

Pointer-Guided Pre-Training: Infusing Large Language Models with Paragraph-Level Contextual Awareness

Lars Hillebrand^{1,2} (✉), Prabhupad Pradhan¹, Christian Bauckhage^{1,2}, and Rafet Sifa^{1,2}

¹ Fraunhofer IAIS, Germany

² University of Bonn, Germany

`lars.patrick.hillebrand@iais.fraunhofer.de`

Abstract. We introduce “pointer-guided segment ordering” (SO), a novel pre-training technique aimed at enhancing the contextual understanding of paragraph-level text representations in large language models. Our methodology leverages a self-attention-driven pointer network to restore the original sequence of shuffled text segments, addressing the challenge of capturing the structural coherence and contextual dependencies within documents. This pre-training approach is complemented by a fine-tuning methodology that incorporates dynamic sampling, augmenting the diversity of training instances and improving sample efficiency for various downstream applications. We evaluate our method on a diverse set of datasets, demonstrating its efficacy in tasks requiring sequential text classification across scientific literature and financial reporting domains. Our experiments show that pointer-guided pre-training significantly enhances the model’s ability to understand complex document structures, leading to state-of-the-art performance in downstream classification tasks.

Keywords: Language Modeling · Representation Learning · Natural Language Processing · Machine Learning.

1 Introduction

The landscape of natural language processing (NLP) has been profoundly transformed by the emergence of generative large language models (LLM) such as OpenAI’s GPT series [1,4], Mixtral [18], and Llama2 [31]. These models have set new benchmarks across a wide range of NLP tasks, showcasing remarkable capabilities in understanding and generating human language. Despite the significant advancements achieved by these large-scale models, there remains an equally important domain for smaller, specialized language models that excel in fast retrieval and semantic search, particularly those that generate precise paragraph and section representations. This domain is crucial, especially in the context of retrieval augmented generation (RAG) [22], where the integration of

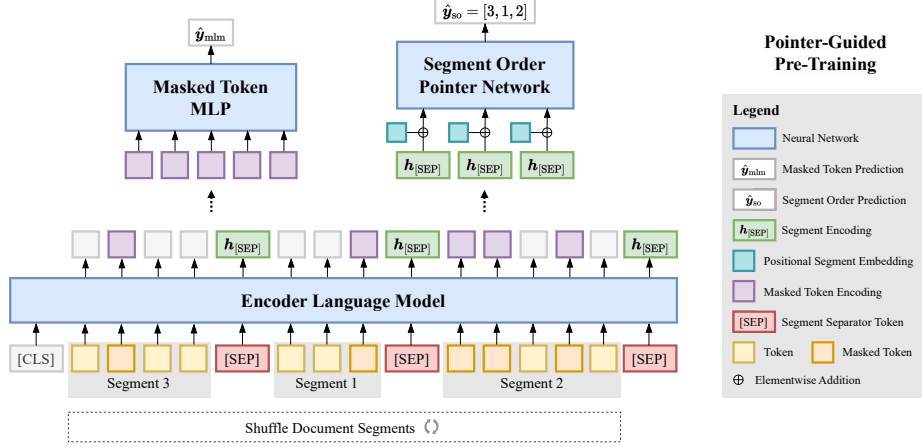


Fig. 1. Schematic visualization of our “Pointer-Guided Pre-Training” methodology. During pre-training a self-attention-based pointer network classification head learns to reconstruct the original order of shuffled text segments based on their hidden state representations ($h_{[\text{SEP}]}$). Employing this segment ordering (SO) pre-training mechanism alongside masked language modeling (MLM) increases the segment level contextual awareness of the encoding language model and subsequently improves its downstream classification capabilities.

retrieval mechanisms with generative models enhances the reliability and informativeness of the output.

In this work, we address the important area of representation learning for improved paragraph-level contextual understanding, which is critical for enhancing the capabilities of NLP systems in sequential text classification and retrieval-based applications such as semantic text search.

At the heart of our approach is the introduction of a novel pre-training methodology, named “pointer-guided segment ordering” (SO). This technique is designed to infuse language models with a deep awareness of paragraph-level context. Utilizing a self-attention-driven pointer network, the pointer-guided SO task challenges the model to reconstruct the original sequence of shuffled text segments (see Figure 1). This complex task enables the model to develop a nuanced comprehension of narrative flow, coherence, and contextual relationships, significantly enhancing its ability to understand and represent paragraph-level context when combined with standard pre-training techniques like masked language modeling.

To complement our pre-training methodology, we further introduce dynamic sampling during the fine-tuning phase. This sampling technique increases the diversity of training instances across epochs, thereby improving sample efficiency. Dynamic sampling is particularly advantageous for smaller fine-tuning datasets characterized by long documents, where it effectively mitigates the risk of overfitting and promotes better generalization.

We demonstrate the effectiveness of these contributions through extensive experiments. We show that models pre-trained with our pointer-guided SO task consistently outperform competing baselines and raise the state-of-the-art across various datasets and tasks in the scientific literature and financial reporting domain. Furthermore, the model-agnostic nature of our methodology positions it to capitalize on future advancements in language model design, promising further improvements in paragraph-level text representation.

In summary, our work not only introduces a novel and effective methodology for enhancing paragraph-level embeddings but also establishes a new benchmark for sequential text classification. By opening new avenues for research in leveraging document structure for enhanced language modeling, our work marks a significant step forward in the ongoing evolution of NLP, with substantial potential impact on retrieval-based applications like semantic text search.

