Supervised Contrastive Learning for Interpretable Long Document Comparison

Akshita Jha  
Virginia Tech, Arlington, VA  
akshitajha@vt.edu

Vineeth Rakesh  
Interdigital, CA  
vineethrakesh@gmail.com

Jaideep Chandrashekhar  
Interdigital, CA  
jaideep.chandrashekar@interdigital.com

Adithya Samavedhi  
Virginia Tech, Arlington, VA  
adithyas@vt.edu

Chandan K. Reddy  
Virginia Tech, Arlington, VA  
reddy@cs.vt.edu

ABSTRACT
Recent advancements in deep learning techniques have transformed the area of semantic text matching. However, most of the state-of-the-art models are designed to operate with short documents such as tweets, user reviews, comments, etc., and have fundamental limitations when applied to long-form documents such as scientific papers, legal documents, and patents. When handling such long documents, there are three primary challenges: (i) The presence of different contexts for the same word throughout the document, (ii) Small sections of contextually similar text between two documents, but dissimilar text in the remaining parts – this defies the basic understanding of "similarity", and (iii) The coarse nature of a single global similarity measure which fails to capture the heterogeneity of the document content. In this paper, we describe CoLDE: Contrastive Long Document Encoder – a transformer-based framework that addresses these challenges and allows for interpretable comparisons of long documents. CoLDE uses unique positional embeddings and a multi-headed chunkwise attention layer in conjunction with a contrastive learning framework to capture similarity at three different levels: (i) high-level similarity scores between a pair of documents, (ii) similarity scores between different sections within and across documents, and (iii) similarity scores between different chunks in the same document and also other documents. These fine-grained similarity scores aid in better interpretability. We evaluate CoLDE on three long document datasets namely, ACL Anthology publications, Wikipedia articles, and USPTO patents. Besides outperforming the state-of-the-art methods on the document comparison task, CoLDE also proves interpretable and robust to changes in document length and text perturbations.

CCS CONCEPTS
• Information Systems → Document Representation; Document Structure; • Computing Methodologies → Information Extraction.

KEYWORDS
semantic text matching, long documents, contrastive learning, attention, embeddings, interpretability, transformer, BERT

1 INTRODUCTION
Semantic text matching (STM) is an important and challenging problem in the field of information retrieval. Given a query, which could range from a few words to a few pages in case of a long document, the objective of STM is to retrieve a set of documents related to the query. While recent advancements in language modeling have certainly produced promising results, their effectiveness has been confined to the following two tasks: (a) matching short-queries to short-documents, or (b) matching short-queries to long-documents. A few examples of short-document matching include contextual text similarity [19], sentence matching [15] and question ranking [3] where one or both documents (i.e., the query and the target document) have limited textual content. However, semantic text matching between two long documents, which is a more challenging problem, has not been studied rigorously in the literature. Some important use cases for long-document matching include ranking of research papers, comparison of patent documents, and clustering of Wikipedia articles spanning several pages. In this paper, we exclusively focus on semantic text matching for long document comparison. There are three main challenges that arise while developing an STM model for long documents.

Challenge 1. It is infeasible to learn the semantics of individual words, phrases, and sentences by considering only a few lines of text (or paragraphs), as longer documents span several pages. In other words, learning a representative embedding that encapsulates the context of an entire document is extremely challenging. Further evidence of this is provided by the state-of-the-art Transformer models, such as BERT [9], which cannot handle more than 512 tokens (or words) during a single feed-forward pass. Current solutions for addressing these limitations include various pooling techniques which have already proven to be ineffective [29].

Challenge 2. Taking into account various levels of (dis)similarity between different sections of text is another challenge. To understand this better, let us consider a scenario where the end-user is interested in finding long documents most similar to the given Query. For the sake of simplicity, we consider only two possible target long documents (Target 1 and Target 2) to choose from (see Figure 1). In the three presented documents (Query, Target 1, and Target 2), there are three levels of similarity: (a) A similar boiler plate text about the popularity of deep learning that does not contribute to the distinguishing content of the documents, (b) Semantically similar words – Query has a ‘semantic match’ with both Target 1 and Target 2 (‘reducing the size’ is equivalent to the word ‘compression’), and (c) Contextually similar content – Query is contextually similar only to Target 2 - both of them discuss the same concepts of ‘compressing neural networks’ unlike Target 1 which discusses ‘image compression’. Here, Query is most similar to Target 2 but
we define interpretability as the weighted similarity scores between compression. All documents homogeneously discuss what makes the documents (dis)similar.

Our Contributions. The primary contributions of this paper are as follows:

- Propose a novel supervised contrastive learning model, CoLDE: Contrastive Long Document Encoder to compare long documents like research papers, patent documents, wikipedia articles, etc.
- Develop new data augmentation techniques for text along with unique position embeddings to capture the long document structure and consequently, learn better representations.
- Demonstrate the interpretability provided by CoLDE – by providing high-level similarity score between documents, in addition to fine-grained similarity score between different sections, and between different text chunks within and across documents.
- Create benchmark long document datasets (using ACL Anthology papers and Wikipedia articles) which will be made publicly available to the research community, thus enabling reproducibility of the proposed work and future methods in this topic.

