
Zhuoyi Peng Yi Yang
The Hong Kong University of Science and Technology
zpengaf@connect.ust.hk imyiyang@ust.hk

Abstract
We study the patent phrase similarity inference task, which measures the semantic similarity between two patent phrases. As patent documents employ legal and highly technical language, existing semantic textual similarity methods that use localized contextual information do not perform satisfactorily in inferring patent phrase similarity. To address this, we introduce a graph-augmented approach to amplify the global contextual information of the patent phrases. For each patent phrase, we construct a phrase graph that links to its focal patents and a list of patents that are either cited by or cite these focal patents. The augmented phrase embedding is then derived from combining its localized contextual embedding with its global embedding within the phrase graph. We further propose a self-supervised learning objective that capitalizes on the retrieved topology to refine both the contextualized embedding and the graph parameters in an end-to-end manner. Experimental results from a unique patent phrase similarity dataset demonstrate that our approach significantly enhances the representation of patent phrases, resulting in marked improvements in similarity inference in a self-supervised fashion. Substantial improvements are also observed in the supervised setting, underscoring the potential benefits of leveraging retrieved phrase graph augmentation.

1 Introduction
Patents are pivotal to the landscape of innovation, safeguarding novel ideas and fostering technological advancements (Hasan and Tucci, 2010). Consequently, understanding patent phrase similarity becomes essential, offering insights into the nuances of intellectual property and aiding in the patent analytics applications, such as patent document categorization, patent retrieval, patent litigation analysis and so on (Tang et al., 2020; Mase et al., 2005; Hall and Ziedonis, 2007). For example, the United States Patent and Trademark Office (USPTO), in its mission to evaluate and grant patents, could leverage the patent phrase similarity task to streamline the examination process, identify prior art more efficiently, and ensure the distinctiveness of newly filed patent applications (Gao et al., 2022).

Patent documents employ legal and highly technical language, featuring context-dependent terms that can deviate significantly from colloquial usage and may vary between different documents. For instance, while a common term like "smartphone" might be easily recognized in everyday language, in patent documents it might be referred to as a "handheld electronic communication device", "mobile telecommunication apparatus", or even a "portable digital assistant with wireless capabilities". Consequently, prevailing semantic textual similarity approaches, such as Sentence-BERT (Reimers and Gurevych, 2019) or SimCSE (Gao et al., 2021), which focus on general text and rely on localized contextualized information for text representation, fall short in inferring patent phrase similarity. Additionally, obtaining a substantial collection of annotations from experts for supervised training presents significant challenges: the process is not only costly but also demands in-depth domain knowledge of the patent innovation landscape. The intricate interplay between technically nuanced terms and the scarcity of labels renders the task especially challenging. A significant research gap persists: How can one effectively infer phrase similarity within the complex language of patents, especially in the absence of training labels?

To address this challenge, we introduce a retrieval-based graph augmentation method to effectively capture phrase representations. Our approach is inspired by the strategies employed by real-world patent experts. For a given patent phrase, we begin by extracting a subgraph from the vast patent universe, such as those registered in the USPTO patent database. The derived subgraph
comprises two node types: the phrase node and the patent node. A connection exists between a phrase node and a patent node if the phrase is present in the patent, and patent nodes interlink based on citation relationships. Hence, for a patent phrase such as "handheld electronic communication device", this subgraph might encompass related patents related to "portable telecommunication gadget". Concurrently, it could also surface interrelated phrases such as "wireless signal transceiver" and "handheld digital communicator". Such related patents and phrases provide a broader context, enabling a deeper understanding of the focal phrase. The information from its focal patent serves as the local context, whereas the extracted phrase graph offers a global context. This extracted phrase graph is subsequently processed through a graph attention network (GAT) (Veličković et al., 2018) to obtain its representative embedding. The final contextualized embedding of a patent phrase is a combination of its textual contextualized embedding and its associated phrase graph embedding. To address the issue of label scarcity, we utilize a self-supervised learning objective that capitalizes on the phrase graph’s topology, facilitating the training of both the textual contextualized embedding and the graph learning parameters in an end-to-end manner.

We evaluate the proposed Retrieval Augmented Patent Phrase Similarity (RA-Sim) on a large patent phrase similarity dataset (Aslanyan and Wetherbee, 2022). In comparison with existing textual semantic similarity approaches—such as word embeddings (Word2Vec (Mikolov et al., 2013), Glove (Pennington et al., 2014), FastText (Bojanowski et al., 2017)), contextualized embeddings (BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019)), and semantic similarity embeddings (Sentence-BERT (Reimers and Gurevych, 2019), Contreever (Izacard et al., 2022), SimCSE (Gao et al., 2021))—our RA-Sim method achieves substantial improvements on inferring patent phrase similarities in a self-supervised manner. Ablation studies and additional analyses further elucidate how RA-Sim enhances the patent phrase similarity inference task. Moreover, RA-Sim, when evaluated in supervised learning setting, consistently outperforms state-of-the-art methods. This highlights the potential advantages of incorporating retrieved graph for contextualized embedding augmentation. We release the codes for RA-Sim, enabling innovation and patent research scholars and practitioners to integrate it into their analytics pipeline.

2 Related Work

Our work is related to several lines of literature.

