Patent Language Model Pretraining with ModernBERT

Anonymous ACL submission

Abstract

Transformer-based language models such as BERT have become foundational in NLP, yet their performance degrades in specialized domains like patents, which contain long, technical, and legally structured text. Prior approaches to patent NLP have primarily relied on fine-tuning general-purpose models or domain-adapted variants pretrained with limited data. In this work, we pretrain 3 domain-specific masked language models for patents, using the ModernBERT architecture and a curated corpus of over 60 million patent records. Our approach incorporates architectural optimizations, including FlashAttention, rotary embeddings, and GLU feed-forward layers. We evaluate our models on four downstream patent classification tasks. Our model, ModernBERT-base-PT, consistently outperforms the general-purpose ModernBERT baseline on three out of four datasets and achieves competitive performance with a baseline PatentBERT. Additional experiments with ModernBERT-base-VX and Mosaic-BERT-large demonstrate that scaling the model size and customizing the tokenizer further enhance performance on selected tasks. Notably, all ModernBERT variants retain substantially faster inference—over 3× that of PatentBERT—underscoring their suitability for time-sensitive applications. These results underscore the benefits of domain-specific pretraining and architectural improvements for patent-focused NLP tasks.

1 Introduction

Since its release in 2018, BERT and other derivative encoder-based transformer models have become a mainstay of modern NLP research and applications (Devlin et al., 2019). Even within an ecosystem apparently dominated by generative Large Language Models (LLMs), BERT-based models continue to be used widely in both traditional encoder applications as well as within some LLM systems e.g., Retrieval Augmented Generation systems (Lewis et al., 2020; Wang et al., 2022).

Optimized primarily for general-domain text, BERT tends to underperform in specialized domains that feature distinct linguistic structures, such as legal or biomedical corpora (Beltagy et al., 2019; Limsopatham, 2021). Many such domains—including patents—exhibit idiosyncratic lexical and syntactic features, warranting domain-specific adaptation. While fine-tuning on downstream tasks remains a popular strategy, several studies have demonstrated that extending pretraining on domain-relevant corpora, or pretraining from scratch, can lead to substantial performance gains (Chalkidis et al., 2020; Rasmy et al., 2021).

Patent documents are a unique blend of legal language and technical exposition, often structured and written in ways that differ sharply from general web or news text. Tasks in this domain—including classification, retrieval, and paragraph highlighting—have been addressed using both traditional ML (Kamateri et al., 2022; Haghighian Roudsari et al., 2022) and transformer-based approaches (Lee and Hsiang, 2020; Bekamiri et al., 2024). However, the majority of BERT-based studies in this space rely on generic pretraining, which may limit their effectiveness given the domain shift.

To date, only one notable effort has reported on pretraining a BERT-style model specifically for patent documents (Srebrovic and Yonamine, 2020a). This indicates a significant gap in the literature, especially given the demonstrated benefits of domain adaptation in comparable fields (Chalkidis et al., 2020; Qiu et al., 2021; Yan and Pei, 2022). Our work aims to address this gap directly. Additionally, several technical enhancements to BERT have recently been proposed and validated, including FlashAttention (Dao et al., 2022; Shah et al., 2024), unpadding (Zeng et al., 2022), optimized MLM masking (Wettig et al., 2023), and rotary

positional embeddings (Su et al., 2024). These improvements not only accelerate training, but also improve model quality and stability (Portes et al., 2023; Warner et al., 2024).

In this work, we build on these architectural and training innovations by using the ModernBERT design in conjunction with MosaicML's Composer framework to pretrain a transformer model specifically for patent-domain tasks. Our contributions are as follows:

- **Architecture and Efficiency:** We adopt the ModernBERT architecture and incorporate recent training optimizations, such as FlashAttention (Dao et al., 2022), dynamic masking ratios (Wettig et al., 2023), and rotary positional embeddings (Su et al., 2024), as facilitated by MosaicML's Composer framework. These enhancements significantly reduce pretraining time and compute cost compared to traditional BERT, making the model both more accessible and scalable.

- **Training Data:** We construct a diverse and representative corpus that blends public patent data with proprietary patent text. The proprietary data are sections of original patent text that have been rewritten in less obscure language by subject-matter experts. This hybrid dataset enhances the model's ability to generalize across patent categories and jurisdictions.

- **Downstream Evaluation:** We benchmark our pretrained model on multiple downstream patent classification tasks and compare it against both a generic ModernBERT base model and Google's PatentBERT (Srebrovic and Yonamine, 2020a). Our results demonstrate that pretraining on domain-specific patent data improves task accuracy over ModernBERT-base and leads to more efficient fine-tuning and inference in comparison with PatentBERT. Moreover, results with ModernBERT-base-VX and Mosaic-BERT-large indicate that both architectural scaling and tokenizer customization can yield further gains on selected datasets, suggesting complementary strategies for optimizing patent-domain transformers.

2 Methods

2.1 Data and Preprocessing

Pretraining a domain-specific language model with MLM loss requires a substantial volume of domain-specific data in order to compete with or exceed the performance of more generic pretrained encoder models. We construct our dataset from ~100M patent documents (mean length ~700 words), comprising original published applications and corresponding proprietary samples extracted from the Derwent World Patents Index (DWPI)[1]. This DWPI data has been constructed by subject matter experts to provide more insightful descriptions of patent content.

To prepare the data, we applied several preprocessing and quality enhancement steps. From the broader corpus, we extracted abstracts and the first independent claim from each patent, and appended DWPI titles, abstracts, and claims by matching publication numbers. Non-essential text (e.g., figure references, non-English characters) was removed. Only the first independent claim per patent was retained.

Following strategies used in FineWeb (Penedo et al., 2024b), we applied a language filter to retain only English patents and implemented text quality and repetition filters (Rae et al., 2021), using the DataTrove toolkit[2] (Penedo et al., 2024b). Additional deduplication was critical given that patents are often clustered in families or legal hierarchies: we applied MinHash-based fuzzy deduplication (Penedo et al., 2024b), which computes n-gram signatures and filters out duplicates or near-duplicates. This procedure reduced the dataset from ~100M to ~64M unique patents.

Our final training dataset contains ~64M patents (~30.8B tokens), with a held-out test set of 3.4M patents (~1.6B tokens). This training corpus is approximately half the size of Google PatentBERT (Srebrovic and Yonamine, 2020a). Full preprocessing details are reported in Appendix B.

2.2 Tokenization and Vocabulary Construction

To support MLM pre-training from scratch on English-language patent text, we developed a domain-specific subword tokenizer for

[1] https://clarivate.com/intellectual-property/patent-intelligence/derwent-world-patents-index/

[2] https://github.com/huggingface/datatrove

MosaicBERT-large (2.2.2) using the tokenizers library (Wolf et al., 2020). The tokenizer implements a *Byte-Pair Encoding* (BPE) model (Sennrich et al., 2016) initialized with an unknown token `[UNK]`. All input text is normalized by Unicode NFKC normalization followed by lowercasing, reducing case-induced sparsity and consolidating visually or canonically distinct code points. Pre-tokenization uses whitespace splitting; punctuation remains attached to neighboring tokens but can be segmented by the subword model as needed.