2 RELATED WORK

Long Document Matching. Guo et al. [13] propose a joint deep learning based architecture for ad-hoc retrieval when comparing documents. Several works have also used convolutional networks [15, 24, 37], with weighting mechanism [34] to generate a final query-document score. Mitra et al. [23] propose a combination model that uses weighted sum representation-based and interaction-based results. Yang et al. [36] propose a hierarchical attention network for document classification whereas Adhikari et al. [1] use BERT for document classification. Jian et al. [18] propose a multi-depth attention based hierarchical recurrent neural network (SMASH) for long-document comparison. However, Yang et al. [35] pre-train a transformer based hierarchical model (SMITH) for text...
matching that outperforms them across multiple datasets. While our proposed CoLDE model is also based on transformers, we provide fine-grained similarity scores unlike the previous models that only classify whether a document is relevant or irrelevant based on a (coarse) single document level similarity score.

**Long Document Encoding.** Earlier works used Doc2Vec [5, 22] for long document representation. A growing body of literature is now examining the use of transformer based models for long document encoding [7, 14, 21, 26]. Longformer [4] adapt transformers to use an attention mechanism that scales linearly with sequence length. Big Bird [38] proposes a sparse attention mechanism that reduces BERT’s quadratic dependency on the sequence length to linear. CogLTX [10] presents a framework that uses multi-step reasoning over key sentences to overcome the insufficient long-range attentions in BERT by using text blocks for rehearsal and decay. Transformer-XL [8] and Compressive Transformers [28] compress the transformers to use attentive sequence over long text. Our work is significantly different from these approaches since we primarily focus on end-to-end long document comparison.

**Interpretability for Text.** Researchers have devised approaches to measure the feature importance in text [30, 31]. There is an increased interest in using causal frameworks for understanding black-box predictions [2]. Some researchers claim that attention is not explanation [17]. However, Wiegreffe et al. [33], Yang et al. [36] and Ghaeni et al. [11] demonstrate how attention scores can be used to provide insights into attention-based models. We use multi-headed chunkwise attention in conjunction with supervised contrastive loss to provide fine-grained similarity scores that help in interpreting the final decision made by the model.

3 THE PROPOSED CoLDE FRAMEWORK

Inspired by the recent success of contrastive learning algorithms in the computer vision domain [6, 20], we build a contrastive learning framework for long-document matching. Intuitively, CoLDE maximizes the similarity between latent representations of sections within the same document and minimizes the similarity between representations of sections from different documents. Additionally, it ensures that representations of documents belonging to the same class are closer to each other in the latent embedding space when compared to the documents from a different class. CoLDE consists of three primary components: (i) Data Augmentation, (ii) Data Encoder, and (iii) Contrastive Loss Function. Figure 2 presents the complete architecture of the proposed model.

**Problem Statement.** The problem being tackled in this paper is defined as follows. Given a query document $s$, and a set of target documents $D_T$, the goal is to estimate the semantic similarity between $s$ and a target document $t \in D_T$ at three different levels: (i) Document Level $y_q^t = sim(s, t)$ for every document pair, (ii) Section Level $y_{pq} = sim(s_i, t_j)$, where $s_i \in s$ and $t_j \in t$ for every section pair, and (iii) Chunk Level $y_{qc} = sim(c_q, c_q)$, where $c_{q_i} \in s_i$ and $c_{q_j} \in t_j$ for every chunk pair. The semantically similar target documents, sections, and chunks will have a higher similarity score.

3.1 Data Augmentation

Long documents, unlike short-text, follow an inherent structure and are organized into sections and sub-sections (e.g., introduction, related work, methodology, etc.). The task of learning meaningful representations for long documents, can be broken down into learning better representations for their sections and sub-sections. To achieve this, we divide a long document into different sections as follows: $x_i, x_j = Aug(x)$ where $Aug(.)$ is the augmentation module that takes a long document $x$ as input and extracts two different sections $x_i$ and $x_j$ from the document. This is represented as Step 1 in Figure 2. These sections contain a subset of the text and represent a subset of information present in the original long document. One of the major limitations of BERT is its inability to handle more than 512 tokens at a time, thus making it ineffective for encoding long documents. To overcome this limitation, we divide a document section into multiple chunks of non-overlapping 512 tokens. Additionally, we enhance these input section chunks using unique positional embeddings (described below) to encode the long document structure.

**Unique Positional Embeddings.** The token embeddings from BERT are combined with long document structure aware positional embeddings before providing them as input to the BERT model. These positional embeddings are explained below (shown in Fig. 3).

- **Section Embeddings:** Section embeddings $S_p$ are unique values given to each section where $p$ represents the section number. Since we divide a long document into two sections, these embeddings take two possible values: 0 and 1. Embeddings for section $x_i$ take the value of 0 and that for section $x_j$ take the value of 1.
- **Chunk Embeddings:** Since BERT can only handle 512 tokens at a time, each section is further divided into several chunks of 512 tokens. Each of these chunks is given a unique ID $C_q$ where $q$ represents the chunk number.
- **Token Embeddings:** Token embeddings $T_r$ are the index $r$ of the token in the input sequence. They are the same as the ones seen in the standard BERT model.

