

A.V. KONSTANTINOV, L.V. UTKIN, S.P. KHOMETS
**A GUMBEL-SOFTMAX-BASED APPROACH FOR SELECTING
IMPORTANT TOKENS IN SELF-ATTENTION MECHANISM**

Konstantinov A.V., Utkin L.V., Khomets S.P. A Gumbel-Softmax-Based Approach for Selecting Important Tokens in Self-Attention Mechanism.

Abstract. The quadratic computational complexity of self-attention in transformers remains a significant bottleneck for efficiently processing long sequences. This paper introduces a plug-in method for reducing the computational cost of attention in an existing transformer backbone via differentiable mask learning, enabling dynamic, input-dependent token selection while preserving the model's ability to capture global contextual information. The proposed approach is designed for integration into an already trained transformer model with limited modification of the attention block. Rather than introducing another standalone efficient-attention architecture, the proposed approach is formulated as a plug-in token-selection module for an existing transformer backbone. It learns task-specific hard token-selection masks in a differentiable manner and is intended to reduce the effective attention cost of an already trained model at inference time. Our approach employs a Gumbel-Softmax-based mechanism to learn hard token-selection masks in a differentiable manner, offering a practical way to reduce inference time while preserving most of the baseline model quality in the setting considered. Experiments on four sentiment-analysis benchmarks with a DistilBERT backbone show that the proposed method achieves an 8–12% reduction in inference time while preserving most of the baseline quality in the setting considered. The learned masks show qualitatively interpretable selection patterns in the presented examples, often prioritizing sentiment-bearing tokens and supporting more efficient inference. The code implementing the model for experiments is publicly available at <https://github.com/NTAILab/GSBA-SIT-SAM>.

Keywords: transformer, self-attention, Gumbel-Softmax, token selection, differentiable mask.

1. Introduction. The transformer architecture [1, 2] has revolutionized natural language processing through its powerful self-attention mechanism, enabling models to capture long-range dependencies and contextual relationships in text. However, the quadratic computational complexity of the self-attention with respect to sequence length remains a fundamental limitation, particularly when processing long documents or deploying models in resource-constrained environments. While several approaches have attempted to address this challenge through sparse attention patterns or approximation methods, these solutions often compromise the model's ability to focus dynamically on the most relevant tokens for a given task. Existing approaches for reducing the quadratic complexity of self-attention can be broadly categorized into sparse attention methods, approximation-based approaches, and token pruning techniques. Comprehensive surveys of these directions can be found in [3–9]. These reviews present comprehensive studies of various algorithms, recent techniques and methods devised to extend the sequence length in transformers and LLMs.

In this work, we focus on reducing the effective computational cost of self-attention by dynamically selecting a subset of informative tokens on a per-input basis, while preserving the original pretrained backbone. The goal is not to propose another standalone efficient-transformer architecture, but to study a plug-in token-selection module that can be attached to an already trained model. Our method introduces selective token inclusion in the attention mechanism through differentiable mask learning, enabling the model to dynamically identify and prioritize the most informative tokens during both training and inference. Unlike static sparsity patterns or fixed window approaches, our technique preserves the transformer's ability to capture global dependencies while reducing the effective sequence length processed by each attention head.

It is important to distinguish the present setting from standalone efficient-transformer architectures such as Longformer, Linformer, Performer, or BigBird, which redesign the attention mechanism itself. In contrast, the present work studies a post-hoc token-selection module for an already trained backbone. For this reason, the main experimental comparison in this paper is performed against the same pretrained backbone without the proposed module, which isolates the effect of the token-selection mechanism itself.

The main contributions of our work are threefold:

1. **Differentiable Mask Learning:** we propose a Gumbel-Softmax-based mechanism for learning token-selection masks that enables hard selection of the most relevant tokens during forward passes while retaining differentiable training of the scoring module.

2. **Plug-in Token Selection for Existing Backbones:** our method is designed as a lightweight token-selection module that can be integrated into an already trained transformer backbone with limited modification of the attention block, offering a tunable trade-off between computational savings and predictive performance.

3. **Empirical Validation:** through experiments on several benchmark sentiment-analysis datasets (IMDb, SST-2, Rotten Tomatoes, and Yelp Polarity), we demonstrate that the proposed approach achieves an 8–12% reduction in inference time while preserving most of the baseline quality in the considered setting, outperforms random masking approaches, and exhibits stable training dynamics across different mask sizes and layer configurations.

Our results show particular advantages in processing long documents, where the method demonstrates strong performance on the Yelp Polarity dataset (average length 382 tokens) with only 50% of the tokens participating in attention computations. The learned masks show qualitatively interpretable

selection patterns in the presented examples, often prioritizing sentiment-bearing tokens while preserving important structural elements of the input.

The code implementing the model for experiments is publicly available at <https://github.com/NTAILab/GSBA-SIT-SAM>.

2. Related Work. Numerous approaches and methods have been developed to reduce the computational complexity of the attention mechanisms in transformers. In particular, Longformer [10] scales linearly with sequence length, making it easy to process documents of thousands of tokens or longer. Another model [11] uses sparse factorization of the attention matrix to simplify the attention. Linformer [12] applies the self-attention approximated by a low-rank matrix. Performer [13] is based on approximation of the softmax attention-kernels by using the Fast Attention Via positive Orthogonal Random features approach. Transformer-XL [14] consists of a segment-level recurrence mechanism. The Compressive Transformer [15] is an attentive sequence model which compresses past memories for long-range sequence learning. BigBird [16] applies a sparse attention mechanism that reduces this quadratic dependency to linear. Reformer [17] is based on using two techniques to improve the efficiency of transformers. The first replaces the dot-product attention with the locality-sensitive hashing. The second uses reversible residual layers instead of the standard residuals. The Token Pruning model is presented in [18], which introduces a novel token reduction method dubbed Learned Token Pruning which adaptively removes unimportant tokens. Two key challenges of standard transformer architectures, including the scaling input length and encoding structured inputs are considered in the ETC model [19]. Interesting solutions for reducing the transformer complexity are provided in [20–23]. In [24], a transformer architecture is proposed that combines the efficient parallelizable training of transformers with the efficient inference of RNNs. Popular approaches to make transformers faster and lighter are reviewed in [3]. Macformer is proposed in [25]. It employs random Maclaurin features to approximate various dot-product kernels. Linear transformers under conditions of corrupted training data and noise are studied in [26]. The self-attention expressed as a linear dot-product of kernel feature maps by using the associativity property of matrix products is proposed in [27]. The linear transformer called cosFormer is presented in [28]. A binarization paradigm customized to high-dimensional softmax attention via kernelized hashing and called EcoFormer is introduced in [29].

The above represents only a very small fraction of all currently proposed or existing approaches, algorithms, and models for reducing the computational complexity of transformers. Existing efficient-transformer methods reduce attention cost at different levels: some redesign the attention

pattern or kernel approximation, while others progressively prune tokens as representations pass through the layers. The present work is closer to learned token-selection methods, but differs in its focus on a fixed-budget key/value token-selection module that can be attached to an already trained backbone and then executed at inference time through actual gather-based pruning. In contrast to sparse and low-rank attention approximations, our method does not modify the attention formulation itself and can be applied as a post-hoc optimization to pretrained models.