In the following, we first review related work, before describing our modeling approach in Section 3. In Section 4, we outline our experiments, describe our datasets, and discuss the results. Section 5 then draws a conclusion and provides an outlook into conceivable future work.

2 Related Work

Traditional language modeling tasks such as masked language modeling (MLM) and next token prediction have been instrumental in learning token-level representations. Models like RoBERTa [23], ELECTRA [8], and GPT variants [1,4,31] have shown significant success in these areas. However, these models often lack mechanisms to enforce the learning of meaningful segment-level representations, crucial for understanding paragraph-level context. BERT [14] introduced the next sentence prediction (NSP) task to bridge this gap, but its simplicity limited its effectiveness. Subsequent models, such as RoBERTa, abandoned NSP due to its limited contribution to model performance. Unlike these approaches, our work introduces a pointer-guided segment ordering methodology that directly leverages the inherent structure of textual data, offering a novel way to enhance paragraph-level understanding without relying on external data sources like Wikipedia article links in LinkBERT [37] or knowledge-graph reasoning [36]. The underlying architecture of our method, the pointer network [33], has been successfully used for stand-alone sequence ordering tasks, as demonstrated by [6,10,24]. However, to the best of our knowledge we are the first to employ a novel self-attention-driven pointer network for segment ordering in conjunction with large language model pre-training.

We evaluate our pre-training technique on several sequential text classification tasks. Previous studies have tackled these challenges with domain-adapted and fine-tuned BERT models [9,17] and the incorporation of hierarchical LSTMs, attention mechanisms, and CRF layers for improved sequential label dependency handling [3,19,29,35].

3 Methodology

In this section, we provide a comprehensive description of our methodological contributions aimed at enhancing the contextual sensitivity of paragraph-level text representations, as well as their optimization for various downstream applications. Initially, we describe our novel “pointer-guided segment ordering” approach, a versatile pre-training strategy that employs a self-attention-driven pointer network to accurately restore the original sequence of shuffled text segments. Subsequently, we detail our fine-tuning methodology, which incorporates dynamic sampling to augment the diversity of training instances throughout successive training epochs, thereby improving sample efficiency.

3.1 Pointer-guided Segment Ordering

A text document is inherently composed of consecutive text segments, which can range from whole paragraphs and individual sentences to enumerations, tables, and headlines. These segments are typically contextually interdependent, forming a coherent narrative in various types of documents, such as news articles, fiction novels, annual reports, or legal contracts.

To capture the essence of this structural coherence, we propose a novel self-supervised pre-training technique denoted as pointer-guided segment ordering (SO) that is capable of leveraging large amounts of unlabeled text data to infuse language models with additional embeddings for individual text segments. Concretely, we employ a self-attention-based pointer network to reconstruct the original order of a randomly shuffled sequence of text segments. This non-trivial pre-training task becomes exponentially more complex as the number of segments increases. Given a document of N consecutive text segments the number of possible segment permutations grows factorially to $N!$. This inherent complexity requires the model to gain a deep contextual understanding, picking up on nuanced intricacies like coherence, chronological order, and causal relationships to ensure that the narrative flows logically and maintains continuity from beginning to end.

To address the fact that transformer-based language models [32] typically exhibit an upper limit on the maximum token context size³, denoted as C , we start with dissecting long text documents into individual training samples. Concretely, a training sample consists of K text segments s , where K is the maximum number of segments that fit within the language model’s context window. Each segment is appended with a special delimiting token, [SEP], that indicates the segment’s end. Note the value of K varies for each sample, depending on the token length of the individual segments.

We enable the segment ordering task by randomly shuffling the segments within each training sample before encoding the entire sequence with a bidirectional language model denoted as BiLM. Specifically, we first apply WordPiece

³ The self-attention mechanism incurs a computational cost that scales quadratically in sequence length, imposing practical limits on the processable context size.

[28] tokenization to transform an exemplary input sample consisting of K segments into a sequence of sub-word tokens $t = ([\text{CLS}], s_1, [\text{SEP}]_1, s_2, [\text{SEP}]_2, \dots, s_K, [\text{SEP}]_K)$. $[\text{CLS}]$ denotes the special start of sequence token and an individual segment $s_i = (t_1, \dots, t_m)$ consists of m sub-word tokens, where m can differ between segments.

We couple our segment ordering pre-training task with masked language modeling (MLM) to enhance the model’s understanding of context and word relationships. In line with the insights from [11] and [34], we implement whole word masking and mask 15% of randomly selected whole words. For the remaining MLM pre-training methodology of predicting the correct sub-words from a given vocabulary for all masked tokens we refer to [14].

The tokenized, masked, and permuted input sequence is then encoded by a BiLM, which yields a series of d -dimensional hidden state vectors $\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_T) \in \mathbb{R}^{T \times d}$ corresponding to each token t_i for a sequence of length T .

For the segment reordering task, we collect the hidden states corresponding to the $[\text{SEP}]$ tokens, denoted as $\mathbf{H}_{[\text{SEP}]} = (\mathbf{h}_{[\text{SEP}]}^1, \mathbf{h}_{[\text{SEP}]}^2, \dots, \mathbf{h}_{[\text{SEP}]}^K) \in \mathbb{R}^{K \times d}$. We add learnable absolute positional embeddings $\mathbf{E} = (\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_K)$ to each segment hidden state, yielding the enhanced segment representations $\mathbf{H}'_{[\text{SEP}]} = \mathbf{H}_{[\text{SEP}]} + \mathbf{E}$. Naturally, the added positional bias encodes the new segment position after shuffling, preventing the reordering task from being compromised.