The BPE vocabulary was trained from a streaming iterator over the pretraining corpus with a target size of 49,152 subwords and a minimum frequency threshold of 2. The following special tokens were reserved for MLM training: `[UNK]`, `[PAD]`, `[CLS]`, `[SEP]`, and `[MASK]`. The final vocabulary and merge rules were serialized and reused for all pretraining and evaluation. At inference time, documents are normalized, whitespace-tokenized, and mapped to subword indices; frequent technical terms and morphemes appear as single tokens, while rarer terms decompose into multiple subwords. Out-of-vocabulary character sequences are mapped to `[UNK]`.

For comparison, we also experimented with WordPiece-style tokenization (Schuster and Nakajima, 2012; Devlin et al., 2019), including section-aware variants with special markers for abstracts, claims, and DWPI text (Srebrovic and Yonamine, 2020b). These tokenizers were trained with varying hyperparameters and evaluated using average token entropy (Zouhar et al., 2023; Dagan et al., 2024) and unknown-token rates. However, preliminary end-to-end experiments showed weaker downstream performance relative to our BPE tokenizer. Since detailed results are out of scope, and all models reported in this paper use either our BPE tokenizer or ModernBERT defaults, we do not report further on these WordPiece experiments.

The other two pretrained models, ModernBERT-base-PT and ModernBERT-base-VX, employ their original publicly released tokenizers (Warner et al., 2024) without modification.

Model suffixes (corpus variants). We suffix model names to indicate the pretraining corpus variant (Table 1). **`-PT`** (*clean*) denotes pretraining on the Phase 2 cleaned and near-deduplicated patent corpus, which applies English language identification, repetition/quality filters, a FineWeb-style quality filter, and MinHash-based near-duplicate removal (Appendix B.2). **`-VX`** (*raw*) denotes pretraining on the corpus prior to Phase 2 filtering—i.e., after Phase 1 extraction/normalization and family-level deduplication only (Appendix B.1). The raw variant is larger but noisier: Phase 2 reduces tokens from $\sim$ 47.7B tokens for *ModernBERT-base-VX* to $\sim$ 31.6B tokens for *ModernBERT-base-PT* (-33.79%; Appendix B.3). See Appendix C.

2.2.1 Model Architecture

We adopt the FlexBERT-base architecture from the *ModernBERT* repository[3], which is a modernized BERT-based encoder tailored for efficient training (Warner et al., 2024). This architecture consists of 22 encoder blocks with a hidden state size of 768 and 12 self-attention heads per layer. Gaussian Error Linear Unit (GELU) is the activation function for all feed-forward layers and Layer Normalization is applied at the beginning of each sublayer to improve training stability. The original ModernBERT has a maximum sequence length of 8192 tokens but we have limited this to 1024 for our requirements.

Following MosaicBERT (Portes et al., 2023), our implementation integrates several efficiency optimizations. First, the model uses FlashAttention for the self-attention layers, enabling larger batch sizes and longer sequences (Dao et al., 2022; Shah et al., 2024). Second, we adopt ALiBi (Attention with Linear Biases) positional encoding instead of learned positional embeddings (Press et al., 2021). ALiBi adds a relative position bias to attention scores rather than using absolute position embeddings, which not only improves final model performance but also enables the model to generalize to sequences longer than those seen in training. Third, feed-forward layers are implemented with a Gated Linear Unit (GLU) (Portes et al., 2023). This implementation sees the intermediate dense layer's output split into two parts. One part is passed through a GELU activation and the other part is used as a gating signal (after a linear transformation) that multiplicatively modulates the activated part, enabling more flexible and expressive representations. Fourth, we scale residual connections by a factor of 0.5 following the Pre-LayerNorm setup, which stabilizes training in deeper architectures. Notably, dropout is removed entirely from the architecture without adverse effects on generalization or training stability. These model enhancements have been

[3] `https://github.com/AnswerDotAI/ModernBERT`

Model	Sfx	Corpus	Tokens	Definition / Notes
ModernBERT-base-PT	-PT	Clean (Phase 2)	∼31.6B	Phase 2 cleaned & near-deduplicated corpus: English LID; MassiveText/Gopher repetition/quality filters; FineWeb; MinHash (Appendix B.2). Phase-2 total ≈31.64B (Appendix B.3).
ModernBERT-base-VX	-VX	Raw (pre–Phase 2)	∼47.7B	Pre-Phase 2 corpus (after Phase 1 & family-level dedup only; Appendix B.1). More tokens, higher noise; aligns with ≈47.79B (Appendix B.3).
MosaicBERT-large (BPE)	—	Raw (pre–Phase 2)[†]	∼47.7B	Custom BPE tokenizer (§2.2); long context; trained on ∼47.7B tokens (Appendix C).

Table 1: **Model suffix legend and corpus variants.** -PT uses the Phase 2 *clean* corpus; -VX uses the *raw* corpus prior to Phase 2.

shown to improve training speed and stability without sacrificing model performance (Portes et al., 2023).

In addition to this baseline, we pretrained two further model variants. First, a ModernBERT-base-VX configuration, which retains 22 layers and 12 attention heads but introduces adjusted intermediate dimensions and GLU expansion consistent with the Composer framework. Specifically, this variant uses a hidden size of 768, intermediate size of 1152, and GLU expansion to 2304, with RoPE θ set to 160,000 and a local attention RoPE θ of 10,000. While the total parameter count remains approximately 149M, training differed slightly from the baseline in terms of optimizer configuration and batch scheduling (see Appendix C). This provides a controlled point of comparison against our domain-pretrained ModernBERT-base-PT.

Second, a Mosaic-BERT-large model was trained using our custom BPE tokenizer (section 2.2.2). This model scales to 28 encoder layers with a hidden size of 1024 and 16 attention heads, yielding a total of approximately 395M parameters. The feed-forward intermediate dimension is expanded to 2624, with GLU expansion to 5248. Like the base variants, it integrates FlashAttention, rotary embeddings, and Pre-LayerNorm residual scaling, but supports a maximum sequence length of 8192 tokens, making it particularly suited for patent texts with long contexts. Training followed a trapezoidal learning rate schedule with StableAdamW, distributed across 8×H100 GPUs, with dropout applied only to attention outputs (0.1). This model allows us to assess both the effect of scaling and the contribution of tokenizer customization.