3. Background: Randomized Baselines and Key/Value Selection.

Rather than replacing self-attention with an alternative mechanism of lower asymptotic complexity, we investigate complementary strategies that reduce the *effective* computational cost of attention by restricting the set of key/value tokens involved in the computation. This perspective offers two practical advantages. First, it enables converting existing pretrained transformer models into more efficient ones by replacing only the attention layers while reusing the original parameters; the resulting adaptation typically requires lightweight fine-tuning and is substantially cheaper than training a new architecture from scratch. Second, token-selection methods can be combined with orthogonal acceleration techniques (e.g., Flash Attention), yielding additive speedups.

Our analysis adopts the standard premise that, for many inputs, the contextual representation of a token can be recovered accurately by interacting with only a limited subset of informative “neighbors.”

3.1. Randomized baselines. We first consider a fully randomized approach that constructs subsets of keys (and the corresponding values) uniformly at random. This baseline is conceptually simple and exposes a fundamental limitation: under realistic compute budgets, random sampling is unlikely to include all task-critical tokens when their number is non-negligible.

3.2. From random sampling to learned key/value selection. The limitations of the randomized baseline motivate a second family of approaches, where the subset of key/value tokens is selected in a data-dependent manner based on the layer-wise token embeddings. The remainder of this section formalizes the randomized baseline and quantifies when it is expected to fail.

The first direction reduces complexity by approximating full attention using only a subset of keys and values and, if necessary, improving accuracy by repeating the procedure multiple times. Assume self-attention is applied to a sequence of length n . The main hyperparameter is the subset size k (the number of selected key/value tokens). We sample t subsets

of size k , compute attention for each subset, and then aggregate the results (e.g., by averaging). As t increases, the approximation quality improves but eventually saturates.

If the base attention has quadratic complexity $O(n^2)$, then computing attention with n queries and only k keys/values has complexity $O(nk)$ per repetition; after t repetitions, the total cost becomes $O(nkt)$. Thus, to obtain a computational advantage, one must satisfy:

$$kt < n. \quad (1)$$

If the sequence contains a absolutely relevant tokens that need to be considered collectively for each query token, then the probability of selecting all such tokens when taking a subset of size k is given by the hypergeometric distribution and equal to $\binom{n-a}{k-a} / \binom{n}{k}$. Then the probability of obtaining at least one subset where the result of applying the attention mechanism will be correct (using all a absolutely relevant tokens) becomes:

$$p = 1 - \left(1 - \frac{\binom{n-a}{k-a}}{\binom{n}{k}} \right)^t. \quad (2)$$

We consider a commonly used training sequence length of $n = 1024$, and compare probabilities of obtaining at least one correct subset (which will significantly affect the upper bound of the model's accuracy) for various parameter values: $k \in \{64, 128, 256\}$, $a \in \{1, 2\}$. These dependencies are shown in Fig. 1, where the red vertical line corresponds to the number of repetitions t beyond which computational efficiency superiority compared to the classical mechanism will be lost.

The results indicate that, under the condition of improved efficiency relative to the base attention mechanism, the probability of randomly selecting the necessary tokens (even for the minimal number of them) turns out to be substantially less than one. Fig. 2 shows the dependency of the probability on the number of absolutely relevant tokens a with the parameter $t = n/k$, i.e., in the case where parity in terms of efficiency arises between the considered model and the base attention mechanism.

The number of absolutely relevant tokens extremely significantly influences the probability of success, and for $a \geq 5$ the probability becomes extremely low for all considered subset sizes.

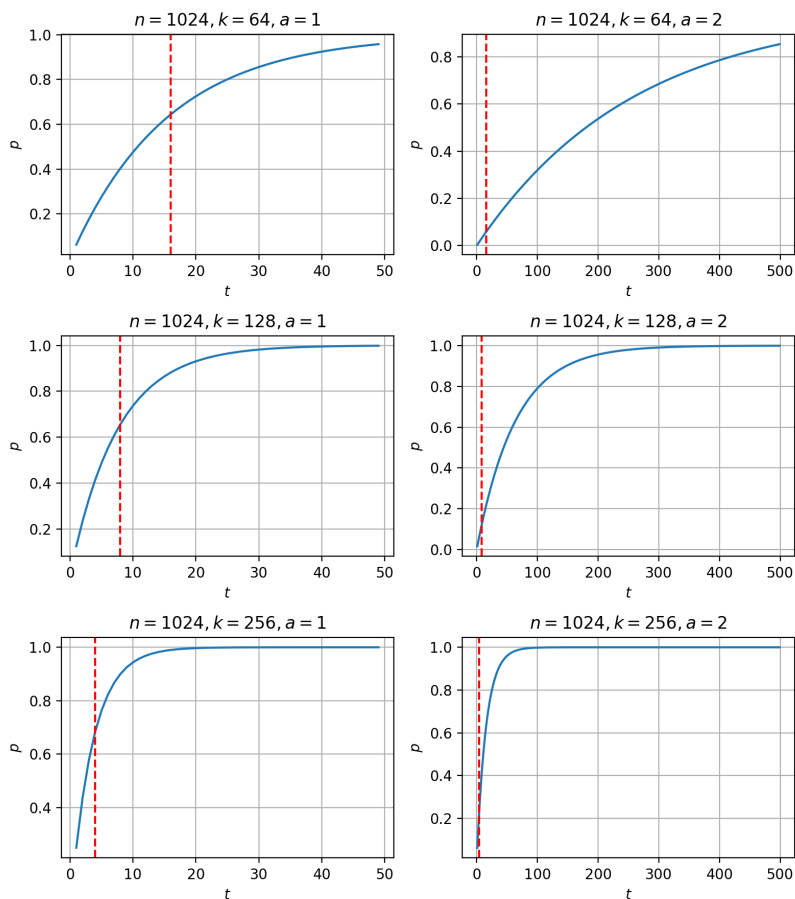


Fig. 1. Dependencies of the probability of a correct subset appearing

These results indicate that purely random subset selection becomes unreliable once the number of absolutely relevant tokens is moderate (e.g., $a \geq 5$): under realistic efficiency constraints, the probability of sampling a subset that contains all required tokens drops to a critically low level. Consequently, random sampling alone is unlikely to yield a method

that is simultaneously faster and accurate when correct attention behavior requires jointly considering many essential tokens.