Subsequently, we pass $\mathbf{H}'_{[\text{SEP}]}$ to a pointer network [33], which is particularly suited for our SO task due to the varying number of segments per sample K , which precludes the use of a static output layer with a fixed number of classes. The network calculates each segment’s probability distribution over the original segment positions using a multiplicative self-attention mechanism, defined as follows:

$$\mathbf{A} = \text{softmax} \left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{q}} \right), \quad \mathbf{Q} = \mathbf{H}'_{[\text{SEP}]} \mathbf{W}_{\text{query}}^\top, \quad \mathbf{K} = \mathbf{H}'_{[\text{SEP}]} \mathbf{W}_{\text{key}}^\top, \quad (1)$$

$$\mathbf{W}_{\text{query}} \in \mathbb{R}^{q \times d}, \quad \mathbf{W}_{\text{key}} \in \mathbb{R}^{q \times d}, \quad \mathbf{Q} \in \mathbb{R}^{K \times q}, \quad \mathbf{K} \in \mathbb{R}^{K \times q}, \quad \mathbf{A} \in \mathbb{R}^{K \times K}, \quad (2)$$

where $\mathbf{W}_{\text{query}}$ and \mathbf{W}_{key} are learnable query and key weight matrices, $q = d/4$ is their respective row dimension, and \mathbf{A} is the row-stochastic⁴ square attention matrix, with each element a_{ij} denoting the predicted probability that segment i originated from position j . It follows that the predicted position of segment i equals $\hat{y}_i = \arg \max_j (\mathbf{a}_i)$.

The loss for the segment ordering task is computed using negative log-likelihood, $\mathcal{L}_{\text{SO}}(\mathbf{A}, \mathbf{y}) = -\sum_{i=1}^K \log(a_{i, y_i})$, where y_i denotes the ground truth position of segment i .

3.2 Sample-efficient Fine-Tuning using Dynamic Sampling

Based on the previously detailed concept of combining multiple text segments to improve contextual understanding, this section focuses on the associated benefits in sample efficiency and introduces dynamic sampling to enhance data diversity.

⁴ $\sum_{j=1}^K a_{ij} = 1 \quad \forall i \in \{1, 2, \dots, K\}$.

Traditional text classification fine-tuning approaches for encoder-only language models like BERT often treat each text segment as an independent sample, which can lead to sub-optimal context utilization and unnecessary computational overhead, especially for short segments. Our method dynamically combines multiple text segments into a single sample, thereby maximizing the use of the model’s context capacity C and enhancing training efficiency.

For an average segment length of \bar{T} tokens, the maximum number of segments per sample $K = \lfloor C/\bar{T} \rfloor$ represents the efficiency gain factor, which quantifies the improvement over processing segments individually. This gain is more evident when using large batch sizes B , as the longest sample in a batch dictates the memory and computational requirements.

A drawback of uniting multiple segments in one sample is the reduction in sample diversity, which is particularly problematic in small datasets. To mitigate this issue and promote sample diversity, we introduce dynamic sampling for fine-tuning in scenarios with scarce data. Instead of deterministically merging the maximum number of consecutive segments K , we randomly select the number of combined segments L , sampling from a uniform distribution $\mathcal{U}(L_{\min}, K)$, where L_{\min} denotes the minimum number of merged segments. While this reduces the expected computational efficiency gain per sample, it introduces beneficial randomness into the training process. By exposing the model to varying consecutive segment combinations of different length during each epoch, we encourage better generalization and reduce the risk of overfitting.

Subsequently, our experiments validate that combining segment order pre-training with sample-efficient fine-tuning using dynamic sampling significantly enhances performance in downstream text classification tasks that require a comprehensive understanding of complex and extended document structures.

4 Experiments

We split our experiments in two parts. First, we focus on our pointer-guided pre-training setup before quantitatively evaluating its impact on five downstream tasks requiring sequential text classification.

The experiments were conducted on a GPU cluster equipped with eight 32GB Nvidia Tesla V100 GPUs. The cumulative pre-training duration of all models amounted to 380 GPU hours. Our code is implemented in PyTorch, with the initial weights of pre-trained models being loaded from HuggingFace. We open-source our code base on GitHub⁵.

4.1 Pre-Training

In the following, we briefly introduce our pre-training datasets and discuss the overall training setup including baselines and results.

⁵ <https://github.com/LarsHill/pointer-guided-pre-training>.

Table 1. Descriptive statistics of pre-training datasets with document, segment, sample, and token counts in English and German, including total and average values (M = million, B = billion). Token and sample statistics are calculated based on the multilingual word-piece vocabulary, custom-100K (see Table 2), created from all pre-training datasets.

	Wikipedia		Bundesanzeiger	News		Sum
	EN	DE	DE	EN	DE	
Documents (M)	5.85	2.44	1.91	2.02	0.65	12.88
Segments (M)	99.37	19.63	85.06	36.25	17.51	257.81
Samples (M)	13.83	4.80	9.53	5.23	1.23	34.62
Tokens (B)	4.48	1.62	3.74	1.97	0.44	12.24
Tokens (%)	36.56	13.26	30.52	16.10	3.56	100

Data Table 1 details the mixture of our pre-training datasets and reports various descriptive statistics like the number of tokens and segments per dataset.

The **Wikipedia** datasets comprise 5.9 (English) and 2.4 (German) million articles respectively that were directly retrieved from their rendered HTML pages. In contrast to the commonly used Wikimedia XML dumps, our corpus resolves Wikipedia’s templating syntax embedded in the dumps, and thus represent the articles in their original form leading to improved data quality.

