for instance word-context or pair-pattern relations [TP10]. Although these metrics are fast to compute, they do not regard word order, and the representations of count-based models can become very large and sparse as the vectors and matrices depend on the vocabulary and corpus size [Ost23].

2.2.3 Embeddings

Embeddings are distributed numerical representations of language entries such as words, sentences, or documents that capture their morphological, lexical, syntactic, and semantic properties in a dense vector mapping the text into an n-dimensional semantic space and enabling efficient processing for downstream NLP tasks [WZJ20; COF23; YS16]. They are learned from the distribution within a training corpus using neural network based methods [PC20] such as word2vec (see section 2.2.4). A limitation of word2vec is that it creates a static embedding for each word that merges multiple meanings of that word into a single point in the semantic space, also described as the *meaning conflation deficiency* [CP18]. More complex models from the field of deep learning allow to take the context into account and create embeddings representing different meanings for words and sentences according to the input [WZJ20]. Muennighoff et al. observed that larger model sizes correlate with higher quality embeddings, leading to better performance of downstream tasks [Mue+23]. These dynamic embeddings encode task-agnostic properties of language [Liu+19], capturing the syntactic and semantic properties with regard to the context, thus solving the meaning conflation deficiency [Ost23]. However, the embeddings of those models are anisotropic, leading to representations that are not uniformly distributed in all directions but only occupy a narrow cone in the vector space with words not well separated in embedding space [Eth19; Gao+19]. Figure 2.1 illustrates the anisotropic word embedding space of a Transformer [Vas+17] compared to the word embeddings of the word2vec model [Mik+13a; Mik+13b] distributed around the origin with a 2D visualization. The

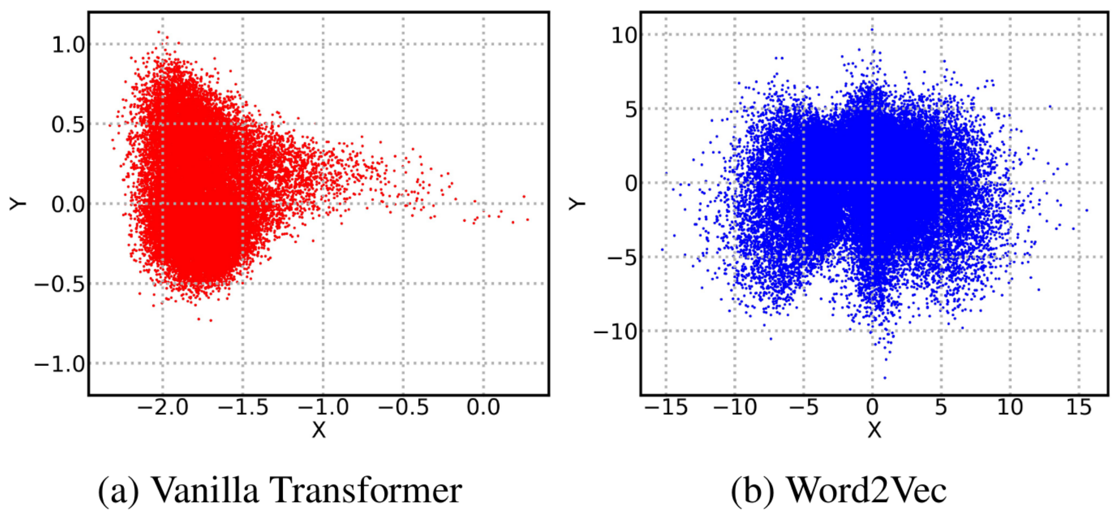


Figure 2.1: 2D visualization of word embeddings [Gao+19]

anisotropic embedding space has a bias towards word frequencies such that high-frequency words are densely distributed and low-frequency words disperse sparsely [Li+20]. This can deteriorate the performance of downstream NLP tasks that need good representations of semantic meanings. Methods to address this are, for instance, normalizing flow [1] [Li+20], whitening [2] [Su+21], and contrastive learning [3] [GYC21].

Other ways to improve embeddings are to leverage a Pre-trained Language Model (PLM) [LLS20] and translate their learned weights to other tasks and domains with finetuning [BLC19] or instruction-based finetuning [Su+23a]. The use of additional information can also be beneficial. For instance, with morphological information, internal knowledge can be derived from the text itself, such as affixes that can help to distinguish between different meanings of a word [LLS20]. Also, external knowledge such as WordNet [Mil95] or citations as an inter-document signal to improve general-purpose embeddings can be integrated into the models [Far+15; Coh+20]. Developing embeddings that have a good performance on diverse NLP tasks is important as it can reduce costs when applied to large corpora [Coh+20].

Document Embeddings

With current algorithms, it is possible to create meaningful embeddings for words and sentences regarding the context [WZJ20]. However, for documents it is more difficult as they are much longer and span over multiple sections and different topics [Ost23]. One approach is to use compositional semantics, where *"the meaning of the whole is a function of the meanings of the parts and of the way they are syntactically combined"* [Par95]. With the averaging or the additive model [LLS20] there exist simple approaches, but they do not consider the syntactical order with the same drawbacks as the BOW model (section 2.2.2). Also it is formulated that *"[...] the meaning of the parts is derived from the whole"* [Fre84; ML10] and that *"the meaning of the whole is greater than the meaning of the parts"* [Lak77; ML10]. Thus, it is necessary to look at the context the words appear in as well as external knowledge. Therefore, the algorithms that can be used for word and sentence embeddings need to be adapted for document embeddings. For instance, the Paragraph Vector-Distributed Memory (PV-DM) model [LM14] extends word2vec by introducing a memory vector representing the paragraph (see section 2.2.5). Large Language Models (LLMs) using the context from the textual features can also be employed. But the length of the documents necessitates methods, capable of processing

[1] Mapping of the embeddings to a Gaussian distribution
[2] Decorrelation of the features and changing variance to one
[3] Bringing related elements closer to each other in the embedding space and unrelated further apart

longer sequences [BPC20], even though the current implementations of such language models often perform worse [PVS22] than PLMs with shorter input sequences such as BERT (Bidirectional Encoder Representations from Transformers) (see section 2.2.7). Many approaches represent documents with a single vector [Ost23], thus generating a single representation for different topics, leading to the meaning conflation deficiency [CP18] for documents. When using a single representation for a document, they can also have a bias towards a specific aspect [Ost23], reducing their performance for downstream tasks. To improve document embeddings, approaches using additional information such as citations [Coh+20], optimized training objectives [LL23], or instruction-based finetuning to generate task- and domain-aware embeddings [Su+23a] have been developed.

2.2.4 Word2Vec

The word2vec [Mik+13a; Mik+13b] algorithm uses shallow neural network models to learn word embeddings from a large corpus. With Continous Bag-of-Words (CBOW) and continuous skip-gram, two distinct model architectures are implemented that learn their embeddings based solely on the local context the words appear in. The CBOW approach predicts a target word w_t by using its surrounding words $w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c}$ with c being the context size. During training the objective of CBOW is to maximize the average log probability in Equation 2.3:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} log\, p(w_t|w_{t+j}) \tag{2.3}$$

Skip-gram on the other hand learns word embeddings to predict the surrounding words such that its objective is to maximize the average log probability in Equation 2.4:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} log\, p(w_{t+j}|w_t) \tag{2.4}$$

The basic formulation uses the softmax function for the prediction task $p(w_{ouput}|w_{input})$ [Mik+13b], which is inefficient because it sums across the whole vocabulary. To train on a large corpus it is replaced, for instance, with a hierarchical softmax function [MB05]. A major limitation of word2vec is that it can only create static representations, not taking the context into account after the training is finished, thus it is not possible to solve the meaning conflation deficiency [CP18].

2.2.5 Paragraph Vectors

The word2vec algorithm is designed to learn embeddings of words, not sentences or documents. A simple approach to overcome this limitation is to average all embeddings of the words within a sentence or document [ML10]. Other word vector pooling methods such as minimum or maximum operations are also possible [Ost23; ML10]. A similar approach is used to create sentence embeddings using the BERT model [RG19]. However, these pooling operations do not consider word order [LM14]. Paragraph Vector [LM14] is an unsupervised algorithm to create embeddings of variable-length text that extends word2vec implementing two different models. The PV-DM model predicts the next word in a context comparable to the objective of CBOW in Equation 2.3, but it uses an additional, unique vector for each paragraph that learns the paragraph's topic. For the prediction, the paragraph vector is either concatenated or averaged with all word vectors for each context sampled from a sliding window over the paragraph. It serves as a feature representing the text outside the size of the current context. Analogous, the *Paragraph Vector-Distributed Bag-of-Words* model's objective is similar to Equation 2.4 of skip-gram, only using the paragraph vector as input to predict the words from the paragraph. This variant requires less storage as the word vectors do not need to be stored.

2.2.6 Transformer

The Transformer [Vas+17] is a sequence transduction model that utilizes the attention mechanism [BCB16] to relate different positions in a sequence to each other instead of using recurrent or convolutional approaches. Therefore, it can be trained with more parallelization, reducing training time. It is built on an encoder-decoder architecture, both of which can be used independently of each other, leading the Transformer to be the base model of many State of the Art (SOTA) solutions for NLP tasks [Mue+23], including the embedding of scientific publications [Sin+23].

Attention

Attention [BCB16] is a mechanism that aims to indicate the important elements of a given input. In NLP, it assigns a weight for each word in an input sequence relative to each other word by computing a compatibility function. The Transformer model employs two forms of attention to achieve this: scaled dot-product attention and multi-head attention.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.5}$$

Equation 2.5 shows the **scaled dot-product attention** with Q, K, and V being the sets of queries, keys of dimension d_k and values of dimension d_v respectively, and $\sqrt{d_k}$ the scaling factor. The set of queries, keys, and values each consist of all the words from an input sequence $x = (x_1, \ldots, x_n)$ of length $n \in \mathbb{N}$. The similarity between a query word and all other key words in the input is calculated as the dot product, leading to higher values for more similar vectors. The result is scaled with $\sqrt{d_k}$ to decrease the variance and improve the stability of the model. The scaled dot product is passed to the softmax function that determines how much attention for a query word should be paid to each other word. This is multiplied with the value vectors for each word to get the attention weighted output.

The Transformer extends this with **multi-head attention** by transforming the original representation of the words with dimension d_{model} into d_k and d_v dimensional vectors using learnable parameter matrices $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}, W_i^K \in \mathbb{R}^{d_{model} \times d_k}, W_i^V \in \mathbb{R}^{d_{model} \times d_v}$ for each attention head $i \in \mathbb{N}$. These matrices differ for each attention head such that the model can utilize information coming from different representations at different positions in the input sequence. For instance, one head can represent syntactic similarities while the other learns the distance between words [Ost23]. The transformation to vectors with dimensions d_k and d_v also reduces the dimensionality and therefore the computational cost. The results of each attention head are concatenated and projected to a final output using a matrix $W^O \in \mathbb{R}^{id_v \times d_{model}}$.

Architecture

The original architecture shown in Figure 2.2 consists of $N \in \mathbb{N}$ stacked encoder-decoder blocks. The encoder receives the initial input sequence, which is converted into a previously learned embedding of dimension d_{model}. The embedding is augmented with a positional encoding by adding sine and cosine functions. Each encoder block consists of a Multi-Head Attention block and a Feed Forward network as sublayers. The sublayers are both connected with residual connections [He+16] and layer normalization [BCB16]. This translates the input into a continuous representation $z = (z_1, \ldots, z_n)$ used in the decoder to generate the output. The decoder duplicates the structure of the encoder but with one additional Masked Multi-Head Attention layer that relates the output of the network with the whole sequence. This enables the model to auto-regressively generate new output tokens that depend on the previous output. The next tokens are predicted using a linear transformation and the softmax function.

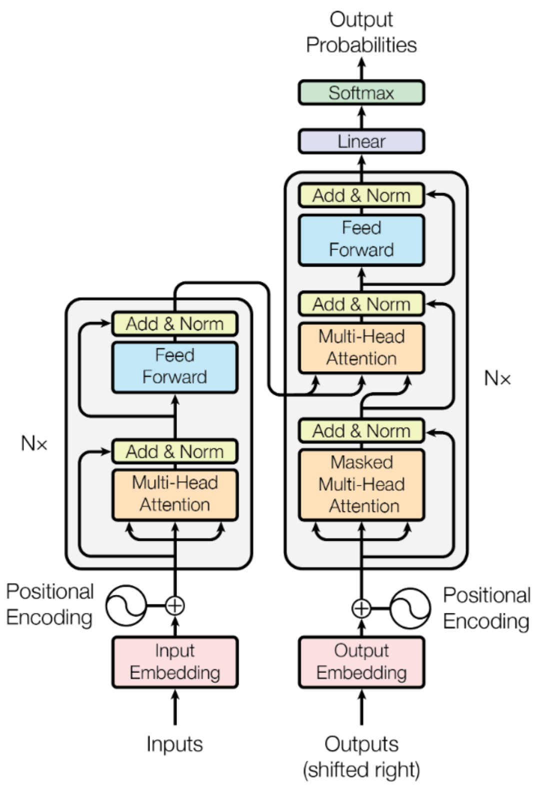


Figure 2.2: Original Transformer architecture [Vas+17]

2.2.7 BERT

BERT [Dev+19] is a model built to learn meaningful representations of words and sentences. It does that by replacing the unidirectional training objective from previous methods that create contextualized word embeddings such as ELMo [Pet+18] with bidirectional pre-training using the left and right context simultaneously. This enables the model to learn rich representations and to be easily fine-tuned for different NLP tasks [LLS20].
As input to BERT, text sequences are converted into WordPiece embeddings [Wu+16], which reduce vocabulary size and can handle OOV words using sub-word tokens. In addition, each sequence begins with the special classification token [CLS] and is separated by the [SEP] token to distinguish between different sentences. For example: `[CLS] Hello world! [SEP]`. To indicate what sentence each token belongs to, a learned segment embedding and a trained positional encoding are used.
The model architecture of BERT consists only of the multi-layer Transformer encoder as described in subsubsection 2.2.6. The pre-training is accomplished with two different tasks being the Masked Language Model (MLM) and Next Sentence Prediction (NSP). In the

MLM task, a certain percentage of the input sequence is randomly masked. Afterwards, the original token is predicted with cross entropy loss. This facilitates the model to learn the bidirectional representations. The NSP task is used for the model to learn how to relate different sentences to each other.
SciBERT [BLC19] transfers this approach to the scientific domain using the same model architecture as BERT but conducting the pre-training on scientific data. This improves the performance of downstream scientific NLP tasks that otherwise are limited due to scarce high-quality data [BLC19].