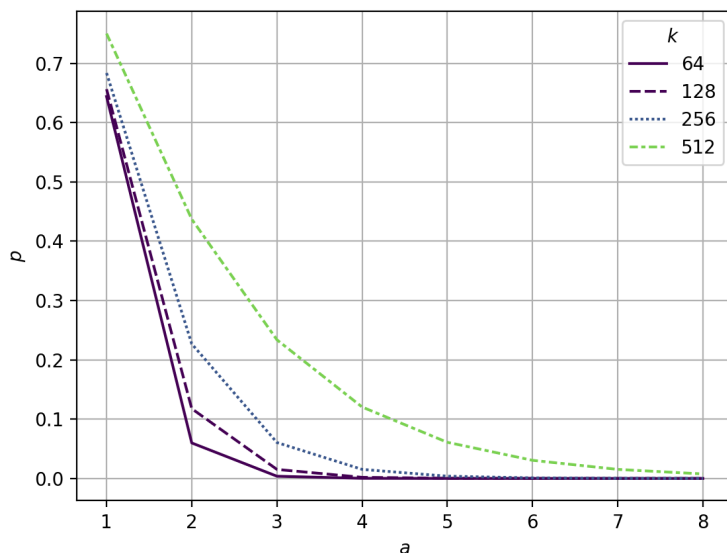


Fig. 2. Dependence of the probability of a correct subset at comparable efficiency

At the same time, the effective value of a is task- and layer-dependent. Therefore, we retain randomized sampling as a simple and informative baseline and use it for comparison throughout the experimental section.

Implication. The above analysis motivates replacing uniform random sampling with a *learned, data-dependent* key/value selection mechanism. In what follows, we introduce a differentiable token-selection method that constructs a single binary mask per layer (shared across queries), so that keys/values can be physically pruned at inference time while preserving end-to-end trainability.

4. Method: Differentiable Key/Value Token Selection.

4.1. Mask learning problem. To determine the most relevant tokens in a layer and subsequently build a model trainable end-to-end, various strategies were considered. A natural idea is to use the query vector itself to determine which keys should be considered for its processing. However, in this case, a separate mask is determined for each query, its values indicating whether a key vector is used for this query vector. This leads to the formation of a quadratic number of masks, one mask per query token, making significant

memory or computation savings impossible. Therefore, another approach was proposed.

An alternative is constructing a single mask for all query vectors. Such a mask does not depend on queries but relies on key vectors, and can be implemented, for example, as a random mask or a position-dependent mask (so-called padding mask). To avoid quadratic complexity in mask generation and simultaneously make the mask dependent on keys, an auxiliary model based on a small neural network is introduced. It works independently for each token embedding and produces a scalar importance score. Then these scores are used to construct a single binary mask, which is applied to all query vectors in the layer.

Let's consider the formulation of the subproblems of constructing and applying the mask based on key vectors. Let $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ be the set of input embeddings of dimensionality d . The standard self-attention mechanism computes the attention matrix $A \in \mathbb{R}^{n \times n}$, which has a computational complexity $O(n^2)$. The goal of the developed method is to approximate full attention with a sparse analogue with a fixed computation budget $k \ll n$. The task of selecting an ordered set of indices $J \subset \{1, \dots, n\}$ of cardinality $|J| = k$ can be formulated as an optimization problem of finding a binary mask $M \in \{0, 1\}^n$, where $\sum_{i=1}^n M_i = k$. The following theoretically possible approaches to forming such a mask were analyzed:

1. *“Soft” selection of token embeddings (so-called “Soft Gating”).* The most obvious approach is to use a sigmoid activation to weight token importance: $\tilde{\mathbf{x}}_i = \sigma(f(\mathbf{x}_i)) \cdot \mathbf{x}_i$. The token embedding can be automatically reduced, which affects its subsequent participation in the attention mechanism (small attention weights, and consequently, small contribution to the result). However, this approach preserves the computational complexity $O(n^2)$, because these tokens are still included in the sequence and participate in the computation of the attention matrix A , despite making no actual contribution to the result. Moreover, the sigmoid function does not guarantee obtaining exactly k significant tokens, and the distribution of embeddings for the next layer may differ from the distribution during training, which negatively affects model accuracy.

2. *Stochastic Bernoulli sampling and reinforcement learning.* Building a mask is possible through independent sampling of each mask component from a Bernoulli distribution $m_i \sim \text{Bernoulli}(p_i)$, where the probability p_i is determined based on an auxiliary model. This approach has two drawbacks. First, it does not guarantee a fixed subset size k , which creates technical problems during batch processing since subsets of key matrices for different

training examples must have the same size (or an upper bound, in this case less than n). However, in this case, the dimensionality is dynamic, which not only complicates computations but requires introducing penalty components into the loss function; otherwise, the auxiliary model could be optimized such that all mask components have a probability of 1, meaning computational efficiency would not be improved. Second, the gradient estimate of the loss function with respect to parameters in this case is characterized by high variance, making training unstable (e.g., when using the REINFORCE algorithm).

3. *l_1 -regularization of attention weights (LASSO method).* Adding a penalty on the l_1 -norm of attention weights to encourage sparsity can lead to zero attention weights for many keys. However, from the perspective of neural network training, this constraint (on zero weights) is “soft”, meaning it is not guaranteed that sparsity will arise at the prediction stage. Furthermore, to determine which attention weights are close to zero, one first needs to fully compute the matrix A . Additionally, this approach does not guarantee obtaining exactly k non-zero elements and requires complex hyperparameter tuning of the penalty to balance sparsity and quality. Forming a mask based on near-zero elements requires applying a threshold function at the prediction stage, which significantly differs from computing the attention mechanism without a threshold function during training. Consequently, the distribution of embeddings for the next layer may differ from the distribution during training, which also negatively impacts model accuracy. Thus, using l_1 -regularization can be considered as an auxiliary penalty but is not a suitable solution for building token masks.

4. *Defining the derivative of the top- k token-selection operation through the extended “Gumbel-Softmax” mechanism.* To overcome the described limitations, a method was chosen that combines determinism of the output set size (selecting exactly k tokens with the highest scores) with the possibility of end-to-end differentiable training. Instead of independent evaluation of each token, a categorical distribution over all tokens in the sequence is introduced. This allows the selection task to be considered as sampling k elements without replacement, where the selection probability depends on the computed token significance score.

The fourth option was chosen as the main approach, namely defining the derivative of the top- k token-selection operation through the extended “Gumbel-Softmax” mechanism. It is important to note that during training and prediction, the result of this operation does not differ, and consequently does not reduce accuracy due to distribution shift for subsequent layers at the prediction stage.

4.2. Differentiable top- k selection via Gumbel-Softmax.

An auxiliary scoring network is introduced, which for each token $\mathbf{x}_i$ predicts an unnormalized logit μ_i . However, direct use of the $\text{argtop-}k$ operation is non-differentiable and does not allow training the scoring network via gradient descent. To estimate the gradient, a continuous relaxation via the Jacobian of the softmax operation is used.

During the forward pass at the *prediction (inference) stage*, a “hard” selection of indices is applied to form the ordered set J , which allows the physical discarding of unused tokens and reducing the complexity of attention computation to $O(n \cdot k)$. For the backward pass during training, a continuous approximation is used.

Let $\mathbf{m}$ be the mask of selected tokens obtained via the $\text{argtop-}k$ operation. This mask can be approximated using a probability vector π (corresponding to a discrete probability distribution over tokens), obtained via softmax with the temperature τ :

$$\pi_i = \text{GumbelSoftmax}_i(\mu) = \frac{\exp((\mu_i + g_i)/\tau)}{\sum_{j=1}^n \exp((\mu_j + g_j)/\tau)}, \quad (3)$$

where g_i is noise from a Gumbel distribution.