Bundesanzeiger contains 1.9 million German corporate annual reports from the Bundesanzeiger, a platform where German companies are mandated to publish their legally required documents. Compared to Wikipedia the average document length is roughly three times higher resulting in around 2,000 tokens per report.

Lastly, we include two proprietary datasets of English (2 million) and German (650 thousand) **news** articles that raise the total number of pre-training tokens to 12.24 billion.

For all datasets, we employ a 90-10 split for training and validation. We parse raw HTML articles using the lxml Python library, distinguishing headlines, paragraphs, tables, enumerations and more.

Training Setup and Results We evaluate the efficacy of our pointer-guided segment ordering (SO) task by pre-training and fine-tuning three variants of the BERT language model: BERT, RoBERTa, and our proposed PointerBERT. The BERT model adheres to the original design by [14], employing self-supervised MLM and next sentence prediction (NSP). RoBERTa [23] modifies the BERT pre-training scheme by omitting NSP and maximizing the use of context by concatenating multiple text segments. PointerBERT extends RoBERTa with the inclusion of our SO task, as detailed in Section 3.1. Note the application of SO is architecture agnostic and can be used to generally enhance the paragraph-level contextual comprehension of bidirectional encoder language models like RNNs [7], DeBERTa [15] and Electra [8].

Table 2. Training configurations and validation accuracies for all language model variations and their pre-training tasks, masked language modeling (MLM), next sentence prediction (NSP) and segment ordering (SO). The scores represent averaged batch accuracies across the validation set.

Architecture	Datasets	Pre-trained	Train steps	Tokenizer	Accuracy \uparrow (%)		
					MLM	NSP	SO
BERT	wiki-en	✗	1×10^5	bert-cased	30.14	73.85	—
RoBERTa	wiki-en	✗	1×10^5	bert-cased	44.77	—	—
PointerBERT	wiki-en	✗	1×10^5	bert-cased	43.10	—	34.90
RoBERTa	all	✗	2×10^5	custom-100K	57.66	—	—
PointerBERT	all	✗	2×10^5	custom-100K	55.11	—	39.49
PointerSciBERT	wiki-en	✓	1×10^5	scibert-uncased	58.27	—	52.45
PointerBERT	all-de	✓	1×10^5	bert-cased _{dbmdz}	73.62	—	57.35

Table 2 presents the pre-training configurations for each model variant. All variants are based on Google’s BERT_{BASE} architecture as encoding backbone, differing only in their tokenizer and vocabulary construction.

Concretely, we distinguish three scenarios. First, we train each model from scratch on the English Wikipedia corpus (wiki-en) using Google’s bert-base-cased tokenizer. The validation results demonstrate that PointerBERT correctly reorders shuffled segments in 35% of cases. Although this may appear modest, it is significantly higher than the baseline random guess accuracy of approximately $1/5040 \approx 2 \times 10^{-4}$, given an average of 7 segments per sample (see Table 1) and 5040 possible permutations. Importantly, the inclusion of SO does not compromise MLM performance, with only a marginal decrease in validation accuracy. Lastly, we observe that combining MLM with NSP significantly diminishes MLM performance, likely due to the reduced sample efficiency and the underutilization of the model’s context capacity.

Second, we train RoBERTa and PointerBERT on the combined multilingual datasets, denoted “all”, using a newly developed 100K token word-piece vocabulary created from the pre-training data. This step aims to assess the impact of SO in a multilingual context.

Third, we build upon the pre-trained checkpoints of Allen AI’s SciBERT [2] and the German BERT model released by the MDZ Digital Library team (dbmdz)⁶ and continue pre-training, incorporating both MLM and SO. Here we aim to demonstrate the applicability of SO for already pre-trained language models and show that continuous pre-training with MLM and SO not only further increases the MLM performance but also manages to induce improved paragraph-level text understanding thanks to pointer-guided SO.

⁶ <https://huggingface.co/dbmdz/bert-base-german-cased>.

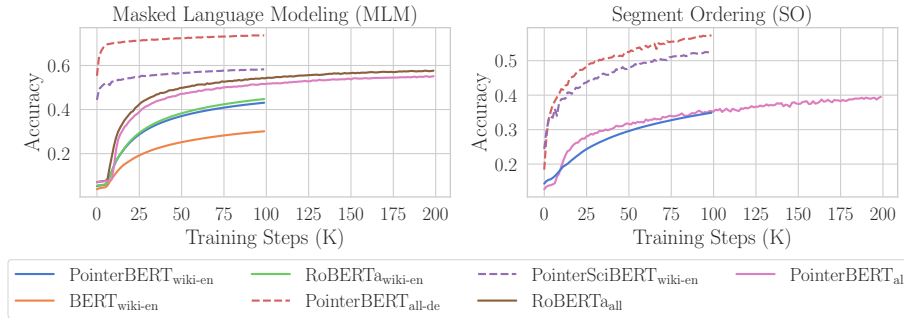


Fig. 2. Pre-training progress for all model variants, showcasing validation accuracy curves for masked language modeling (MLM) and segment ordering (SO).

All models are trained using gradient descent with the AdamW optimizer [25], featuring a 10% linear warmup and a decaying learning rate schedule. We apply a weight decay of 0.01 and clip gradients at a maximum value of 1. The peak learning rate is set to 1×10^{-4} , with a batch size of 16 and gradient accumulation over 4 steps, resulting in an effective batch size of 64. Model performance is evaluated on a hold-out validation set every 5,000 steps. Figure 2 provides a detailed view of the training progress, depicting both MLM and SO validation losses and accuracies.