These additional embeddings are summed up together and form the augmented token embeddings $\tilde{T}_{pq}$ given by:

$$\tilde{T}_{pq} = \sum_{p=0}^{n_p} \sum_{q=0}^{512} \sum_{r=0}^{512} (S_p + C_{pq} + T_{pr})$$

where $n_p$ is the total number of sections; and $n_q$ is the total number of chunks in a section. There are 512 tokens in each chunk. These augmented token embeddings are given to the Data Encoder for further processing.

3.2 Data Encoder

CoLDE’s data encoder $f(.)$ is built on BERT [9]. We use a pre-trained BERT model that is fine-tuned during training. The augmented input chunks of 512 tokens are given as input to BERT. Zero-padding is done for chunks smaller than 512 tokens. The encoded BERT chunk embeddings are given to a bidirectional-LSTM (Bi-LSTM) [12] layer for aggregation which uses a unique ‘multi-headed chunkwise attention’ component (described below).

**Multi-headed Chunkwise Attention.** To get chunk level similarity scores, we introduce a multi-headed chunkwise attention layer. This is a self-attention [32] layer that computes attention between the BERT embeddings of different chunks of 512 tokens.
Figure 2: Model architecture for CoLDE. The contrastive learning framework consists of three primary components: (i) Data Augmentation, (ii) Data Encoder, and (iii) Contrastive Loss Function. It takes as input long documents and divides them into several sections. Each section is further split into chunks of 512 tokens and enhanced with unique positional embeddings (Fig. 3) before being given to the data encoder module. The contrastive loss function in conjunction with multi-headed chunk-wise attention provide fine-grained similarity scores within and across sections and chunks.

Figure 3: Visualization of the proposed unique positional embeddings for a long-document with two sections. In this example, each section has two chunks of 512 tokens each. The embeddings are summed up together to form the augmented token embeddings for BERT. They capture the structural information present in the long document.

within and across sections. As a consequence, we can obtain a fine-grained understanding of which chunk attends the most to which other text chunk in the document. The chunk with the highest attention weight w.r.t. a Query chunk plays the most important part in computing the similarity score. We compute the multi-headed chunkwise attention between sections by treating each chunk in section $p$ as a Query ($Q$). The chunks of section $p+1$ are treated as Keys ($K$) against which the attention is computed. The chunks of section $p+1$ are also used in calculating the Values ($V$). These vectors are of dimension $d$. The weights for calculating the query $Q$, keys $K$, and values $V$ are learned during training. The matrix of the outputs is computed as follows: $\text{Attention}(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d}} \right)V$.

**Dimensionality Reduction.** The encoded BERT output of different chunks, along with the multi-headed chunkwise attention, is given sequentially to a Bi-LSTM layer. The Bi-LSTM layer is required for aggregation of the segmented chunk representations in order to obtain an intermediate high-dimensional section representation $\tilde{z}_l$ corresponding to section $\tilde{x}_l$, where $l$ is the section number.
This intermediate section representation $\tilde{z}_i = f(\tilde{x}_i)$ is further given to a projection layer $\text{Proj}(\cdot)$ for reducing the dimensionality which results in a final section-level representation $z_j = \text{Proj}(\tilde{z}_j)$. The final section representations $z_j$ is used to compute the contrastive loss between sections across query and target documents. The loss is then back-propagated through the network during training.

### 3.3 Contrastive Loss Function

**Supervised Contrastive Loss.** Using the contrastive learning framework, we bring representations of sections within the same documents closer and obtain a similarity score between sections within and across documents. CoLDE also provides document level similarity scores and ensures that documents belonging to the same class are closer to each other in the latent embedding space. To achieve this, we use supervised contrastive loss [20] as our objective function. We randomly sample a set of $N$ long documents and their corresponding labels $\{x_k, y_k\}_{k=1}$. This results in $2N$ data points since each document is divided into two sections. In this paper, we refer to the set of $N$ documents as a ‘batch’ and the set of $2N$ documents as an ‘augmented batch’. The supervised contrastive loss function is described as follows:

$$L^{sup} = \left(1 - \frac{1}{2N_{y_j}} \right) \sum_{i=1}^{2N} \exp\left(\frac{z_i \cdot z_j}{\tau}\right)$$

In the above equation, $N$ is the batch size; $z_j = \text{Proj}(f(\tilde{x}_j))$ where $\tilde{x}_j$ is a section; $f(\cdot)$ is the encoder; and $\text{Proj}(\cdot)$ is the projection layer. The three indicator functions: (i) $\mathbb{1}_{i \neq j} \in \{0, 1\}$ evaluate to 1 iff $i \neq j$; (ii) $\mathbb{1}_{y_i = y_j} \in \{0, 1\}$ evaluate to 1 when the labels for two sections are the same; and (iii) $\mathbb{1}_{i \neq k} \in \{0, 1\}$ evaluate to 1 iff $i \neq k$. The symbol $\cdot$ indicates inner (dot) product; $\tau$ denotes a temperature parameter. The final loss is summed across all the samples. The triplet loss, one of the widely used losses for supervised training, is a special case of the contrastive loss when the batch size $N = 2$ and contains only one positive and one negative sample.