2.2.2 Training Procedure

Pretraining used the MLM objective. We did not include the Next Sentence Prediction task as it has been shown to provide minimal downstream performance improvement (Liu et al., 2019; Izsak et al., 2021). We used a masking probability of 30% as recent findings indicate that the original probability of 15% used in BERT is suboptimal (Wettig et al., 2023). Training was optimized using StableAdamW (Wortsman et al., 2023), an enhancement of AdamW (Loshchilov and Hutter, 2018) that incorporates update clipping on a per-parameter basis similar to Adafactor (Shazeer and Stern, 2018). Hyperparameter details are provided in Appendix D.

To avoid initial instability the learning rate schedule employed used a linear warmup followed by linear decay. Specifically, the learning rate was linearly ramped up from 0 to 3×10^{-4} over the first 6% of training steps during a warmup period and then linearly decayed for the remaining 94% of training steps, reaching about 2% of the peak learning rate by the end of training. We use a global batch size of 4096 sequences per step during training. This required each GPU processing a micro-batch of 128 sequences, with gradients synchronized across GPUs prior to each optimizer update. The maximum sequence length during training was set to 1024 tokens per example.

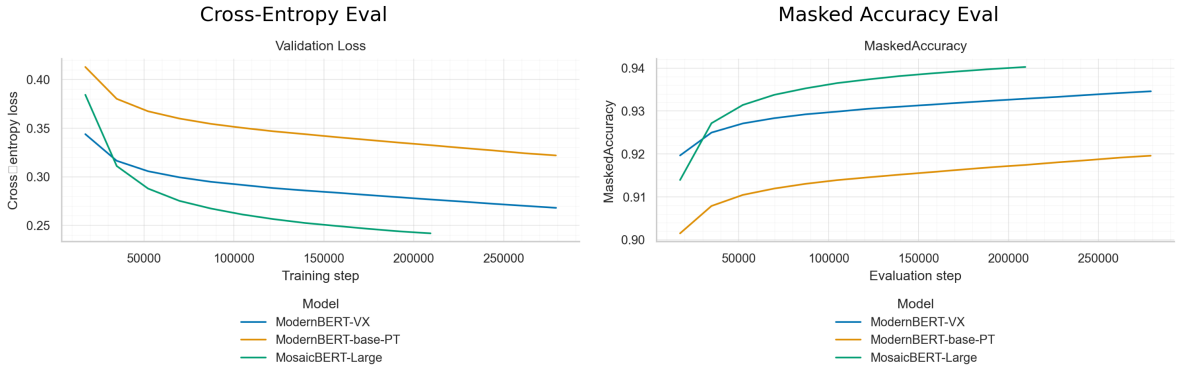

Figure 1: Pretraining loss and MLM accuracy for ModernBERT-base-PT, ModernBERT-base-VX and Mosaic-BERT-large.

Dataset	Labels	Train	Val.	Test
WIPO	14	1731	424	533
WIPOEC	43	11349	2798	3524
HUPD	8	20772	5194	6000
DatasetCLV	5	1481	371	400

Table 2: Dataset statistics for downstream evaluation tasks.

3 Results

3.1 Pretraining Evaluation

Figure 1 presents the training loss and masked accuracy for ModernBERT-base-VX, ModerBERT-base-PT and MosaicBERT-large tokenizer. The model trained with the custom BPE tokenizer exhibited a steeper decline in MLM loss in the early stages of pretraining and maintained a consistently lower loss throughout training (Figure 1). Similarly, masked token prediction accuracy rose more sharply and stabilized at a higher level compared to the WordPiece-based counterparts (Figure 1).

Pretraining results suggest that the BPE tokenizer provided a more efficient and compact representation of the patent text. BPE is known for its ability to balance vocabulary size and token granularity, particularly by effectively encoding frequent subword patterns and maintaining coherence for rare or domain-specific terms. In contrast, the standard WordPiece tokenizer—optimized for general-domain corpora—often splits complex technical terms into less informative units, making masked token prediction more difficult.

3.2 Downstream Evaluation

We validated our pretrained patent language models on downstream classification tasks using four different datasets (Table 2; Figure 3 with confidence intervals). The first two datasets come from the World International Patent Office (WIPO)[4]. Another dataset is the Harvard USPTO Patent Dataset (HUPD) (Suzgun et al., 2023)[5]. The final dataset is a proprietary dataset consisting of 5 category labels relating to data storage and networking technology. Each was split into train/validation/test sets with experiments run four times using different seeding. Reported f1 scores in Table 3 are averages of these experiments. Details on the category labels of each of the three open datasets can be found in Appendix A.

ModernBERT-base-PT improved upon the performance of ModernBERT-base on three of the four datasets. ModernBERT-base-VX achieved broadly similar results, with slightly higher performance on the DatasetCLV dataset. Mosaic-BERT-large, trained with our custom BPE tokenizer, also demonstrated strong performance, though without consistently surpassing the smaller domain-pretrained models. PatentBERT remained strongest on HUPD, reflecting its use of USPTO data during pretraining. Taken together, these results indicate that domain-specific pretraining, architectural scaling, and tokenizer customization each provide complementary advantages for patent-specific NLP, though no single strategy yet establishes a decisive state of the art across all benchmarks.

In figure 4, evaluation f1 scores across 20 epochs of finetuning are shown (evaluation loss plots

[4] https://www.wipo.int/portal/en/index.html
[5] https://huggingface.co/datasets/HUPD/hupd