Semantic Textual Similarity. Semantic Textual Similarity (STS) is a classic natural language processing task. Word2Vec (Mikolov et al., 2013), FastText (Bojanowski et al., 2017) train word-level embeddings unsupervisedly, which can be used for similarity inference. BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) pretrain language model, which provide powerful embeddings for similarity inference. Additionally, SBERT (Reimers and Gurevych, 2019) leverages dual-tower architecture to enhance sentence-level embeddings. SimCSE (Gao et al., 2021) proposes a self-supervised loss for similarity, by taking different views of the same sentence as contrastive positive (Jaiswal et al., 2020). Usually, their embeddings are evaluated on Semantic Textual Similarity datasets (Agirre et al., 2012, 2013, 2014, 2015, 2016; Cer et al., 2017). Recently, domain-specific semantic similarity tasks have drawn attention due to the potentially unique characteristics of domain language for similarity (Liu et al., 2024). Patent phrase similarity inference also imposes some unique challenges. Firstly, methods that perform well for general text, such as SBERT and SimCSE, and even domain-specific model PatentBERT (Srebrovic and Yonamine, 2020), perform poorly due to the interplay of technical language and phrase brevity, implying a need for domain-specific modelling. Moreover, labelling patent phrase similarity requires a high expertise in understanding the patent landscape, which is often lacking in practice, resulting in label absence for model training.

Retrieval-based NLP Generation. A growing body of research incorporates a retrieval system for NLP generation tasks (Asai et al., 2023; Yogatama et al., 2021; Borgeaud et al., 2022; Zhong et al., 2022; Tang and Yang, 2024). Specific applications include question answering (Kumar et al., 2016; de Masson D’Autume et al., 2019; Chen et al., 2023), dialogue (Fan et al., 2021) and other traditional NLP tasks (Lewis et al., 2020). No prior work has utilized a retrieval module for the tasks in patent domain.

Graph Learning for Patent Analysis. Graph neural networks (Kipf and Welling, 2017; Hamilton et al., 2017; Veličković et al., 2018) have been used for analyzing patents (Tang et al., 2020; Fang et al., 2021; Siddharth et al., 2022; Zuo et al., 2022)
or other technical text like customer requirements (Shbita et al., 2023). The potential of graph is still underexplored in patent phrase similarity inference.

This paper presents the first self-supervised framework for patent phrase similarity inference, by retrieving a domain graph to amplify the global contextual information for patent phrases.

3 Retrieval Augmented Patent Phrase Similarity

Patent phrases typically employ legal and highly technical language and may deviate significantly from colloquial usage. Moreover, patent phrases are typically short, containing only a few tokens. This brevity makes it even more challenging to meaningfully represent them due to the lack of contextual information. For example, consider the technical phrases "acid absorption" and "chemically soaked" from the Patent Phrase Similarity dataset released by Google (Aslanyan and Wetherbee, 2022). Even though these two phrases don’t share any words, patent experts rate them as domain-related, assigning a similarity score of 0.25 on a scale of [0,1]. Conversely, the phrases "acid absorption" and "acid reflux" are rated as "not related" with a similarity score of 0.1.

**Design Rationale:** To enhance patent phrase similarity inference, we propose to integrate additional contextual information to bolster phrase representations. There are two primary avenues to augment this contextual information. First, the patents in which a phrase appears can offer crucial context to elucidate the phrase’s meaning. Second, the citation network linking the patents can further refine the representations of these patents, which, in turn, enriches the contextual understanding of the associated phrases. Building on this premise, we introduce a retrieval-augmented patent phrase similarity method, termed RA-Sim. The design framework of RA-Sim is illustrated in Figure 1.

3.1 Constructing Patent-Phrase Universe

We begin with a patent database that includes granted patents along with their citation information. In this study, our emphasis is on the U.S. granted patent data from USPTO. It’s worth noting that while our focus is on U.S. patents, RA-Sim is not exclusive to them and can be generalized to patents granted in other jurisdictions as well.

Formally, we have a set of $N$ patents $\mathcal{V}$. For the $i$-th patent $v_i \in \mathcal{V}$, its text (e.g., patent abstract) is denoted as $d_i$. We represent $\mathcal{E} \in \mathbb{R}^{N \times N}$ as the adjacency matrix indicating citation relations between patents. Regarding the citation relation, $\mathcal{E}_{ij} = 1$ if patent $v_i$ cites patent $v_j$, or 0 otherwise. In our context, the USPTO patent dataset contains 7,619,250 utility patents with 102,674,056 citations.

The original patent database does not include a specific list of phrases associated with each patent. Given our emphasis on patent phrase similarity, we augment the patent citation graph with phrase nodes. For the sake of efficiency, we employ the Rapid Automatic Keyword Extraction (RAKE) algorithm (Rose et al., 2010) to extract key phrases from the patent set $\mathcal{V}$. This process yields a set of $M$ phrase $\mathcal{U}$. We present some phrase examples in Appendix E.1.

After obtaining the patent set $\mathcal{V}$ and phrase set $\mathcal{U}$, we establish relations between these two sets. Specifically, for a given patent phrase $u \in \mathcal{U}$, we utilize a retrieval algorithm to fetch the top- $k$ patents from the patent set $\mathcal{V}$. Following prior work (Zhong et al., 2022), we adopt BM25 as the retrieval algorithm, due to efficiency and capability.

We denote $\mathcal{E}^r \in \mathbb{R}^{N \times M}$ as the adjacency matrix, indicating the relationships where a patent is retrieved based on a patent phrase query. The retrieval examples are shown in Appendix E.2.

In the patent-phrase universe, there are two types of nodes: the phrase node and the patent node. A connection is established between a phrase node and a patent node if the phrase appears in the patent. Additionally, patent nodes are interconnected based on their citation relationships. These associated patents and phrases offer a broader context, enabling a deeper understanding of the focal phrase. The patent phrase universe is denoted as $\mathcal{G} = (\mathcal{U}, \mathcal{V}, \mathcal{D}, \mathcal{E}^r, \mathcal{E}^c)$.

3.2 Phrase Representation Augmentation with Phrase Ego Graph

The semantic information of a patent phrase retains a wealth of knowledge pertinent to phrase similarity inference. Without loss of generality, let’s denote a textual encoder $f$ that maps a patent phrase $u$ to a $d$-dimensional numerical vector $h_u$, i.e., $f(u) \mapsto h_u \in \mathbb{R}^d, \forall u \in \mathcal{U}$. For instance, one can choose BERT (Devlin et al., 2019) or RoBERTa (Liu et al., 2019).
2019) as the textual encoder.