The numerator incorporates all positive sections in an augmented batch, i.e., every section with the same label in the augmented batch is treated as a positive sample. The denominator, on the other hand, performs a summation over the negative samples. To understand the positive and negative samples better, let us consider a simple scenario where $N = 3$. The batch has three documents: $D_1$, $D_2$, and $D_3$ with labels 1, 1, and 0, respectively. Each document is split into two sections which results in an augmented batch of $2N = 6$. With respect to the source document $D_1$, sections of $D_1$ and $D_2$ are considered to be positive samples for each other. Sections from $D_3$ are considered to be negative samples. Our loss function encourages the encoder to output section representations for $D_1$ and $D_2$ to be closer to each other in the latent embedding space. Although, the above example only describes the binary class scenario, the loss generalizes well to a multi-class setting as well. In a multi-class setting, the positive and negative samples are computed w.r.t. the label of the source document.

**Negative Samples.** For a batch size of $N$, all documents having the same label are treated as positive samples and the others are considered to be negative samples w.r.t. a source document. The negative samples belong to one of the three categories: (i) hard negatives, (ii) semi-hard negatives, and (iii) easy negatives. These three categories represent the distance between the representation of the source document and the negative samples in the latent embedding space, ranging from very close for hard negatives to very far for easy negatives. Most models explicitly require these different kinds of negatives as inputs. This is not the case for CoLDE. When using the supervised contrastive loss, hard negative mining is implicitly performed by the model [20]. The discriminatory power of the encoder increases with the increase in the number of negative samples w.r.t. a source document, i.e., with increase in batch size, the number of negative samples increase.

### 4 EMPIRICAL EVALUATION

In this section, we empirically evaluate the proposed CoLDE framework by studying the following research questions:

- **RQ1:** Does CoLDE perform better than existing methods for downstream tasks such as long document comparison?
- **RQ2:** How does the supervised contrastive loss function and the proposed multi-headed attention layer aid in interpretability?
- **RQ3:** How robust is the model to changes in batch size, document length, and text perturbation?
- **RQ4:** What is the impact of different components on the performance of CoLDE?

#### 4.1 Methodology

**4.1.1 Datasets.** For our experiments, we consider three different long document datasets: (i) ACL Anthology Network Corpus (AAN), (ii) Wikipedia (WIKI), and (iii) Patent (PAT) datasets. We will publicly release the pre-processed AAN and the Wikipedia datasets and make it available to the research community in order to enable further work in this topic.

- **AACL Anthology Network Corpus (AAN):** The AAN corpus [27] consists of 23,766 papers written by 18,862 authors in 373 venues related to NLP and forms a citation network. Each paper is represented by a node with directed edges connecting a paper (the parent node) to all its cited papers (children nodes). Papers that have been cited by the parent paper are treated as similar samples [18]. For every similar sample, an irrelevant paper is randomly chosen to create a balanced dataset. Sets of similar papers are given the same labels. To prevent leakage of information and make the task more difficult, the references and the abstract sections are removed. Papers without any content are also removed. The dataset consists of 12,665 randomly selected research papers split into two sections. On average, each section has 6 chunks.

- **Wikipedia (WIKI):** We use the Wikipedia dump, and adopt a similar methodology proposed by Jiang et al. [18] to process this data. We create a dataset of similar Wikipedia articles by assuming that similar articles have similar outgoing links. The Jaccard similarity between the outgoing links of the source and the target articles is calculated. If the Jaccard similarity > 0.5, the documents are assumed to be similar, otherwise they are considered dissimilar. Only articles with two or more similar articles are selected. Sets of similar articles are given the same

1. [http://aan.how/download/#aanNetworkCorpus](http://aan.how/download/#aanNetworkCorpus)
2. [https://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2](https://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2)
Table 1: Comparison of different methods on various long-document datasets: (i) AAN, (ii) WIKI, and (iii) PAT. Different metrics such as Precision (P), Recall (R), F1-Score (F1), and Accuracy (Acc) for document comparison task are reported. The best result is highlighted in boldface and the second best is underlined.