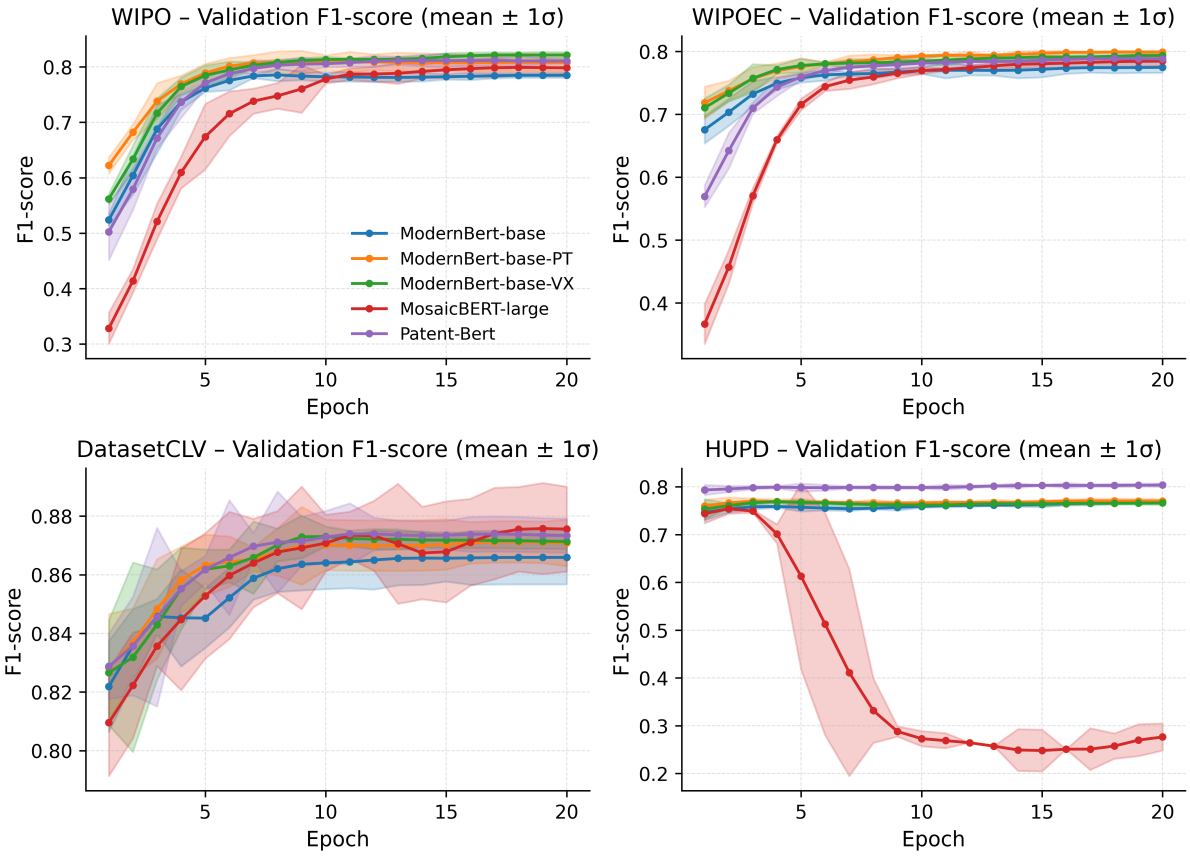

Figure 2: Validation f1-scores for finetuning tasks across 20 epochs.

Model	WIPO	WIPOEC	HUPD	DatasetCLV
ModernBERT-base	**0.806** ± 0.011	0.786 ± 0.005	0.773 ± 0.004	0.822 ± 0.019
ModernBERT-base-PT	0.802 ± 0.011	**0.814** ± 0.004	0.782 ± 0.003	0.843 ± 0.009
ModernBERT-base-VX	0.796 ± 0.003	0.801 ± 0.008	0.776 ± 0.005	0.852 ± 0.005
MosaicBERT-large	0.787 ± 0.010	0.796 ± 0.007	0.772 ± 0.006	0.850 ± 0.008
PatentBERT	0.801 ± 0.013	0.807 ± 0.006	**0.810** ± 0.001	**0.854** ± 0.012

Table 3: F1-score (mean ± std across 4 runs) for downstream classification tasks on four patent datasets.

are in Appendix E). All four plots exhibit our ModernBERT-base-PT with higher evaluation f1 scores in comparison with ModernBERT-base at all data points. This suggests there is value in domain-specific pretraininig for patent-specific downstream tasks. The plot depicting the HUPD f1 scores is noteworthy as PatentBERT outperforms the other models by a degree not seen in the other plots. We hypothize the reason for this is that USPTO data has been used during the pretraining of PatentBERT.

3.3 Finetuning Efficiency

Consistent with prior reports of ModernBERT's efficiency (Warner et al., 2024), our measurements corroborate high throughput across four patent datasets (WIPO, WIPOEC, HUPD, DatasetCLV). Per-dataset mean ± std over four seeds is reported for inference (Table 4) and training (Table 5); confidence-interval plots appear in Appendix E. *Implementation details.* In our setup, **ModernBERT-base**, **ModernBERT-base-PT**, and **ModernBERT-base-VX** use *FlashAttention*; **MosaicBERT-large** uses *ALiBi* positional bias with non-FlashAttention kernels; and **PatentBERT** uses a pre-FlashAttention (vanilla scaled dot-product) attention stack. *Aggregate speed.* Macro-averaged across datasets, the FlashAttention-based ModernBERT family attains **2.32**× higher training throughput and **3.55**× higher

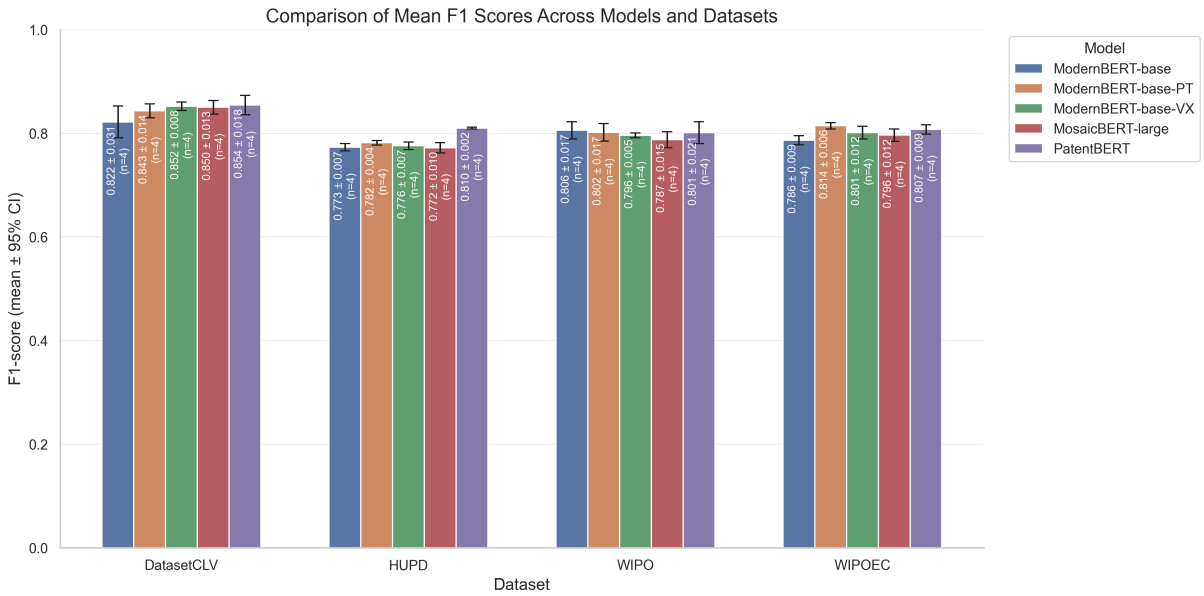

Figure 3: **Test micro-averaged F1-scores** across models and datasets, reported as mean $\pm$ 95% confidence interval computed over **4 random seeds** per (dataset, model). Confidence intervals are estimated using the *Student's t* distribution and clipped to $[0, 1]$ to respect the bounded range of F1.

inference throughput than PatentBERT (+132% and +255%, respectively), and **2.89**$\times$ / **2.93**$\times$ over MosaicBERT-large in training/inference (+189% / +193%). Looking at individual variants, **ModernBERT-base** is the fastest in training, running **1.48**$\times$ and **1.40**$\times$ faster than ModernBERT-base-PT and ModernBERT-base-VX, respectively; inference among the three ModernBERT models is essentially tied (PT and VX are $\approx$**1.02**$\times$ ModernBERT-base). Relative to non-Flash baselines, ModernBERT-base is **3.63**$\times$ faster in training and **2.90**$\times$ in inference than MosaicBERT-large, and **2.91**$\times$ / **3.51**$\times$ faster than PatentBERT (training/inference). These results support FlashAttention-based ModernBERT as a competitive alternative to other BERT variants, particularly for time-sensitive inference. *Note:* Throughput differences reflect both attention kernels and parameter counts (149M for ModernBERT vs. 340–346M for MosaicBERT/PatentBERT), so part of the gain is architectural while part is model size.