Notably continued pointer-guided pre-training not only further improves MLM accuracy but also yields robust SO performance, which translates into enhanced results on downstream classification tasks, as discussed in the following section.

4.2 Downstream Fine-Tuning for Sequential Text Classification

In this section, we delve into the application of our pre-trained models to a series of fine-tuning downstream tasks that necessitate sequential text classification. We explore a diverse array of datasets, spanning both scientific literature and financial reporting domains, to assess the models’ capabilities in categorizing text segments in different scenarios.

Datasets Our evaluation encompasses five datasets, three from the scientific literature corpus and two from the financial reporting sector, each presenting unique challenges for text segment classification.

Within the scientific literature, we examine the **CSAbstract** dataset [9], which comprises 2,189 computer science abstracts with sentences annotated to discern their rhetorical roles. The **PubMed-RCT** dataset [13] extends our evaluation to 20,000 biomedical abstracts from PubMed, segmented into five rhetorical categories, following the preprocessing methodology outlined by [19]. The **Nicta** dataset [21] further contributes with 1,000 biomedical abstracts, where sentences

Table 3. Descriptive statistics of scientific and financial fine-tuning datasets. Sample statistics are calculated based on the custom-100K vocabulary (see Table 2).

	Scientific Abstracts			Financial Reports	
	CSAbstract	PubMed-RCT	Nicta	IFRS _{EN}	GRI _{DE}
Documents	2189	20,000	1000	45	92
Segments	14,708	240,386	9771	19,573	89,412
Samples	2191	23,289	1061	4773	21,306
Is multi-label	✗	✗	✗	✓	✓
Classes	5	5	6	543	89
Segments labeled (%)	100.00	100.00	100.00	78.37	8.57

are classified according to the PICO framework (Population, Intervention, Comparison, Outcome) [27].

Transitioning to the financial sector, we incorporate the GRI_{DE} dataset [17], which consists of 92 sustainability reports from leading German companies. These reports were initially obtained as PDFs from corporate websites and subsequently annotated by experts to correspond with the Global Reporting Initiative (GRI) standards, covering 89 indicators across economic, environmental, and social dimensions. The IFRS_{EN} dataset is composed of 45 English annual reports adhering to the International Financial Reporting Standards (IFRS). Provided by an auditing firm, these reports contain annotations that map paragraphs to 543 distinct legal requirements, with some paragraphs addressing multiple items.

A summary of the datasets’ descriptive statistics is presented in Table 3. The scientific abstract datasets (CSAbstract, PubMed-RCT, and Nicta) contrast with the financial report datasets (GRI_{DE} and IFRS_{EN}) in terms of structure. The former category includes a higher volume of documents, each with approximately 10 sentences, typically fitting within the model’s context window of 512 tokens. This is reflected in the average number of samples per document being close to 1 for these datasets. They feature a smaller set of categories and require multi-class classification, where every text segment is annotated and assigned to a single category.

Conversely, the financial datasets contain fewer but significantly longer documents, averaging over 400 segments. The resulting classification complexity is further amplified by a larger number of checklist categories and annotation scarcity, with only 8.5% of paragraphs in the GRI_{DE} dataset linked to a GRI requirement. Moreover, both datasets contain segments that refer to multiple checklist items, making this task a multi-label classification challenge that resembles the difficulty of information retrieval due to the severe class imbalance and annotation sparsity.

The variation in class distribution across all datasets is graphically depicted in Figure 3. For all datasets, we adhere to the established training, validation, and test splits introduced in prior work. For the new IFRS_{EN} dataset, we employ a random split of 35 training, 5 validation, and 5 test documents.

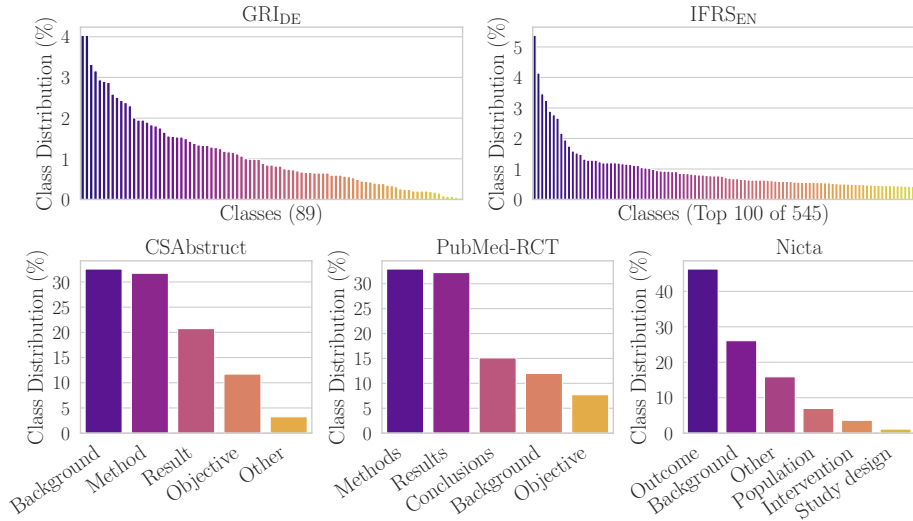


Fig. 3. Class distributions across all datasets showcasing label imbalances.