| Model            | AAN       | WIKI       | PAT        |
|------------------|-----------|------------|------------|
|                  | P     | R     | F1    | Acc  | P     | R     | F1    | Acc  | P     | R     | F1    | Acc  |
| DSSM             | 0.6763 | 0.7167 | 0.7112 | 0.6868 | 0.7364 | 0.8306 | 0.7807 | 0.7886 | 0.5380 | 0.5380 | 0.5380 | 0.5212 |
| ARC-I            | 0.6475 | 0.6837 | 0.6651 | 0.6557 | 0.7996 | 0.6667 | 0.7271 | 0.7643 | 0.7824 | 0.6951 | 0.7362 | 0.7080 |
| HAN              | 0.6415 | 0.7435 | 0.6887 | 0.6640 | 0.6761 | 0.9546 | 0.7916 | 0.7486 | 0.5207 | 0.5207 | 0.5207 | 0.5381 |
| S-BERT           | 0.6652 | 0.6667 | 0.6569 | 0.5663 | 0.6651 | 0.6672 | 0.6661 | 0.4993 | 0.5362 | 0.4958 | 0.5152 | 0.5206 |
| SMITH            | 0.7329 | 0.6038 | 0.6621 | 0.6918 | 0.6402 | 0.9800 | 0.7749 | 0.7146 | 0.5241 | 0.4853 | 0.5023 | 0.5093 |
| S-LONG           | 0.6331 | 0.7235 | 0.6752 | 0.6996 | 0.7947 | 0.7331 | 0.7626 | 0.7527 | 0.8489 | 0.5534 | 0.6651 | 0.5213 |
| CoLDE w/o Aug    | 0.6238 | 0.6294 | 0.6235 | 0.6166 | 0.6061 | 0.5800 | 0.5928 | 0.6061 | 0.7787 | 0.8096 | 0.7903 | 0.7573 |
| CoLDE w/o LSTM   | 0.7185 | 0.7016 | 0.7099 | 0.7063 | 0.7803 | 0.7542 | 0.7668 | 0.7862 | 0.7954 | 0.8432 | 0.8186 | 0.7921 |
| CoLDE w/o CL     | 0.5995 | 0.4999 | 0.5452 | 0.5955 | 0.9439 | 0.5104 | 0.6589 | 0.5003 | 0.6142 | 0.5848 | 0.5954 | 0.5866 |
| CoLDE            | 0.7199 | 0.7562 | 0.7344 | 0.7420 | 0.8604 | 0.7678 | 0.8092 | 0.8006 | 0.8299 | 0.8491 | 0.8393 | 0.8333 |

Note: CoLDE = Coherence Long Document Encoder; AAN = Attention-based Autoencoder; WIKI = Wikipedia; PAT = USPTO Patent.

4.2 RQ1: Evaluation on Document Comparison

We evaluate the performance of our model on the downstream task of long-document comparison. The document comparison problem can be treated as a classification task. During evaluation, the model is given a pair of documents and the task is to predict whether the two input documents are similar or dissimilar. The training is done using the model architecture described in Figure 2. During testing, we do not split the document into sections and use only the Data Encoder module to get a single representation for the query and the target documents, respectively. We then use cosine distance to compute the similarity between these representations. If the document level similarity score is above a certain

---

1https://github.com/google/patents-public-data
2https://www.uspto.gov/web/patents/classification/cpc/html/cpc.html
3Since the pre-processed dataset used in [18, 35, 36] is not public, our results are not exactly aligned with theirs. We use their publicly available code and fine-tune the model on our dataset.
4https://huggingface.co/transformers/model_doc/bert.html
5We use classification metrics instead of information-retrieval metrics due to the limitations of the dataset which has very few positive samples (2-3) for every document.
To demonstrate the interpretability provided by CoLDE, we select \( R \) author contributions. Document \( D \) analysis using the text from these documents is shown in Figure 5. by CoLDE using the architecture describe in Figure 2. Qualitative three levels of interpretability (in terms of similarity scores) offered hence, \( D \) cites \( D \), \( D \), and \( D \) are similar to each other. Document \( D \) is different from the other two. \( I_1 \) – \( I_3 \) correspond to the 'Introduction'; \( RW_1 \) – \( RW_3 \) correspond to the 'Related Work' sections of documents \( D_1 \) – \( D_3 \), respectively. Qualitative analysis for Fig. 4c is shown in Fig. 5.

\[ \text{(a) Level 1: Document Level Similarity} \]

\[ \text{(b) Level 2: Section Level Similarity} \]

\[ \text{(c) Level 3: Chunk Level Similarity} \]

Figure 4: Three levels of interpretability provided by CoLDE: (a) Document Level Similarity, (b) Section Level Similarity, and (c) Chunk Level Similarity. We consider three research papers: \( D_1 \), \( D_2 \), and \( D_3 \). \( D_1 \) and \( D_2 \) are similar to each other. Document \( D_3 \) is different from the other two. \( I_1 \) – \( I_3 \) correspond to the 'Introduction'; \( RW_1 \) – \( RW_3 \) correspond to the 'Related Work' sections of documents \( D_1 \) – \( D_3 \), respectively. Qualitative analysis for Fig. 4c is shown in Fig. 5.

\[ \text{Figure 5: Qualitative Analysis: Documents } D_1 \text{ and } D_2 \text{ are similar. The introduction chunk } I_{11} \text{ from } D_1 \text{ is about ‘topic segmentation and extractive summarisation’. The introduction chunk } I_{21} \text{ from } D_2 \text{ is about ‘text segmentation’; the related work chunk } RW_{21} \text{ from } D_2 \text{ presents prior work in the area of ‘text segmentation’. The introduction chunk } I_{12} \text{ from } D_1 \text{ summarizes the author contributions. Document } D_3 \text{ is on question answering and is different. Darker colored lines indicate higher similarity.} \]

\[ \text{4.3 RQ2: Case Study on Model Interpretation} \]

To demonstrate the interpretability provided by CoLDE, we select three documents, \( D_1 \), \( D_2 \), and \( D_3 \) with the following properties. \( D_1 \) cites \( D_2 \); hence, they are similar. \( D_3 \) is not cited by either \( D_1 \) or \( D_2 \); hence, \( D_3 \) is different from other two documents. Figure 4 shows the three levels of interpretability (in terms of similarity scores) offered by CoLDE using the architecture describe in Figure 2. Qualitative analysis using the text from these documents is shown in Figure 5.