4 Limitations

Language Our pretraining corpus is limited to English-language patents. Given the global nature of patent filings, expanding to multilingual corpora could significantly improve model utility in international contexts.

MLM-only objective Following Warner et al., we rely solely on the MLM objective for pretraining. While MLM has proven effective, it does not capture sentence-level semantics or inter-document relationships. Future work could explore complementary pretraining tasks such as contrastive objectives, span prediction, or retrieval-based learning tailored for patent data.

Scaling Our model was trained on a compute budget significantly smaller than that of large-scale commercial or academic models. While this demonstrates efficiency, it may limit representational capacity compared to larger-scale alternatives. Additionally, despite pretraining on 60M patents, there are potential performance increases available from further scaling of the training data.

Limited Downstream Tasks Our downstream evaluation focuses on patent classification tasks. While results are strong, they do not yet demonstrate generalization to other patent-related tasks such as similarity ranking, summarization, claim matching, or novelty detection.

5 Conclusion

In this work, we presented ModernBERT-PT, a domain-specific masked language model pretrained from scratch on a curated corpus of over 60 million patent documents. Leveraging architectural innovations such as FlashAttention, ALiBi positional embeddings, and GLU-based feed-forward layers, we demonstrated that pretraining on patent-specific data yields tangible benefits in both clas-

Model	Parameters	WIPO	WIPOEC	HUPD	DatasetCLV
ModernBERT-base-PT	149M	83.68 ± 0.09	**85.54 ± 0.15**	368.42 ± 0.33	90.99 ± 0.83
PatentBERT	346M	36.09 ± 0.06	35.50 ± 0.12	68.84 ± 0.38	35.51 ± 0.05
ModernBERT-base	149M	**84.48 ± 0.04**	62.66 ± 0.52	**377.41 ± 1.73**	**93.45 ± 0.42**
ModernBERT-base-VX	149M	84.06 ± 1.04	84.51 ± 0.38	368.09 ± 0.75	91.87 ± 0.23
MosaicBERT-large	340M	27.44 ± 0.09	26.75 ± 0.56	127.39 ± 0.60	31.74 ± 0.30

Table 4: Inference throughput (samples/sec; mean ± std over runs) for four patent datasets.

Model	Parameters	WIPO	WIPOEC	HUPD	DatasetCLV
ModernBERT-base-PT	149M	21.11 ± 0.43	21.48 ± 1.01	24.98 ± 1.58	23.33 ± 0.16
PatentBERT	346M	8.65 ± 0.14	9.29 ± 0.06	19.29 ± 0.56	8.87 ± 0.07
ModernBERT-base	149M	**21.51 ± 0.54**	**23.17 ± 0.93**	**65.04 ± 2.13**	**24.55 ± 0.20**
ModernBERT-base-VX	149M	20.85 ± 0.38	21.15 ± 0.80	30.38 ± 2.67	23.50 ± 0.13
MosaicBERT-large	340M	6.19 ± 0.12	6.53 ± 0.16	17.33 ± 0.71	6.92 ± 0.07

Table 5: Training throughput (samples/sec; mean ± std over runs) for four patent datasets.

sification accuracy and computational efficiency. ModernBERT-base-PT outperformed a general-purpose ModernBERT baseline on three out of four downstream classification tasks and achieved competitive performance with PatentBERT, while offering substantially faster inference speeds. Additional comparisons with ModernBERT-base-VX and Mosaic-BERT-large suggest that scaling model capacity and introducing customized tokenization provide further gains on certain datasets, particularly those with longer or more technical contexts. These findings highlight the value of combining domain-specific pretraining with targeted architectural and tokenization strategies for advancing patent-domain NLP.

These findings highlight the importance of domain-specific pretraining and architecture optimization for specialized text corpora like patents. While our results validate ModernBERT-base-PT as a strong foundation for patent NLP tasks, several avenues remain for future improvement. These include scaling up the training corpus, incorporating multilingual patents, and expanding evaluations to a broader set of downstream tasks such as retrieval, summarization, and novelty detection.

With repsect to tokenization, results support growing evidence that tokenizer customization, particularly with BPE, can significantly enhance pretraining efficiency and downstream performance in domain-specific NLP. Future work could explore comparisons between BPE, SentencePiece unigram models, or byte-level tokenization schemes to further isolate the effects of subword modeling strategies in specialized corpora like patents. Overall, our work underscores the value of tailored model development for unlocking the full potential of NLP in technical and legally structured domains.

6 Acknowledgements

We would like to thank *anonymous* for supporting this project and facilitating the compute resources. We would also like to thank *anonymous* and *anonymous* for their support and feedback throughout the project.

References

Hamid Bekamiri, Daniel S Hain, and Roman Jurowetzki. 2024. Patentsberta: A deep nlp based hybrid model for patent distance and classification using augmented sbert. *Technological Forecasting and Social Change*, 206:123536.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. Legal-bert: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904.

Gautier Dagan, Gabriel Synnaeve, and Baptiste Roziere. 2024. Getting the most out of your tokenizer for pre-training and domain adaptation. *arXiv preprint arXiv:2402.01035*.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35:16344–16359.

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)*, pages 4171–4186.

Arousha Haghighian Roudsari, Jafar Afshar, Wookey Lee, and Suan Lee. 2022. Patentnet: multi-label classification of patent documents using deep learning based language understanding. *Scientometrics*, 127(1):207–231.

Peter Izsak, Moshe Berchansky, and Omer Levy. 2021. How to train bert with an academic budget. *arXiv preprint arXiv:2104.07705*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Eleni Kamateri, Vasileios Stamatis, Konstantinos Diamantaras, and Michail Salampasis. 2022. Automated single-label patent classification using ensemble classifiers. In *Proceedings of the 2022 14th International Conference on Machine Learning and Computing*, pages 324–330.