As $\tau \rightarrow 0$, the distribution π tends to a degenerate (discrete one-hot) distribution, and at high values of τ , to a uniform one. This property allows during training to smoothly transition from a near-uniform distribution at high temperature to a hard distribution at low temperature, while preserving the ability to compute the gradient of the loss function with respect to the scoring network parameters $\nabla_{\theta} L$.

Note that the classical Gumbel-Softmax approach allows randomly selecting an element in accordance with a given probability distribution by determining the index of the element with the maximum probability value: $\text{argmax } \pi_i$. To form the mask $\mathbf{m}$, it is necessary to select k elements without replacement, so instead of argmax , $\text{argtop-}k$ is used. The approximation of the mask $\mathbf{m}$ by the probability vector π can be done at best up to a multiplicative constant k , since $\sum_i \pi_i = 1$, $\sum_i m_i = k$. The temperature should not be too close to zero, because otherwise only one component of the vector π will be non-zero. With a correct choice of the τ parameter for estimating partial derivatives of the mask components m_i with respect to logits μ_j , it is valid to use the derivatives of the GumbelSoftmax function.

Thus, the proposed architecture solves three key problems:

1. Ensuring constant performance (due to fixed subset size). Exactly k tokens are always selected, ensuring predictable memory consumption and computation time.

2. Training the model with gradient descent. Using Gumbel-Softmax allows training the masking parameters jointly with the main model in an end-to-end manner.

3. Linear complexity of mask generation. The importance evaluation of key tokens occurs once for the entire attention layer in $O(n)$, reducing the overall computational complexity to $O(n(k+1))$.

4.3. Token-Importance Scoring Network. A key component of the architecture is the mechanism mapping each token embedding $\mathbf{x}_i \in \mathbb{R}^d$ to a scalar score of its significance, called the scoring network ϕ_θ . For parameterizing it, the following options were considered: self-attention mechanism, convolutional neural networks, fully connected neural networks with shared parameters.

The self-attention mechanism can be used as the scoring network ϕ_θ as follows. Each token $\mathbf{x}_i$ is assigned a number characterizing the relevance of this token for processing the entire sequence. Accordingly, vectors of dimension 1 are used as value vectors, which after linear combination with attention weights (in the ϕ_θ model) form logits of relevance scores. This option, besides the current token $\mathbf{x}_i$, considers the context of the entire sequence and is the most general option, but requires computing an attention matrix of size $n \times n$, leading back to the original quadratic complexity $O(n^2)$ and negating the performance gain. It is worth noting that for evaluating the absolute relevance of tokens, a simplified attention mechanism can be used: with a lower number of parameters or a linearized attention mechanism. However, the computation time of the scoring function ϕ_θ in that case would be comparable to computing the mechanism whose efficiency we aim to improve, negatively impacting performance.

The second option, convolutional neural networks, can allow considering local context (neighboring tokens) through the application of a one-dimensional convolution operation. This option has linear complexity $O(n \cdot w)$, where w is the window width, but has two significant drawbacks. First, applying convolutions, while allowing context consideration to some degree, has a fixed window width, which can lead to significant differences in such scores with minor permutations of sequence elements (where an element moves out of the window), complicating training and increasing the risk of overfitting. Second, applying convolution significantly limits the applicability to tasks with a constant sampling step. In other words, the convolution operation

is only applicable for analyzing sequences where the position of each element is its index, whereas the attention mechanism is also applied to more general tasks, such as analyzing time series with a variable sampling step. Positional encoding can define the spatial arrangement of objects, which is not directly related to token index, making the application of convolution impossible in this case.

The third option consists of using fully connected neural networks with shared parameters to determine the relevance score logit for each token individually. Namely, it was proposed to use a fully connected network that processes each token independently:

$$\mu_i = \phi_\theta(\mathbf{x}_i), \quad (4)$$

where ϕ_θ is a classical fully connected neural network with a scalar output whose parameters do not depend on the token index.

Note that since in a transformer containing multiple attention layers, the vector $\mathbf{x}_i$ at layer l is built considering context, i.e., aggregates information from other tokens at previous layers $1, \dots, l-1$, additional context consideration when applying the scoring network is not necessary. At the same time, an element-wise applied small fully connected neural network has minimal computational complexity, allowing mask generation with insignificant overhead.

Also, the neural network does not require learning an additive constant on the output, because the obtained logits $\mu = (\mu_1, \dots, \mu_n)$ are subsequently transformed into a discrete probability distribution π via Gumbel-Softmax, which is invariant to shift.

4.4. Token-Reduced Self-Attention. After forming the ordered set of indices J (where $|J| = k$), the attention operation must be performed. The standard formula for the attention mechanism is defined as [1, 2]:

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d}} \right) V. \quad (5)$$

In a naive implementation, simply zeroing (masking) unselected elements by replacing corresponding components in the QK^T matrix with $-\infty$ does not lead to acceleration, because the matrix multiplication is still performed for matrices with dimensions $(n \times d), (d \times n)$. To achieve real acceleration, the proposed method applies an index gathering operation

(Gather). Let W_Q, W_K, W_V be the projection matrices of the attention layer (without loss of generality, single-head attention is considered). Full queries $Q = XW_Q$ are computed, but keys and values are computed only for the subset of indices J :

$$\hat{K} = \text{Gather}(X, J)W_K \in \mathbb{R}^{k \times d}, \quad \hat{V} = \text{Gather}(X, J)W_V \in \mathbb{R}^{k \times d}, \quad (6)$$

where the index gathering operation *Gather* is defined as concatenation of row vectors from the processed matrix with indices from the ordered set J :

$$\text{Gather}(X, J) = (\mathbf{x}_i)_{i \in J} = \begin{pmatrix} \mathbf{x}_{J_1} \\ \vdots \\ \mathbf{x}_{J_k} \end{pmatrix}. \quad (7)$$

The attention scores matrix is computed as:

$$\hat{S} = \frac{Q\hat{K}^T}{\sqrt{d}} \in \mathbb{R}^{n \times k}. \quad (8)$$

Note the dimensionality of the result: instead of a matrix of size $n \times n$, a matrix of size $n \times k$ is obtained. Under the condition $k \ll n$, the number of floating point operations (FLOPs) is reduced proportionally to the compression ratio $r = k/n$. Attention weights are calculated via softmax along the k dimension:

$$\hat{\alpha}_{ij} = \frac{\exp(S_{ij})}{\sum_{m=1}^k \exp(S_{im})}, \quad \text{where } j \in \{1, \dots, k\}. \quad (9)$$

The final result of the proposed attention mechanism is computed analogously to the classical mechanism:

$$\tilde{X} = \hat{A}\hat{V} \in \mathbb{R}^{n \times d}. \quad (10)$$

Thus, the computational complexity of the attention mechanism is reduced from quadratic $O(n^2d)$ to linear relative to the sequence length: $O(nkd)$.

4.5. Overall Model Architecture. The structure of the modified transformer layer *during training* is illustrated in Fig. 3. The left part of the scheme shows the modified attention block, where attention weights are sparsified by a hard mask and then renormalized; the right part shows the auxiliary block that generates this mask (scoring network followed by the Gumbel-Softmax/ $\text{argtop-}k$ procedure).