3 Related Work

3.1 SPECTER

With SPECTER[1] [Coh+20] a method was developed to generate general-purpose document embeddings of scientific publications that do not need task-specific fine-tuning but can be directly used as features for downstream NLP tasks. To achieve this, they use SciBERT [BLC19] as SOTA model for language modeling in the scientific domain and fine-tune its weights on a large corpus. Specifically, they use a contrastive learning objective [RA23] formulated in Equation 3.1 with a triplet margin loss [SKP15].

$$\mathcal{L} = \max\Big\{\Big(s(q, d^+) - s(q, d^-) + m\Big), 0\Big\} \tag{3.1}$$

This loss function takes the embeddings of an anchor document together with a positive and negative sample as input. It learns a representation where the anchor document is closer to all positive publications that are textually related than all other negative samples. To measure the similarity s the L2 norm is used and the hyperparameter m is the margin between positive and negative samples. The training is conducted on a total of 684K training and 145K validation triplets. Each triplet consists of a query paper, a positive paper, and a negative paper such that (q, d^+, d^-) are all training triplets. For each query paper, up to a maximum of five positive and negative publications are sampled, utilizing citations as a naturally occurring inter-document signal, which indicates the relatedness of documents. Positives are publications cited by the query paper, whereas negatives are randomly selected publications that are not cited by the query. Also, another set of "hard negatives" is used to improve the embeddings compared to sole random sampling [Coh+20; Rob+20]. These are slightly related and thus more difficult to learn publications. They are not cited by the query but by a positive sample.

$$s(q, d^+) + m < s(q, d^-) \tag{3.2}$$

[1]Scientific Paper Embeddings using Citation-informed TransformERs

Sampling appropriate positive and negative papers is important as they are key to the contrastive learning objective [Rob+20; Tia+20]. This is also shown in Equation 3.2, where the model does not learn if the samples fulfill the inequality, thus deteriorating convergence. As input to the model the title and the abstract are used as they are written to summarize the publication. No benefits were found using additional fields such as venues and authors [Coh+20]. This is similar to findings in [PVS22], where key information is found in the first tokens of the documents. The title and the abstract of the publication are tokenized with the WordPiece tokenizer [Wu+16] and concatenated with the special [CLS] and [SEP] tokens (see section 2.2.7). Once passed through the model, the document representation itself is taken from the output of the [CLS] token. To evaluate the model, the authors also created an evaluation benchmark with seven document-level tasks called SciDocs. SPECTER outperforms baseline models such as SciBERT [BLC19] in most tasks.

3.2 SciNCL

The sampling strategy presented in SPECTER [Coh+20] uses direct citations to create training triplets. However, this approach does not prevent collisions. For instance, if publication A cites B but not vice versa, B is considered a positive paper for A, while A is viewed as a negative paper for B. This contradiction leads to deteriorating performance [Sau+19]. Also, the sampling strategy provides a discrete similarity signal, which cannot relate similar publications without a direct citation. Further, using raw citations, the training triplets can suffer from noise (see section 2.1). Ostendorff et al. [Ost+22] extend the sampling approach of SPECTER to improve upon these shortcomings and find valuable positive and negative samples for the contrastive learning objective. They train their model SciNCL[2], reusing the training setup of SPECTER but with the data created from the new sampling strategy. SciNCL trains an additional graph embedding model with PyTorch BigGraph [Ler+19] on the citations. These citation embeddings make it possible to sample from a continuous inter-document signal. Therewith, positive papers can be sampled close to the query paper while being dissimilar enough to avoid gradient collapse [WI20] whereas negative papers can be separated with a sample induced margin from the positives to avoid collisions. To create the training triplets, the k-Nearest Neighbors (k-NN) [JDJ21] algorithm is used. This induces a sorting of samples close to the query paper and therefore enables the controlled sampling of positives and negatives within specific ranges and a margin between positive and negative publications can be defined. Alternative sampling strategies applied are random and filtered random[3] sampling. SciNCL is evaluated using

[2]Neighborhood Contrastive Learning for Scientific Document Representations with Citation Embeddings
[3]Random strategy without the samples already selected with k-NN sampling

the SciDocs benchmark [Coh+20] with an overall performance improvement.

3.3 SciRepEval

With SciRepEval,[4] Singh et al. [Sin+23] introduced a benchmark that consists of 24 tasks across the formats of classification, regression, ranking, and search to evaluate representations of scientific documents. It was designed to assess a more diverse range of tasks compared to prior benchmarks, such as SciDocs [Coh+20]. To facilitate and standardize comparison between methods evaluated on the benchmark, they provide training and evaluation datasets. In companion to the benchmark, they released an improved version of SPECTER (see section 3.1) that was pretrained utilizing data across 23 domains named SPECTER2 base. Under the assumption that a single vector is not sufficient to generalize across diverse NLP tasks, different model variations specialized for various task formats are proposed with this base model. Hereto, the base model is finetuned in a multi-task setup on several tasks provided with the SciRepEval benchmark specifically for training. The variations are a model without alterations, a model augmented with *control codes* [Kes+19], and one altered with adapters [Hou+19]. The *control codes* are tokens that are combined with the input as an additional signal indicating the task format. Adapters are additional trainable parameters that are added to each layer in the Transformer model to learn task format specific weights. The results show that the task format specific embeddings of SPECTER2 improve performance.

3.4 INSTRUCTOR

Instruction-based finetuning [Mis+22; Min+24] is a technique where the training data is combined with a natural language instruction that describes the specific tasks. This can improve downstream task performance and zero-shot generalization [Zho+21; Nav+23]. Su et al. [Su+23a] developed INSTRUCTOR to investigate instruction-based finetuning for a general-purpose embedding model. Leveraging such task and domain descriptions, this model can create different embeddings for a single input adapted to the downstream tasks. To train the model, they created MEDI,[5] a collection of 330 datasets for text embedding annotated with instructions. 300 datasets originate from SuperNaturalInstructions [Wan+22], which already are annotated with instructions but are not split into positive and negative training pairs for the contrastive learning objective. To construct these pairs, they created

[4]A Multi-Format Benchmark for Scientific Document Representations
[5]Multitask Embedding Dataset with Instructions

embeddings for the 300 datasets using a Sentence-T5 model [Ni+22a]. From this embedding space, they split the datasets into the positive and negative training pairs by their cosine similarity scores. The remaining 30 datasets stem from the Sentence Transformers embedding data,[6] which they manually annotated with instructions for each dataset. The samples in MEDI are tuples (x, I_x, y, I_y), with x and y as the input and output and I_x, I_y their associated instructions. INSTRUCTOR is based on the GTR model [Ni+22b] and was trained on MEDI using an in-batch sampled softmax loss from Ni et al. [Ni+22b] depicted in Equation 3.3. Here, τ is the softmax temperature and $\mathcal{B}$ is the union between the positive and $k \in \mathbb{N}$ negative pairs. The input for the model $\mathbf{E}_I(I, x)$ is the concatenation of instruction with its input text $I \oplus x$ and the output is calculated as mean pooling over the tokens in x disregarding the instruction tokens.

$$\mathcal{L} = \frac{e^{s(x,y^+)/\tau}}{\sum_{y \in \mathcal{B}} e^{s(x,y)/\tau}}, \text{ with } \quad s(x,y) = cos(\mathbf{E}_I(I_x \oplus x), \mathbf{E}_I(I_y \oplus y)) \tag{3.3}$$

INSTRUCTOR was evaluated on Massive Text Embedding Benchmark (MTEB) [Mue+23], Billboard [Kas+22], and prompt retrieval [Su+23b] and achieved SOTA performance.

3.5 GTE

The GTE[7] model by Li et al. [Li+23] is a general-purpose text embedding model utilizing large-scale and diverse data for multi-stage contrastive learning comprised of unsupervised pre-training and supervised finetuning.

$$\mathcal{L} = -log \frac{e^{s(q,d^+)/\tau}}{e^{s(q,d^+)/\tau} + \sum_{i=1}^{n} e^{s(q,d_i^-)/\tau}} \tag{3.4}$$

To train their model, they use a contrastive loss function similar to the InfoNCE loss [OLV19] in Equation 3.4, which uses a cross-entropy loss to rank one positive sample over n negative samples. This increases a lower bound of mutual information between input data and its embedding. Compared to the triplet margin loss (Equation 3.1), this objective does not depend on a directly defined margin to distinguish between positives and negatives. However, the temperature τ also affects the distribution of the embeddings [Hua+24].

$$\mathcal{L} = -\frac{1}{n} \sum_{i=1}^{n} log \frac{e^{s(q_i,d_i)/\tau}}{Z} \tag{3.5}$$

[6] https://huggingface.co/datasets/sentence-transformers/embedding-training-data (accessed: 06.08.2024)
[7] General Text Embedding

$$Z = \sum_{j} e^{s(q_i,d_j)/\tau} + \sum_{j \neq i} e^{s(q_i,q_j)/\tau} + \sum_{j} e^{s(q_j,d_i)/\tau} + \sum_{j \neq i} e^{s(d_j,d_i)/\tau} \tag{3.6}$$

The contrastive learning objective in Equation 3.5 enlarges the negative samples with bidirectional in-batch queries and documents as shown in the partitioning function in Equation 3.6. The running indices for the elements within a batch are i and j and (q_i, d_j) are the positive text pairs if i=j, else negative. The BERT model [Dev+19] (section 2.2.7) was used as the backbone for GTE. In the first stage, it was pre-trained with ∼800M weak supervised correlation text pairs from open-source data. The weak supervision derives from the inherent connection between the text pairs, such as questions and answers in QA forums, or titles and abstracts of scientific publications. As the number of samples differs between data sources, they employed a multinomial distribution to sample batches of up to 10.000 samples, which they found to improve performance compared to uniform sampling or a direct combination of all samples. The large batch size was used to increase the number of negatives for the contrastive learning objective in Equation 3.5 using in-batch negative sampling. Afterwards, they finetuned the model with the same training objective using ∼3M pairs from symmetric and asymmetric tasks[8] from diverse datasets with human annotations. With the hard negatives from this data, the finetuning could be performed using a smaller batch size. GTE was evaluated on multiple benchmarks, resulting in comparable performance with much larger models.

3.6 AnglE

In contrastive learning, often the cosine similarity is used to assess the differences between vector representations of text and therefore how to adapt the model parameters so that embeddings of similar texts are closer in the embedding space and unrelated texts further apart. However, the cosine similarity faces the challenge of vanishing gradient due to saturation zones of the cosine function [LL23]. Li & Li [LL23] approach this with an optimization of the objective function in complex space. Specifically, they reformulate the CoSENT [Hua+24] loss function from Equation 3.7. Its similarity label *sim* of the input text can be an explicit score or be inferred from the data itself, thus, CoSENT can be applied to various datasets. The indices i, j and k, l depict the positive and negative pairs respectively. Whereas θ indicates the embedded text and τ is a hyperparameter for amplification. The aim of this objective is to rank the similarity of text pairs rather than

[8]Tasks with same or different encoding objective for input and output, e.g. document-document or sentence-document tasks

the similarity between single text samples. It trains a consistent ranking between similar pairs rather than bringing similar texts closer and dissimilar texts further apart in the embedding space.

$$\mathcal{L} = log\Big(1 + \sum_{sim(i,j)>sim(k,l)} e^{\tau(cos(\theta_k,\theta_l)-cos(\theta_i,\theta_j))}\Big) \tag{3.7}$$

To optimize this function, the embedding as the output of models such as BERT (section 2.2.7) is divided into a real $\mathbf{X}_i^{re}$ and an imaginary $\mathbf{X}_i^{im}$ part [Sun+18]. Using this, the angle difference can be calculated as the normalized division in complex space in polar coordinates. To create the AnglE model, the original CoSENT loss function is combined with the optimized version and an in-batch negative objective. These are used to train the model on the MNLI [WNB18] and SNLI [Bow+15] datasets. They evaluated AnglE on various Semantic Textual Similarity (STS) benchmarks for short and long text, with the best performance gains for the combination of the cosine objective, the in-batch negative objective, and the angle optimized objective.

3.7 Summary and Research Gap

LLMs are a prominent approach to generate embeddings as they have the capabilities to incorporate contextual information. Using these models without pre-training or task specific finetuning to create the embeddings, however, can lead to detrimental downstream task performance that is comparable to simpler methods. This chapter presented multiple approaches that were selected as they implement different strategies to overcome this challenge and improve the current state of embedding models, resulting in SOTA solutions on the MTEB [Mue+23]. At the time of writing, the accompanying models were surpassed by various new solutions[9]. In Table 3.1 an overview of the selected methods is presented, including the improved base model, the evaluation benchmarks (see chapter 6) used to assess performance, and the respective approaches. The selected methods employ contrastive learning to enhance the learned representations. It has been demonstrated that leveraging hard negatives can reduce training time and improve the results for this objective [RA23]. SPECTER, SciNCL, and SciRepEval (section 3.1 - section 3.3) use citations as an inter-document signal to augment the sample quality within the scientific domain. At this, the continuous citation signal proposed by Ostendorff et al. [Ost+22] can be beneficial as hard negatives can be collected more effectively. In SciRepEval, the pre-training is extended with multi-task finetuning, as specialized embeddings for different tasks can further increase

[9] https://huggingface.co/spaces/mteb/leaderboard (accessed: 06.08.2024)

Table 3.1: Overview Embedding Models - Describes approaches that are used to finetune the base models and create SOTA solutions on the respective evaluation benchmarks

Model	Base Model	Evaluation Benchmark	Approach
SPECTER [Coh+20]	SciBERT	SciDocs	Discrete citation signal
SciNCL [Ost+22]	SciBERT	SciDocs	Continous citation signal
SPECTER 2 [Sin+23]	SciBERT	SciRepEval	Multi-task finetuning
INSTRUCTOR [Su+23a]	GTR	MTEB, Billboard, Promp Retrieval	Instruction finetuning
GTE [Li+23]	BERT	SST-2, BEIR, MTEB, CodeSearchNet	Multi-stage learning
AnglE [LL23]	BERT & LLAMA	GLUE, STS, GitHub Issues Similarity Dataset	Angle-optimization

performance. Similarly, INSTRUCTOR (section 3.4) is trained to create task and domain aware embeddings with the help of instructions instead of special task tokens or adapters. These instructions are natural language text that provides additional information for task and domain. To create a general-purpose embedding model that needs no additional input and therefore reduces complexity, GTE (section 3.5) leverages large-scale and diverse data to pre-train and finetune a BERT model in a multi-stage learning setup. In AnglE (section 3.6), the cosine similarity often used during training is optimized in complex space. Otherwise, the saturation zones of the cosine function can impose a challenge for the gradient updates.

Although the finetuning is conducted with different contrastive learning objectives, few evaluations between different kinds of loss functions such as triplet margin loss [SKP15] (Equation 3.1), InfoNCE [OLV19] (Equation 3.4), or CoSENT [Hua+24] (Equation 3.7) could be found in the literature, especially not for the embedding of scientific publications and training datasets comprised of triplets. With few exceptions, only comparisons between the performance of an initial loss function and its altered variation are conducted. For instance, Huang et al. [Hua+24] directly compare the performance of CoSENT, MSE, and Softmax on the same datasets.

During the literature review, publications were identified that considered different citation information to sample positive and negative documents. For instance, a discrete citation signal, derived either from direct citations [Coh+20; Sin+23] or co-citations [MCH22], and a continuous citation signal [Ost+22], derived from direct citations. However, no study was found investigating the performance of a continuous co-citation signal to sample training triplets.

4 Concept

To identify viable approaches for creating embeddings of scientific publications and to address the research question, a comprehensive literature review was conducted (chapter 2, chapter 3). Many SOTA solutions to create document embeddings include LLMs capable of generating contextualized representations. A common method to enhance those models is contrastive learning, revising the capabilities of the LLMs to distinguish between related and unrelated documents and encoding these features within the embedding space. However, few studies exist evaluating the influence of different contrastive loss functions. Also, a continuous co-citation signal to sample training triplets has not yet been investigated. Based on the literature review and the research gaps identified in section 3.7, the following objectives are formulated:

Objective 1 Compare the influence of multiple contrastive learning loss functions on the training of LLMs and their embeddings of scientific publications.

Objective 2 Develop an approach that utilizes a continuous co-citation signal.

The remainder of this chapter will introduce detailed approaches to both tasks.

4.1 Objective 1: Contrastive Loss

As identified in section 3.7 few prior studies directly compare the influence of diverse contrastive loss functions on the training progress and the performance of the resulting model. During the literature review, no comparison of the triplet margin loss with other loss functions has been found. Efforts are being made to develop general-purpose embeddings, as demonstrated by the works of Cohan et al. [Coh+20] and Li et al. [Li+23]. This research direction encourages the use and development of loss functions that are applicable across a variety of datasets, as it could enable the training of a model on different datasets without changing the training objective [Hua+24].
To address this, a collection of commonly used loss functions that are gathered from the literature are implemented and compared to each other. This includes the triplet margin loss [SKP15], the InfoNCE loss [OLV19], the CoSENT loss [Hua+24], and the AnglE loss

[LL23]. Apart from their mathematical formulation, they differ in how many negatives they use to contrast a positive sample and whether they compare single positive and negative samples or sample pairs [Hua+24].
The following sections explain which dataset and models are used for the training and how the training process is configured to compare the influence of the contrastive learning objectives on the training of the LLMs and the quality of the resulting embeddings for the scientific publications.