Jieh-Sheng Lee and Jieh Hsiang. 2020. Patent classification by fine-tuning bert language model. *World Patent Information*, 61:101965.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Nut Limsopatham. 2021. Effectively leveraging bert for legal document classification. In *Proceedings of the Natural Legal Language Processing Workshop 2021*, pages 210–216.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Guilherme Penedo, Hynek Kydlíček, Alessandro Cappelli, Mario Sasko, and Thomas Wolf. 2024a. Datatrove: large scale data processing.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin A Raffel, Leandro Von Werra, Thomas Wolf, and 1 others. 2024b. The fineweb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*, 37:30811–30849.

Jacob Portes, Alexander Trott, Sam Havens, Daniel King, Abhinav Venigalla, Moin Nadeem, Nikhil Sardana, Daya Khudia, and Jonathan Frankle. 2023. Mosaicbert: A bidirectional encoder optimized for fast pretraining. *Advances in Neural Information Processing Systems*, 36:3106–3130.

Ofir Press, Noah A Smith, and Mike Lewis. 2021. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*.

Zhaopeng Qiu, Xian Wu, Jingyue Gao, and Wei Fan. 2021. U-bert: Pre-training user representations for improved recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4320–4327.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, and 1 others. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Laila Rasmy, Yang Xiang, Ziqian Xie, Cui Tao, and Degui Zhi. 2021. Med-bert: pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *NPJ digital medicine*, 4(1):86.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. 2024. Flashattention-3: Fast and accurate attention with asynchrony and low-precision. *Advances in Neural Information Processing Systems*, 37:68658–68685.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR.

Rob Srebrovic and Jay Yonamine. 2020a. Leveraging the bert algorithm for patents with tensorflow and bigquery. *White paper*.

Rob Srebrovic and Jay Yonamine. 2020b. Leveraging the bert algorithm for patents with tensorflow and bigquery. Technical report, Google Patents.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Mirac Suzgun, Luke Melas-Kyriazi, Suproteem Sarkar, Scott D Kominers, and Stuart Shieber. 2023. The harvard uspto patent dataset: A large-scale, well-structured, and multi-purpose corpus of patent applications. *Advances in neural information processing systems*, 36:57908–57946.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, and 1 others. 2024. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. *arXiv preprint arXiv:2412.13663*.

Alexander Wettig, Tianyu Gao, Zexuan Zhong, and Danqi Chen. 2023. Should you mask 15% in masked language modeling? In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2985–3000.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Mitchell Wortsman, Tim Dettmers, Luke Zettlemoyer, Ari Morcos, Ali Farhadi, and Ludwig Schmidt. 2023. Stable and low-precision training for large-scale vision-language models. *Advances in Neural Information Processing Systems*, 36:10271–10298.

Bin Yan and Mingtao Pei. 2022. Clinical-bert: Vision-language pre-training for radiograph diagnosis and reports generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2982–2990.

Jinle Zeng, Min Li, Zhihua Wu, Jiaqi Liu, Yuang Liu, Dianhai Yu, and Yanjun Ma. 2022. Boosting distributed training performance of the unpadded bert model. *arXiv preprint arXiv:2208.08124*.

Vilém Zouhar, Clara Meister, Juan Luis Gastaldi, Li Du, Mrinmaya Sachan, and Ryan Cotterell. 2023. Tokenization and the noiseless channel. *arXiv preprint arXiv:2306.16842*.

A Dataset Information

WIPO Vision Dataset (WIPO)

Categories:

- Adaptive Focus
- Artificial Iris
- Artificial Silicon Retina (ASR) / Retinal Prostheses
- Augmented Reality Devices
- Bionic Eye (System)
- Cortical Implants
- Drug Delivery (Vision-related)
- Hand Wearables
- Intraocular Lenses (IOL) with Sensors
- Intracorneal Lenses
- Multifocal
- Smart Eyewear
- Telescopic Lenses
- Virtual Reality Devices

WIPOEC Dataset

Categories:

- Accessories for changing body position or lifting persons – mounted in combination with a bathtub
- Accessories for changing body position or lifting persons – mounted in combination with a toilet
- Assistive Products for Animal Care
- Assistive Products for Camping
- Assistive Products for Creating Arts and Crafts
- Assistive Products for Hunting and Fishing
- Assistive Products for Play
- Assistive Products for Playing and Composing Music
- Assistive Products for Vertical Accessibility
- Bathroom Accessories
- Bathroom/Toilet Unit
- Bathtub and Accessories
- Beds and Their Accessories
- Building Structural Components
- Entry/Exit and Openings
- Environment Alarms
- Fall Detectors
- Food Preparation
- Golf
- Handrails and Grab Bars
- Laundry
- Light Fixtures
- Mountaineering
- Other Furniture Accessories
- Personal Emergency Alarm Systems and Medical Alert ID
- Portable Travel Aids
- Shower Unit and Accessories
- Shower/Bathroom and Toilet Chairs
- Sitting Arrangements and Their Accessories
- Sports Wheelchairs
- Storage

- Swimming
- Tables and Their Accessories
- Tennis or Table Tennis
- Toilet Seat and Accessories
- Urinals
- Wandering and Locating of Persons/Items
- Wash Basin and Accessories
- Winter Sports
- Workplace and Domestic Safety
- Workplace/Domestic Machinery
- Workplace/Domestic Object Conveyance, Hoisting or Repositioning, Crane
- Workplace/Domestic Object Securing, Gripping, Holding, Carrying and Handling

HUPD Dataset
Categories:

- C: Chemistry; Metallurgy
- A: Human Necessities
- H: Electricity
- G: Physics
- B: Performing Operations; Transporting
- E: Fixed Constructions
- F: Mechanical Engineering; Lighting; Heating; Weapons; Blasting
- D: Textiles; Paper

DatasetCLV

Categories: Five categories related to data storage and networking.

B Data Preprocessing

This appendix documents our two-phase preprocessing pipeline for large-scale patent texts used to pretrain masked language models. Phase 1 focuses on extraction, basic normalization, and family-level deduplication; Phase 2 performs FineWeb-inspired filtering and MinHash-based near-duplicate removal implemented with HuggingFace *DataTrove* on Databricks.

Data fields. For each record we retain: publication number, the *abstract*, the *first independent claim* (selected from the full claim set), and the *DWPI*. Phase 2 is applied **per field** (abstract, first claim, DWPI) and results are re-assembled at the record level.

B.1 Phase 1: Extraction, Normalization, and Family-Level Deduplication

Scope. We extract fields, remove markup (HTML/XML), normalize or drop corrupted characters, strip repetitive boilerplate phrases in abstracts/claims, and remove references to figures/images. We then deduplicate by patent *family*: for each family we keep the most recent publication.

Scale impact (documents). From $\approx$**150 M** English-language patents to $\approx$**70 M** unique family representatives.