Baselines and Classification Tasks In the following comparative analysis, we benchmark our PointerBERT model variants against the various pre-trained baselines described in Section 4.1. In contrast to the other baselines, BERT_{wiki-en} processes each text segment individually and we utilize the hidden state vector of its special classifier token [CLS] as input for the downstream classification head. The remaining RoBERTa-based models handle multiple segments at a time, which is why we leverage the hidden states of their segment separating special tokens [SEP] for subsequent category prediction. The final output layer of each model differs depending on the dataset and its respective classification task. The scientific abstract datasets require multi-class classification which implies a softmax output layer that uniquely maps each segment s_i to a single category $r_j \in \mathcal{R}$. Conversely, the financial report datasets necessitate a sigmoidal output layer for multi-label classification, where a segment s_i receives relevance scores for all checklist requirements in \mathcal{R} .

We also compare our methodology with external state-of-the-art models that have not been pre-trained by us. For the scientific abstract datasets, we draw comparisons with [9] who utilize a pre-trained SciBERT [2] model and fine-tune it in the fashion of our RoBERTa baselines using the model’s [SEP] tokens as input for the classification layer. Additionally, we report results for the latest sequential sentence classification models [3,19,29,35] that are particularly optimized for incorporating sequential label dependencies while decoding. Lastly, we compare our methodology with a dedicated recommender model for sustainability reports that was introduced by [17]. It leverages a pre-trained BERT architecture equipped with a multi-layer perceptron (MLP) to identify the most relevant checklist requirements for each text segment in the GRI_{DE} dataset.

Table 4. Selected hyperparameters per dataset for our PointerBERT models based on the best validation-set micro F_1 and MAP@3 performances.

Hyperparameter	CSAbstract	PubMed-RCT	Nicta	IFRS _{EN}	GRI _{DE}
Dropout	0.1	0.2	0.2	0.2	0.2
Batch size	8	4	4	4	4
Learning rate	1×10^{-4}	5×10^{-5}	5×10^{-5}	5×10^{-5}	1×10^{-5}
Epochs	2	2	3	30	3
Loss weighting	\times	\times	\times	\checkmark	\checkmark
Random oversampling	\times	\times	\times	\times	\checkmark
Dynamic sampling	\times	\times	\times	\checkmark	\checkmark
Classification head	RNN	Linear	Linear	Linear	Linear
Label embedding dim	32	—	—	—	—

Training Setup For all models and datasets, we conduct an exhaustive grid search across a wide range of hyperparameters to identify the best parameter combinations, evaluated on the hold-out validation set micro F_1 (scientific abstracts) and mean average precision (MAP) @3 scores (financial documents). Initially, we perform a broad parameter sweep to establish a viable starting point for each architecture and dataset, followed by a more detailed fine-tuning within the proximity of these initial parameters. The outcome of this rigorous process is detailed in Table 4, presenting the optimal configurations for our PointerBERT model across datasets.

In the following, we highlight a few insights from Table 4. First, we compare different classification heads for the single-label prediction tasks. Besides a standard linear output layer that classifies each segment simultaneously, we evaluate the performance of employing a gated recurrent unit (GRU) [5] that incorporates the previously predicted label information for the subsequent segment prediction (see [16] for more details). Surprisingly, this more elaborate decoding method only slightly improves results for CSAbstract, which indicates that label dependencies are already sufficiently encoded in the separator token hidden states.

Second, we mitigate the challenges of annotation scarcity and data imbalance in the financial report datasets, by utilizing class-based loss weighting, adjusting the binary cross entropy loss according to the segments’ inverse class frequencies. Following [16], we also adopt random oversampling for the GRI_{DE} dataset, enhancing model exposure to annotated segments.

Third, we employ dynamic segment sampling (Section 3.2), which increases sample diversity across epochs for models that are capable of processing multiple segments together. This sampling technique proves especially useful for the financial report datasets characterized by a small number of long documents. For our experiments we set the minimum number of randomly selected segments per sample to $L_{\min} = 3$. We refrain from applying dynamic sampling on the scientific abstract datasets because of their substantially larger size and almost all documents comfortably fitting within our models’ 512-token context window.

Table 5. Test set results for sequential text classification on scientific abstract and financial document datasets. PointerBERT outperforms all competing baselines in micro and macro F₁ score as well as top 3/5 mean average precision (MAP). We report mean (best scores in bold) and standard deviation values from 10 independently seeded training runs for robust test set evaluation.

in %	Scientific Abstracts						Financial Reports			
	CSAbstract		PubMed-RCT		Nicta		IFRS _{EN}		GRI _{DE}	
	F ₁		F ₁		F ₁		MAP		MAP	
	Micro	Macro	Micro	Macro	Micro	Macro	@3	@5	@3	@5
Architecture										
Jin et al. [19]	81.30	—	92.60	—	84.70	—	—	—	—	—
Cohan et al. [9]	83.10	—	92.90	—	84.80	—	—	—	—	—
Yama. et al. [35]	—	—	93.10	—	84.40	—	—	—	—	—
Shang et al. [29]	—	—	92.80	—	86.80	—	—	—	—	—
Brack et al. [3]	—	—	93.00	—	86.00	—	—	—	—	—
Pointer- SciBERT _{wiki-en}	83.21	82.05	93.56	89.56	85.07	76.22	62.75	63.99	—	—
	±0.31	±0.57	±0.10	±0.16	±0.33	±1.13	±1.00	±0.94		
Hillebrand et al. [17]	—	—	—	—	—	—	—	—	33.37	35.29
									±0.95	±0.91
Pointer- BERT _{all-de}	—	—	—	—	—	—	—	—	34.25	36.19
									±1.07	±1.06
BERT _{wiki-en}	73.25	73.83	85.98	80.72	72.74	65.80	56.77	57.61	—	—
	±0.72	±0.60	±0.07	±0.09	±0.48	±0.82	±0.71	±0.67		
RoBERTa _{wiki-en}	81.21	79.37	92.48	88.17	80.98	69.92	54.68	56.05	—	—
	±0.70	±1.02	±0.05	±0.10	±0.56	±1.28	±0.94	±0.90		
Pointer- BERT _{wiki-en}	81.91	80.42	92.63	88.29	81.25	70.82	57.17	58.43	—	—
	±0.41	±0.66	±0.06	±0.08	±0.27	±0.73	±1.17	±1.11		
RoBERTa _{all}	81.60	79.92	92.77	88.53	81.67	70.42	59.07	60.47	27.83	29.88
	±0.46	±0.72	±0.10	±0.12	±0.38	±0.90	±0.72	±0.89	±1.79	±1.91
PointerBERT _{all}	81.82	80.36	92.87	88.65	81.92	71.52	59.50	60.52	28.84	30.99
	±0.71	±0.83	±0.10	±0.18	±0.33	±1.06	±0.94	±0.87	±1.80	±1.75