4.1.1 Training Dataset

As training dataset, the data created for the pre-training of SPECTER2 base [Sin+23] is used. It consists of 6.2 million training and 175 thousand test triplets, containing scientific publications from 23 domains in the format of query, positive, and negative publication. The positive and negative publications are sampled from direct citations indicating related and unrelated papers. The samples generated for the training of SciNCL [Ost+22] are included as a subset. Each publication is identified by its corpus ID [Lo+20] and is comprised of its title and abstract. While full document data could improve the quality of the embeddings, it necessitates methods that increase the context size of LLMs. Processing whole documents could also increase training costs. However, models with a shorter context size can perform better if key information exists within the data [PVS22]. In scientific publications, this is provided by the abstract. Thus, a dataset comprising titles and abstracts is adequate for this task. The triplet format dataset allows for the application of all selected loss functions with minor adaptations. The triplet margin loss [SKP15] can be directly applied to this data. For the CoSENT [Hua+24] and AnglE [LL23] loss functions, binary labels need to be inferred to rank the similarity between positive and negative pairs. The typical approach utilizing in-batch negatives for InfoNCE loss cannot be applied as this would introduce noise to the loss functions. To illustrate, if all negatives within a single batch containing the triplets $(q_1, p_1, n_1), (q_2, p_2, n_2)$ are paired with the same query q_1, the publication n_2 is noise if $n_2 = p_1$. Thus, randomly sampled batches could induce negative samples for contrasting with the query that collide with the positive samples. Therefore, either a single negative sample can be used, making the InfoNCE loss similar to other loss functions, the noise could be tolerated, or the data needs to be sorted so that within one batch multiple triplets with the same query are grouped. The sorting enables to utilize multiple negatives for one query document, although at a small number. As the sorting approach does not introduce noise and still utilizes multiple negatives, it was implemented for the comparison.

4.1.2 Model Architecture

LLMs based on the Transformer architecture can take the context into account when creating embeddings of scientific publications. PLMs can be finetuned to reduce overall training time and increase downstream task performance [LLS20]. Prior studies achieved SOTA performance using this approach together with a contrastive learning objective [Coh+20; LL23]. In order to select an appropriate model for the task of comparing different loss functions, model architecture and model size, which impact training time and performance, are considered. From the MTEB leaderboard [Mue+23] and SciRepEval benchmark [Sin+23] good performing baseline embedding models are selected. As the GTE base model [Li+23] is with 110M parameters relatively small but better performing than many larger models, it is selected for the finetuning. It is an encoder-only Transformer model initialized with the BERT [Dev+19] weights and trained using a contrastive learning objective with the average pooling strategy to create the embeddings (see section 3.5). To test if the influence of the loss functions on the training generalizes to other models and is not dependent on the model's pre-training, SciBERT [BLC19] is used as a secondary model. It uses a similar architecture, however, it has different initial weights as it was trained specifically for the scientific domain using the masked language modeling and next sentence prediction task [Dev+19], instead of contrastive learning.

4.1.3 Training Setup

To compare the selected contrastive learning loss functions, the GTE [Li+23] and SciBERT [BLC19] models are finetuned on the triplet dataset with the same number of samples[1] and the same hyperparameters. Afterwards, their convergence speed is compared and their performance is measured using the SciRepEval benchmark [Sin+23]. The finetuning of the LLMs is repeated multiple times with three different random seeds for each loss function to estimate variance and assess the robustness of the training objectives. To increase the validity, this could be repeated more often, but the number was chosen small as a tradeoff with training time. The hyperparameters are mainly reused from SPECTER [Coh+20]. This includes AdamW [LH18] as optimizer with a slanted triangular learning rate of 2e-5 with a warmup of five percent of the training data, a batch size of 32 with gradient accumulation of 4 to achieve an effective batch size of 128[2]. The input size for the models is set to 512 tokens. As pooling strategies average pooling and CLS pooling were implemented. SciBERT was trained with both strategies in separate runs, whereas GTE was trained only using the average pooling strategy, as this was used during the prior

[1]Exception for InfoNCE due to sorted batches (subsubsection 5.3)
[2]Used in GTE [Li+23] with similar hardware

stages of its contrastive pre-training and finetuning. Due to the size of the training dataset, the model was trained for a single epoch as in Li et al. [Li+23]. Therefore, the validation was performed at fixed intervals instead of at the end of an epoch.

4.2 Objective 2: Continuous Co-Citations

The aim of the second objective is to develop a novel approach for the creation of document embeddings of scientific publications. To achieve that, information from the citation graph is utilized. Cohan et al. [Coh+20] and Mysore at al. [MCH22] already use direct citations and co-citations to sample positive and negative publications for the contrastive learning objective. Ostendorff et al. [Ost+22] extended the discrete signal from the direct citations to a continuous signal. In their experiments, they found this to improve the textual sample efficiency. This section describes how this approach is transferred to co-citations (section 2.1). It consists of the collection of the co-citation data itself, a graph embedding model to learn embeddings for all documents within the citation graph, the sampling strategy to create positives and negatives, and the creation of training triplets with titles and abstracts. After the creation of the co-citation dataset, the SciBERT model [BLC19] is trained using the triplet margin loss [SKP15] and the CLS pooling strategy. This is the same model and configuration as in SPECTER [Coh+20] and SciNCL [Ost+22], only changing the dataset. Therefore, the continuous co-citation sampling can be compared to the continuous direct citation sampling [Ost+22].

4.2.1 Co-Citations

To create a co-citation dataset that facilitates the comparison of the co-citation signal with the direct citation signal present in the dataset from *Objective 1* (section 4.1.1) the same query documents are used. Publications that are cited together with these query documents are used as positives instead of publications cited by the queries. The citation information is collected from the Semantic Scholar Academic Graph Datasets [Kin+23] using the corpus ID of each query entry. The current version of the corpus is split into several datasets. For this task, the *'citations'* dataset was used, which is split into 217 files. Each file contains entries with a *citingcorpusid* and *citedcorpusid*[3]. To access this data, an API key of the Semantic Scholar platform [Kin+23] is required. Requesting the information for each individual publication in the dataset would take too long with the request limit of 1 request per second of that endpoint. To acquire the co-citations of the

[3] *citationid*, *isinfluential*, citation *contexts*, and citation *intents* are not needed for this task

queries, the references of their citing elements are extracted. Afterwards, the co-citations for a subset of the queries are generated.

4.2.2 Graph Embedding

Pytorch-Biggraph [Ler+19] is used to learn embeddings for the documents within the co-citation graph. This creates a continuous co-citation signal and enables it to relate publications that are not directly co-cited together with each other. It utilizes a multi-relational directed graph $G = (V, R, E)$ with V as nodes, R and E a set of relations and edges respectively. Each node represents a publication in the co-citation graph, which is connected to other publications with a co-citation as an edge. The relation type for all edges is *co-citation*. Each edge $e = (s, r, d) \in E$ connects a source and a destination node with the relation of type r where $s, d \in V$ and $r \in R$. As Pytorch-Biggraph is a directed graph, it is necessary to duplicate all undirected co-citations such that both formats (a, r, b) and (b, r, a) of the co-cited publications a and b are in the set E. The nodes and relations are represented with an embedding θ that with a scoring function $f(\theta_s, \theta_r, \theta_d)$ is maximized for all edges $e \in E$, else minimized. To optimize the embeddings, the margin-based ranking objective from Pytorch-Biggraph is used. It is formulated in Equation 4.1, where the negative edges $e' \in S'_e = \{(s', r, d) | s' \in V\} \cup \{(s, r, d') | d' \in V\}$ are generated by changing either the source or destination node of the edges e. This is similar to the triplet margin loss (Equation 3.1), however, it uses two edges as input instead of a query, positive, and negative publication.

$$\mathcal{L} = \sum_{e \in G} \sum_{e' \in S'_e} max(f(e) - f(e') + m, 0)) \tag{4.1}$$

The training configuration is adopted from SciNCL [Ost+22] to better compare both approaches. It uses an embedding size of 768, a margin of $m = 0.15$, a learning rate of $\eta = 0.1$, and the dot product as a similarity measure.

4.2.3 Triplet Sampling

To sample positive and negative publications from the graph embeddings, a k-nearest-neighbor search is implemented. Other strategies, such as *similarity thresholds* or *k-means*, have inferior performance [Ost+22]. The k-nearest-neighbor search sorts the samples around a query paper. A fixed number c of positive and negative samples is drawn from different ranges $(k^+ - c, k^+]$ and $(k^- - c, k^-]$ with respective sample difficulties k^+ and k^-. The number of elements between k^+ and k^- describes an adjustable sample induced margin between all positives and negatives. Randomly collecting samples from larger ranges does

not improve quality and is more difficult to optimize due to the additional randomness [Ost+22]. To measure the impact of sample difficulty, multiple combinations with different margins are created.
To curate the final training triplets, the titles and abstracts are extracted from the Semantic Scholar Academic Graph Datasets [Kin+23]. As not necessarily all titles and abstracts can be found in the corpus, the length of the dataset for each sample difficulty is truncated so that all resulting datasets have the same number of triplets. Subsequently, the triplets are split into a training and validation set. The validation sets are created separately for each sample difficulty, but a uniform validation set is required to facilitate comparison of the sample difficulties. Therefore, a combined validation set is created by concatenating all sets for each sample difficulty.

5 Implementation

This chapter presents details about the benchmark, the creation of the continuous co-citation dataset, and the implementation of the training with multiple loss functions. The code is executed on a HPC-Cluster[1] with Nvidia A100 (80GB) and H100 (94GB) GPUs and SLURM [YJG03] as job scheduler[2].

5.1 Benchmark

The SciRepEval benchmark [Sin+23] was chosen to assess the capabilities of the trained models in generating embeddings of scientific publications. The structure of SciRepEval is depicted in Figure 5.1. The `SciRepEval` class orchestrates the evaluation, setting up the tasks and their corresponding parameters as defined in the configuration file, initializing the task-format specific `Evaluator` classes, and saving the results of all evaluators as a file. The `Evaluator` uses an `EmbeddingsGenerator` to either generate or load prior embeddings for the data encapsulated with the `SimpleDataset` or `IRDataset` classes. The functionality to change between task-format specific models and the code provided for multi-task training in the benchmark are not discussed, as they are not relevant to the scope of this work. The original implementation only supports the CLS pooling strategy and uses a hard-coded seed for its nondeterministic algorithms such as LinearSVC[3] with coordinate descent in the dual [Hsi+08].
When executing the benchmark on the HPC-Cluster, time-out errors are observed. Also, the embeddings for the *Paper-Reviewer Matching* task are not always generated. However, no error message is presented. The S2AND[4] [Sub+21] task requires a separate configuration. Consequently, both tasks are excluded. The alterations to the benchmark are made within a fork of the codebase[5]. The code uses the `Datasets` library [Lho+21] to load either local or remote training and evaluation data for each task of the benchmark. The local data

[1]High-Performance Computing-Cluster
[2]To automatically generate the `.job` files, util functions are provided
[3]`https://contrib.scikit-learn.org/lightning/generated/lightning.classification.LinearSVC.html` (accessed: 06.08.2024)
[4]Semantic Scholar's Author Name Disambiguation
[5]`https://github.com/allenai/scirepeval` (accessed: 06.08.2024)

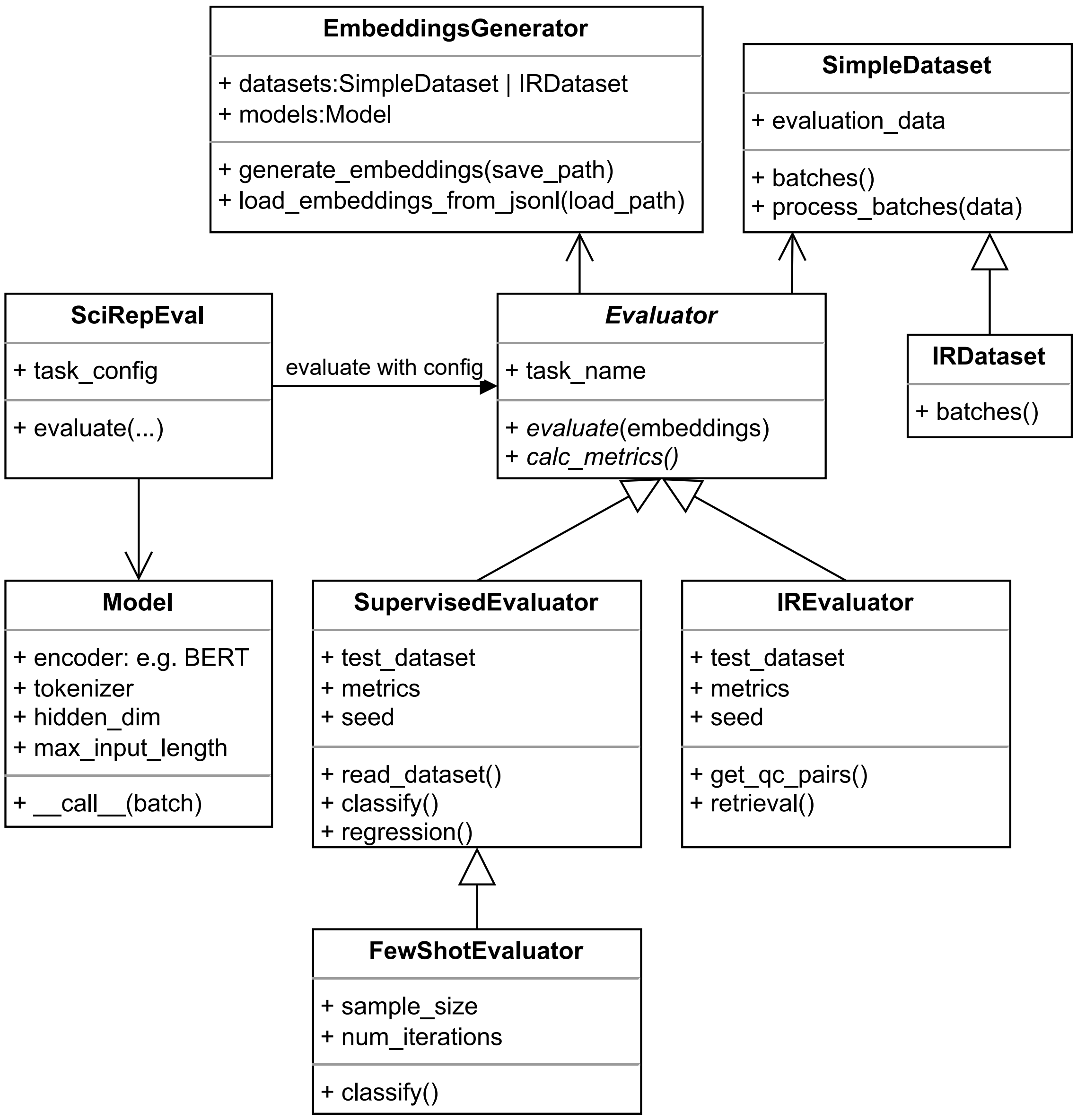


Figure 5.1: Class diagram of SciRepEval Benchmark [Sin+23]

needs to be downloaded from the Amazon cloud, the remote data is accessed through the Huggingface platform. However, the preprocessing between the data differs, resulting in different numbers of data points and single datasets that cannot be parsed using the local data. As no preprocessing pipeline is provided, the data is loaded from Huggingface. The `load_dataset()` function by default caches the data locally. However, without specifically exporting the `HF_DATASETS_OFFLINE` environment variable, requests to access the data are sent, leading to timeouts. To cache the data without an error, a function iterating

over all datasets is implemented. Before this solution was found, a *waiting* mechanism was implemented. Also, partial results are saved and already computed tasks are skipped.
As the LLMs are trained using the CLS and average pooling strategies (section 5.3) the benchmark should generate these different embeddings as well. Therefore, the `Model` and `EmbeddingsGenerator` classes are adapted by also calculating the average of the `last_hidden_layer` of the encoder output and appending the alternative encoding to a `list`. Afterwards, the evaluation is computed on both embedding types per task by invoking the evaluate function with the different embeddings.
The hard-coded seed is changed into a parameter that could be adjusted. The whole benchmark is run three times for each model to account for variance within the evaluation algorithms. As the embeddings do not change, they are computed only once and written to disk. The runtime of the benchmark with the GPU accelerator is approximately 15 hours.

