Words That Matter: Vocabulary Pruning in ModernBERT

Wout De Rijck, Dr. Ir. Karel D'Oosterlinck,
Prof. Dr. Ir. Thomas Demeester, Prof. Dr. Ir. Chris Develder

Abstract

Large language models like ModernBERT face deployment challenges due to their computational demands and memory footprint, limiting their use in resource-constrained environments. While various pruning techniques exist, most require resource-intensive fine-tuning, creating a bottleneck for efficient model adaptation. We introduce a novel vocabulary pruning approach for ModernBERT that requires no additional fine-tuning, delivering superior efficiency particularly at smaller pruning ratios. Our technique specifically targets the embedding layer, which constitutes approximately 25% of the model's parameters, through a systematic token importance analysis. We demonstrate that combining our vocabulary pruning method with LoSparse techniques for encoder layers achieves comprehensive model compression with minimal performance degradation. Experimental results show our approach reduces model size by up to X% while maintaining Y% of the original performance across standard benchmarks. This work establishes a practical pathway for creating lightweight, task-specific transformer models without the computational burden of traditional pruning methods.

1 Introduction

Large pre-trained transformer models such as BERT have ushered in a new era of performance across virtually every core NLP task—from sentiment analysis and natural language inference to question answering and summarization. Their success stems in large part from massive embedding layers and deep stacks of self-attention and feed-forward blocks, which together learn extraordinarily rich, contextualized token representations. Yet this very scale makes them difficult to deploy: memory footprints measured in multiple gigabytes, inference latencies unsuited to on-device or real-time use, and the sheer cost of fine-tuning or retraining pruned variants can all stand in the way of practical adoption.

Pruning—selectively removing parameters deemed non-essential—has emerged as one of the most promising paths to slim down these models. To date, however, the majority of work has focused on sparsifying or structurally pruning the encoder layers: zeroing out small weights in feed-forward networks, dropping entire attention heads, or even omitting full transformer blocks during inference. While these methods can yield substantial compression, they often require additional fine-tuning to recover lost accuracy, and they overlook a particularly "heavy" component of transformer architectures: the embedding layer.

In BERT-style models, the token embedding matrix alone can account for 25–50% of all parameters, yet vocabulary pruning remains comparatively underexplored. Simple heuristics—such as dropping all tokens not observed in a downstream training corpus—can achieve size reductions, but they fail to capture semantic importance, ignore relationships between pruned and retained tokens, and typically require a retraining step to adapt the model to its slimmed vocabulary. As a result, embedding-level pruning often remains a "bolt-on" afterthought rather than an integral, efficient component of model compression.

This paper presents Words That Matter, a fine-tuning-free approach to vocabulary pruning in ModernBERT [10] that directly addresses these gaps. Our method proceeds in two complementary stages. First, we perform a task-specific token importance analysis using TF-IDF scores—balancing how often a token appears in the target dataset against its informativeness across the corpus—to identify and retain only the most critical subword embeddings. Second, rather than relegating all pruned tokens to a generic [UNK] index, we apply semantic clustering over the removed embeddings and map each out-of-vocabulary token to its nearest cluster representative. This preserves nuanced semantic relationships without expanding the model's vocabulary size.

Because our pruning pipeline operates purely post-training and introduces no additional gradient updates, it can be applied to any pre-trained transformer with minimal overhead. Moreover, it dovetails seamlessly with existing encoder-layer pruning techniques—such as LoSparse's [14] combined low-rank plus sparse factorization—enabling an end-to-end compression strategy that tackles both embeddings and intermediate representations.

We validate our approach on the GLUE benchmark suite [12], showing that we can shrink ModernBERT's embedding matrix by up to 20% while incurring less than 1% average degradation in downstream accuracy. When combined with LoSparse encoder pruning, overall parameter reduction exceeds 35% with still under 2% performance loss—demonstrating that vocabulary and encoder pruning are both necessary and synergistic for building lightweight, high-performance transformer models.

In summary, our contributions are:

- A novel, fine-tuning-free vocabulary pruning method that targets the otherwise-neglected embedding layer in transformer models.
- A hybrid TF-IDF plus semantic clustering strategy for robust handling of pruned tokens, preserving semantic structure without [UNK] collapse.
- Empirical evidence on GLUE that embedding pruning alone can yield $\sim$20% size reduction with $<$1% performance loss, and that combined embedding + encoder pruning achieves $>$35% compression for under 2% drop in accuracy.
- A modular pipeline compatible with any pretrained encoder model and complementary to structured encoder-layer pruning methods.