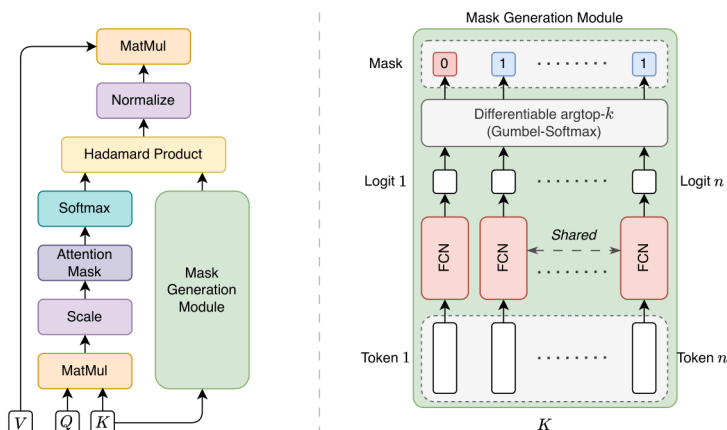


Fig. 3. Integration of the proposed model into the transformer architecture

In the mask-generation block (right part of Fig. 3), the input tokens are passed independently through a neural network, resulting in a logit for each token characterizing its significance. A softmax operation is applied to the logits to obtain a probability distribution, based on which a set of token indices (without replacement) of size k is generated.

At the *prediction stage*, the forward pass of each modified transformer layer is summarized in Algorithm 1.

Thus, the asymptotic computational complexity of the entire model at the prediction stage is reduced from $O(n^2)$ to $O(nk)$.

Although the proposed method reduces the attention computation from $O(n^2)$ to $O(nk)$ in the selected part of the computation, the observed end-to-end speedup is more moderate because attention is only one component of total inference time. The measured runtime also includes the overhead of mask generation, gather operations, and the remaining transformer computations outside the attention score matrix.

Algorithm 1 Prediction-stage forward pass of a modified transformer layer

Require: Token embeddings $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, subset size k , projections W_Q, W_K, W_V , scoring network ϕ_θ

Ensure: Output embeddings $\tilde{X} \in \mathbb{R}^{n \times d}$

- 1: $\mu \leftarrow (\phi_\theta(\mathbf{x}_i))_{i=1}^n$ ▷ Compute logits (token importance scores)
 - 2: $J \leftarrow \text{argtop}_k(\mu)$ ▷ Select keys/values and compute token-reduced attention
 - 3: $Q \leftarrow XW_Q$
 - 4: $\hat{K} \leftarrow \text{Gather}(X, J)W_K$
 - 5: $\hat{V} \leftarrow \text{Gather}(X, J)W_V$
 - 6: $\hat{S} \leftarrow \frac{Q\hat{K}^T}{\sqrt{d}}$
 - 7: $\hat{A} \leftarrow \text{softmax}(\hat{S})$ ▷ Row-wise over k keys
 - 8: **return** $\hat{A}\hat{V}$
-

To solve this problem, during training we compute the attention weights using all n key-value pairs (thus keeping the standard $O(n^2)$ computation for this step), and apply the hard mask $\mathbf{m}$ to the attention weights. Specifically, based on the discrete probability distribution π computed via Gumbel-Softmax, a “hard” mask $\mathbf{m} = (m_1, \dots, m_n)$ is constructed such that $m_i = 1 \iff i \in J$. Then the attention weights of the base mechanism are sparsified by element-wise multiplication with the mask followed by renormalization, so that non-relevant tokens receive strictly zero weights:

$$\tilde{\alpha}_{ij} = \frac{\alpha_{ij} \cdot m_j}{\sum_{g=1}^n \alpha_{ig} \cdot m_g}, \quad \tilde{A} = (\tilde{\alpha}_{ij}) \in \mathbb{R}^{n \times n}. \quad (11)$$

The result after multiplication by the value matrix will be completely equivalent to the result with the $\text{Gather}(X, J)$ operation:

$$\tilde{A}V = \hat{A}\hat{V} \in \mathbb{R}^{n \times d}, \quad (12)$$

since for each individual row it holds that:

$$\tilde{\alpha}_i V = \sum_{j=1}^n \frac{\alpha_{ij} \cdot m_j}{\sum_{g=1}^n \alpha_{ig} \cdot m_g} v_j = \sum_{j \in J} \hat{\alpha}_{ij} v_j = \hat{\alpha}_i \hat{V}. \quad (13)$$

Thus, this mechanism is an alternative implementation of the scheme used at the prediction stage, but the result functionally depends on the vector $\mathbf{m}$. Consequently, using automatic differentiation, derivatives with respect to the components of vector $\mathbf{m}$ can be calculated, which in turn depends on Gumbel-Softmax that takes the logit vector μ as input, depending on the parameters of the scoring network θ . Using the previously discussed method for defining the derivative of the operation generating the mask $\mathbf{m}$ and the automatic differentiation algorithm, partial derivatives of the loss function with respect to parameters θ are calculated.

At the *training stage*, the forward pass of each modified transformer layer is summarized in Algorithm 2.

During the backward pass at the training stage, all the same steps are used (in reverse order), except for step 2, where instead of the derivatives of the *argtop- k* operation, derivatives of the Gumbel-Softmax function with the same Gumbel noise as in the forward pass are used.

There is a complete correspondence between the presented training and prediction schemes from the perspective of computation results, due to the equivalence shown above of the attention mechanisms based on the non-differentiable operation $\text{Gather}(X, J)$ and the differentiable multiplication of attention weights by the mask (whose derivatives with respect to parameters are defined using the Gumbel-Softmax technique). Consequently, when applying the developed method to new data from the same distribution as the training data, the model's accuracy will at least not decrease due to a mismatch between training and prediction operations. At the same time, from a computational point of view, the proposed training scheme is less efficient. However, it is important to note the properties of this scheme that allow for more efficient training. First, for layers where the training of the auxiliary networks can be considered complete (no significant parameter updates occur), the forward pass procedure of the training stage can be replaced by the non-trainable but equivalent procedure for the prediction stage. Second, the proposed scheme with token relevance scoring does not explicitly take positional information into account, making it possible to train auxiliary networks only on short sequences where quadratic computational complexity is not critical, and to apply the efficient non-trainable procedure on long sequences.