Furthermore, for a patent phrase \( u \in \mathcal{U} \), we retrieve its corresponding ego graph from the patent phrase universe, referred as phrase ego graph, \( \mathcal{G}_u = (\mathcal{U}_u, \mathcal{V}_u, \mathcal{E}_u^r, \mathcal{E}_u^c) \), using a recursive neighbor sampling approach. Specifically, to generate the ego graph for the patent phrase \( u \), we follow these steps: 1) Initialize a node set that includes only \( u \) and an empty edge set; 2) Sample neighbors for each node within the node set. Add the resulting neighbor nodes to the node set and their corresponding neighboring edges to the edge set; 3) Repeat step 2 for a predetermined number of iterations. By iteratively expanding both the node set and edge set, we construct a patent phrase ego graph, \( \mathcal{G}_u \), comprising patents where the focal phrase \( u \) is present, along with a set of related patents referenced within those patents.

We subsequently employ a graph neural network \( g \) to transform \( \mathcal{G}_u \) to a \( d \)-dimensional vector \( g(\mathcal{G}_u) \). In essence, \( g \) functions as a readout operation in graph neural networks (Kipf and Welling, 2017; Hamilton et al., 2017; Veli\'ckovi\'c et al., 2018), i.e., \( g(\mathcal{G}_u) \mapsto h^0_u \in \mathbb{R}^d \). This fixed-size representation \( h^0_u \) encodes the ego-graph information of focal patent phrase \( u \), which provides an augmentation to its textual representation.

Finally, the retrieval-augmented phrase embedding for \( \phi(u) \) is modeled as follows:

\[
\phi(u) = f(u) \oplus g(\mathcal{G}_u),
\]

where \( \oplus \) is element-wise addition. For two given patent phrases of interest, the similarity between them is measured by the cosine similarity of corresponding augmented embeddings.

### 3.3 Phrase Ego Graph Representation

In this section, we discuss how to map a phrase \( u \)'s ego graph \( \mathcal{G}_u \) to a fixed-size representation \( g(\mathcal{G}_u) \).

**Initialize Node Embeddings by Textual Encoder.**
Recall that the nodes in \( \mathcal{G}_u \) are either patent phrases or patent texts. Therefore, we reuse the text encoder \( f \) to map the patent phrase node \( i \in \mathcal{U}_u \) and patent node \( j \in \mathcal{V}_u \) in \( \mathcal{G}_u \) to its initial embeddings:

\[
h(i)^0 = f(i), \forall i \in \mathcal{U}_u, \quad h(j)^0 = f(j), \forall j \in \mathcal{V}_u.
\]

**Graph-based Transformation.** We then use Graph Attention Network (GAT) (Veli\'ckovi\'c et al., 2018) to recursively performing feature transformation on the ego graph \( \mathcal{G}_u \). Detailed GAT transformation can be found in (Shi et al., 2021). Note that the graph-based transformation is architecture-agnostic and can use other GNN architectures as well. We use GAT to model the two relations in the ego graph, including retrieval relation and citation relation, respectively in \( l \)-the layer, as follows.
For a given phrase node \( i \) in ego graph, we aggregate neighbors features associating with the \( i \)'s feature at \( l \)th-layer:

\[
h^{(l+1)}_i = \text{GAT}^l \left( h^{(l)}_i, \{ h^{(l)}_j \}_{j \in \mathcal{N}_u^i(i)} \right),
\]

where \( \mathcal{N}_u^i(\cdot) \) returns the retrieved patents for phrase \( i \) in its ego graph \( \mathcal{G}_u \).

For a given patent node \( j \) in ego graph, we model two types of feature aggregation from retrieval and citation perspectives by:

\[
h^{(l+1)}_{\text{retrieval}} = \text{GAT}^l_{r} \left( h^{(l)}_j, \{ h^{(l)}_i \}_{i \in \mathcal{N}_u^c(j)} \right),
\]

and

\[
h^{(l+1)}_{\text{citation}} = \text{GAT}^l_{c} \left( h^{(l)}_j, \{ h^{(l)}_i \}_{i \in \mathcal{N}_u^c(j)} \right),
\]

where \( \mathcal{N}_u^c(\cdot) \) and \( \mathcal{N}_u^c(\cdot) \) are neighbouring lookup functions to return phrase and patent neighbors for patent \( j \) in ego graph \( \mathcal{G}_u \), respectively.

Then, we combine the above two feature transformations to form patent node \( j \)'s representation at the \( l + 1 \) layer as follows:

\[
h^{(l+1)}_j = \text{Mean} \left( h^{(l+1)}_{\text{retrieval}}, h^{(l+1)}_{\text{citation}} \right),
\]

where \( \text{Mean}(\cdot, \cdot) \) is element-wise Mean pooling.

Finally, the retrieval-augmented phrase embedding \( \phi(u) \) in Equation 1, can be written as follows:

\[
\phi(u) = f(u) \oplus g(\mathcal{G}_u) = h^{(l)}_u \oplus h^{(l)}_u,
\]

where \( l \) is the number of GAT layers.

### 3.4 Learning Objective

One challenge in inferring patent phrase similarity is the absence of annotated similarity labels. To address this issue, we propose training both the textual encoder \( f \) and the graph encoder \( g \) using a self-supervised learning objective.

First, we posit that within the ego graph, the representation of a given patent node should be more similar to the phrase node that retrieves the patent than to a random phrase node unlinked to that patent. We adopt the following triplet margin loss to capture this graph topology:

\[
\mathcal{L}_{\text{retrieval}}(a, p, n) = \max \{ \text{dis}(h(a), h(p)) - \text{dis}(h(a), h(n)) + \delta_r, 0 \},
\]

where \( a \in \mathcal{V}_a \) denotes a patent node (anchor), and \( p \in \mathcal{U}_a \) denotes a phrase node (positive) and \( p \in \mathcal{U} \) denotes an in-batch negative phrase (negative). \( \delta_r \) is a hyperparameter denoting the predefined margin. \( \text{dis}(\cdot, \cdot) \) is the Euclidean distance function.