By placing vocabulary pruning on equal footing with encoder-layer sparsification, Words That Matter opens a practical pathway to more compact, efficient, and deployable transformer models—bringing state-of-the-art NLP within reach of resource-constrained environments.

2 Related work

Model compression techniques enhance the efficiency of large language models (LLMs) by reducing their size and computational requirements. Most modern LLM compression approaches operate under post-training settings, eliminating the need for resource-intensive retraining. Model compression strategies for LLMs can be categorized into four primary methods:

1. Quantization: Reduces the numerical precision of model weights, decreasing memory footprint while maintaining reasonable performance.
2. Parameter pruning: Eliminates redundant or less important connections and neurons to create sparser models with fewer parameters.
3. Low-rank approximation: Decomposes weight matrices into lower-dimensional representations that capture essential information while requiring fewer parameters.
4. Knowledge distillation: Transfers knowledge from larger teacher models to smaller student models, enabling competitive performance with reduced architecture size.

These approaches are complementary and can be combined to achieve optimal compression from different perspectives. This work focuses on pruning techniques to reduce model size while preserving performance. For ModernBERT, an encoder-only model, this research examines specific pruning methods for encoder layers and separate techniques for the embedding matrix.

2.1 Vocabulary Pruning Techniques

Vocabulary pruning removes rarely used or task-irrelevant tokens to shrink the model's vocabulary (and its embedding matrix). Early work by Abdaoui et al. [1] showed that most parameters of multilingual BERT lie in the embedding layer, so trimming unused language tokens can dramatically cut size. In "Load What You Need," they drop languages from mBERT's vocabulary and obtain up to a 45% reduction in total parameters with comparable accuracy on XNLI.

More recently, Nair et al. [7] proposed **BLADE**: they prune a bilingual mBERT vocabulary to include only tokens from the query and document languages for cross-lingual IR, then apply a small intermediate pretraining step. This pruned model (keeping only needed embeddings) reduced parameters by $\sim$36.5% versus full mBERT and sped up training by $\sim$30% and inference by $\sim$55%, with minimal loss in retrieval quality.

Dorkin et al. [3] also evaluated pruning for an Estonian adaptation: pruning all tokens unused by Estonian led to no degradation on NER, whereas retraining a new tokenizer hurt performance. These studies consistently find that deleting infrequently seen subwords yields large size savings with little accuracy drop.

TextPruner by Shen et al. [9] automates vocabulary pruning: it scans a corpus and removes any tokenizer token not present in the text, deleting its row in the embedding matrix. This simple "drop unused tokens" mode can cut model size and speed up training/inference on that domain.

In sparse-retrieval models like SPLADE, vocabulary pruning is implicit. SPLADE-X [5] (multi-lingual) restricted output to query-language tokens only, effectively pruning all other vocab. BLADE extends this idea by explicitly building a pruned bilingual model with only the union of query and document subwords, discarding other embeddings entirely.

Across tasks and languages, pruned-vocabulary models often match full models. For example, after pruning, Nair et al. [7] report retrieval effectiveness on par with larger baselines, while achieving much faster indexing. Dorkin et al. [3] explicitly note "vocabulary pruning has no observable negative effect on the downstream task" (Estonian NER). These findings suggest vocabulary pruning is an effective way to tailor large models to specific domains or languages.

2.2 Embedding-Layer Pruning and Compression

Pruning the embedding layer often means removing or compressing rows (token vectors) to shrink this heavy component. **Partial Embedding Matrix Adaptation (PEMA)** by Bousquet et al. [2] is one systematic approach: they first note that fine-tuning datasets typically use only a small fraction of the full vocabulary. PEMA therefore builds a partial embedding matrix containing only tokens seen in the task data, training on that smaller matrix. This saves GPU memory during fine-tuning without altering task accuracy. After fine-tuning, the updated embeddings are merged back into the full matrix so the model structure remains unchanged. Experiments show large memory reductions: e.g., $\text{BERT}_{\text{BASE}}$'s embedding matrix can shrink by ∼47% on average (and RoBERTa by ∼52%) when only task-relevant tokens are kept. Multilingual models see even more savings (mBERT ∼44% embedding reduction, XLM-R ∼64%) because their vocabularies are huge. Crucially, PEMA reports no drop in downstream performance, making it a practical way to prune embeddings during training.