The results of each benchmark run are saved to different files according to the model configurations and the seed used for the benchmark, which is repeated three times due to non-deterministic evaluation methods. This results in a total of nine benchmark results per model, with three seeds used during training of the models. A script is implemented with pandas [tea23] to automatically summarize the results using the same aggregation methods as provided by an excel sheet[6]. All metrics of the benchmark are scaled between 0 and 1, with 1 representing the optimal score. Thus, a task-wise average can be calculated. Singh et al. [Sin+23] report task-wise averages for the categories: "in-train", "out-of-train", and "SciDocs" [Coh+20] as well as for each task-format (classification, regression, proximity, search) and an overall average score. The in-train category describes the tasks that are provided for (multi-task) finetuning and adaptation to the different task formats in the benchmark. The out-of-train tasks are exclusively utilized for evaluation. The same task-wise averages are calculated after the results for each seed are averaged.

5.2 Co-citation Dataset

The creation of the co-citation dataset is split into several sub-tasks, including the extraction of the co-citations from the Semantic Scholar Academic Graph Datasets [Kin+23], the training of a Pytorch Biggraph model [Ler+19], the k-NN search, and the triplet generation. To download the citations, papers, and abstracts datasets (release: 16.07.2024), a HTTP GET request is sent with an API key of the Semantic Scholar platform to the specific endpoint of each dataset using the `requests` library. The response body contains pre-signed download links for the dataset files, which are then downloaded using the `wget` library. The

[6]`https://docs.google.com/spreadsheets/d/1JMq-jR4M8KU119cvglUDmMwwzd60Z3vyvn3VqhPn9EY/view#gid=1450677429` (accessed: 06.08.2024)

download process is split between multiple nodes on the SLURM cluster as the archived data is several hundred gigabytes large. These archives are then extracted, resulting in roughly a terabyte of data (706GB citations, 183GB papers, 133GB abstracts).

5.2.1 Co-citation Extraction

After iterating over all files to extract the citing publication's references for a subset of the query publications, the co-citations are created as paired combinations of the sorted references. The resulting `tuples` are used as unique keys within a `dictionary` and a counter as value is increased each time that co-citation is found. This is only done for the selected subset of queries, otherwise, the number of co-citations grows too quickly with an upper bound of $(length - 1) * (length/2)$. Unrestricted runs create `.tsv` files of over 100GB for 1% of the query publications.

5.2.2 Graph Embedding

The Pytorch-Biggraph framework [Ler+19] provides out-of-the-box functionality to train a graph embedding model. Only the `get_torchbiggraph_config()` function needs to be implemented that returns the configurations within a `dictionary`. Important options include the path to the training data, its entities, and relations, as well as parameters for the training itself such as the number of epochs, learning rate, and the margin between embeddings. To prevent `out_of_memory` errors, the number of edge chunks needs to be set, otherwise, the training needs over 1.5TB of memory. Using the `torchbiggraph_import_from_tsv` command, the framework prepares the data for the training. With the extracted co-citations, this results in unique 31.544.787 entities (= publications) and 250.874.460 edges (= co-citations). Afterwards, the training is commenced with the `torchbiggraph_train` command, which creates the embeddings for each entity and a `training_stats.json` file containing statistics about the training process.

5.2.3 Triplet Sampling

The graph embeddings enable the sampling of positive and negative publications from a continuous co-citation signal. To select positives and negatives, a k-NN search is implemented with the FAISS library [Dou+24]. The first step is to create an offset list of the query publication to load their embeddings. Thereafter, an `IndexFlatL2` index is created, which implements an exhaustive search. From this index, the nearest $k = 5000$ neighbors are identified and a result file is saved. From these 5000 nearest neighbors, seven positive and negative publications are sampled using multiple margins to assess the impact of sample difficulty[7]. Similar to the extraction of the co-citations, the title and abstracts are collected from the *papers* and *abstracts* datasets from the Semantic Scholar Datasets [Kin+23]. With a random seed, the training and validation splits are created with `sklearn's` train_test_split() function. The validation data is combined for all different margins used so that the validation data is the same for all margins during training.

5.3 Training

The training of the SciBERT[8] and GTE[9] models is implemented using Pytorch Lightning[10]. This framework provides a `Trainer` class, which automates the training process and manages the logging and execution of callbacks. The logging is realized with Lightning's `TensorBoardLogger` and model checkpoints are created after each validation run, monitoring the validation loss. From the `Trainer` only the `fit()` function is used as the testing is done with the benchmark. It takes a `LightningModule` and the training and evaluation `Dataloader` as input. The `SciPubEmb` class is implemented for this thesis. It extends the `LightningModule` and defines the key components of the training loop:

`__init()__` Configures training and loads model and tokenizer from Huggingface [Wol+20].

`configure_optimizers()` Initializes the optimizer and the learning rate scheduler.

`forward()` Tokenizes a batch of input documents, passes them through the model, and returns embeddings using either CLS or average pooling.

`training_step()` Splits batch into query, positive, and negative samples and creates embeddings with forward function. Then, it applies the loss function, logs results, and returns the loss.

`validation_step()` Same as `training_step()` but for validation data.

The hyperparameters used during training are as follows: batch size = 32, gradient accumulation = 4, epoch = 1 (3 if training with the continuous co-citation datasets), learning rate = 2e-5, optimizer = AdamW [LH18], scheduler = slanted triangular learning rate with 5% warm-up. In the `get_data_loader()` function, the `datasets` library is used to load and shuffle the datasets with a random seed and apply a mapping. This includes the training triplets from SPECTER2 base [Sin+23], its adapted version for

[7] $Positive = \{25, 200, 500\}$, $Negative = \{1000, 3000, 5000\}$
[8] `https://huggingface.co/allenai/scibert_scivocab_uncased` (accessed: 06.08.2024)
[9] `https://huggingface.co/thenlper/gte-base` (accessed: 06.08.2024)
[10] `https://lightning.ai/docs/pytorch/stable/` (accessed: 06.08.2024)

the InfoNCE loss function, and the co-citation data of *Objective 2*. The `map()` function concatenates the title and the abstract with the [SEP] token of the tokenizer for each triplet. The datasets are streamed instead of computing this mapping all at once. Afterwards, the streamed datasets are wrapped within a `Dataloader`. The training using the A100 or H100 Nvidia GPUs can last from two days to a week on the largest dataset. Due to the maximum runtime of one day on the cluster configured with SLURM [YJG03], it is interrupted. To save the training progress and continue afterwards, an additional `InterimCheckpointCallback` was implemented. It checks the remaining time at the start of a training batch and the validation loop with the `on_train_batch_start()` and the `on_validation_start()` hooks. After 23.5 hours, it saves an interim checkpoint and stops the training. However, the state of the `Dataloader` is not saved. Therefore, a `SkipIterableDataset` is implemented that creates an `Iterator` of the dataset and skips the first $n = global\_step * batch\_size * gradient\_accumulation$ samples. The following sections describe how each loss function is applied to the triplet format datasets.

Triplet Margin Loss

The Triplet Margin Loss [SKP15] is copied without changes from the SPECTER repository[11]. To measure the distance between embeddings the L2-norm is used, which is implemented with Pytorch's `pairwise_distance()`. The loss is computed as described in Equation 3.1 using $m = 1$ as margin with a $ReLU$ function. The losses for each pair within the batch are aggregated using a mean reduction method.

InfoNCE

The InfoNCE loss [OLV19] in Equation 3.4 contrasts the similarity between a *(query, positive)* pair with the similarity of $n \in \mathbb{N}$ *(query, negative)* pairs. This can be implemented with a cross-entropy loss function. To compute the InfoNCE loss for larger batches where triplets have different query publications, each negative is matched with its corresponding query[12]. It computes the cosine similarity between all queries and positives and all queries and negatives as the dot product of their normalized embeddings. The resulting logits are concatenated and labels are created and passed to the cross-entropy function scaled with $\tau = 0.1$ as temperature. To aggregate the loss of each mini-batch, the mean reduction method is used.
It is important to ensure that within a mini-batch the InfoNCE loss is calculated using the same query document for all similarity pairs. Otherwise, the query document is not

[11] `https://github.com/allenai/specter` (accessed: 06.08.2024)
[12] `https://github.com/RElbers/info-nce-pytorch/tree/main` (accessed: 06.08.2024)

directly contrasted with the positive and negative samples. In the triplet format dataset, each negative sample is assigned to a single query document. To utilize the correct query document, the dataset is sorted into blocks of a fixed size with one query document throughout all samples in the block. This approach reduces the number of queries that can be used, as not all queries within the original dataset have the same number of positive and negative samples. To keep the number of queries comparable, a block size of four is chosen. Shuffling is applied between and within the blocks.

CoSENT

The CoSENT loss function[13] in Equation 3.7 [Hua+24] compares the cosine similarity of embedding pairs depending on their similarity label. The similarity labels are inferred from the triplet dataset, assuming that the similarity of a (query, positive) pair is higher than the similarity of a (query, negative) pair. This is reproduced by creating binary labels for all positive and negative pairs. All elements within the label vector are compared to each other, resulting in a label matrix. In a second step, the similarity of all *(query, positive)* and *(query, negative)* pairs is computed utilizing the `cosine_similarity()` function of `torch`. The result is scaled with $\tau = 20$. Subtracting the similarity pairs from one another as in Equation 3.7 is implemented as a matrix operation. The resulting similarity matrix and the label matrix are combined, removing similarities with label zero by subtracting 1e12. This results in a convergence to zero with the exponential function. The loss is computed with the `logsumexp()` function of `torch`.

AnglE

The implementation of the AnglE loss[14] [LL23] is similar to the CoSENT loss (subsubsection 5.3) except for the optimization in complex space. Here, the chunking strategy to obtain the real and imaginary parts [Sun+18] was transferred to the triplet dataset format instead of using a zigzag-style input, creating chunks for query, positives, and negatives separately. The division in complex space in polar coordinates to compute the angle difference is reused from Li et al. [LL23].

[13]Adapted from `https://github.com/RingBDStack/CoSENT/blob/main/loss.py` (accessed: 06.08.2024)
[14]`https://github.com/SeanLee97/AnglE/blob/main/angle_emb/angle.py` (accessed: 06.08.2024)

6 Evaluation

This chapter introduces approaches and metrics used to assess the performance of algorithms that create document embeddings of scientific publications. Afterwards, the results for the developed methods are presented. To assess the quality of the embeddings an intrinsic or extrinsic evaluation can be conducted. The **intrinsic evaluation** encompasses methods that directly evaluate the quality and coherence of a vector space, regardless of its performance in downstream NLP tasks [Sch+15]. These methods evaluate embeddings on the syntactic or semantic relations between language entries such as words or documents by comparing how well they align with human intuition [Bak18; Sch+15]. This ensures that similar words or documents are represented mathematically close to each other. This includes, for instance, the prediction of semantic similar language entries using a distance measure such as the cosine similarity. These similar entries express the same sense such as synonyms [LLS20]. An overview of intrinsic evaluation methods can be found in Bakarov [Bak18]. With intrinsic evaluation methods the relationships between embeddings can be analyzed and using e.g. visualizations internal properties can be explored in a lower-dimensional space [Gao+19; Coh+20; Hei+22]. A limitation of intrinsic evaluation methods is that they are often limited by small datasets and that they do not always correlate with downstream task performance as different tasks need different notions of similarity [Far+16; PC20]. However, with intrinsic evaluation methods, the relationships between embeddings can be analyzed, and using e.g. visualizations internal properties can be explored in a lower-dimensional space [Gao+19; Coh+20; Hei+22]. **Extrinsic evaluation** uses embeddings as input features for downstream tasks to measure the quantitive performance of the vector representations using task-specific metrics [Sch+15]. This approach shows the effectiveness of embeddings in real-world applications [PC20]. Many tasks such as sentiment analysis or topic categorization can be used [PC20]. Limitations of extrinsic evaluation are that the performance of embeddings is not consistent across tasks [Sch+15] and the result is dependent on many factors such as the configuration of the algorithm used for testing [PC20]. Therefore, it is necessary to evaluate embeddings across diverse tasks and datasets to approximate their general applicability or restrict their use to specific tasks [Bak18].

6.1 Evaluation Approach

The evaluation considers the following components: (i) the performance of the graph embedding, (ii) the influence of the contrastive learning loss functions on the convergence for the training of the LLMs, and (iii) the resulting performance on the SciRepEval benchmark, which uses a mixture of intrinsic and extrinsic tasks. To estimate the variance, all experiments are conducted three times with different random seeds and mean and standard deviation are reported. Furthermore, the following configurations are compared: (i.A) multiple embedding sizes and (i.B) distance measures for the training of the graph embedding models; (i.C) the sample difficulty with different margin hyperparameters; (ii.A) the influence of different initial weights of PLMs and (ii.B) pooling strategies.

6.1.1 Metrics

This section provides an overview of the metrics used to evaluate the performance of the graph embedding models and those used within the SciRepEval benchmark to assess the quality of embedding models in the scientific domain. In classification tasks, the F-score measures the balance between the precision and recall of the predicted classes. To assess the quality of the embeddings as input to a regression model, the prediction of that model is compared with the ground truth using Kendall's τ. The Mean Average Precision (MAP) evaluates if information about a publication can be correctly identified using only the embeddings. For a more nuanced measure, Normalized Discounted Cumulative Gain (nDCG) assesses the relevancy of publications ranked with distance measure in the embedding space. With Hits@k it is tested if a relevant publication is found in the first k ranked publications and Mean Reciporal Rank (MRR) measures how quickly a relevant publication can be found. Except for the Hits and MAP the following sections describe those metrics in more detail.

F-score

For the evaluation of a classifier, a common measure of predictive performance is the F_β-score and its F_1-score variant [CHK23]. It is defined in Equation 6.1, combining two aspects of classifier performance with precision P and recall R. However, the F-score has some drawbacks such as it ignores true negatives, it can have the same value for different precision and recall values, and is not a representational measure (only a numerical value, not an objective feature of the classification) [CHK23].

$$F_\beta = \frac{(\beta^2 + 1)P * R}{\beta^2 * P + R} \tag{6.1}$$

Specifically, the *Macro F1* metric is used, which averages the F1 scores calculated independently for each class resulting in more robustness to class imbalance [OB21].

Kendall's τ

Kendall's τ [KEN38] computes a rank correlation on non-parametric data, where it only needs the rank of the values as input and not the numerical values. With n samples of paired observations (X, Y)[1] there exist $m = \frac{n(n-1)}{2}$ possible comparisons[2]. An observation pair is considered concordant if, for the rank order $i < j$, both $X_j - X_i$ and $Y_j - Y_i$ have the same sign, otherwise, the pair is discordant. With C and D as the number of concordant and discordant pairs respectively, Kendall's τ can be calculated as shown in Equation 6.2.

$$\tau = \frac{C - D}{m} \tag{6.2}$$

It scales between [-1,1] so that when the value is 1 the rankings are the same, when -1 the order is reversed. Compared to Spearmans ρ it can be interpreted as the probability that the observed data is in the same order [Puk11].