Second, the representation of a given patent node should be more similar to another patent node with which it shares a citation, compared to a random patent node that has no linkage to that patent. We capture the citation connections by:

\[
\mathcal{L}_{\text{citation}}(a, p, n) = \max \{ \text{dis}(h(a), h(p)) - \text{dis}(h(a), h(n)) + \delta_c, 0 \},
\]

where \( a, p \in \mathcal{V}_a \) denote two patents with a citation relation, and \( n \in \mathcal{V} \) is an in-batch negative patent. \( \delta_c \) is the margin hyperparameter.

Our final learning objective is shown as follows (the notations for sampled positive \( p \) and negative \( n \) are dropped for simplicity and readability):

\[
\mathcal{L} = \alpha \sum_{u \in \mathcal{U}_a} \mathcal{L}_{\text{retrieval}}(u) + (1 - \alpha) \sum_{v \in \mathcal{V}_a} \mathcal{L}_{\text{citation}}(v),
\]

where \( \alpha \) is a coefficient to balance the importance between two optimization goals. Note that the framework is trained in a end-to-end fashion where the parameters in \( f \) and \( g \) are jointly optimized.

### 4 Experiment

We evaluate proposed RA-Sim method on Patent Phrase Similarity Dataset (Aslanyan and Wetherbee, 2022), which is curated by Google and rated by domain experts in patents.

#### 4.1 Data

**Patent Phrase Similarity** (Aslanyan and Wetherbee, 2022). The dataset comprises annotated patent phrase pairs, including 36,473 for training, 2,843 for validation, and 9,232 for testing. Given that our method employs self-supervised learning and does not utilize annotated patent similarities, we restrict our evaluation to the testing set. Later in the experiments, we will explore a supervised setting for RA-Sim, where labeled training data are utilized.

As highlighted previously, an ego phrase graph is retrieved from the patent-phrase universe, which is constructed by the following two external datasets.

**USPTO Patent Dataset.** We download the data of US granted patents and their citations from the PatentsView\(^3\). This dataset encompasses approxi-
mately seven million patents issued from January 6, 1976, to December 28, 2022. From this collection, our sample includes 7,619,250 utility patents. Additionally, we retrieve 102,674,056 citation records. Patent abstracts are extracted from the database to serve as the source of patent textual information.

**RAKE Phrase Set.** We also create a patent phrase dataset using RAKE algorithm (Rose et al., 2010), for efficient phrase generation for the large patent database. For each patent abstract, we extract 3 key phrases to obtain 22,852,178 phrases and remove simple digits or single alphabets. We also apply WordNet Lemmatizer\(^4\) to lemmatize phrases. Then we remove the phrases whose frequency is less than 25 which drops a large amount of rare phrases. This process results in a phrase set of size 26,555. We summarize the token statistics in Appendix A.

### 4.2 Metrics

Following prior research (Aslanyan and Wetherbee, 2022), we use two metrics to evaluate similarity inference performance: Pearson correlation and Spearman correlation which measure the relatedness between inferred and labelled similarities. A higher score indicates better alignment with patent experts and thus better inference performance. The reported results are averaged over 3 runs.

### 4.3 Training Setup

Our model is trained on 4 RTX 3090 GPUs. We use Sentence-BERT model with all-mpnet-base-v2\(^5\) as the textual encoder. The full training details can be found in the Appendix B.

### 4.4 Baselines

We compare RA-Sim with existing methods and two proposed baselines. The full details of the baselines are provided in the Appendix D.

The pretrained baselines include base/large BERT (Devlin et al., 2019), base/large RoBERTa (Liu et al., 2019), Sentence-BERT (Reimers and Gurevych, 2019) (SBERT), and pretrained Contriever (Izacard et al., 2022) for dense information retrieval. We use mean pooling to obtain phrase-level embeddings. We also evaluate Patent-BERT, which is pretrained on patent data, for comparison. Moreover, we compare multi-stage contrastive embedding method GTE (Li et al., 2023) and the instruction-finetuned model instructor-xl (Su et al., 2022) with our method.

We fine tune base/large BERT, base/large RoBERTa and SBERT with contrastive loss SimCSE (Gao et al., 2021), and derive baselines SimCSE-BERT\(_{base}\), SimCSE-BERT\(_{large}\), SimCSE-RoBERTa\(_{base}\), SimCSE-RoBERTa\(_{large}\) and SimCSE-SBERT. We also compare established off-the-shelf word-level embeddings, including Glove (Pennington et al., 2014), Word2Vec (Mikolov et al., 2013), and FastText (Bojanowski et al., 2017).

We propose a retrieval baseline and a graph baseline: RetrieveAvg and Graph-Only. RetrieveAvg is a retrieval-based method that retrieves the most relevant patent for a given phrase using BM25 and obtains an augmented embedding by weighted averaging the phrase embedding and the most relevant patent’s embedding. Graph-Only is a graph-based baseline that replaces the initial GAT embeddings in RA-Sim with random embeddings, to remove the effects of phrase and patent text information.

### 4.5 Main Results

We show the main results of experiments in Table 1, leading to the following observations.