Algorithm 1 Phase 1: Extract, Clean, and Family-Deduplicate (pseudocode)

Input: Patent DB with fields {`pub_no`, `abstract`, `claims`, `DWPI`, `family_id`, `pub_date`}
Output: Clean, family-deduplicated records {`pub_no`, `abstract`*, `first_ind_claim`*, `DWPI`*}
for each record r **do**
    Extract `pub_no`, `DWPI`, `abstract`, and select `first_ind_claim` from `claims`
    Clean text fields: remove HTML/XML tags; normalize/drop corrupted characters
    Remove repetitive boilerplate prefixes; drop image/figure references
end for
Group records by `family_id`; within each group, keep the record with the most recent `pub_date`
return unified table with cleaned `abstract`*, `first_ind_claim`*, `DWPI`*

B.2 Phase 2: FineWeb-Inspired Filtering and MinHash Near-Deduplication (Per Field)

Filtering (Phase 2a). For each field independently, we apply:

- **Language filtering** via a fastText-based LID model (retain English only) (Joulin et al., 2016).
- **Quality and repetition filters** inspired by MassiveText/Gopher heuristics (Rae et al., 2021).
- **FineWeb-style quality filter** to remove subtle low-quality artifacts (Penedo et al., 2024b).

Near-duplicate removal (Phase 2b). We use MinHash LSH over 5-gram shingles with 64-bit hashes and a 14×8 banding scheme (14 buckets, 8 hashes per bucket) to cluster and remove near-duplicates, keeping one representative per cluster, following DataTrove components for signatures, bucketing, clustering, and filtering (Penedo et al., 2024a).

Execution environment. The pipeline runs on **Databricks** with parallel tasks and Parquet I/O to object storage for intermediate artifacts and logs.

Algorithm 2 Phase 2: Filter and Near-Deduplicate Per Field (pseudocode)

Input: Phase 1 output (cleaned `abstract`, `first_ind_claim`, `DWPI`)
Output: High-quality, near-deduplicated fields
for each field $f \in$ {`abstract`, `first_ind_claim`, `DWPI`} **do**
    for each document d in f **do**
        if $\mathrm{LanguageID}(d) \neq$ English **then**
            discard
        end if
        if $\mathrm{RepetitionHeuristics}(d) =$ fail **then**
            discard
        end if
        if $\mathrm{QualityHeuristics}(d) =$ fail **then**
            discard
        end if
        if $\mathrm{FineWebStyleHeuristics}(d) =$ fail **then**
            discard
        end if
    end for
    Compute MinHash signatures on 5-gram shingles (64-bit); LSH-bucket into 14×8 bands
    Cluster colliding documents; keep one representative per cluster; mark others as near-duplicates
end for
Re-assemble fields at the record level; count tokens before/after deduplication

B.3 Summary of Reductions

Table 6: Dataset size reductions across phases.

Phase	Before	After	Reduction
Phase 1 (patents)	≈150,000,000	≈70,000,000	≈80,000,000
Phase 2 (tokens)	47,791,818,029	31,644,981,330	16,146,836,699 (**–33.79%**)

B.4 References for Filtering and Deduplication

We follow the large-scale text cleaning and deduplication practices documented in FineWeb (Penedo et al., 2024b), MassiveText/Gopher (Rae et al., 2021), and use fastText language identification (Joulin et al., 2016). Our implementation is built on the *DataTrove* library (Penedo et al., 2024a).

C Training Recipe

Model	N tokens	Epochs	Train Time	Training Hardware	Training Strategy
ModernBERT-base-PT	31.6B	16	2.352 day	8x H100	Distributed DataParallel
ModernBERT-base-VX	47.7B	16	3.264 day	8x H100	Distributed DataParallel
MosaicBERT-large	47.7B	12	8.707 day	8x H100	Distributed DataParallel

Table 7: Training Statistics.

D Hyperparameters

Pretraining Hyperparameters

Model	Optimizer	LR	β	ε	Warmup	MLM	Sliding Window (FA)	ALiBi start size
ModernBERT-base-PT	StableAdamW	3e-4	(0.90, 0.98)	1e-06	6%	30%	256	n/a
ModernBERT-base-VX	StableAdamW	3e-4	(0.90, 0.98)	1e-06	6%	30%	256	n/a
MosaicBERT-large	StableAdamW	2e-4	(0.90, 0.98)	1e-06	6%	30%	n/a	1024

Table 8: Pretraining Hyperparameters.

Finetuning Hyperparameters

Model	Optimizer	LR	beta	weight decay	epochs	seeds
ModernBERT-base-PT	`adamw_torch_fused`	3e-5	(0.9, 0.999)	3e-6	20	4
ModernBERT-base-VX	`adamw_torch_fused`	3e-5	(0.9, 0.999)	3e-6	20	4
MosaicBERT-large	`adamw_torch_fused`	3e-5	(0.9, 0.999)	3e-6	20	4
PatentBERT	`adamw_torch`	2e-5	(0.9, 0.999)	1e-2	20	4
ModernBERT-base	`adamw_torch`	5e-5	(0.9, 0.999)	5e-6	20	4

Table 9: Finetuning Hyperparameters.