Algorithm 2 Training-stage forward pass of a modified transformer layer

Require: Token embeddings $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, subset size k , temperature τ , projections W_Q, W_K, W_V , scoring network ϕ_θ

Ensure: Output embeddings $\tilde{X} \in \mathbb{R}^{n \times d}$

```

1:  $\mu \leftarrow (\phi_\theta(\mathbf{x}_i))_{i=1}^n$   $\triangleright$  Compute logits (token importance scores)
2:  $\mathbf{g} \sim \text{Gumbel}_n(0, 1)$ 
3:  $\pi \leftarrow \text{softmax}((\mu + \mathbf{g})/\tau)$ 
4:  $J \leftarrow \text{argtop}_k(\mu)$ 
5:  $\mathbf{m} \leftarrow (\mathbb{I}[j \in J])_{j=1}^n + \pi.\text{detach}() - \pi$   $\triangleright$  Backward pass uses the
   Jacobian of  $\pi$ 
6:  $Q \leftarrow XW_Q, K \leftarrow XW_K, V \leftarrow XW_V$   $\triangleright$  Any attention, e.g. SDPA
7:  $S \leftarrow \frac{QK^T}{\sqrt{d}}$ 
8:  $A \leftarrow \text{softmax}(S)$   $\triangleright$  Row-wise over  $n$  keys
9: for  $i \leftarrow 1$  to  $n$  do
10:    $\tilde{\alpha}_i \leftarrow \frac{\alpha_i \odot \mathbf{m}}{(\alpha_i \odot \mathbf{m})^T \mathbf{1}}$   $\triangleright$  Apply mask and renormalize
11: end for
12: return  $\tilde{A}V$ 

```

Thus, the developed method significantly reduces the asymptotic computational complexity of the algorithm from $O(n^2)$ to $O(nk)$, where k is a tunable hyperparameter, for an arbitrary base attention mechanism, by selecting the most relevant tokens. The selection is performed using a trainable small scoring neural network, the application of which has complexity $O(n)$ and does not require significant additional computation. This approach can be applied to improve efficiency both at the prediction stage and at the training stage, provided that suitable parameters are defined for the scoring neural network.

5. Experiments. In numerical experiments, our hypotheses are validated by considering the DistilBERT model [30] and its modified variants, including the addition of random and trainable masks to the attention mechanisms. In contrast to trainable masks derived in accordance with the proposed method, the random mask is based on the uniform random selection of k tokens from n ones, i.e., subset J is randomly formed. The random mask model does not require additional training time, since it does not compute the probability distribution π for selecting k tokens. It is used in experiments in order to illustrate a difference between models and outperformance of the proposed model.

The base model is a pre-trained DistilBERT¹, fine-tuned on a selected dataset for classification.

For the trainable mask, the overall training procedure consists of two stages. First, the base transformer is fine-tuned on the selected downstream dataset. Second, the parameters of the fine-tuned transformer are frozen, and further optimization is performed exclusively for the lightweight fully connected scoring network. For the random mask, no additional model training is conducted. Therefore, the additional training cost introduced by the proposed method is limited to the second stage rather than requiring retraining of the full backbone.

For reproducibility, we note that the experiments follow this two-stage training procedure throughout. The Adam optimizer with an initial learning rate of 10^{-5} and a linear scheduler is used for the second stage, and the Gumbel-Softmax temperature parameter is set to 10. The reported results correspond to the stated experimental setup without repeated-run variance estimates.

To compare the proposed models, four datasets from the sentiment analysis domain are selected. All datasets are obtained using the Hugging Face library.

The *IMDb* dataset [31] contains data for movie review classification (positive/negative). It includes 25,000 examples for training and 25,000 for testing. The average sentence length is 230 tokens. The dataset has an equal number of examples for each class.

The *SST-2* dataset [32] contains data for sentiment analysis of sentences (positive/negative sentiment) based on the dependency tree annotations. It includes both complete sentences and their parts. There are 67,349 examples for training and 1,821 for testing. The average sentence length is 10-20 tokens.

The *Rotten Tomatoes* dataset [33] contains data for classifying movie reviews (positive/negative). It differs from IMDb in its more “professional” evaluations, use of cinema-specific vocabulary, and complex mixed reviews. There are 8,530 examples for training and 1,066 for testing. The average sentence length is 20-30 tokens.

The *Yelp Polarity* dataset [34] includes data for classifying reviews on the Yelp platform. Reviews with 1-2 stars are considered negative, and those with 4-5 stars are positive. Examples in the dataset were filtered to increase the average sentence length. Only sentences with lengths between 300 and 512 words were retained. The filtered dataset contains 64,445 examples for training and 4,340 for testing. The average sentence length increased to 382 tokens.

¹From the Hugging Face Transformers library https://huggingface.co/docs/transformers/model_doc/distilbert

Results of numerical experiments with the above datasets are represented in Table 1, where three models are compared: the original DistilBERT, the DistilBERT with the random mask, and the DistilBERT with the trainable mask. For every model, we provide the mean time of processing the whole test set and the ROC AUC metric. The mask size is changed from 40% to 90% of the sentence length.

Table 1. Comparison results by using ROC AUC metric and runtime for models: DistilBERT and two proposed modifications with the trainable mask (TM) and the random mask (RM) of various sizes. All 6 layers are replaced

Dataset	Mask size	DistilBERT		DistilBERT+RM		DistilBERT+TM	
		Time	ROC AUC	Time	ROC AUC	Time	ROC AUC
IMDb	0.4	45.36	0.979	40.16	0.9414	40.58	0.9666
	0.5			41.24	0.9563	41.37	0.9734
	0.6			42.35	0.9658	42.63	0.9762
	0.7			43.79	0.9715	43.84	0.9772
	0.8			44.78	0.9754	45.11	0.9777
	0.9			46.05	0.9779	46.15	0.9779
SST-2	0.4	0.560	0.966	0.511	0.8454	0.533	0.8982
	0.5			0.537	0.8722	0.537	0.9139
	0.6			0.561	0.9082	0.551	0.9294
	0.7			0.566	0.9308	0.552	0.9550
	0.8			0.573	0.9493	0.556	0.9506
	0.9			0.577	0.9594	0.557	0.9638
Rotten Tomatoes	0.4	0.698	0.924	0.638	0.8011	0.622	0.8386
	0.5			0.647	0.8423	0.634	0.7927
	0.6			0.6567	0.8761	0.634	0.8814
	0.7			0.6654	0.8890	0.647	0.8961
	0.8			0.6814	0.9063	0.657	0.9239
	0.9			0.6846	0.9179	0.671	0.9241
Yelp Polarity	0.4	21.19	0.992	18.155	0.9722	18.491	0.9795
	0.5			18.692	0.9787	18.970	0.9882
	0.6			19.121	0.9839	19.449	0.9901
	0.7			19.649	0.9874	19.736	0.9912
	0.8			20.324	0.9894	20.100	0.9906
	0.9			21.012	0.9911	20.705	0.9915

Table 1 shows that as the mask size decreases, the performance gap between the trainable mask and random mask models widens significantly. For instance, on the SST-2 dataset with a mask size of 0.4, the trainable mask model achieves a ROC AUC of 0.8982, a substantial 5.3% absolute improvement over the random mask model (0.8454). This demonstrates that at high levels of sparsity, intelligent, learned selection becomes paramount. The trainable mask model effectively identifies and retains the most salient tokens for sentiment classification, while random removal severely degrades model reasoning. The fact that the trainable mask model nearly matches

the performance of the full DistilBERT model on the lengthy Yelp Polarity reviews with a mask size of just 0.5 (ROC AUC of 0.9882 vs. 0.9923) is particularly noteworthy, highlighting the method's strength in processing long-form content.