<table>
<thead>
<tr>
<th>Model</th>
<th>Dim.</th>
<th>Pear. Cor.</th>
<th>Spear. Cor.</th>
</tr>
</thead>
<tbody>
<tr>
<td>GloVe(^†)</td>
<td>300</td>
<td>0.429</td>
<td>0.444</td>
</tr>
<tr>
<td>FastText(^†)</td>
<td>300</td>
<td>0.402</td>
<td>0.467</td>
</tr>
<tr>
<td>Word2Vec(^†)</td>
<td>250</td>
<td>0.437</td>
<td>0.483</td>
</tr>
<tr>
<td>Patent-BERT(^†)</td>
<td>1024</td>
<td>0.528</td>
<td>0.535</td>
</tr>
<tr>
<td>Conttriever</td>
<td>768</td>
<td>0.528</td>
<td>0.498</td>
</tr>
<tr>
<td>BERT(_{base})</td>
<td>768</td>
<td>0.413</td>
<td>0.418</td>
</tr>
<tr>
<td>BERT(_{large})</td>
<td>1024</td>
<td>0.422</td>
<td>0.405</td>
</tr>
<tr>
<td>RoBERTa(_{base})</td>
<td>768</td>
<td>0.313</td>
<td>0.329</td>
</tr>
<tr>
<td>RoBERTa(_{large})</td>
<td>1024</td>
<td>0.364</td>
<td>0.372</td>
</tr>
<tr>
<td>SimCSE-BERT(_{base})</td>
<td>768</td>
<td>0.525</td>
<td>0.516</td>
</tr>
<tr>
<td>SimCSE-RoBERTa(_{base})</td>
<td>768</td>
<td>0.471</td>
<td>0.435</td>
</tr>
<tr>
<td>SimCSE-BERT(_{large})</td>
<td>1024</td>
<td>0.534</td>
<td>0.510</td>
</tr>
<tr>
<td>SimCSE-RoBERTa(_{large})</td>
<td>1024</td>
<td>0.484</td>
<td>0.460</td>
</tr>
<tr>
<td>SimCSE-SBERT</td>
<td>768</td>
<td>0.562</td>
<td>0.542</td>
</tr>
<tr>
<td>GTE(_{base})</td>
<td>768</td>
<td>0.586</td>
<td>0.562</td>
</tr>
<tr>
<td>GTE(_{large})</td>
<td>1024</td>
<td>0.599</td>
<td>0.573</td>
</tr>
<tr>
<td>Instructor-xl</td>
<td>768</td>
<td>0.600</td>
<td>0.584</td>
</tr>
<tr>
<td>SBERT</td>
<td>768</td>
<td>0.598</td>
<td>0.577</td>
</tr>
<tr>
<td>Graph-Only</td>
<td>768</td>
<td>0.258</td>
<td>0.146</td>
</tr>
<tr>
<td>RetrieveAvg</td>
<td>768</td>
<td>0.622</td>
<td>0.595</td>
</tr>
<tr>
<td>RA-Sim</td>
<td>768</td>
<td>0.633</td>
<td>0.629</td>
</tr>
</tbody>
</table>

Table 1: Patent phrase similarity inference performance under self-supervised setup, in terms of Pearson Correlation and Spearman Correlation. \(^†\) denotes the scores reported by previous work (Aslanyan and Wetherbee, 2022). Results are averaged over 3 runs.

RA-Sim outperforms existing state-of-the-art methods. Our method leverages a phrase ego graph

---

\(^4\)https://www.nltk.org/_modules/nltk/stem/wordnet.html

\(^5\)We examine model performance under different textual encoders like GTE (Li et al., 2023) and E5 (Wang et al., 2022), and all-mpnet-base-v2 performs the best.
to augment embeddings and achieves a significant improvement over the second-best model, SBERT, with a 5.8% increase in Pearson R and a 9% gain in Spearman R. Notably, the state-of-the-art method in general text similarity, SimCSE, exhibits poor performance with BERT, RoBERTa, and SBERT as base models, with a decline from 0.598 to 0.562 in Pearson R in SimCSE (SBERT). This suggests that SimCSE is not an appropriate fine-tuning objective for inferring patent phrase similarity, highlighting a need for self-supervised objective in patent phrase similarity task. Moreover, we find that the multi-stage contrastive learning method GTE and the instruction-finetuned model instructor-xl do not significantly outperform SBERT, while our RA-Sim outperforms them in a large margin. This implies that improving model capability in general text does not necessarily improve the performance in patent phrase similarity inference. Lastly, RA-Sim beats the state-of-the-art method in dense information retrieval (i.e., Contreiver) and a series of word embedding methods (e.g., Word2Vec) by a large margin, demonstrating the superiority of RA-Sim.

Retrieval module is helpful. Retrieval-based methods, including our method and the proposed RetrieveAvg baseline, underscore the effectiveness of retrieval in improving patent phrase similarity inference. For example, our proposed baseline RetrieveAvg can improve SBERT from 0.598 to 0.622 in Pearson R. These results indicate that the retrieval module is of vital importance.

Phrase graph is informative. From the result of Graph-Only, we observe that incorporating structural information (i.e., relations) alone contributes to patent phrase similarity inference to a certain extent, leading to a Pearson R of 0.258. This observation highlights the informative role of graph topology in augmenting phrase embeddings.

Domain patent texts matter. The patent document text information from USPTO is indispensable for RA-Sim, as demonstrated by the large performance degradation of the Graph-Only baseline due to lacking domain text information, only presenting a Spearman Correlation of 0.146.