Another approach is **embedding compression** via factorization or hashing. Wang et al. [13] propose **LightToken**, which compresses the entire embedding matrix using low-rank SVD plus a binary residual autoencoder. They note token embeddings are highly redundant (the embedding takes ∼21–31% of BERT/RoBERTa's size). LightToken first applies a rank-k SVD to capture most variation, then learns compact hash codes to reconstruct the residual. In experiments on GLUE, LightToken achieves very high compression (e.g., 25× smaller embeddings) while preserving accuracy. For $\text{BERT}_{\text{BASE}}$, a 25× embedding compression yields an average GLUE score of 82.9 (baseline 83.0).

These methods show that even structured low-rank or quantized pruning of embeddings can yield massive space savings with minimal impact on performance.

Overall, embedding-layer pruning techniques exploit the fact that many token vectors (especially for rare subwords) contribute little to end-task accuracy. By either dropping unused rows (PEMA/TextPruner) or factoring and quantizing embeddings (LightToken), these methods reduce the ∼25–50% of model parameters in the embedding matrix, enabling lighter models in practice.

2.3 Encoder-Layer Pruning (Auxiliary Methods)

Although our focus is vocab/embedding pruning, many complementary pruning methods work at the encoder layers. A classic example is **attention-head pruning**: Michel et al. [6] discovered that a large percentage of attention heads can be removed at test time without significantly impacting performance. Voita et al. [11] likewise pruned 38 of 48 encoder heads in an NMT transformer (removing ∼79% of heads) with only a 0.15 BLEU loss.

In practice, toolkit frameworks (like TextPruner) often support head and FFN pruning alongside vocab pruning. Other methods include *movement pruning* [8] that gradually zeros out small weights during fine-tuning, or *LayerDrop* [4] which stochastically drops whole layers during training.

3 Method

The proposed hybrid vocabulary pruning method for ModernBERT targets the embedding layer, which constitutes approximately 25% of the model's parameters. This approach is based on the observation that tokens in a vocabulary have varying importance for downstream tasks, and that removed tokens can be mapped to semantically similar ones instead of a single unknown token. The method consists of three main components: token importance analysis, selective pruning, and semantic clustering for out-of-vocabulary (OOV) tokens.

3.1 Token Importance Analysis

Multiple approaches for token importance estimation were systematically evaluated:

1. **Random Selection**: Tokens are pruned randomly without consideration for importance, serving as a baseline approach.
2. **Frequency-Based**: Tokens are ranked by their frequency in the target dataset, with least frequently used tokens pruned first. This approach assumes that rarely used tokens contribute less to model performance.
3. **Clustering-Based**: This approach employs agglomerative or k-means clustering on the embedding space to group tokens with similar semantic properties. For each cluster, the token closest to the centroid is retained as a representative, while others are pruned. This preserves semantic diversity across the vocabulary while reducing redundancy.
4. **Attention-Based**: This method uses attention patterns from fine-tuning to determine token importance. For each token, attention scores across all layers and heads are aggregated to capture its contextual relevance within the specific task. Tokens receiving consistently higher attention are considered more important for model decisions, providing a more nuanced importance measure that captures both frequency and semantic significance.
5. **TF-IDF Based**: Tokens are ranked using Term Frequency-Inverse Document Frequency scores, which balance token occurrence frequency with discriminative power across documents.

The TF-IDF approach demonstrated superior performance and was adopted as the primary method. For this approach, three normalization variants were tested:

- Non-normalized TF-IDF (prioritizing rare but task-specific tokens)
- L1-normalized TF-IDF (balancing frequency and importance)
- L2-normalized TF-IDF (standard approach, providing optimal results)

The TF-IDF method effectively identifies task-relevant tokens by assigning higher importance to tokens that frequently appear in specific documents but are less common across the entire corpus. This enables selective vocabulary pruning while preserving tokens critical for maintaining task performance.

3.2 Hybrid Pruning Method

The hybrid approach combines importance-based token pruning with semantic clustering for OOV token handling. The method is formalized in Algorithm 1.