E Results

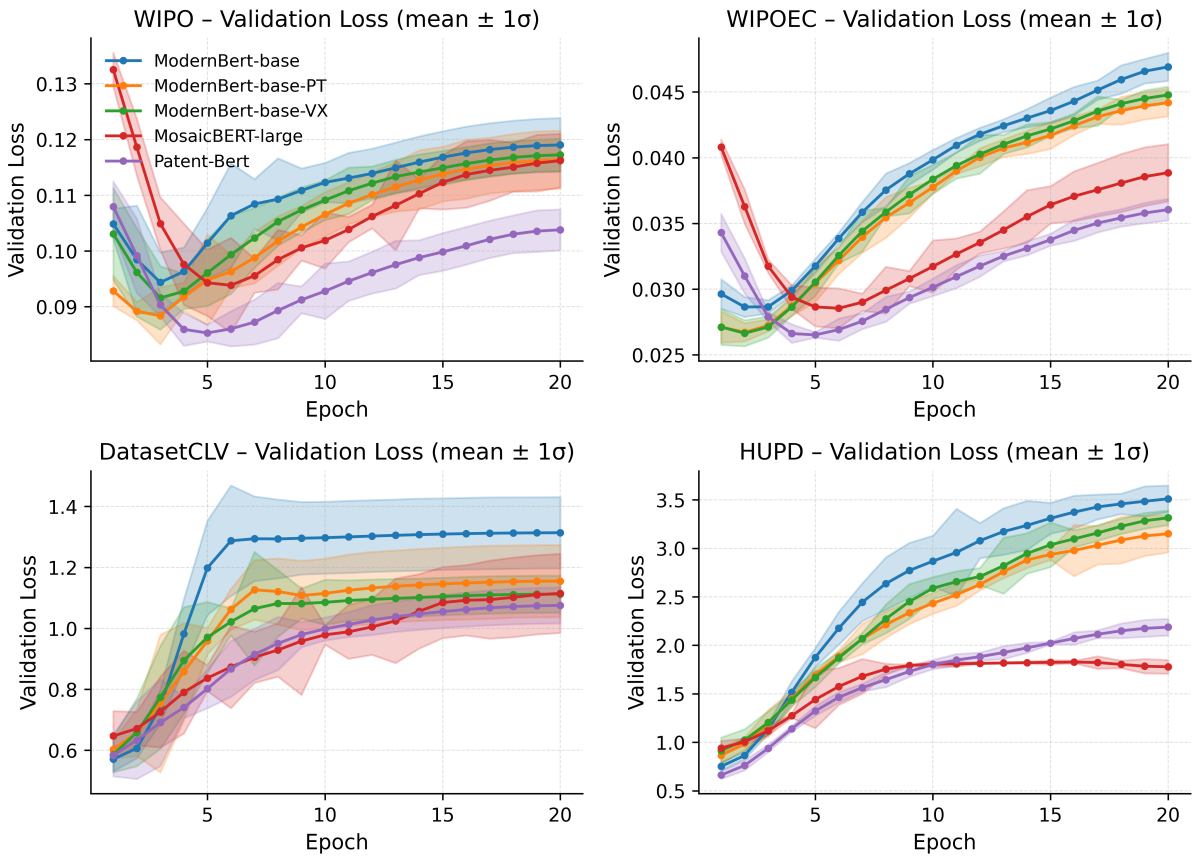

Figure 4: Validation loss curves over 20 epochs for four independent fine-tuning runs.

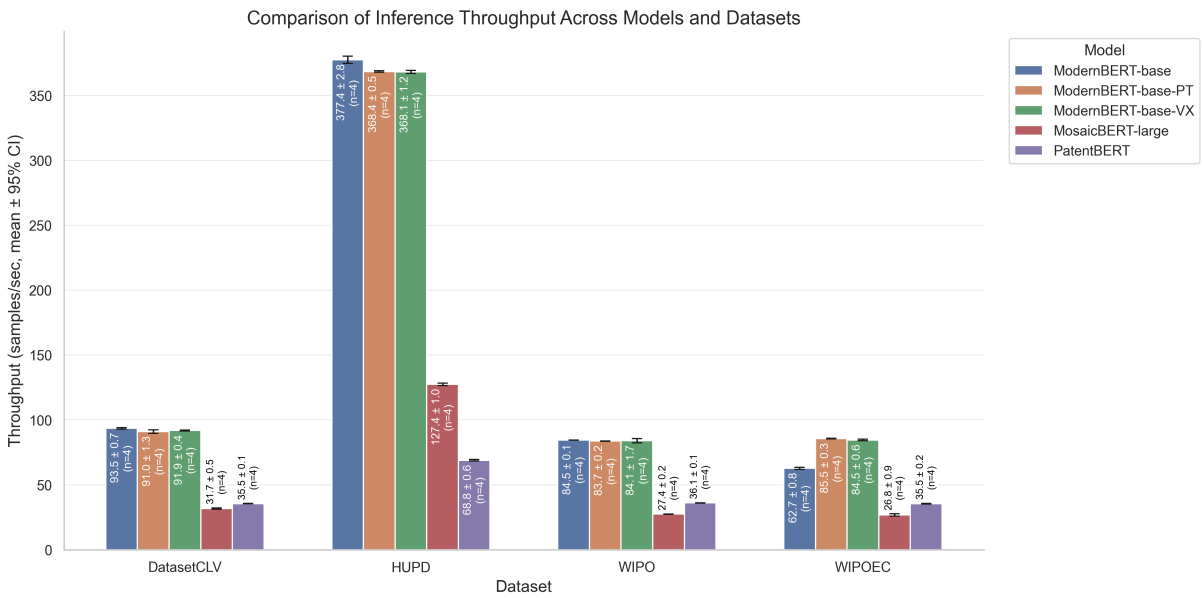

Figure 5: **Test throughput** (*samples per second*) across models and datasets, reported as mean ± 95% confidence interval computed over **4 random seeds** per (dataset, model). Confidence intervals are estimated using the *Student's t* distribution. Higher values indicate greater inference efficiency.

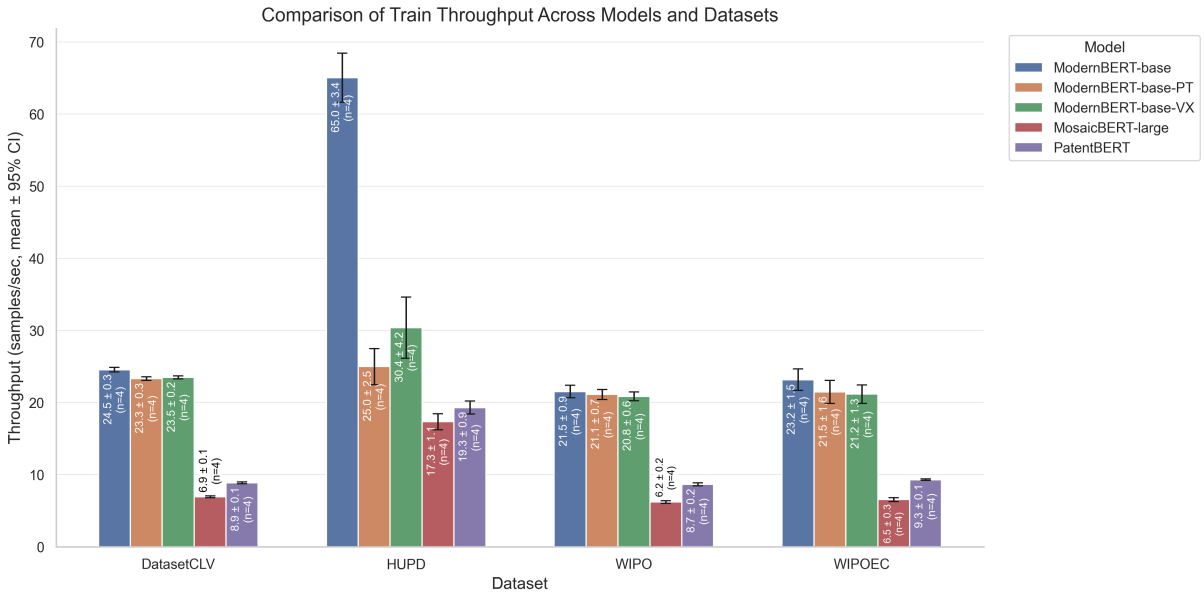

Figure 6: **Train throughput** (*samples per second*) across models and datasets, reported as mean ± 95% confidence interval computed over **4 random seeds** per (dataset, model). Confidence intervals are estimated using the *Student's t* distribution. Higher values indicate greater inference efficiency.