4.6 Ablations
A series of ablation results are shown in Table 2.

<table>
<thead>
<tr>
<th>Ablation</th>
<th>Pear. Cor.</th>
<th>Spear. Cor.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1% RAKE Phrases</td>
<td>0.616</td>
<td>0.613</td>
</tr>
<tr>
<td>10% RAKE Phrases</td>
<td>0.633</td>
<td>0.621</td>
</tr>
<tr>
<td>k=3 in BM25</td>
<td>0.626</td>
<td>0.621</td>
</tr>
<tr>
<td>k=7 in BM25</td>
<td>0.629</td>
<td>0.626</td>
</tr>
<tr>
<td>k=50 in BM25</td>
<td>0.621</td>
<td>0.624</td>
</tr>
<tr>
<td>Expansion Iter=3</td>
<td>0.620</td>
<td>0.619</td>
</tr>
<tr>
<td>L_retrieval Only</td>
<td>0.551</td>
<td>0.547</td>
</tr>
<tr>
<td>L_simplified Only</td>
<td>0.608</td>
<td>0.588</td>
</tr>
<tr>
<td>Hard Negatives</td>
<td>0.602</td>
<td>0.582</td>
</tr>
<tr>
<td>Additive Attention</td>
<td>0.613</td>
<td>0.604</td>
</tr>
<tr>
<td>GCN</td>
<td>0.605</td>
<td>0.601</td>
</tr>
<tr>
<td>GCNII</td>
<td>0.612</td>
<td>0.615</td>
</tr>
<tr>
<td>w/o Lemmatization</td>
<td>0.629</td>
<td>0.624</td>
</tr>
<tr>
<td>Use Train Set Phrases</td>
<td>0.635</td>
<td>0.631</td>
</tr>
<tr>
<td><strong>RA-Sim (k=5 in BM25)</strong></td>
<td><strong>0.633</strong></td>
<td><strong>0.629</strong></td>
</tr>
</tbody>
</table>

Table 2: Ablation results.

robustness. One may also concern about the bias introduced by RAKE. When we replace the phrase set with the phrases from the training similarity dataset (keeping the training labels absent), we observe no significant change in terms of performance. When dropping lemmatization, the performance remains. Thus, our constructed RAKE phrase set is capable for effectively training RA-Sim.

Retrieval Size in BM25. We evaluate our method with different BM25 retrieval sizes (best performance is achieved when k=5). The results imply that: the receptive field size of the phrase ego graph (controlled by k) should be neither too small nor too large. If the retrieved ego graph is too small, it may not provide enough information to enhance the embedding. Conversely, if the k in BM25 is too large, the retrieved two graphs may become too blurry to distinguish two different phrases.

Iteration Time for Expanding Graph. We change the iteration time for expanding the phrase graph to three iterations. This change results in a decrease in performance compared to the setup with two iterations used in the main result setup. It appears that increasing the iteration time leads to the inclusion of irrelevant or noisy information, thereby diluting the power of the constructed phrase graphs.

Hard Negatives. One may consider a more advanced negative scheme. We replace the negatives in Eq. 8 and 9 with the structure-aware negatives in prior work (Ahrabian et al., 2020). The intuition behind is to utilize the ego graph structure to generate hard negatives from the anchor’s n-hop neighborhood. However, when using 3-hop negatives, we find a performance degradation. We suspect that: in an ego graph, the neighbors within a 3-hop distance and the anchor node itself are highly relevant,
and should not be treated as hard negatives.

**Graph Layer.** We also implement the GAT layer with Additive Attention (Veličković et al., 2018) scheme or use GCN (Kipf and Welling, 2017) or GCNII (Chen et al., 2020). However, the graph layer with the attention scheme we used achieves the best empirical performance, consistent with prior findings (Shi et al., 2021).

**Effectiveness of Two Losses.** Table 2 shows that there is only a minor improvement when we solely use retrieval loss. This illustrates the importance of citation information. When solely using citation loss, the performance of the model degrades largely in Pearson R (from 0.633 to 0.551). This reveals that the model can not understand how to aggregate information from phrase graph into focal phrase embedding without retrieval loss.

### 4.7 Supervised Learning with Training Set

One may consider using RA-Sim in a supervised learning setting. We conduct supervised learning experiments with the training dataset, and present model performance in Table 3, with training details in Appendix C. Generally, the results are consistent with the observations in self-supervised setting: RA-Sim outperforms other benchmarks. Notably, we emphasize that our focus is on the more challenging self-supervised setup with label absence.

### 5 Analysis

We conduct further analyses to understand the inner workings of self-supervised RA-Sim.

#### 5.1 Alignment and Uniformity

We follow prior work (Wang and Isola, 2020) to use alignment and uniformity losses to analyze the underlying mechanism. The a smaller loss in alignment or uniformity implies a better representation capability. Intuitively, a smaller alignment loss indicates that positive pairs are more similar, and a smaller uniformity loss indicates better information retention. We obtain the positive samples in alignment by the similarity threshold of 0.756. The scatter plot is shown in Figure 2.

Specifically, we observe that: (1) RA-Sim preserves uniformity and enhances alignment, as evidenced by the relative positions between RA-Sim and SBERT; (2) as SimCSE training improves uniformity, it performs poorly in alignment (i.e., inferring positive examples in patent phrases are challenging for SimCSE), which contradicts that SimCSE presents good alignment in general text; (3) a simple retrieval scheme in RetrieveAvg enhances alignment; (4) retrieving the graph yields better alignment than simply retrieving related patents.

#### 5.2 Qualitative Results

Using cosine similarity and trained embeddings, we randomly select phrases as queries to retrieve similar phrases from the testing data. Specifically, we validate RA-Sim from different perspectives and show some results in Table 4. While "planting tree" is a commonly used in daily language, "eye quadrants" is a highly technical term. When retrieving "planting tree", the top-5 results in RA-Sim include "artificial tree" and "plant growth", "tree topology" and "plant species". When retrieving "eye quadrants", the top-5 results include "eye region anatomy", "rotate eyes", and "eye image".

---

**Table 3:** Inference performance in supervised learning.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT_large</td>
<td>1024</td>
<td>0.704</td>
<td>0.683</td>
</tr>
<tr>
<td>RoBERTA_large</td>
<td>1024</td>
<td>0.651</td>
<td>0.633</td>
</tr>
<tr>
<td>SBERT</td>
<td>768</td>
<td>0.724</td>
<td>0.706</td>
</tr>
<tr>
<td>RA-Sim</td>
<td>768</td>
<td>0.741</td>
<td>0.721</td>
</tr>
</tbody>
</table>

---

**Figure 2:** Alignment and Uniformity losses scatter plot. Colors represent similarity inference performance. For clarity, we do not include methods like Word2Vec and Graph-Only in the plot due to poor performance.