Mean Reciporal Rank

The MRR [Cra09] is defined in Equation 6.3 with the set of all queries Q. It calculates the average multiplicative inverse of the rank of the first relevant objects. When each query has a single relevant document it is equivalent to the *Mean Average Precision*.

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i} \tag{6.3}$$

It is easy to compute, however, it does not take multiple relevant documents into account and can suffer from early errors. It is used to assess the trained graph embeddings, measuring how quickly the first co-cited publication can be found in the embedding space. A better score means that the model ranks relevant publications higher.

Normalized Discounted Cumulative Gain

The nDCG [Wan+13] is a family of ranking measures that employs a discount function $D(r)$ describing a decreasing function of the rank r of an object. The logarithmic discount $\frac{1}{log(1+r)}$ is an often applied variation. In Equation 6.4 the discounted cumulative gain is

[1] The observations are the pairs: (prediction, label).
[2] The m is used as normalizing coefficient

depicted where S_n is a dataset and y_r^f is the corresponding relevancy to ranked object x_r^f using f as a ranking function. With $IDCG_D(S_n) = max_f(DCG(f, S_n))$ the nDCG is defined with $NDCG_D(f, S_n) = \frac{DCG_D(f,S_n)}{IDCG(S_n)}$. The nDCG allows for degrees of relevancy instead of a binary measure, considers higher-ranked objects with more importance, and the normalization enables the comparison of its output between different queries.

$$DCG_D(f, S_n) = \sum_{r=1}^{n} y_r^f D(r) \tag{6.4}$$

6.1.2 Benchmark

To measure the quantitative performance of the trained LLMs using different contrastive learning loss functions, they are evaluated on the SciRepEval benchmark. This specific benchmark is chosen, as it is aimed at the embedding of scientific publications and it employs different task formats and datasets. This is important to assess the general applicability of the embedding models. It consists of 24 tasks across four task formats (classification, regression, ranking, and search). For each task, the embeddings are created using the combined title and abstract of a publication. Other formats such as question answering and generation are not present in SciRepEval.

Classification Tasks

Classifying publications into distinct categories is important for managing and retrieving documents, as it helps in assigning a publication to a specific topic, assessing its relevance (spam filtering) and more [COF23; Zha+21]. To evaluate the quality of the generated embeddings, the benchmark trains a linear support vector classifier[3] with the embeddings as input features, and the performance is measured with the *Macro F1-score*, except for the binary *Biomimicry* task where the Binary F1-score is calculated. In total, they use six tasks, including Biomimicry,Disease Research State Model (DRSM), Fields of Study (FoS), Medical Subject Headings (MeSH) Descriptors, SciDocs Microsoft Academic Graph (MAG) and MeSH. A short description of each task is presented in Table 6.1.

Regression Tasks

To assess the regression performance a linear support vector regression model[4] is trained and the *Kendall's τ rank correlation* between ground truth and prediction is reported. The

[3] https://contrib.scikit-learn.org/lightning/generated/lightning.classification.LinearSVC.html (accessed: 06.08.2024)
[4] https://contrib.scikit-learn.org/lightning/generated/lightning.regression.LinearSVR.html (accessed: 06.08.2024)

Table 6.1: SciRepEval - Classification Tasks

Task Name	**Metric**	**Description**
Biomimicry	Binary F1	Decision whether publication from the PeTaL database [Shy+19] is about biomimicry or not
DRSM	Macro F1	Assign publications to six specific aspects of rare diseases
FoS	Macro F1	Assign publications to a at most three of their associated fields-of-study
MeSH Descriptors	Macro F1	Assign publication to the descriptors in MeSH forming a hierarchy
SciDocs MAG	Macro F1	Assign publication to its predefined topic categories
SciDocs MeSH Diseases	Macro F1	Assign publication to the descriptors in a subset of MeSH

regression format enables the assessment of continuous tasks, such as quality or influence scores. The specific tasks used are described in Table 6.2.

Table 6.2: SciRepEval - Regression Tasks

Task Name	**Metric**	**Description**
Citation count	Kendall's τ	Predict number of citations of publications from 2016
h-Index of Authors	Kendall's τ	Predict maximum h-Index of any authors of a publication with labels
Peer Review Score	Kendall's τ	Predict average rating of publication from OpenReviewAPI metadata
Tweet Mentions	Kendall's τ	Predict the number of mentions and retweets of a publication
Year of Publication	Kendall's τ	Predict the year of publication

Proximity Tasks

In the proximity task format publications related to a query document are ranked. This is used in retrieval and recommendation of scientific publications where similar publications are selected from the embedding space based on a distance measure[5]. SciRepEval evaluates embeddings for this format on ten different tasks, which are described in Table 6.3.

[5]SciRepEval uses the Euclidean distance

Table 6.3: SciRepEval - Proximity Tasks

Task Name	Metric	Description
Highly Influential Citations	MAP	Find publications that are highly influential for the query, which are articles that are cited at least 4 times by the query
Paper-Reviewer Matching	Precision	Find reviewer for a query publication by comparing the query with multiple articles of the reviewer and using the highest average similarity score
RELISH	nDCG	Find relevant publications on PubMed for a query
Same Author Detection	MAP	Find out of three publications the article pair written by the same author
SciDocs Cite	MAP, nDCG	Find cited publications based on a query publication
SciDocs Co-Cite	MAP, nDCG	Find co-cited publications based on a query publication
SciDocs Co-read	MAP, nDCG	Find publications that are downloaded together in one session by a user of an academic search engine
SciDocs Co-view	MAP, nDCG	Find publications viewed together in one session by a user of an academic search engine

Ad-Hoc Search Tasks

The Ad-Hoc Search is the asymmetric counterpart to proximity where publications are ranked given a short query sentence instead of a document. Therefore, it is primarily used to discover publications, similar to a search engine. SciRepEval employs three different tasks to evaluate this format, depicted in Table 6.4

Table 6.4: SciRepEval - Ad-Hoc Search Tasks

Task Name	Metric	Description
NFCorpus	nDCG	Find publications related to nutrition from PubMed
TREC-CoVID	nDCG	Find literature related to COVID-19
Search	nDCG	Find publications that are most often selected by users of the Semantic Scholar platform for a specific query

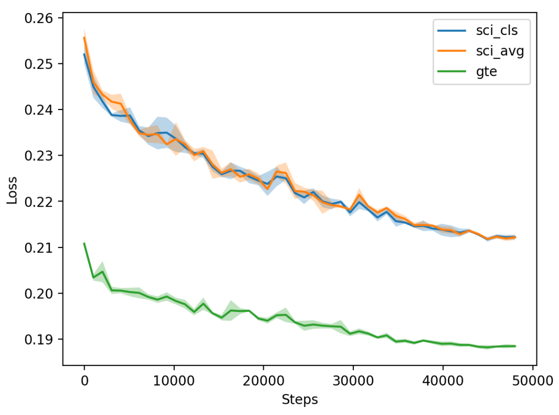
6.2 Evaluation Objective 1: Contrastive Loss

To evaluate *Objective 1*, the influence of multiple contrastive learning loss functions on the training of LLMs and the resulting embeddings for scientific publications, the SciBERT and GTE models are trained on the SPECTER2-base pre-training dataset. The training with each loss function is repeated three times to estimate variance. An extrinsic evaluation with the SciRepEval is the primary approach to compare the influence of the loss function on downstream task performance. To assess differences in the training process itself, the convergence of the validation loss is shown in Figure 6.1 using the average loss across all training runs and the standard deviation across the different seeds for each model. It can be observed that the loss for the GTE model is generally lower than for SciBERT and that the influence of the pooling strategy is marginal. No overfitting can be seen within the first epoch of training. The standard deviation for all loss functions is relatively low (Triplet = 0.0009, InfoNCE = 0.006, CoSENT = 0.004, AnglE = 0.0009). The higher standard deviation for the InfoNCE loss is most likely due to the fewer number of samples used within each batch compared to the other loss functions. The loss of a single training run with the CoSENT loss and the GTE model increased within the initial steps, however, afterwards it converged with the other training runs resulting in a lower validation loss on average similar to the other loss functions.

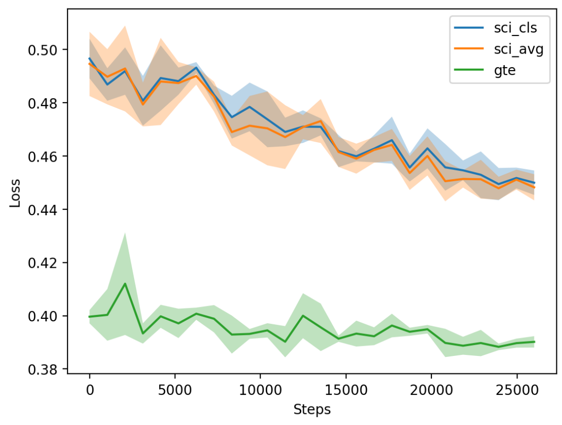
Benchmark Results

With the SciRepEval benchmark, the loss functions and their influence on downstream task performance are evaluated. In Table 6.5 the results on the different task formats are listed as the mean and standard deviation across three model and benchmark seeds[6]. The best results are highlighted in **bold**, differences between the loss functions <u>underlined</u>. For comparison, the results of SciBERT, SPECTER, and SciNCL from Singh et al.[7] [Sin+23] and the GTE model are used as they are trained with a similar contrastive learning objective and data, or used for the training with the loss functions.
The general-purpose embedding model GTE achieves the best overall performance on the benchmark due to its performance on the *Proximity* and the *Ad-Hoc Search* formats. Especially, GTE outperforms all non-GTE models by an average of ∼6% on the *Ad-Hoc Search* format, which can be attributed to its use of asymmetric tasks during pre-training. However, when further training GTE for a single epoch, its performance slightly decreases

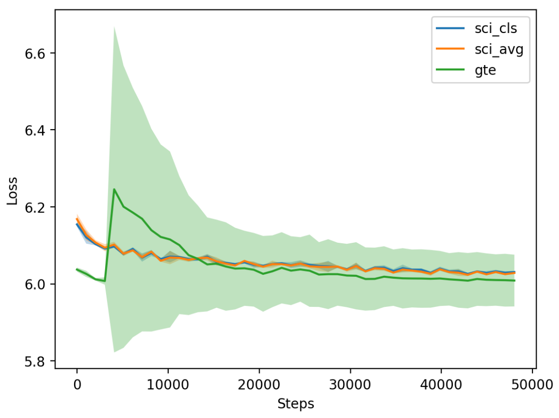
[6]The evaluation results on *In-Train*, *Out-of-Train*, and *SciDocs* split are excluded from this section as the models are not finetuned withe the In-Train data (see Table A.1)
[7]`https://docs.google.com/spreadsheets/d/1JMq-jR4M8KU119cvglUDmMwwzd60Z3vyvn3VqhPn9EY/view#gid=1450677429` (accessed: 06.08.2024)



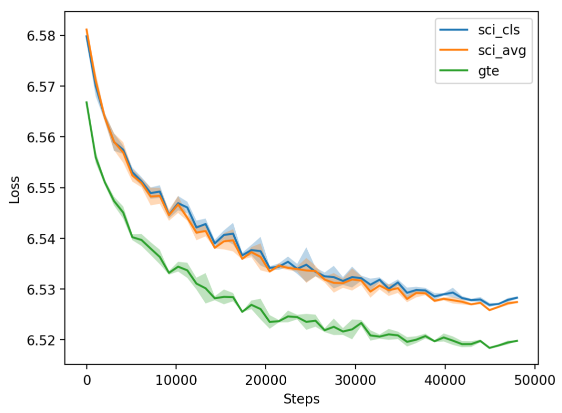
(a) Triplet Loss



(b) InfoNCE Loss



(c) CoSENT Loss



(d) AnglE Loss

Figure 6.1: Contrastive Loss Functions - Validation loss of the training with the GTE (green) and SciBERT model (orange, blue), which is trained with the CLS and the average pooling strategy. For each loss function the validation loss is presented as the average across three random seeds with the standard deviation

on those tasks. For the *Classification* format, the best results are achieved with the InfoNCE loss across all models. Yet, it has the worst results on the *Regression* format, decreasing performance. In this format, the triplet margin loss attains the most improvement. The CoSENT loss acquires the highest scores across all trained models on the *Proximity* format. The AnglE loss receives the lowest scores in most formats. These results do not show a clear best loss function with the Triplet Margin Loss and the CoSENT loss sharing the same best overall score, however, the results indicate differences between the performance on specific task formats. No consistent difference between the different pooling strategies could be identified.

Table 6.5: SciRepEval benchmark results across four task formats of the GTE and SciBERT models trained with the contrastive loss functions reported as the average and standard deviation of nine runs. The values for SciBERT, SPECTER, and SciNCL originate from the authors of SciRepEval

Model	**Classification**	**Regression**	**Proximity**	**Ad-Hoc Search**	**Average**
SciBERT	63.7	23.2	66.1	68.2	57.2
SPECTER	67.5	22.2	85.8	74.9	67.4
SciNCL	67.8	23.9	87.2	77.4	67.8
GTE	67.2 (±0.0)	20.5 (±1.6)	**87.8** (±1.1)	**81.9** (±0.0)	**68.7** (±0.8)
GTE-avg Triplet	66.2 (±0.2)	21.9 (±2.0)	87.3 (±1.1)	80.4 (±0.0)	68.3 (±0.9)
GTE-avg InfoNCE	**67.9** (±0.3)	15.8 (±2.5)	87.4 (±1.1)	80.8 (±0.2)	67.6 (±1.1)
GTE-avg CoSENT	66.5 (±0.5)	21.5 (±1.8)	87.7 (±1.1)	80.5 (±0.2)	68.5 (±1.0)
GTE-avg AnglE	67.7 (±0.1)	17.4 (±1.4)	84.8 (±1.3)	76.8 (±0.1)	66.3 (±0.9)
SciBERT-avg Triplet	66.0 (±0.3)	25.9 (±0.7)	86.8 (±1.1)	76.9 (±0.3)	68.4 (±0.7)
SciBERT-avg InfoNCE	67.5 (±0.4)	14.6 (±2.0)	86.8 (±1.2)	77.3 (±0.2)	66.6 (±1.0)
SciBERT-avg CoSENT	67.0 (±0.3)	22.4 (±2.1)	87.5 (±1.0)	77.2 (±0.2)	68.3 (±1.0)
SciBERT-avg AnglE	66.9 (±0.2)	14.6 (±2.2)	84.9 (±1.3)	73.9 (±0.1)	65.2 (±1.1)
SciBERT-cls Triplet	66.0 (±0.4)	**26.3** (±1.0)	86.9 (±1.1)	77.0 (±0.2)	68.6 (±0.8)
SciBERT-cls InfoNCE	67.4 (±0.5)	13.1 (±2.2)	86.9 (±1.1)	76.8 (±0.3)	66.2 (±1.1)
SciBERT-cls CoSENT	66.7 (±0.3)	23.8 (±2.1)	87.3 (±1.1)	76.7 (±0.2)	68.4 (±1.0)
SciBERT-cls AnglE	66.7 (±0.2)	14.1 (±2.2)	85.0 (±1.3)	74.0 (±0.1)	65.1 (±1.0)
Triplet Average	66.1 (±0.1)	<u>24.7</u> (±2.4)	87.0 (±0.3)	78.1 (±2.0)	<u>68.4</u> (±0.2)
InfoNCE Average	<u>67.6</u> (±0.3)	14.5 (±1.4)	87.0 (±0.3)	<u>78.3</u> (±2.2)	66.8 (±0.7)
CoSENT Average	66.7 (±0.3)	22.6 (±1.2)	<u>87.5</u> (±0.2)	78.1 (±2.1)	<u>68.4</u> (±0.1)
AnglE Average	67.1 (±0.5)	15.4 (±1.8)	84.9 (±0.1)	74.9 (±1.6)	65.5 (±0.7)

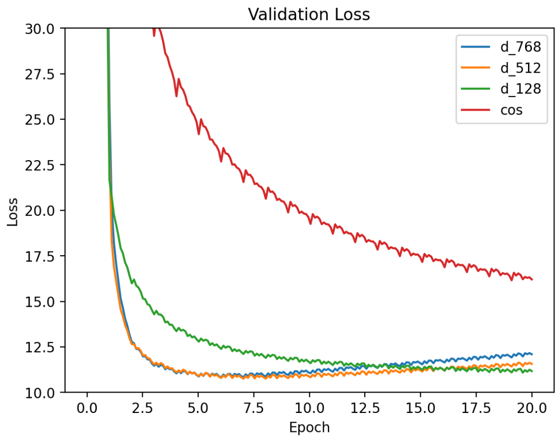
6.3 Evaluation Objective 2: Continuous Co-Citations

The evaluation of the continuous co-citations signal is separated into the evaluation of the graph embeddings and the results of the trained LLM on the SciRepEval benchmark.