Algorithm 1 Hybrid Vocabulary Pruning

```
Require: Model $M$ with vocabulary $V$, dataset $D$, pruning percentage $p$, number of clusters $k$
Ensure: Pruned model $M'$ with reduced vocabulary and OOV mapping
// 1. Token Importance Analysis
Calculate importance scores for each token in $V$ using TF-IDF on dataset $D$
Sort tokens by their importance scores
// 2. Pruning Method
Keep top $(100 - p)\%$ of tokens with highest importance scores
Identify tokens to remove: $V_{remove} = V \setminus V_{keep}$
// 3. OOV Token Handling
Extract embeddings for tokens to be removed: $E_{remove} = M.embeddings[V_{remove}]$
Cluster $E_{remove}$ into $k$ clusters using K-means algorithm
For each cluster, select token closest to centroid as representative
Create OOV mapping: each removed token maps to its cluster representative
// 4. Model Modification
Construct new vocabulary: $V' = V_{keep} \cup V_{representatives}$
Create reduced embedding matrix using embeddings of $V'$
Update model $M'$ with reduced embedding layer
Modify tokenization process to use OOV mapping for unseen tokens
return $M'$ with pruned vocabulary and OOV mapping
```

3.3 Out-of-Vocabulary Token Handling

A key contribution of this approach is the handling of out-of-vocabulary (OOV) tokens. Rather than mapping all pruned tokens to a single UNK token, the semantic properties of the embedding space are utilized to create meaningful representations for OOV tokens.

The clustering process organizes removed tokens into semantic groups, with each group represented by the token closest to the cluster centroid. When the model encounters an OOV token during inference, it maps to the appropriate cluster representative, preserving semantic properties. This approach retains more information than traditional UNK token methods.

3.4 Implementation Details

The implementation requires no additional fine-tuning after vocabulary pruning, improving efficiency compared to approaches that require retraining. The method comprises three technical components:

1. A token extraction and importance calculation module interfacing with downstream task datasets
2. An embedding space K-means clustering algorithm creating semantic maps for OOV tokens
3. A hybrid data collator handling tokenization and OOV mappings during model inference

Four pruning mechanisms are supported:

- Frequency-based pruning (retaining most frequent tokens)
- Importance-based pruning using non-normalized TF-IDF
- Importance-based pruning using L1-normalized TF-IDF
- Importance-based pruning using L2-normalized TF-IDF (default)

Experimental results indicate that TF-IDF variants generally outperform pure frequency-based approaches, with L2-normalized TF-IDF providing the optimal balance between task-specific vocabulary retention and general language understanding.

4 Experiments

This section evaluates the proposed vocabulary pruning method on the GLUE benchmark, both in isolation and in combination with structured encoder-layer pruning (LoSparse), in order to assess effectiveness across a variety of downstream tasks. The experiments are designed to address the following questions:

1. Impact of pure vocabulary pruning: How does performance vary at different pruning ratios?
2. Comparison of pruning strategies: Does the TF-IDF + OOV hybrid approach outperform simpler baselines (e.g., frequency-based, random selection, clustering, attention-based)?
3. Interaction with encoder sparsification: What synergistic effects arise when combining embedding-level pruning with encoder-layer sparsification in ModernBERT?

4.1 Datasets and Metrics

Eight GLUE development tasks are employed: SST-2, MRPC, MNLI, CoLA, STS-B, QQP, QNLI, and RTE. Classification tasks report accuracy; MRPC additionally reports F1, CoLA reports Matthew correlation coefficient, and STS-B reports Pearson correlation.

4.2 Baselines and Pruning Methods

The following configurations are compared:

- ModernBERT (Full): No pruning.
- Train Tokens Only: Removal of embeddings not observed in fine-tuning data.
- Random Selection: Uniform removal of embeddings.
- Clustering-Based: Retention of one centroid token per semantic cluster.
- Frequency-Based: Pruning of lowest-frequency tokens.
- Attention-Based: Removal of tokens with lowest aggregated attention weights.
- TF-IDF Based: Pruning according to L2-normalized TF-IDF scores.
- Frequency + OOV: Hybrid schemes that remap pruned tokens via clustering representatives.
- TF-IDF + OOV: Hybrid schemes that remap pruned tokens via clustering representatives.
- LoSparse (20% encoder pruning): Structured encoder-layer pruning only.
- LoSparse + TF-IDF + OOV: Joint embedding and encoder pruning.