- **Level 1-Document Level Similarity**: We obtain the high-level similarity score using CoLDE. Figure 4a illustrates the level of similarity between the three documents \( D_1 \), \( D_2 \), and \( D_3 \). A document is most similar to itself. Documents \( D_1 \) and \( D_2 \) have a higher similarity score when compared to document \( D_3 \). Qualitatively, \( D_1 \) and \( D_2 \) are about ‘automatic segmentation of speech’ whereas \( D_3 \) is about ‘question answering’ (see Figure 5).

- **Level 2-Section Level Similarity**: While document-level similarity is certainly useful to get a global overview, it does not demonstrate the reason behind (dis)similarity. Therefore, CoLDE goes a level deeper by providing section level similarity. To explain this outcome, we split each document \( \{D_k\}^3 \) into two sections \( \{(I_k, RW_k)\}^3 \), where \( I_k \) and \( RW_k \) indicate the Introduction and Related Work of \( k \)th document, respectively. Figure 4b shows how these are (dis)similar. First, sections from the same document have a high similarity score \( (I_1 \text{ and } RW_1; I_2 \text{ and } RW_2; I_3 \text{ and } RW_3) \). Second, since documents \( D_1 \) and \( D_2 \) are similar, their introductions \( I_1 \) and \( I_2 \) are more similar compared to \( I_3 \) from \( D_3 \). This is also true for sections \( RW_1 \) and \( RW_2 \), which are more similar to each other compared to \( RW_3 \). Figure 5 shows some of the content from \( I_1 \), \( I_2 \), and \( RW_2 \). The introductions \( I_1 \) and \( I_2 \) discuss ‘topic/text segmentation in dialogues and meetings’, and \( RW_2 \) presents the related work in the area.

- **Level 3-Chunk Level Similarity**: Besides providing the section-level similarity, CoLDE also explains which segment (or chunk) of a section is important. To illustrate this outcome, we divide
Figure 6: Effect of batch size on CoLDE’s accuracy for document comparison task on the three datasets.

Figure 7: Effect of document length on CoLDE’s accuracy for document comparison task on the three datasets.

\((I_k, RW_k)\) into \(j\) chunks, which is denoted by \((I_{jk}, RW_{jk})\). In our experiments, we set \(j = 2\); in other words, each document is divided into 4 chunks. Figure 4e shows some interesting traits of chunk-level interpretability between documents \(D_1\) and \(D_2\). The introduction chunk \(I_{11}\) from document \(D_1\) is similar to the first introduction chunk \(I_{21}\) from \(D_2\). In Figure 5, we see that \(I_{11}\) broadly discusses ‘the system components that perform topic segmentation and extractive summarisation’, and \(I_{21}\) discusses ‘text segmentation and its applications’. Additionally, the related work chunk \(RW_{21}\) from \(D_2\) is very similar to both \(I_{11}\) and \(I_{21}\) from documents \(D_1\) and \(D_2\), respectively. Qualitatively, \(RW_{21}\) presents prior work in the area of ‘text segmentation’. However, we observe that the second introduction chunk \(I_{12}\) from document \(D_1\) has very low similarity score with the above chunks \((I_{11}, I_{21}, \text{and} RW_{21})\) and interestingly, \(I_{12}\) is about ‘visual aids for interpretability and a summary of the author contributions’.

CoLDE, therefore, provides fine-grained inter-document and intra-document (dis)similarity scores at three different levels which makes the model more interpretable.

4.4 RQ3: Robustness Analysis of CoLDE

4.4.1 Effect of Batch Size on CoLDE. Recent works on contrastive learning in the computer vision domain \([6, 20, 25]\) have shown that the discriminating ability of the model increases with increase in batch size since there will be an increase in the number of informative negative samples. We empirically validate this hypothesis and demonstrate that this holds true for the long document comparison task. Figure 6 shows the impact of batch size on model accuracy when it is trained for the same number of epochs on three datasets: (i) AAN, (ii) WIKI, and (iii) PAT. For the same number of epochs, larger batch-size performs better for AAN and PAT datasets. However, because of the presence of hard as well as easy negatives in the WIKI dataset, a larger batch size will have many more easy negatives making the task more challenging. Although, CoLDE does eventually converge to the optimal accuracy of \(\sim 80\%\) for all batch sizes on the WIKI dataset, it takes longer for bigger batch sizes than for smaller ones. Irrespective of the dataset, the best performance of CoLDE steadily increases with the number of training epochs.