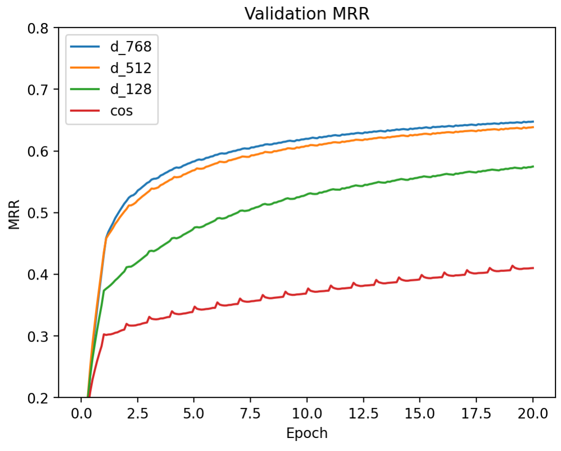
6.3.1 Graph Embedding

The graph embedding model learns the relation within the citation graph and creates the continuous co-citation signal. For validation, a hold-out set of 5% of the training data is used and the metrics (loss, MRR, Hits@1,@10,@50, AUC) are tracked during training. In total four different configurations are tested including the distance measures *cosine similarity* and *dot product*, as well as the dimensions of the graph embeddings (128, 512, 768). Figure 6.2 shows the validation loss and MRR[8]. The validation loss has its minimum value of 10.76 for the model with $dim = 512$, however, it indicates overfitting for the models

[8]All other metrics rank similarly to MRR and can be found in Figure A.1



(a)



(b)

Figure 6.2: Graph Embedding Training - Validation loss and MRR for embedding models trained with distance measure dot product and dimensions 128, 512, and 768, as well as cosine similarity and dimension 768

with dimensions 512 and 768, whereas MRR still improves. The best results for all metrics of each model are listed in Table 6.6. The best scores are highlighted in **bold**. As the model with $dim = 768$ and the *dot product* as distance measure achieved the highest results, its embeddings are used as the continuous co-citation signal.

Table 6.6: Graph Embedding Models - Validation Metrics

Model	**Loss** ↓	**Mean Rank** ↓	**MRR** ↑	**Hits@1** ↑	**Hits@10** ↑	**Hits@50** ↑	**AUC** ↑
Cos, dim=768	16.16	16.53	0.41	0.27	0.75	0.97	0.99
Dot, dim=128	11.09	13.87	0.57	0.41	0.89	0.97	0.99
Dot, dim=512	**10.76**	12.42	0.64	0.48	0.92	0.98	0.99
Dot, dim=768	10.83	**12.32**	**0.65**	**0.49**	**0.92**	**0.98**	**0.99**

6.3.2 Co-Citations

From the graph embeddings, five datasets using different sample difficulties are created and the SciBERT model trained. The sample difficulties $k^+ \in \{25, 200, 500\}$ for positive and $k^- \in \{1000, 3000, 5000\}$ for negative publications are based on the results of SciNCL, to test if similar sample difficulties can be used for the continuous co-citations. The progression of the validation loss is shown in Figure 6.3. The curves between the different datasets behave similarly and it can be observed that the training tends to overfit after the first epoch (550

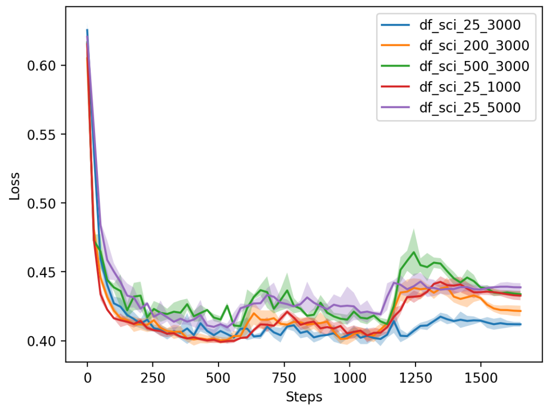


Figure 6.3: Co-Citation Training - Validation loss of SciBERT models trained with continuous co-citation datasets with different sample difficulties

steps). Table 6.7 shows the results on the SciRepEval benchmark for models with the lowest validation loss. The training on the continuous co-citation signal improved performance in all formats compared to the initial SciBERT model, however, the results are inferior to the SPECTER, SciNCL, and GTE model in the *Proximity* and *Ad-Hoc Search* formats. Nonetheless, for the sample difficulties $k^+ = 25, k^- = 5000$ an improvement of 1.4 and 2.9 could be achieved on the *Classification* and *Regression* formats respectively compared to the best baseline model. On average, it achieved the best performance, similar to the results of the GTE model, and improved the score of SciNCL by 0.9% while using only 70k training triplets instead of 684k. This drastically reduces training time (approximately one hour compared to 50 hours with the SPECTER2-base pre-training dataset (~6.2 million triplets).

Table 6.7: SciRepEval benchmark results across four task formats of SciBERT models trained with the continuous co-citation datasets from different sample difficulties

Model	**Classification**	**Regression**	**Proximity**	**Ad-Hoc Search**	**Average**
SciBERT	63.7	23.2	66.1	68.2	57.2
SPECTER	67.5	22.2	85.8	74.9	67.4
SciNCL	67.8	23.9	87.2	77.4	67.8
GTE	67.2 (±0.0)	20.5 (±1.6)	**87.8** (±1.1)	**81.9** (±0.0)	68.7 (±0.8)
CoCite 25-3000	67.4 (±0.4)	25.5 (±0.6)	85.2 (±1.5)	75.8 (±0.5)	67.9 (±0.9)
CoCite 200-3000	68.4 (±0.7)	26.3 (±0.7)	85.3 (±1.4)	75.4 (±0.4)	68.3 (±1.0)
CoCite 500-3000	69.0 (±0.4)	26.7 (±0.6)	85.0 (±1.5)	74.7 (±0.4)	68.3 (±0.9)
CoCite 25-1000	68.0 (±0.9)	26.0 (±0.7)	85.3 (±1.5)	75.2 (±0.5)	68.1 (±1.1)
CoCite 25-5000	**69.2** (±0.6)	**26.8** (±0.8)	85.6 (±1.4)	75.5 (±0.3)	**68.7** (±0.9)

7 Discussion

This thesis explores different contrastive learning loss functions as well as the development of a novel approach to generate embeddings of scientific publications and how they can be evaluated compared to SOTA solutions. Specifically, it addresses two main objectives: comparing the influence of multiple contrastive learning loss functions on the training of LLM and their embeddings of scientific publications, and developing an approach that utilizes a continuous co-citation signal to learn these embeddings.

7.1 Benchmark

To evaluate the models trained for both objectives the SciRepEval benchmark [Sin+23] is employed. It enables the comparison of the performance of embedding models within the scientific domain across multiple task formats, ensuring a comprehensive assessment of their general applicability. However, the benchmark has multiple tasks that focus on biomedical publications introducing a bias, as tasks related to other fields are underrepresented. For instance, two of the *Classification* tasks are concerned with MeSH. Furthermore, the number of tasks within each format differs, and only for the SciDocs Cite, Co-Cite, Co-View, and Co-Read tasks multiple metrics are computed. This attributes disproportionately to the overall benchmark score. These limitations suggest that while the SciRepEval benchmark is valuable, its validity could be enhanced by incorporating a broader range of tasks from other domains and employing the same number of metrics for each task, or weighing them before aggregating an overall score.

7.2 Discussion Objective 1: Contrastive Loss

The results of the first objective (section 6.2) indicate that different contrastive learning loss functions have varying influence on the performance of LLM and their embeddings, particularly when applied to the different task formats. For instance, the models trained using the triplet margin loss yield the best score in the *Regression* format and the CoSENT loss in the *Proximity* format on the SciRepEval benchmark. These findings are consistent

across both models used in the comparison, suggesting that the final scores are robust across different initial model weights and that the choice of loss function is an important factor when optimizing for a specific task format.
Whereas no clear advantage is observed between different pooling strategies, the influence of the PLM selected for training is significant. Although both the GTE and SciBERT models yield a similar overall performance score in the benchmark, the GTE model is superior in the *Ad-Hoc Search* format whereas SciBERT outperformed in the *Regression* format. This highlights the importance of model selection regarding the intended application. To determine if the results can be replicated with other training datasets, model architectures, and benchmarks a similar setup with, for instance, a decoder-only Transformer and the MTEB, which also has different task formats, can be used.
In addition to the benchmark, the convergence of the loss functions during training is compared using the validation loss. However, the loss functions scale differently and the lower validation loss of the GTE model does not result in overall better downstream task performance. Thus, other metrics such as Spearmans's rank correlation must be tracked during training to better compare performance gains over time.

7.3 Discussion Objective 2: Continuous Co-Citations

Regarding the second objective, the results show that the continuous sampling strategy could successfully be transferred from direct to co-citations while also improving overall performance with a smaller number of training samples thus reducing training time. This shows that general model performance can be improved without task specific finetuning. The additional training of a graph embedding model and tuning the sample difficulty adds to the complexity of this approach [Ost+22]. However, the reduced training time alleviates this effect and the dataset with samples from the continuous co-citation signal must be created only once. To further estimate the effectiveness of this approach it can be validated with other LLM architectures.
A limitation compared to the use of a direct citation signal is the size of the co-citation graph, which can become very large. Therefore, only a subset of the co-citations is used based on the query publications present in the SPECTER2-base pre-training data. This changes the learned graph embeddings, where relations between publications can be lost. The resulting effects can be investigated in future studies. Also, the co-citation strength is not considered in the training of the graph embedding model but it could be valuable as it indicates the degree of how much publications are related to each other. The size of the created dataset using the continuous co-citation strategy is relatively small, and the observed overfitting in validation loss indicates that a larger dataset could improve training outcomes. Further,

with the continuous co-citation signal and data from the Semantic Scholar datasets three triplets are generated for each query on average. Increasing the number of triplets and, consequently, the number of negative samples, could enhance model performance [RA23].

8 Conclusion and Outlook

This thesis demonstrated the impact different contrastive learning loss functions can have on the performance of LLMs across various task formats, emphasizing the importance of selecting an appropriate loss function for specific applications. The results also indicate that the selection of the PLM is a critical factor in achieving the best performance, with different models excelling in different task formats.

Moreover, to the insights gained from comparing contrastive learning loss functions, this thesis introduced a novel approach utilizing a continuous co-citation signal for sampling training triplets. The results of the trained model show that this approach improves the overall performance of the embeddings on the SciRepEval benchmark compared to previous SOTA solutions while also reducing the required training time with a smaller number of samples needed.

These findings also highlight several areas for future research, including the extension of the SciRepEval benchmark to incorporate more scientific domains, adjusting the influence of different tasks, and providing tasks with full-text documents for the evaluation of models with a larger context sizes.

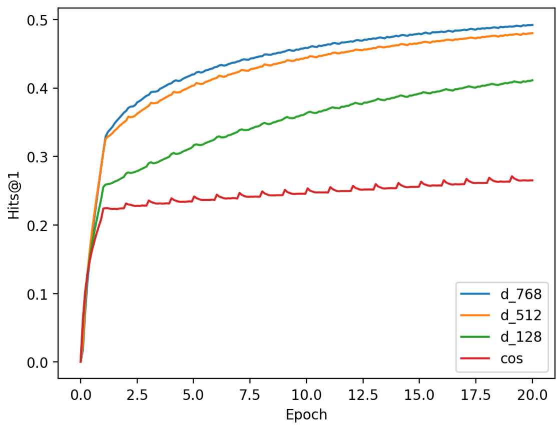
The comparison of contrastive learning loss functions could be conducted within a different setting, utilizing other training data, models, and benchmarks to replicate the results. If similar results are found training methods that consistently increase the performance on diverse tasks formats could be investigated.

As a subset of the co-citation graph was used to generate the training triplets, the impact of different sized graphs as input to the graph embedding model could be analyzed. Similarly, the number of training triplets and the total size of the resulting dataset can be evaluated, decreasing the size for even faster training or increasing it to potentially improve overall performance. Further, the direct and co-citation signals could be combined for a more diverse sampling signal. Other approaches could also use the co-citation strength and extend the co-citations with Citation Proximity Analysis, where publications that are cited together in close proximity indicate a stronger relation.

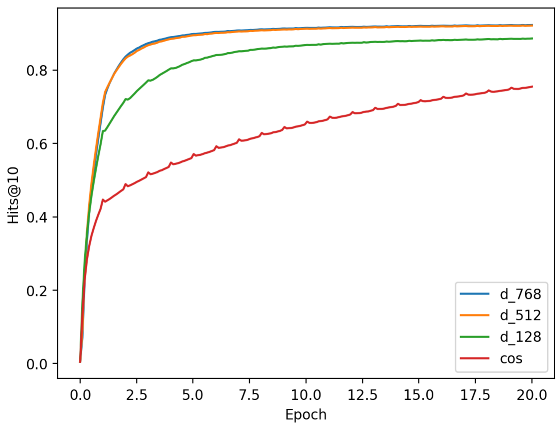
A Appendix

Table A.1: SciRepEval benchmark results across In-Train, Out-of-Train, and SciDocs task splits for GTE and SciBERT models trained with different contrastive loss functions and the continuous co-citation datasets with different sample difficulties. Reported as the average and standard deviation of nine runs. The results show that the best performance could be achieved with the continuous co-citation approach and a sample difficulty of $k^{+} = 25, k^{-} = 5000$. In the SciDocs category, the GTE model trained with the CoSENT loss achieves the best results.