**Table 4:** Retrieved top-5 phrases from testing set.

<table>
<thead>
<tr>
<th>Query Phrase</th>
<th>RA-Sim</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1 artificial tree</td>
<td>artificial tree</td>
</tr>
<tr>
<td>#2 leaf of tree</td>
<td>plant growth</td>
</tr>
<tr>
<td>#3 jackfruit tree</td>
<td>man made plant</td>
</tr>
<tr>
<td>#4 tree topology</td>
<td>plant</td>
</tr>
<tr>
<td>#5 disjoint or tree</td>
<td>plant species</td>
</tr>
<tr>
<td>Query Phrase</td>
<td>eye quadrants</td>
</tr>
<tr>
<td>#1 quadrants</td>
<td>eye region anatomy</td>
</tr>
<tr>
<td>#2 eye region anatomy</td>
<td>rotate eyes</td>
</tr>
<tr>
<td>#3 quadrant</td>
<td>eye position plan</td>
</tr>
<tr>
<td>#4 hand quadrant</td>
<td>eye image</td>
</tr>
<tr>
<td>#5 shape quadrant</td>
<td>apertures of sclera</td>
</tr>
</tbody>
</table>
which directly relevant to the query phrase. As for SBERT, the results are noisy with graph topology terms like "tree topology". As for "eye quadrants", both methods have an overlap of "eye region anatomy," which is directly correlated to the query. However, RA-Sim includes more precise results, such as "rotate eyes" and "eye position plan", which are specifically related to the anatomy, positioning, and images of the eye. SBERT includes results like "hand quadrant", which may be less directly related to the eye region anatomy. These results demonstrate the effectiveness of RA-Sim embeddings.

6 Conclusion

We explore the challenge of inferring patent phrase similarity, a task distinct from classic semantic textual similarity due to the highly technical language inherent to the patent domain. In this study, we introduce a retrieval-augmented phrase similarity method, termed RA-Sim. This method enhances the contextualized textual information of a patent phrase by incorporating a phrase graph that encompasses its focal patent, related patents via the citation network, and associated phrases through these related patents. RA-Sim is trained using self-supervised learning objectives centered on the phrase graph. Experimental results demonstrate RA-Sim’s superior performance, highlighting the benefits of retrieval augmentation for contextualized embeddings in domain problems.

7 Limitations

Our method has several limitations to improve in the future. Firstly, we only test our approach on patent data, and further research is needed to evaluate its potential for scaling up to other application domains, such as scientific articles. Secondly, the computational cost of generating a phrase set for the vast patent universe is high, which is why we utilized the efficient RAKE method. In future work, we plan to explore more computationally expensive phrase generation methods. Lastly, compared to non-graph-based methods, our approach incurs additional computational costs in learning from the retrieved graph module.

References


A Token Statistics

We summarize the token statistics in the following Table 5.

<table>
<thead>
<tr>
<th></th>
<th>Max</th>
<th>Min</th>
<th>Average</th>
<th>Std.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phrase by RAKE</td>
<td>11</td>
<td>1</td>
<td>3.3</td>
<td>0.86</td>
</tr>
<tr>
<td>Patent Abstract</td>
<td>9,134</td>
<td>1</td>
<td>114.9</td>
<td>49.64</td>
</tr>
</tbody>
</table>

Table 5: Statistics of token number.

B Full Training Details

We train the model with a learning rate from \( \{2e^{-6}, 2e^{-5}, 2e^{-4}\} \), a batch size from \( \{2, 4, 6, 8\} \), and a maximum of 2 epochs. We experiment with graph layers from 1 to 3 and find that 2 layers performed the best. Our model is trained on 4 GeForce RTX 3090 GPUs, each with 24G memory, using Pytorch\(^7\) and PyTorch Geometric\(^8\) for implementation. We use the Python library Pyserini\(^9\) to implement BM25 with different retrieval sizes of \( k \) from \( \{3, 5, 7, 50\} \). For neighbor sampling, we experiment our method with sampling iteration times from \( \{1, 2, 3\} \) and per-iteration sampled neighbors from \( \{1, 3, 5\} \). Margin parameter \( \delta_v \) and \( \delta_e \) are chosen from \( \{0.01, 0.02, 0.05, 0.1, 0.2, 0.5\} \). Adam (Kingma and Ba, 2015) is used to optimize the model parameters in a end-to-end fashion. We assess the model’s performance every 100 training steps by evaluating it on the validation set. We choose the best checkpoint according to validation performance and finally evaluate model on testing phrase set. We use Sentence-BERT model with all-mpnet-base-v2\(^{10}\) as the base textual encoder. The reported results are averaged over 3 runs.

C Supervised Learning Setup

We conduct supervised learning experiments on BERT, RoBERTa, and SBERT to compare with our method. We obtain embeddings for two phrases and use Mean Squared Error loss to guide training, following prior work (Reimers and Gurevych, 2019). We compute the loss for a phrase pair \((u_1, u_2)\) sampled from the training dataset:

\[
\mathcal{L}_{\text{supervised}}(u_1, u_2) = MSE(\hat{y}, y),
\]

where \( \hat{y} = \text{sim}(\phi(u_1), \phi(u_2)) \) is inferred similarity for phrases \( u_1 \) and \( u_2 \), and \( y \) is ground truth similarity labelled by domain experts. As for our method, we train RA-Sim jointly with supervised loss Eq. 11 and proposed self-supervised loss Eq. 10 after training with supervised loss Eq. 11 solely for 2 epochs. As for other benchmarks, we solely use Eq. 11 for training. We limit overall training budget to 5 epochs and show model performance in Table 3.