All additional training parameters not specified in Table 4 align with the pre-training configurations described in Section 4.1, except for gradient accumulation, which is not needed during fine-tuning due to smaller batch sizes. Also, to ensure a level playing field, all pre-trained baseline models undergo the same hyperparameter selection process and benefit from the described training enhancements, which enables fair test-set evaluations.

Results Table 5 presents the test-set performance metrics for all evaluated models across each dataset. To ensure reliability of our results, each model-dataset pairing has been evaluated 10 times using different seeds, with the average and standard deviation of these runs reported. We categorize the comparisons into three distinct groups. In the first category, our continuously pre-trained Point-

erBERT models are compared with current state-of-the-art models, revealing that the English PointerSciBERT model surpasses previous benchmarks on two out of three scientific abstract datasets. Notably, we did not focus on incorporating improved decoding mechanisms like advanced attention and CRF output layers, as included in the external baselines. Joining these methods with our PointerBERT methodology would likely further improve results. Additionally, the German PointerBERT_{all-de} model exhibits enhanced retrieval performance on the German GRI_{DE} sustainability report dataset. Due to the distinct vocabularies and monolingual pre-training approach, English models were not evaluated on German datasets and vice versa.

The second comparison group evaluates the PointerBERT architecture against RoBERTa and BERT models in a controlled setting. Both sets of models have been identically pre-trained from scratch, allowing any performance discrepancies to be attributed to architectural differences. This comparison underscores the superiority of our pointer-guided SO task, with PointerBERT consistently outperforming the other architectures across all English datasets.

In the final comparison group, the multilingual PointerBERT_{all} model is pitted against the RoBERTa_{all} baseline. Both models have been pre-trained from scratch as well using the same datasets, vocabulary, and training configurations (see Table 2. Evaluated across all five datasets, the PointerBERT_{all} model consistently outperforms the RoBERTa_{all} baseline, showcasing the efficacy of our pointer-guided architecture.

Overall, our findings demonstrate that the pointer-guided SO methodology, combined with dynamic sampling for efficient fine-tuning, surpasses all competing baselines across a diverse array of datasets and classification tasks.

4.3 Limitations

Despite the promising results, our current experiments have a few practical limitations that we plan to improve upon in future work. Firstly, the models are constrained by a context window size of only 512 tokens and employ absolute positional embeddings. Incorporating advanced attention mechanisms [12], along with relative positional embeddings [30], enables our pre-training approach to accommodate longer input sequences during inference. This enhancement will not only increase the complexity and effectiveness of the SO pre-training task but also enable the model to capture more distant paragraph-level context.

Besides scaling up our pre-training methodology in terms of larger model sizes, increased number of training steps and larger datasets, we also aim to extend our evaluation and fine-tuning efforts to information retrieval and semantic search tasks [20,26]. Specifically, we seek to assess our methodology’s effectiveness in identifying semantically relevant passages from long documents in response to natural language queries. We assume that our method’s enhanced understanding of paragraph-level context and its ability to jointly embed subsequent text segments has the potential to improve semantic search and thereby retrieval augmented generation (RAG) [22].

5 Conclusion

We introduce a novel approach to enhance the contextual sensitivity of paragraph-level text representations through a pointer-guided segment ordering (SO) pre-training strategy and dynamic sampling for fine-tuning. Our methodology aims at improving the understanding of document structure and coherence, which is crucial for a wide range of downstream NLP applications, including text classification and information retrieval.

Our pre-training methodology leverages a self-attention-driven pointer network to restore the original sequence of shuffled text segments, thereby requiring the model to develop a deep understanding of narrative flow, coherence, and contextual relationships. This task, combined with masked language modeling, shows to significantly enhance the model’s ability to comprehend and represent paragraph-level context. We further establish dynamic sampling during the fine-tuning phase to increase the diversity of training instances across epochs and improve sample efficiency. This sampling technique proves particularly beneficial for small datasets with long documents, as it helps to mitigate overfitting and to foster better generalization.

Our experiments demonstrate that models pre-trained with our pointer-guided SO task outperform existing baselines across a variety of datasets and tasks. Notably, our PointerBERT models achieve superior performance on both scientific literature and financial reporting datasets, showcasing the versatility and effectiveness of our approach. Looking ahead, we aim to overcome current limitations by incorporating more sophisticated language model backbones and broadening our evaluation framework to include information retrieval and semantic search tasks.

In conclusion, our work contributes an advancement in representation learning for paragraph-level text, setting a new benchmark for sequential text classification and paving the way for future research in document structure-based language modeling.