4.4.2 Effect of Document Length on CoLDE. In order to study CoLDE’s robustness to varying document lengths, we limit the maximum number of tokens in each section during training and observe the final test accuracy on the document comparison task. The model is given the entire long document during evaluation. Figure 7 shows the model accuracy for the maximum lengths of 1000, 4000 and 8000 tokens. We compare CoLDE’s performance with other methods for similar document lengths. We observe that CoLDE outperforms the baselines and its performance steadily increases with increase in document length, emphasising the need for explicit long document comparison models.
We study the effect of different components of CoLDE during training by conducting ablation study experiments. The results of the ablation study on the document comparison task are shown in Table 1.

4.5.1 Effect of Data Augmentation. We remove the Data Augmentation module to study its impact on the overall model performance. This model variant is termed as CoLDE-w/o Aug. We do not split a long document into different sections. We also do not add the proposed positional embeddings to the input document. Instead, the entire long document is split into chunks of 512 tokens and given to the data encoder module. The objective function used is supervised contrastive loss. We observe that there is a significant drop in the performance of DSSM, ARC-I, and HAN. We observe that, although, other transformer-based models are sensitive to text perturbation, CoLDE is robust. The proposed unique positional embeddings used to enhance the tokens during the data augmentation phase are long document aware, and hence, even after shuffling the text sections, they retain the original long document structure information, thus preserving the accuracy on the downstream task.

4.5.2 Effect of LSTM and Multi-Headed Chunkwise Attention. BERT can only handle 512 tokens at a time. To overcome this limitation and get section representations, we use a Bi-LSTM layer for aggregation. We study its effectiveness by removing only the Bi-LSTM and the multi-headed chunkwise attention layer from the data encoder module. We term this variant as CoLDE-w/o LSTM. The section representation is the ‘average’ of the BERT output representations for all chunks of 512 tokens within a section. This is then given to the projection layer for dimensionality reduction. The objective function used is supervised contrastive loss. We observe that there is a performance drop on the document comparison task when not using the Bi-LSTM layer for aggregation. Bi-LSTM layer effectively captures the sequential nature of the chunks within a section which is lost when computing the average of the BERT output representations for text chunks.

4.5.3 Effect of using Contrastive Loss. We conduct an ablation study by replacing the supervised contrastive loss with cosine similarity. Since, we do not require documents to be split into sections for computing the cosine similarity between them, removing the contrastive loss function entails disabling the data augmentation module as well. This variant is termed as CoLDE-w/o CL. The input documents are split into multiple chunks of 512 tokens and given to the data encoder module. The final representations from this module are used to compute the cosine similarity. From Table 1, we observe that there is a significant drop in performance when using cosine similarity as opposed to using the supervised contrastive loss as our objective function. Supervised contrastive loss helps in providing fine-grained similarity scores and ensures that the latent representations of sections within the same document are closer to each other. Additionally, representations of documents belonging to the same class are also closer in the latent embedding space.

4.5.4 Effect of Number of Input Sections. We study the effect of number of sections on the performance of CoLDE by splitting the source and the target documents into sections ranging from 2 to 5, before being given as input to the model. CoLDE was robust to these changes and the performance was consistent with the ones shown in Table 1 for different input sections for all datasets. We do not plot the results due to space constraints.

5 CONCLUSION
In this work, we introduce a contrastive learning framework CoLDE for long document comparison. CoLDE leverages the long document structure in conjunction with supervised contrastive loss to provide fine-grained similarity scores within and across documents at three different levels: (i) Document Level, (ii) Section Level, and (iii) Chunk Level. The overall framework consists of three primary components: (i) Data Augmentation, (ii) Data Encoder, and (iii) Contrastive Loss Function. We present a detailed case study for model interpretation and show that CoLDE outperforms the existing state-of-the-art models for long document comparison. We also empirically demonstrate CoLDE’s robustness to batch size, document length, and text perturbation compared to other methods. Moreover, we conduct an ablation study to examine the effectiveness of various components of CoLDE. In summary, we show CoLDE’s contrastive learning approach is effective for interpretable long document comparison.
REFERENCES

[1] Ashutosh Adhikari, Achyuth Ram, Raphael Tang, and Jimmy Lin. 2019. DocBERT: BERT for document classification. arXiv preprint arXiv:1904.08389 (2019).

[2] David Alvarez-Melis and Tommi Jaakkola. 2017. A causal framework for explaining the predictions of black-box sequence-to-sequence models. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 412–421.

[3] Hadi Amini, Philip Resnik, Jordan Boyd-Graber, and Hal Daumé III. 2016. Learning text pair similarity with context-sensitive autoencoders. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 1882–1892.

[4] J. Beliagyi, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. arXiv preprint arXiv:2004.05150 (2020).

[5] Minmin Chen. 2017. Efficient Vector Representation for Documents through Corruption. In ICLR (Poster).

[6] Ting Chen, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In International conference on machine learning. PMLR, 1597–1607.

[7] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. arXiv preprint arXiv:1904.16509 (2019).