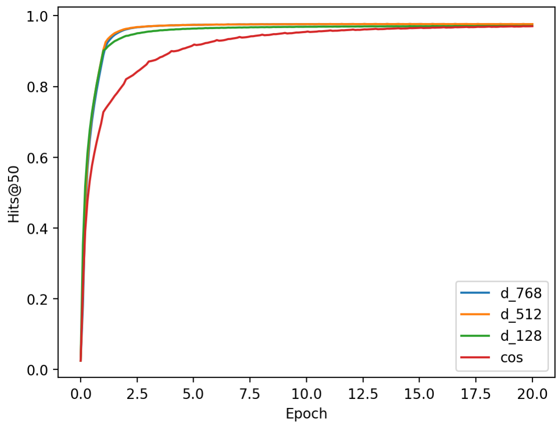
Model	**In-Train**	**Out-of-Train**	**SciDocs**	**Average**
SciBERT	51.5	47.4	69.0	57.2
SPECTER	54.7	51.3	89.1	67.4
SciNCL	55.6	52.6	90.8	67.8
GTE-base	55.4 (±0.4)	52.4 (±0.7)	91.0 (±1.2)	68.7 (±0.8)
GTE-avg Triplet	55.2 (±0.5)	51.5 (±1.0)	91.0 (±1.2)	68.3 (±0.9)
GTE-avg InfoNCE	52.8 (±0.7)	51.4 (±1.3)	91.0 (±1.2)	67.6 (±1.1)
GTE-avg CoSENT	55.3 (±0.5)	51.8 (±1.1)	**91.1** (±1.2)	68.5 (±1.0)
GTE-avg AnglE	52.6 (±0.3)	49.9 (±0.8)	89.0 (±1.4)	66.3 (±0.9)
SciBERT-avg Triplet	55.7 (±0.3)	52.2 (±0.5)	90.3 (±1.3)	68.4 (±0.7)
SciBERT-avg InfoNCE	51.4 (±0.7)	50.3 (±1.0)	90.2 (±1.3)	66.6 (±1.0)
SciBERT-avg CoSENT	54.8 (±0.9)	51.9 (±0.8)	90.8 (±1.2)	68.3 (±1.0)
SciBERT-avg AnglE	50.5 (±0.5)	48.5 (±1.1)	88.8 (±1.5)	65.2 (±1.1)
SciBERT-cls Triplet	55.9 (±0.2)	52.5 (±0.8)	90.3 (±1.3)	68.6 (±0.8)
SciBERT-cls InfoNCE	51.0 (±0.7)	49.5 (±1.2)	90.3 (±1.3)	66.2 (±1.1)
SciBERT-cls CoSENT	54.8 (±1.0)	52.4 (±0.6)	90.7 (±1.2)	68.4 (±1.0)
SciBERT-cls AnglE	50.1 (±0.6)	48.5 (±0.9)	88.9 (±1.4)	65.1 (±1.0)
CoCite 25-3000	55.8 (±0.4)	52.4 (±0.6)	88.7 (±1.6)	67.9 (±0.9)
CoCite 200-3000	56.1 (±0.5)	53.1 (±0.6)	88.9 (±1.6)	68.3 (±1.0)
CoCite 500-3000	56.1 (±0.3)	53.3 (±0.6)	88.8 (±1.6)	68.3 (±0.9)
CoCite 25-1000	55.9 (±0.5)	52.8 (±0.9)	88.8 (±1.7)	68.1 (±1.1)
CoCite 25-5000	**56.6** (±0.3)	**53.5** (±0.7)	89.3 (±1.5)	**68.7** (±0.9)



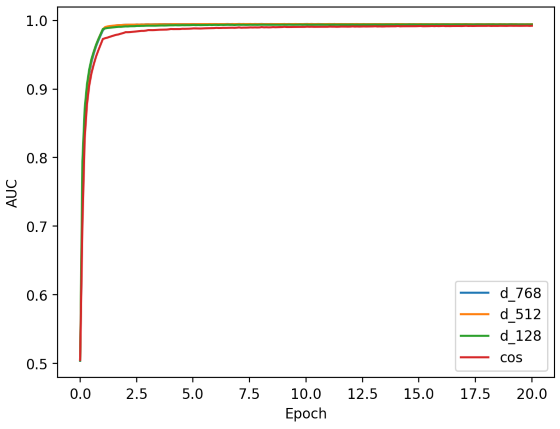
(a) Hits@1



(b) Hits@10



(c) Hits@50



(d) AUC

Figure A.1: Graph-Embedding - Validation metrics Hits@k and AUC of the training for embedding models trained with distance measure dot product and dimensions 128, 512, and 768, as well as cosine similarity and dimension 768. The curves show a similar ranking of model performance across metrics, where embedding models trained with smaller embedding sizes or cosine similarity converge slower. Hits@1 and Hits@10 show that the embedding model trained with dot product and dimension 768 achieves the best performance, while Hits@50 and AUC metrics show less difference between all models, converging with perfect performance.

List of Figures

List of Tables

Bibliography

[Bak18] Amir Bakarov. *A Survey of Word Embeddings Evaluation Methods*. Jan. 21, 2018. arXiv: 1801.09536[cs].

[BCB16] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. May 19, 2016. DOI: 10.48550/arXiv.1409.0473.

[BLC19] Iz Beltagy, Kyle Lo, and Arman Cohan. "SciBERT: A Pretrained Language Model for Scientific Text". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3615–3620. DOI: 10.18653/v1/D19-1371.

[Bow+15] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. "A large annotated corpus for learning natural language inference". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, Sept. 2015, pp. 632–642. DOI: 10.18653/v1/D15-1075.

[BPC20] Iz Beltagy, Matthew E. Peters, and Arman Cohan. *Longformer: The Long-Document Transformer*. Dec. 2, 2020. DOI: 10.48550/arXiv.2004.05150.

[CH89] Kenneth Ward Church and Patrick Hanks. "Word association norms, mutual information, and lexicography". In: *Proceedings of the 27th annual meeting on Association for Computational Linguistics*. June 26, 1989, pp. 76–83. DOI: 10.3115/981623.981633.

[Chi+19] Matteo Chinazzi, Bruno Gonçalves, Qian Zhang, and Alessandro Vespignani. "Mapping the physics research space: a machine learning approach". In: *EPJ Data Science* 8.1 (Dec. 2019). Number: 1 Publisher: SpringerOpen, pp. 1–18. ISSN: 2193-1127. DOI: 10.1140/epjds/s13688-019-0210-z.

[CHK23] Peter Christen, David J. Hand, and Nishadi Kirielle. “A Review of the F-Measure: Its History, Properties, Criticism, and Alternatives”. In: *ACM Comput. Surv.* 56.3 (Oct. 6, 2023), 73:1–73:24. ISSN: 0360-0300. DOI: 10.1145/3606367.

[COF23] Liliane Soares da Costa, Italo L. Oliveira, and Renato Fileto. “Text classification using embeddings: a survey”. In: *Knowledge and Information Systems* 65.7 (July 1, 2023), pp. 2761–2803. ISSN: 0219-3116. DOI: 10.1007/s10115-023-01856-z.

[Coh+20] Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel Weld. “SPECTER: Document-level Representation Learning using Citation-informed Transformers”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. ACL 2020. Online: Association for Computational Linguistics, July 2020, pp. 2270–2282. DOI: 10.18653/v1/2020.acl-main.207.

[CP18] Jose Camacho-Collados and Mohammad Taher Pilehvar. “From Word To Sense Embeddings: A Survey on Vector Representations of Meaning”. In: *Journal of Artificial Intelligence Research* 63 (Dec. 6, 2018), pp. 743–788. ISSN: 1076-9757. DOI: 10.1613/jair.1.11259.

[Cra09] Nick Craswell. “Mean Reciprocal Rank”. In: *Encyclopedia of Database Systems*. Ed. by LING LIU and M. TAMER ÖZSU. Boston, MA: Springer US, 2009, pp. 1703–1703. ISBN: 978-0-387-39940-9. DOI: 10.1007/978-0-387-39940-9_488.

[Dev+19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423.

[Dom16] T. N. Domnina. “A megajournal as a new type of scientific publication”. In: *Scientific and Technical Information Processing* 43.4 (Oct. 1, 2016), pp. 241–250. ISSN: 1934-8118. DOI: 10.3103/S0147688216040079.

[Dou+24] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé

Jégou. *The Faiss library.* Jan. 16, 2024. doi: 10.48550/arXiv.2401.08281. arXiv: 2401.08281[cs].

[Eth19] Kawin Ethayarajh. "How Contextual are Contextualized Word Representations? Comparing the Geometry of BERT, ELMo, and GPT-2 Embeddings". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP).* Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 55–65. doi: 10.18653/v1/D19-1006.

[EW16] Lisa Ehrlinger and Wolfram Wöß. "Towards a Definition of Knowledge Graphs". In: *SEMANTiCS* Vol-1695 (Sept. 2016). issn: 1613-0073.

[Far+15] Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. "Retrofitting Word Vectors to Semantic Lexicons". In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.* Denver, Colorado: Association for Computational Linguistics, 2015, pp. 1606–1615. doi: 10.3115/v1/N15-1184.

[Far+16] Manaal Faruqui, Yulia Tsvetkov, Pushpendre Rastogi, and Chris Dyer. "Problems With Evaluation of Word Embeddings Using Word Similarity Tasks". In: *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP.* RepEval 2016. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 30–35. doi: 10.18653/v1/W16-2506.

[Fre84] Gottlob Frege. "Die Grundlagen der Arithmetik. Eine logisch mathematische Untersuchung uber den Begriff der Zahl". In: (1884). Publisher: Breslau: Wilhelm Koebner.

[Gao+19] Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. *Representation Degeneration Problem in Training Natural Language Generation Models.* July 27, 2019. doi: 10.48550/arXiv.1907.12009.

[Gur+20] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. "Don't Stop Pretraining: Adapt Language Models to Domains and Tasks". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.* ACL 2020. Online: Association for Computational Linguistics, July 2020, pp. 8342–8360. doi: 10.18653/v1/2020.acl-main.740.

[GYC21] Tianyu Gao, Xingcheng Yao, and Danqi Chen. “SimCSE: Simple Contrastive Learning of Sentence Embeddings”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021, pp. 6894–6910. DOI: 10.18653/v1/2021.emnlp-main.552.

[Har70] Zellig S. Harris. “Distributional Structure”. In: *Papers in Structural and Transformational Linguistics*. Formal Linguistics Series. Dordrecht: Springer Netherlands, 1970, pp. 775–794. ISBN: 978-94-017-6059-1. DOI: 10.1007/978-94-017-6059-1_36.

[He+16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[Hei+22] Florian Heimerl, Christoph Kralj, Torsten Möller, and Michael Gleicher. “embComp: Visual Interactive Comparison of Vector Embeddings”. In: *IEEE Transactions on Visualization and Computer Graphics* 28.8 (Aug. 2022), pp. 2953–2969. ISSN: 1941-0506. DOI: 10.1109/TVCG.2020.3045918.

[Hou+19] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. *Parameter-Efficient Transfer Learning for NLP*. June 13, 2019. arXiv: 1902.00751[cs, stat].

[Hsi+08] Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S. Sathiya Keerthi, and S. Sundararajan. “A dual coordinate descent method for large-scale linear SVM”. In: *Proceedings of the 25th international conference on Machine learning - ICML ’08*. Helsinki, Finland: ACM Press, 2008, pp. 408–415. ISBN: 978-1-60558-205-4. DOI: 10.1145/1390156.1390208.

[Hua+24] Xiang Huang, Hao Peng, Dongcheng Zou, Zhiwei Liu, Jianxin Li, Kay Liu, Jia Wu, Jianlin Su, and Philip S. Yu. “CoSENT: Consistent Sentence Embedding via Similarity Ranking”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 32 (2024), pp. 2800–2813. ISSN: 2329-9304. DOI: 10.1109/TASLP.2024.3402087.

[JDJ21] Jeff Johnson, Matthijs Douze, and Hervé Jégou. “Billion-Scale Similarity Search with GPUs”. In: *IEEE Transactions on Big Data* 7.3 (July 2021), pp. 535–547. ISSN: 2332-7790. DOI: 10.1109/TBDATA.2019.2921572.

[Kas+22] Jungo Kasai, Keisuke Sakaguchi, Ronan Le Bras, Lavinia Dunagan, Jacob Morrison, Alexander Fabbri, Yejin Choi, and Noah A. Smith. “Bidimensional Leaderboards: Generate and Evaluate Language Hand in Hand”. In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, July 2022, pp. 3540–3557. DOI: 10.18653/v1/2022.naacl-main.259.

[KEN38] M. G. KENDALL. “A NEW MEASURE OF RANK CORRELATION”. In: *Biometrika* 30.1 (June 1, 1938), pp. 81–93. ISSN: 0006-3444. DOI: 10.1093/biomet/30.1-2.81.

[Kes+19] Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. *CTRL: A Conditional Transformer Language Model for Controllable Generation*. Sept. 20, 2019. DOI: 10.48550/arXiv.1909.05858.

[Kes63] M. M. Kessler. “Bibliographic coupling between scientific papers”. In: *American Documentation* 14.1 (1963), pp. 10–25. ISSN: 1936-6108. DOI: 10.1002/asi.5090140103.

[Kin+23] Rodney Kinney et al. *The Semantic Scholar Open Data Platform*. Jan. 24, 2023. DOI: 10.48550/arXiv.2301.10140.

[Lak77] George Lakoff. “Linguistic gestalts”. In: *Papers from the... Regional Meeting. Chicago Ling. Soc. Chicago, Ill.* Vol. 13. 1977, pp. 236–287.

[Ler+19] Adam Lerer, Ledell Wu, Jiajun Shen, Timothee Lacroix, Luca Wehrstedt, Abhijit Bose, and Alex Peysakhovich. *PyTorch-BigGraph: A Large-scale Graph Embedding System*. Apr. 9, 2019. DOI: 10.48550/arXiv.1903.12287.

[LH18] Ilya Loshchilov and Frank Hutter. “Decoupled Weight Decay Regularization”. In: International Conference on Learning Representations. Sept. 27, 2018. DOI: 10.48550/arXiv.1711.05101.

[Lho+21] Quentin Lhoest et al. “Datasets: A Community Library for Natural Language Processing”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 175–184. DOI: 10.18653/v1/2021.emnlp-demo.21.

[Li+20] Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. “On the Sentence Embeddings from Pre-trained Language Models”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language*

Processing (EMNLP). Online: Association for Computational Linguistics, Nov. 2020, pp. 9119–9130. DOI: 10.18653/v1/2020.emnlp-main.733.

[Li+23] Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. *Towards General Text Embeddings with Multi-stage Contrastive Learning*. Aug. 6, 2023. arXiv: 2308.03281[cs].

[LI10] Peder Olesen Larsen and Markus von Ins. "The rate of growth in scientific publication and the decline in coverage provided by Science Citation Index". In: *Scientometrics* 84.3 (Sept. 1, 2010), pp. 575–603. ISSN: 1588-2861. DOI: 10.1007/s11192-010-0202-z.

[Liu+19] Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. "Linguistic Knowledge and Transferability of Contextual Representations". In: *Proceedings of the 2019 Conference of the North*. Minneapolis, Minnesota: Association for Computational Linguistics, 2019, pp. 1073–1094. DOI: 10.18653/v1/N19-1112.

[LL23] Xianming Li and Jing Li. *AnglE-optimized Text Embeddings*. Nov. 8, 2023. DOI: 10.48550/arXiv.2309.12871.

[LLS20] Zhiyuan Liu, Yankai Lin, and Maosong Sun. *Representation Learning for Natural Language Processing*. Singapore: Springer Nature Singapore, 2020. ISBN: 9789811555725 9789811555732. DOI: 10.1007/978-981-15-5573-2.

[LM14] Quoc Le and Tomas Mikolov. "Distributed Representations of Sentences and Documents". In: *Proceedings of the 31st International Conference on Machine Learning*. ISSN: 1938-7228. PMLR, June 18, 2014, pp. 1188–1196.

[Lo+20] Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. "S2ORC: The Semantic Scholar Open Research Corpus". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, 2020, pp. 4969–4983. DOI: 10.18653/v1/2020.acl-main.447.

[MB05] Frederic Morin and Yoshua Bengio. "Hierarchical Probabilistic Neural Network Language Model". In: *International Workshop on Artificial Intelligence and Statistics*. ISSN: 2640-3498. PMLR, Jan. 6, 2005, pp. 246–252.

[MCH22] Sheshera Mysore, Arman Cohan, and Tom Hope. "Multi-Vector Models with Textual Guidance for Fine-Grained Scientific Document Similarity". In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, July 2022, pp. 4453–4470. DOI: 10.18653/v1/2022.naacl-main.331.

[Mer73] Robert K. Merton. *The Sociology of Science: Theoretical and Empirical Investigations*. University of Chicago Press, 1973. 639 pp. ISBN: 978-0-226-52092-6.