Fig. 4 illustrates the mask training process on the IMDB dataset. The graph illustrates the change in the loss function on the training and test sets, as well as the loss function value on the test set for the random mask. Hyperparameters: 6 layers are replaced, mask size is 60% of the maximum sentence length, a small neural network is used. The graph illustrates that training is performed correctly, and over time, the loss function value for the trainable mask becomes lower than the loss function for the random mask. The analysis of training dynamics also confirms the stability and effectiveness of the proposed learning procedure. The loss for the trainable mask model decreases steadily on both training and test sets, eventually falling below the constant loss value of the random mask.

Fig. 5 shows the dependence of the ROC AUC metric on the number of replaced layers for various mask sizes on the IMDB dataset for models with random and trainable masks.

Analysis of the graph illustrates that when using the trainable mask, the ROC AUC value remains almost unchanged regardless of the number of replaced layers, with fluctuations within 1%. In contrast, for the random mask, a significant decrease in ROC AUC is observed as the number of modified layers increases.

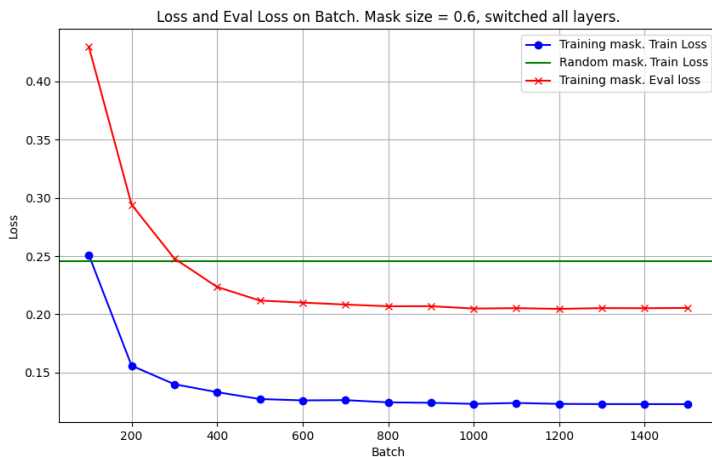


Fig. 4. Illustration of the mask training process using the IMDB dataset

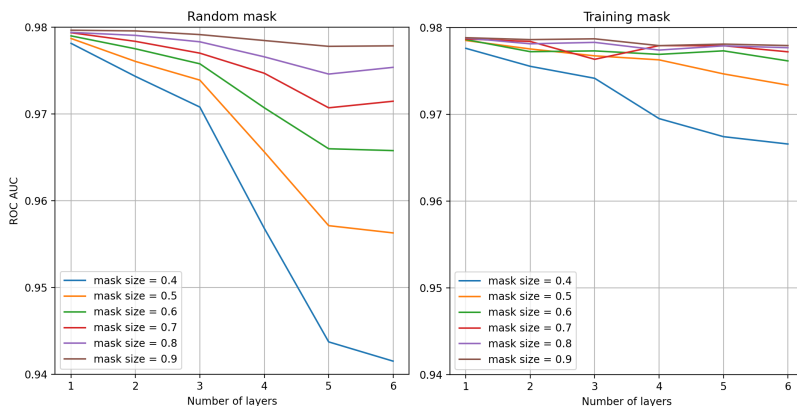


Fig. 5. Dependence of ROC-AUC metric on the number of replaced layers for random (the left graph) and trainable (the right graph) masks. The dataset IMDB is used

Figs. 6-11 display the dependence of the ROC AUC metric on the inference time compared to the unmodified original model for the IMDB and Yelp Polarity datasets. Fig. 6 shows results of experiments for the IMDB dataset conducted with a fixed mask size of 50% of the maximum sentence length. Attention layers are replaced starting from the first layer. For comparison purposes, Figs. 7-8 illustrates results of the same experiments when the fixed mask sizes are 40% (the left graph) and 60% (the right graph). The same experiments for the Yelp Polarity dataset are illustrated in Figs. 9-11.

The presented graphs demonstrate the balance between speed and prediction quality when using random and trainable masks. The results indicate that at comparable acceleration levels, the model with the trainable mask maintains significantly higher ROC AUC values compared to the random mask. Specifically, when all layers are replaced:

- The metric deviation does not exceed 1% from the original model for the trainable mask and 3% for the random mask;
- Time savings are achieved: 8% for the dataset IMDB and over 10% for the dataset Yelp Polarity.

The difference in acceleration is explained by the fact that the dataset Yelp Polarity contains longer sentences due to pre-filtering.

Additionally, the graphs illustrate increasing divergence in quality between the two mask types as the number of replaced layers grows: the ROC AUC gap progressively widens in favor of the trainable approach.

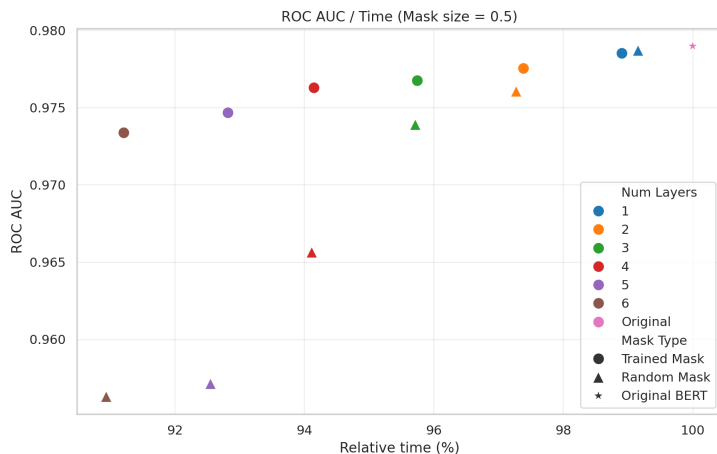


Fig. 6. Differences between the original model and its mask-modified variants with different numbers of replaced layers for the dataset IMDb by two parameters: reduction of ROC-AUC and achieved runtime reduction. The mask size is 0.5.

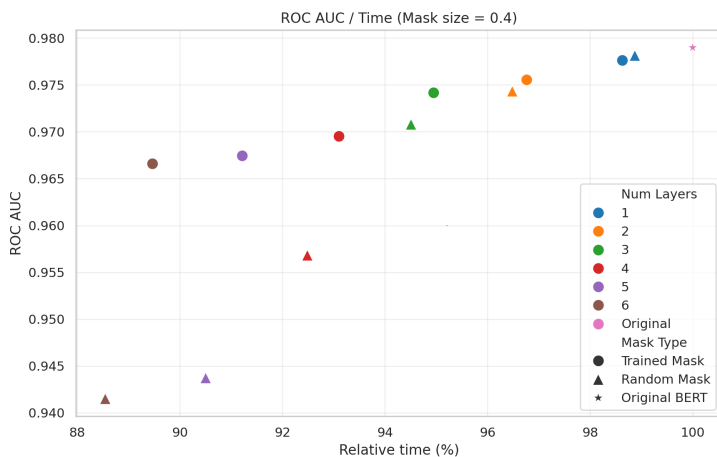


Fig. 7. Differences between the original model and its mask-modified variants with different numbers of replaced layers for the dataset IMDb by two parameters: reduction of ROC-AUC and achieved runtime reduction. The mask size is 0.4