[8] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2975–2988.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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). 4171–4186.

[10] Ming Ding, Chang Zhou, Hongxia Yang, and Jie Tang. 2020. CogLTX: Applying BERT to Long Texts. Advances in Neural Information Processing Systems 33 (2020).

[11] Reza Gheasai, Xiaodi Fern, and Prasad Tadepalli. 2018. Interpreting Recurrent and Attention-Based Neural Models: A Case Study on Natural Language Inference. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. 4952–4957.

[12] Alex Graves and Jürgen Schmidhuber. 2005. Frameworks phoneme classification with bidirectional LSTM networks. In Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005. Vol. 4. IEEE, 2047–2052.

[13] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance modeling model for ad-hoc retrieval. In Proceedings of the 25th ACM international conference on information and knowledge management. 55–64.

[14] Jonathan Ho, Kalchichrenner, Dirk Weissenborn, and Tim Salimans. 2019. Axial attention in multidimensional transformers. arXiv preprint arXiv:1912.12180 (2019).

[15] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. Advances in neural information processing systems 27 (2014), 2042–2050.

[16] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In Proceedings of the 22nd ACM international conference on Information & Knowledge Management. 2333–2338.

[17] Sarthak Jain and Byron C Wallace. 2019. Attention is not Explanation. 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). 3543–3556.

[18] Jyun-Yu Jiang, Mingyang Zhang, Cheng Li, Michael Bendersky, Nadav Golbandi, and Marc Najork. 2019. Semantic text matching for long-form documents. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, San Diego, California, 1480–1489. https://doi.org/10.18653/v1/N19-1174

[19] Jianfei Li, Qiang Qiu, Jing Jiang, Jun Huang, Shuangyang Song, Wei Chu, and Haiping Chen. 2018. Modelling domain relationships for transfer learning on retrieval-based question answering systems in e-commerce. In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, 682–690.

[20] Prannay Khola, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, C. Liu, and Dilip Krishnan. 2020. Supervised Contrastive Learning. Advances in Neural Information Processing Systems 33 (2020).

[21] Nikita Kitaev, Lukasz Kaiser, and Anselm Levkavya. 2019. Reformer: The Efficient Transformer. In International Conference on Learning Representations.

[22] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In International conference on machine learning. PMLR, 1188–1196.

[23] Bhaskar Mitra, Fernando Diaz, and Nick Crawewell. 2017. Learning to match using local and distributed representations of text for web search. In Proceedings of the 26th International Conference on World Wide Web. 1291–1299.

[24] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Shengxian Wan, and Xuezhi Cheng. 2016. Text matching as image recognition. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 30.

[25] Tsung-Hung Park, Alexei A Efros, Richard Zhang, and Jun-Yan Zhu. 2020. Contrastive Learning for unpaired image-to-image translation. In European Conference on Computer Vision. Springer, 319–345.

[26] Jiezhong Qiu, Hao Ma, Omer Levy, Wen-tau Yih, Sanzong Wang, and Jie Tang. 2020. Blockwise Self-Attention for Long Document Understanding. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings. 2555–2565.

[27] Dragomir R. Radev, Pradeep Muthukrishnan, Vahed Qazvinian, and Amjad Abu-Jbara. 2013. The ACL anthology network corpus. Language Resources and Evaluation (2013), 1–26. https://doi.org/10.1007/s10579-012-9211-2

[28] Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, Chille Hiller, and Timothyp F Lillisrep. 2019. Compressive Transformers for Long-Range Sequence modelling. In International Conference on Learning Representations.

[29] Nils Reimers and Iryna Gurevych. 2019. 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). 3973–3983.

[30] Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. 2017. Right for the right reasons: training differentiable models by constraining their explanations. In Proceedings of the 26th International Joint Conference on Artificial Intelligence. 2662–2670.

[31] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks: In International Conference on Machine Learning. PMLR, 3319–3328.

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems. 6000–6010.

[33] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Radev. 2020. BERT to Long Texts. Advances in Neural Information Processing Systems 33 (2020).

[34] Liu Yang, Qingyao Ai, Jiafeng Guo, and W Bruce Croft. 2016. aNMM: Ranking long sequences with sparse transformers. In Proceedings of the 25th International Conference on World Wide Web Conference. 1295–1304.

[35] Liu Yang, Mingyang Zhang, Cheng Li, Michael Bendersky, and Marc Najork. 2020. Beyond 512 Tokens: Siamese Multi-depth Transformer-based Hierarchical Encoder for Long-Form Document Matching. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management. 1725–1734.

[36] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical Attention Networks for Document Classification. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, San Diego, California, 1480–1489. https://doi.org/10.18653/v1/N16-1174

[37] Jianfei Li, Qiang Qiu, Jing Jiang, Jun Huang, Shuangyang Song, Wei Chu, and Haiping Chen. 2018. Modelling domain relationships for transfer learning on retrieval-based question answering systems in e-commerce. In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, 682–690.

[38] Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainsdle, Chris Alberti, Santiago Ontanion, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. arXiv preprint arXiv:2007.14062 (2020).