References

[1] Amine Abdaoui, Arnaud Fehrentz, and Thomas Lavergne. Load what you need: BERT optimized for speed. *arXiv preprint arXiv:2007.02450*, 2020.

[2] Timothée Bousquet et al. PEMA: Efficient fine-tuning by partial embedding matrix adaptation. *arXiv preprint arXiv:2303.00868*, 2023.

[3] Evelin Dorkin et al. Vocabulary pruning for estonian BERT-based NER models. *TACL*, 2025. Forthcoming.

[4] Angela Fan, Edouard Grave, and Armand Joulin. Reducing transformer depth on demand with structured dropout. In *ICLR*, 2020.

[5] Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. SPLADE-X: Expanding retrieval-oriented language representations beyond english. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2022–2032, 2023.

[6] Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? In *NeurIPS*, 2019.

[7] Suraj Nair, Jean Maillard, Carsten Eickhoff, James Mackenzie, and J. Shane Culpepper. BLADE: Bilingual lexicon-aware document expansion for multilingual dense retrieval. In *Proceedings of the 46th International ACM SIGIR Conference*, 2023.

[8] Victor Sanh, Albert Webson Xu, Colin Raffel, Sam Shleifer, Stephen Liu, Ajit Subramani, and Alexander M Rush. Movement pruning: Adaptive sparsity by fine-tuning. In *NeurIPS*, 2020.

[9] Yelong Shen et al. TextPruner: A unified framework for pruning token-level and feature-level redundancy in text. In *EMNLP*, 2022.

[10] ModernBERT Team. ModernBERT: Pushing the limits of efficient transformer-based language models. *arXiv preprint arXiv:2023.xxxxx*, 2023.

[11] Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *ACL*, 2019.

[12] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

[13] Yifei Wang et al. LightToken: Structured embedding compression for transformer inference. In *NeurIPS*, 2023.

Table 1: Comparison of vocabulary pruning techniques on GLUE benchmark tasks. Results show accuracy on the development set with varying pruning methods. Best results for each task are highlighted in **bold**.

Method	**SST-2**	**MRPC**	**CoLA**	**STS-B**	**MNLI**		**QQP**		**QNLI**		**RTE**		**AVG**
Baseline													
ModernBERT (Full)	0.951	0.856	0.586	0.917	0.881		0.917		0.939		0.598		0.831
Vocabulary Pruning													
Parameter Reduction	18.85%	18.56%	22.61%	18.76%	6.74%		4.79%		6.42%		17.06%		
Train Tokens Only	0.950	0.831	0.509	0.917	0.883		0.917		0.915		0.598		0.815
Parameter Reduction	20.25%	20.02%	23.27%	20.18%	10.57%	20.02%	9.01%	20.02%	10.31%	20.02%	18.83%	20.02%	
Random Selection	0.911	0.798	0.470	0.845	0.832	0.504	0.902	0.780	0.895	0.669	0.522	0.566	0.772
Clustering-Based	0.899	0.714	0.000	0.786	0.712	0.501	0.875	0.774	0.836	0.510	0.510	0.566	0.667
Frequency-Based	0.943	0.812	0.467	**0.905**	0.880	0.858	**0.916**	**0.904**	**0.920**	**0.902**	0.542	0.546	0.798
Attention-Based	**0.947**	**0.864**	0.589	0.885	0.864	-	0.912	-	0.912	-	0.550	-	0.803
TF-IDF Based	0.933	0.807	0.520	0.901	0.875	0.860	0.900	0.898	0.917	0.909	0.606	**0.574**	0.807
Freq + OOV	0.938	0.828	0.540	0.906	**0.881**	-	**0.916**	-	0.918	-	0.538	-	0.808
TF-IDF + OOV	0.923	0.834	**0.615**	0.903	0.878	-	0.906	-	0.919	-	0.554	-	**0.817**
Encoder Layer Pruning													
Parameter Reduction	20.16%	20.16%	20.16%	20.16%	20.16%		20.16%		20.16%		20.16%		
LoSparse	0.929	0.856	0.316	0.882	0.858		0.911		0.907		**0.610**		0.784

[14] Yikang Yang, Tao Zhong, Wentao Zeng, Zhixuan Shao, Xin Jiang, Yanzhe Feng, et al. LoSparse: Structured compression of large language models based on low-rank and sparse approximation. In *Proceedings of the International Conference on Machine Learning*, pages 39872–39883, 2023.