Acknowledgments. This research has been funded by the Federal Ministry of Education and Research of Germany and the state of North-Rhine Westphalia as part of the Lamarr-Institute for Machine Learning and Artificial Intelligence, LAMARR22B.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report. arXiv:2303.08774 (2023)
2. Beltagy, I., Lo, K., Cohan, A.: Scibert: A pretrained language model for scientific text. In: Proc. EMNLP (2019)

3. Brack, A., Hoppe, A., Buschermöhle, P., Ewerth, R.: Cross-domain multi-task learning for sequential sentence classification in research papers. In: Proc. JCDL (2022)
4. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Nee-lakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. In: Proc. NeurIPS (2020)
5. Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder–decoder for statistical machine translation. In: Proc. EMNLP (2014)
6. Chowdhury, S.B.R., Brahman, F., Chaturvedi, S.: Is everything in order? a simple way to order sentences. In: Proc. EMNLP (2021)
7. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recur-rent neural networks on sequence modeling. In: Proc. NeurIPS (2014)
8. Clark, K., Luong, M.T., Le, Q.V., Manning, C.D.: Electra: Pre-training text en-coders as discriminators rather than generators. In: Proc. ICLR (2020)
9. Cohan, A., Beltagy, I., King, D., Dalvi, B., Weld, D.: Pretrained language models for sequential sentence classification. In: Proc. EMNLP (2019)
10. Cui, B., Li, Y., Chen, M., Zhang, Z.: Deep attentive sentence ordering network. In: Proc. EMNLP (2018)
11. Cui, Y., Che, W., Liu, T., Qin, B., Yang, Z.: Pre-training with whole word masking for chinese bert. IEEE/ACM TASLP (2021)
12. Dao, T., Fu, D.Y., Ermon, S., Rudra, A., Ré, C.: FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In: Proc. NeurIPS (2022)
13. Dernoncourt, F., Lee, J.Y.: PubMed 200k RCT: a dataset for sequential sentence classification in medical abstracts. In: Proc. IJCNLP (2017)
14. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proc. NAACL (2019)
15. He, P., Gao, J., Chen, W.: DeBERTav3: Improving deBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. In: Proc. ICLR (2023)
16. Hillebrand, L., Deußner, T., Dilmaghani, T., Kliem, B., Loitz, R., Bauckhage, C., Sifa, R.: Kpi-bert: A joint named entity recognition and relation extraction model for financial reports. In: Proc. ICPR (2022)
17. Hillebrand, L., Pielka, M., Leonhard, D., Deußner, T., Dilmaghani, T., Kliem, B., Loitz, R., Morad, M., Temath, C., Bell, T., Stenzel, R., Sifa, R.: sustain.ai: a recommender system to analyze sustainability reports. In: Proc. ICAIL (2023)
18. Jiang, A.Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D.S., Casas, D.d.l., Hanna, E.B., Bressand, F., et al.: Mixtral of experts. arXiv:2401.04088 (2024)
19. Jin, D., Szolovits, P.: Hierarchical neural networks for sequential sentence classifica-tion in medical scientific abstracts. In: Proc. EMNLP (2018)
20. Khattab, O., Zaharia, M.: Colbert: Efficient and effective passage search via con-textualized late interaction over bert. In: Proc. SIGIR (2020)
21. Kim, S.N., Martinez, D., Cavedon, L., Yencken, L.: Automatic classification of sentences to support evidence based medicine. In: BMC Bioinformatics (2011)
22. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.t., Rocktäschel, T., et al.: Retrieval-augmented generation for knowledge-intensive nlp tasks. In: Proc. NeurIPS (2020)
23. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: RoBERTa: A robustly optimized BERT pretraining approach. arXiv:1907.11692 (2019)

24. Logeswaran, L., Lee, H., Radev, D.: Sentence ordering and coherence modeling using recurrent neural networks. In: Proc. AAAI (2018)
25. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: Proc. ICLR (2018)
26. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. In: Proc. EMNLP. pp. 3982–3992 (2019)
27. Richardson, W.S., Wilson, M.C., Nishikawa, J., Hayward, R.S.: The well-built clinical question: a key to evidence-based decisions. ACP journal club (1995)
28. Schuster, M., Nakajima, K.: Japanese and korean voice search. In: Proc. ICASSP (2012)
29. Shang, X., Ma, Q., Lin, Z., Yan, J., Chen, Z.: A span-based dynamic local attention model for sequential sentence classification. In: Proc. ACL/IJCNLP (2021)
30. Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., Liu, Y.: Roformer: Enhanced transformer with rotary position embedding. Neurocomputing (2024)
31. Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al.: Llama 2: Open foundation and fine-tuned chat models. arXiv:2307.09288 (2023)
32. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Proc. NeurIPS (2017)
33. Vinyals, O., Fortunato, M., Jaitly, N.: Pointer networks. In: Proc. NeurIPS (2015)
34. Wettig, A., Gao, T., Zhong, Z., Chen, D.: Should you mask 15% in masked language modeling? In: Proc. EACL (2023)
35. Yamada, K., Hirao, T., Sasano, R., Takeda, K., Nagata, M.: Sequential span classification with neural semi-markov crfs for biomedical abstracts. In: Proc. EMNLP (2020)
36. Yasunaga, M., Bosselut, A., Ren, H., Zhang, X., Manning, C.D., Liang, P.S., Leskovec, J.: Deep bidirectional language-knowledge graph pretraining. In: Proc. NeurIPS (2022)
37. Yasunaga, M., Leskovec, J., Liang, P.: Linkbert: Pretraining language models with document links. In: Proc. ACL (2022)