D Baseline Models

We elaborate on how we implement different baselines for comparison in our evaluation:

Pretrained Model. Existing pretrained language model can effectively map phrases to embeddings with parameters pretrained on large amounts of training data. Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) is a widely-adopted pretrained language model and we use mean pooling to obtain phrase-level embeddings. Sentence-BERT (Reimers and Gurevych, 2019) (SBERT) improves BERT by incorporating dual-tower architecture to obtain sentence embeddings, which is a competitive method in phrase similarity. We use all-mpnet-base-v2 as the base model for Sentence-BERT. Patent-BERT (Srebrovic and Yonamine, 2020) trains BERT on patent data to obtain a pretrained model on patent. Contriever (Izacard et al., 2022) is trained for unsupervised dense information retrieval and we obtain phrase embeddings by mean pooling on hidden states. GTE utilizes a multi-stage contrastive learning method during the pretraining phase (Li et al., 2023), while instructor-xl (Su et al., 2022) employs instruction to fine-tune and enhance embeddings. Abovementioned pretrained models are fetched from Hugging Face\(^{11}\).

Finetune. Simple Contrastive Learning of Sentence Embeddings (SimCSE) finetunes BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) by leveraging an unsupervised loss which views two different embeddings of a same phrase as positive pair in a contrastive objective. We apply SimCSE training to finetune BERT, RoBERTa, SBERT on the mean-pooled hidden states, with the same training phrases as our method. We experiment bert-base-uncased, bert-large-uncased for BERT, termed SimCSE-BERT\_base, SimCSE-BERT\_large. As for RoBERTa, we finetune base models of roberta-base

\(^7\)https://pytorch.org
\(^8\)https://pytorch-geometric.readthedocs.io
\(^9\)https://github.com/castorini/pyserini
\(^10\)We examine model performance under different textual encoders like GTE (Li et al., 2023) and E5 (Wang et al., 2022), and all-mpnet-base-v2 performs the best.

\(^{11}\)https://huggingface.co/models
and roberta-large, termed SimCSE-RoBERTa\textsubscript{base}, SimCSE-RoBERTa\textsubscript{large}. The all-mpnet-base-v2 is used for training SBERT based SimCSE, termed SimCSE-SBERT. All aforementioned base models for training SimCSE are from Hugging Face.

**Word Embedding.** Glove (Pennington et al., 2014), Word2Vec (Mikolov et al., 2013) and FastText (Bojanowski et al., 2017) are established off the shelf word-level embeddings. For GloVe we use the Wikipedia 2014 Gigaword 5 model, for FastText the wiki-news-300d-1M model, and for Word2Vec the Wiki-words-250 from TensorFlow Hub\textsuperscript{12}.

**Retrieval-based Embedding.** We propose a retrieval-based baseline, named RetrieveAvg. RetrieveAvg retrieves the most relevant patent for a given phrase using BM25, and then obtains augmented embedding by weighted averaging phrase embedding and most relevant patent’s embedding. Specifically, we use SBERT to map phrases or most relevant patents to embeddings. We adjust the weight (a scalar) used to combine the two embeddings according to the performance on validation dataset.

**Graph-based Embedding.** We propose Graph-Only, a graph-based baseline to validate the power of graph structure. This method is implemented by simply replacing the initial node embeddings in RA-Sim with random embeddings to remove the effects of phrase and patent text information.

**E Examples of Phrase and BM25 Retrieval Results**

**E.1 Phrase Examples**

We show phrase examples generated by the Rapid Automatic Keyword Extraction (RAKE), shown in Table 6.

**E.2 BM25 Retrieval Example**

We present retrieval results for BM25, which retrieves relevant patents for a given phrase from the USPTO patent database. The sample results are shown in Table 7.

---

\textsuperscript{12}https://tfhub.dev/google/Wiki-words-250/2
A method for selecting an alternate wireless communication system for a wireless communication device is disclosed. The method comprises using a first radio access technology (RAT) by a wireless communication device when scanning (202) for an initial wireless communication network (201). The wireless communication device further registers (215) to the initial wireless communication network (201). Then, the initial wireless communication network (201) determines (206) alternate RAT wireless communication network information for the wireless communication device and sends (235) the alternate RAT wireless communication network information to the wireless communication device. The wireless communication device receives the alternate RAT wireless communication network information and selects (240) a first alternate wireless communication network from within the alternate RAT wireless communication network information, scans for the first alternate wireless communication network using an alternate RAT and registers with the first alternate wireless communication network using the alternate RAT.

A wireless communication device comprises a processing system and a wireless communication transceiver. The processing system is configured to store in a memory system data that associates a geographic identifier, a pseudo network signal of a first wireless communication network, and a wireless communication channel of a second wireless communication network. The wireless communication transceiver is configured to wirelessly exchange first wireless communications with the first wireless communication network. The processing system is configured to, in response to the wireless communication device entering a geographic region associated with the geographic identifier, process the geographic identifier to identify the pseudo network signal of the first wireless communication network and the wireless communication channel of the second wireless communication network. The wireless communication transceiver is configured to wirelessly exchange first wireless communications with the first wireless communication network. The processing system is configured to, in response to the wireless communication device entering a geographic region associated with the geographic identifier, process the geographic identifier to identify the pseudo network signal of the first wireless communication network and the wireless communication channel of the second wireless communication network. The wireless communication transceiver is configured to wirelessly exchange first wireless communications with the first wireless communication network.

According to some embodiments, a method in a wireless device operable in a first wireless communication network and a second wireless communication network comprises receiving, from the first wireless communication network, an identification of network nodes of the second wireless communication network. The network nodes of the second wireless communication network are operable to process traffic for the wireless device. The method further comprises receiving an instruction from the second wireless communication network to move traffic from a first network node of the second wireless communication network to a second network node of the second wireless communication network. The first network node is one of the identified one or more network nodes of the second wireless communication network. The method also comprises determining that an identification of the second network node is not included in the received identification of one or more network nodes of the second wireless communication network.

Table 7: Retrieved patents for phrase "Wireless Communications Network". The three most relevant patents are shown.