[Mie+21] Sabrina J. Mielke et al. *Between words and characters: A Brief History of Open-Vocabulary Modeling and Tokenization in NLP*. Dec. 20, 2021. arXiv: 2112.10508[cs].

[Mik+13a] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. *Efficient Estimation of Word Representations in Vector Space*. Sept. 6, 2013. DOI: 10.48550/arXiv.1301.3781.

[Mik+13b] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. "Distributed Representations of Words and Phrases and their Compositionality". In: *Advances in Neural Information Processing Systems*. Vol. 26. Curran Associates, Inc., 2013.

[Mil95] George A. Miller. "WordNet: a lexical database for English". In: *Communications of the ACM* 38.11 (Nov. 1, 1995), pp. 39–41. ISSN: 0001-0782. DOI: 10.1145/219717.219748.

[Min+24] Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. *Large Language Models: A Survey*. Feb. 20, 2024. DOI: 10.48550/arXiv.2402.06196.

[Mis+22] Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. "Cross-Task Generalization via Natural Language Crowdsourcing Instructions". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, 2022, pp. 3470–3487. DOI: 10.18653/v1/2022.acl-long.244.

[MK21] Bethany A. Myers and Katherine L. Kahn. "Practical publication metrics for academics". In: *Clinical and Translational Science* 14.5 (Sept. 2021), pp. 1705–1712. ISSN: 1752-8054. DOI: 10.1111/cts.13067.

[ML10] Jeff Mitchell and Mirella Lapata. "Composition in Distributional Models of Semantics". In: *Cognitive Science* 34.8 (2010), pp. 1388–1429. ISSN: 1551-6709. DOI: 10.1111/j.1551-6709.2010.01106.x.

[MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Higher Education from Cambridge University Press. July 7, 2008. DOI: 10.1017/CBO9780511809071.

[Mue+23] Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. "MTEB: Massive Text Embedding Benchmark". In: *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. Dubrovnik, Croatia: Association for Computational Linguistics, May 2023, pp. 2014–2037. DOI: 10.18653/v1/2023.eacl-main.148.

[Mus+21] Ghulam Mustafa, Muhammad Usman, Lisu Yu, Muhammad Tanvir Afzal, Muhammad Sulaiman, and Abdul Shahid. "Multi-label classification of research articles using Word2Vec and identification of similarity threshold". In: *Scientific Reports* 11.1 (Nov. 9, 2021), p. 21900. ISSN: 2045-2322. DOI: 10.1038/s41598-021-01460-7.

[Nav+23] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. *A Comprehensive Overview of Large Language Models*. Nov. 23, 2023. DOI: 10.48550/arXiv.2307.06435.

[Ni+22a] Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. "Sentence-T5: Scalable Sentence Encoders from Pre-trained Text-to-Text Models". In: *Findings of the Association for Computational Linguistics: ACL 2022*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 1864–1874. DOI: 10.18653/v1/2022.findings-acl.146.

[Ni+22b] Jianmo Ni et al. "Large Dual Encoders Are Generalizable Retrievers". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 9844–9855. DOI: 10.18653/v1/2022.emnlp-main.669.

[Nic+21] Josh M. Nicholson, Milo Mordaunt, Patrice Lopez, Ashish Uppala, Domenic Rosati, Neves P. Rodrigues, Peter Grabitz, and Sean C. Rife. "scite: A smart citation index that displays the context of citations and classifies their intent using deep learning". In: *Quantitative Science Studies* 2.3 (Nov. 5, 2021), pp. 882–898. ISSN: 2641-3337. DOI: 10.1162/qss_a_00146.

[OB21] Juri Opitz and Sebastian Burst. *Macro F1 and Macro F1*. Feb. 8, 2021. arXiv: 1911.03347[cs,stat].

[OLV19] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. *Representation Learning with Contrastive Predictive Coding.* Jan. 22, 2019. DOI: 10.48550/arXiv.1807.03748.

[Ost+22] Malte Ostendorff, Nils Rethmeier, Isabelle Augenstein, Bela Gipp, and Georg Rehm. "Neighborhood Contrastive Learning for Scientific Document Representations with Citation Embeddings". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing.* Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 11670–11688. DOI: 10.18653/v1/2022.emnlp-main.802.

[Ost23] Malte Ostendorff. "Aspect-based Document Similarity for Literature Recommender Systems". doctoralThesis. June 2023. DOI: 10.53846/goediss-9937.

[Par95] Barbara Partee. "Lexical semantics and compositionality". In: *An invitation to cognitive science: Language* 1 (1995). Publisher: MIT Press Cambridge, MA: pp. 311–360.

[PC20] Mohammad Taher Pilehvar and Jose Camacho-Collados. *Embeddings in Natural Language Processing: Theory and Advances in Vector Representations of Meaning.* Morgan & Claypool Publishers, Nov. 13, 2020. 177 pp. ISBN: 978-1-63639-022-2.

[Pet+18] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. "Deep Contextualized Word Representations". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers).* New Orleans, Louisiana: Association for Computational Linguistics, 2018, pp. 2227–2237. DOI: 10.18653/v1/N18-1202.

[Pet+19] Alexander M. Petersen, Raj K. Pan, Fabio Pammolli, and Santo Fortunato. "Methods to account for citation inflation in research evaluation". In: *Research Policy* 48.7 (Sept. 1, 2019), pp. 1855–1865. ISSN: 0048-7333. DOI: 10.1016/j.respol.2019.04.009.

[Pet20] Eugenio Petrovich. *Science mapping.* 2020. URL: https://www.isko.org/cyclo/science_mapping (visited on 11/27/2023).

[Puk11] Llukan Puka. "Kendall's Tau". In: *International Encyclopedia of Statistical Science.* Ed. by Miodrag Lovric. Berlin, Heidelberg: Springer, 2011, pp. 713–715. ISBN: 978-3-642-04898-2. DOI: 10.1007/978-3-642-04898-2_324.

[PVS22] Hyunji Park, Yogarshi Vyas, and Kashif Shah. “Efficient Classification of Long Documents Using Transformers”. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 702–709. DOI: 10.18653/v1/2022.acl-short.79.

[RA23] Nils Rethmeier and Isabelle Augenstein. “A Primer on Contrastive Pretraining in Language Processing: Methods, Lessons Learned, and Perspectives”. In: *ACM Computing Surveys* 55.10 (Feb. 2, 2023), 203:1–203:17. ISSN: 0360-0300. DOI: 10.1145/3561970.

[RB21] Abigail Rai and Samarjeet Borah. “Study of Various Methods for Tokenization”. In: *Applications of Internet of Things*. Ed. by Jyotsna K. Mandal, Somnath Mukhopadhyay, and Alak Roy. Vol. 137. Series Title: Lecture Notes in Networks and Systems. Singapore: Springer Singapore, 2021, pp. 193–200. ISBN: 9789811561979 9789811561986. DOI: 10.1007/978-981-15-6198-6_18.

[RG19] Nils Reimers and Iryna Gurevych. “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3982–3992. DOI: 10.18653/v1/D19-1410.

[Rob+20] Joshua David Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. “Contrastive Learning with Hard Negative Samples”. In: International Conference on Learning Representations. Oct. 2, 2020.

[Sau+19] Nikunj Saunshi, Orestis Plevrakis, Sanjeev Arora, Mikhail Khodak, and Hrishikesh Khandeparkar. “A Theoretical Analysis of Contrastive Unsupervised Representation Learning”. In: *Proceedings of the 36th International Conference on Machine Learning*. International Conference on Machine Learning. ISSN: 2640-3498. PMLR, May 24, 2019, pp. 5628–5637.

[Sch+15] Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. “Evaluation methods for unsupervised word embeddings”. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, 2015, pp. 298–307. DOI: 10.18653/v1/D15-1036.

[SH10] Andrea Sims and Martin Haspelmath. *Understanding Morphology, 2ed.* May 23, 2010. ISBN: 978-0-340-95001-2. DOI: 10.4324/9780203776506.

[Shy+19] Vikram Shyam et al. "PeTaL (Periodic Table of Life) and Physiomimetics". In: *Designs* 3.3 (Sept. 2019). Number: 3 Publisher: Multidisciplinary Digital Publishing Institute, p. 43. ISSN: 2411-9660. DOI: 10.3390/designs3030043.

[Sin+23] Amanpreet Singh, Mike D'Arcy, Arman Cohan, Doug Downey, and Sergey Feldman. "SciRepEval: A Multi-Format Benchmark for Scientific Document Representations". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing.* Singapore: Association for Computational Linguistics, Dec. 2023, pp. 5548–5566. DOI: 10.18653/v1/2023.emnlp-main.338.

[SKP15] Florian Schroff, Dmitry Kalenichenko, and James Philbin. "FaceNet: A unified embedding for face recognition and clustering". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* ISSN: 1063-6919. June 2015, pp. 815–823. DOI: 10.1109/CVPR.2015.7298682.

[SL20] Ye Sun and Vito Latora. "The evolution of knowledge within and across fields in modern physics". In: *Scientific Reports* 10.1 (July 21, 2020), p. 12097. ISSN: 2045-2322. DOI: 10.1038/s41598-020-68774-w.

[Sma73] Henry Small. "Co-citation in the scientific literature: A new measure of the relationship between two documents". In: *Journal of the American Society for Information Science* 24.4 (1973), pp. 265–269. ISSN: 1097-4571. DOI: 10.1002/asi.4630240406.

[SPA72] KAREN SPARCK JONES. "A STATISTICAL INTERPRETATION OF TERM SPECIFICITY AND ITS APPLICATION IN RETRIEVAL". In: *Journal of Documentation* 28.1 (Jan. 1, 1972). Publisher: MCB UP Ltd, pp. 11–21. ISSN: 0022-0418. DOI: 10.1108/eb026526.

[Su+21] Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. *Whitening Sentence Representations for Better Semantics and Faster Retrieval.* Mar. 28, 2021. arXiv: 2103.15316[cs].

[Su+23a] Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. "One Embedder, Any Task: Instruction-Finetuned Text Embeddings". In: *Findings of the Association for Computational Linguistics: ACL 2023.* Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 1102–1121. DOI: 10.18653/v1/2023.findings-acl.71.

[Su+23b] Hongjin Su et al. "Selective Annotation Makes Language Models Better Few-Shot Learners". In: *International Conference on Learning Representations.* 2023.

[Sub+21] Shivashankar Subramanian, Daniel King, Doug Downey, and Sergey Feldman. "S2AND: A Benchmark and Evaluation System for Author Name Disambiguation". In: *2021 ACM/IEEE Joint Conference on Digital Libraries (JCDL).* Sept. 2021, pp. 170–179. DOI: 10.1109/JCDL52503.2021.00029.

[Sun+18] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. "RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space". In: International Conference on Learning Representations. Sept. 27, 2018.

[SWY75] G. Salton, A. Wong, and C. S. Yang. "A vector space model for automatic indexing". In: *Communications of the ACM* 18.11 (Nov. 1975), pp. 613–620. ISSN: 0001-0782, 1557-7317. DOI: 10.1145/361219.361220.

[TB18] Iman Tahamtan and Lutz Bornmann. "Core elements in the process of citing publications: Conceptual overview of the literature". In: *Journal of Informetrics* 12.1 (Feb. 1, 2018), pp. 203–216. ISSN: 1751-1577. DOI: 10.1016/j.joi.2018.01.002.

[tea23] The pandas development team. *pandas-dev/pandas: Pandas.* Version v2.0.3. June 28, 2023. DOI: 10.5281/zenodo.8092754.

[Tia+20] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. "What Makes for Good Views for Contrastive Learning?" In: *Advances in Neural Information Processing Systems.* Vol. 33. Curran Associates, Inc., 2020, pp. 6827–6839.

[TP10] P. D. Turney and P. Pantel. "From Frequency to Meaning: Vector Space Models of Semantics". In: *Journal of Artificial Intelligence Research* 37 (Feb. 27, 2010), pp. 141–188. ISSN: 1076-9757. DOI: 10.1613/jair.2934.

[Vas+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. "Attention is All you Need". In: *Advances in Neural Information Processing Systems.* Vol. 30. Curran Associates, Inc., 2017.

[Wan+13] Yining Wang, Liwei Wang, Yuanzhi Li, Di He, and Tie-Yan Liu. "A Theoretical Analysis of NDCG Type Ranking Measures". In: *Proceedings of the 26th Annual Conference on Learning Theory.* ISSN: 1938-7228. PMLR, June 13, 2013, pp. 25–54.

[Wan+22] Yizhong Wang et al. “Super-NaturalInstructions: Generalization via Declarative Instructions on 1600+ NLP Tasks”. In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 5085–5109. DOI: 10.18653/v1/2022.emnlp-main.340.

[WI20] Tongzhou Wang and Phillip Isola. “Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere”. In: *Proceedings of the 37th International Conference on Machine Learning*. International Conference on Machine Learning. ISSN: 2640-3498. PMLR, Nov. 21, 2020, pp. 9929–9939.

[WNB18] Adina Williams, Nikita Nangia, and Samuel Bowman. “A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 1112–1122. DOI: 10.18653/v1/N18-1101.

[Wol+20] Thomas Wolf et al. “Transformers: State-of-the-Art Natural Language Processing”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. DOI: 10.18653/v1/2020.emnlp-demos.6.

[Wu+16] Yonghui Wu et al. *Google’s Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*. Oct. 8, 2016. arXiv: 1609.08144[cs].

[WZJ20] Shirui Wang, Wenan Zhou, and Chao Jiang. “A survey of word embeddings based on deep learning”. In: *Computing* 102.3 (Mar. 1, 2020), pp. 717–740. ISSN: 1436-5057. DOI: 10.1007/s00607-019-00768-7.

[YJG03] Andy B. Yoo, Morris A. Jette, and Mark Grondona. “SLURM: Simple Linux Utility for Resource Management”. In: *Job Scheduling Strategies for Parallel Processing*. Berlin, Heidelberg: Springer, 2003, pp. 44–60. ISBN: 978-3-540-39727-4. DOI: 10.1007/10968987_3.

[YM20] Moslem Yousefi and Fatemeh Mardian. “Analyzing Meaning: An Introduction to Semantics and Pragmatics: by Paul R. Kroeger, Berlin: Language Science Press, 2018, xiv+482 pp.” In: *Australian Journal of Linguistics* 40.2 (Apr. 2, 2020), pp. 270–272. issn: 0726-8602, 1469-2996. doi: 10.1080/07268602.2019.1680090.

[YS16] Yadollah Yaghoobzadeh and Hinrich Schütze. “Intrinsic Subspace Evaluation of Word Embedding Representations”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. ACL 2016. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 236–246. doi: 10.18653/v1/P16-1023.

[Zha+21] Yu Zhang, Zhihong Shen, Yuxiao Dong, Kuansan Wang, and Jiawei Han. “MATCH: Metadata-Aware Text Classification in A Large Hierarchy”. In: *Proceedings of the Web Conference 2021*. New York, NY, USA: Association for Computing Machinery, June 3, 2021, pp. 3246–3257. isbn: 978-1-4503-8312-7. doi: 10.1145/3442381.3449979.

[Zho+21] Ruiqi Zhong, Kristy Lee, Zheng Zhang, and Dan Klein. “Adapting Language Models for Zero-shot Learning by Meta-tuning on Dataset and Prompt Collections”. In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021, pp. 2856–2878. doi: 10.18653/v1/2021.findings-emnlp.244.

Statement of Originality

I hereby confirm in lieu of oath, that I have written the accompanying thesis by myself, without contributions from any sources other than those cited in the text and acknowledgments. I certify, that I have followed the general principles of scientific work and publications as written in the guidelines of good research of the Carl von Ossietzky University of Oldenburg. This work has not yet been submitted to any examination office in the same or similar form.

Oldenburg, 3.9.2024 Melchior Schilewa

Oldenburg, September 3, 2024 Melchior Schilewa