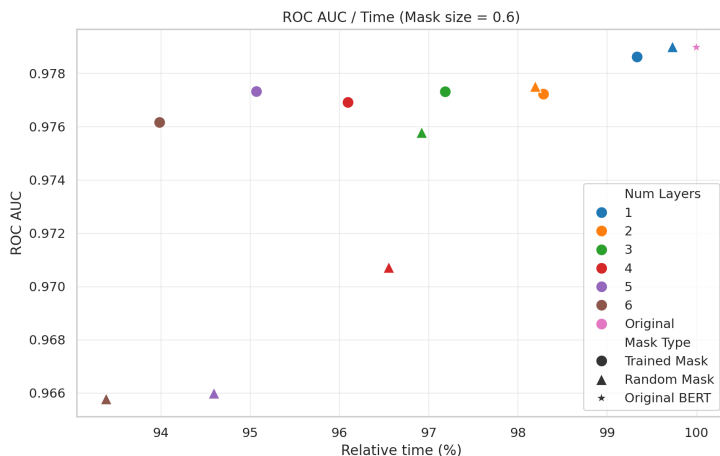


Fig. 8. Differences between the original model and its mask-modified variants with different numbers of replaced layers for the dataset IMDb by two parameters: reduction of ROC-AUC and achieved runtime reduction. The mask sizes is 0.6

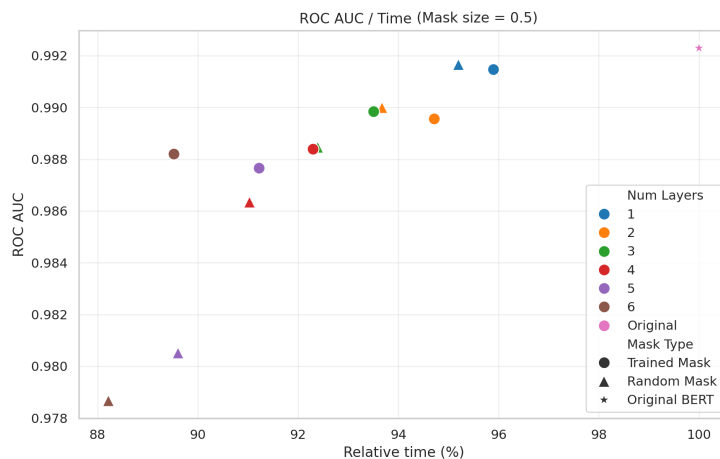


Fig. 9. Differences between the original model and its mask-modified variants with different numbers of replaced layers for the dataset Yelp Polarity by two parameters: reduction of ROC-AUC and achieved runtime reduction. The mask size is 0.5

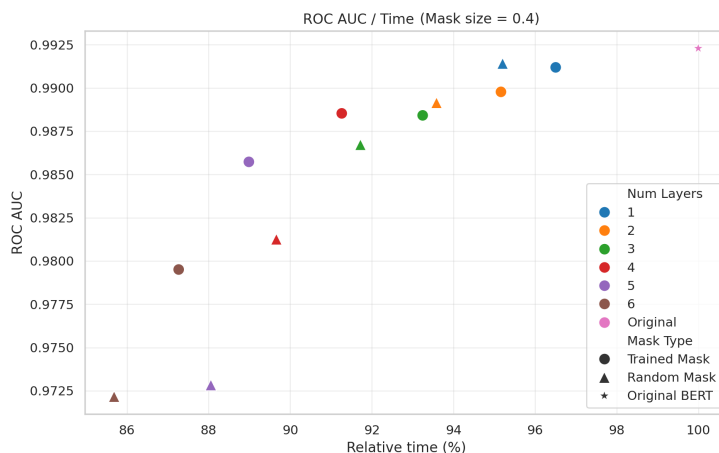


Fig. 10. Differences between the original model and its mask-modified variants with different numbers of replaced layers for the dataset Yelp Polarity by two parameters: reduction of ROC-AUC and achieved runtime reduction. The mask sizes is 0.4

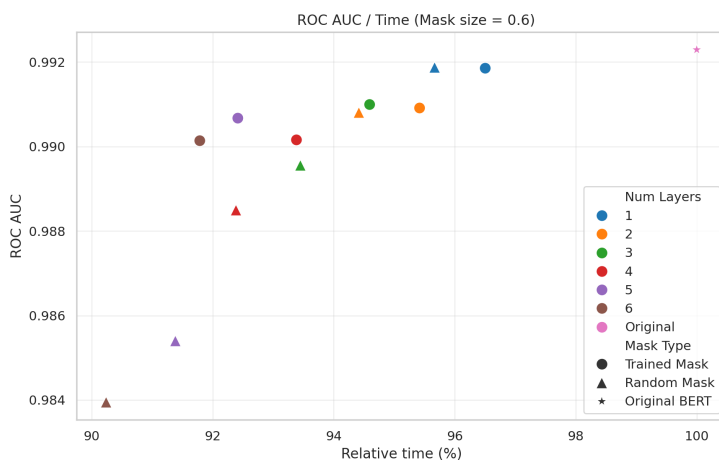


Fig. 11. Differences between the original model and its mask-modified variants with different numbers of replaced layers for the dataset Yelp Polarity by two parameters: reduction of ROC-AUC and achieved runtime reduction. The mask sizes are 0.6

Fig. 12 shows the softmax values calculated by the neural network for tokens from an example in the IMDb dataset. Only the first layer of the transformer architecture was modified in the experiment. As illustrated, the model correctly identifies significant tokens: adjectives expressing emotional tone receive the highest softmax values. With a mask size of 0.4, the selected key tokens are: “amazing”, “brilliant”, “capt”, “ivating”, “[SEP]”.

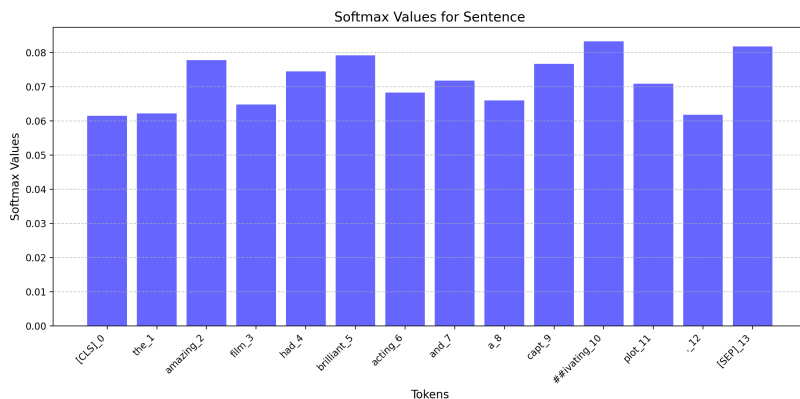


Fig. 12. Softmax values for tokens from an example in the IMDb dataset

Fig. 13 shows the dependence of classification accuracy on the number of tokens for the IMDb dataset. The mask size is 60%, corresponding to approximately 307 tokens. The graph illustrates that the number of tokens does not affect classification quality. This indicates that the model retains important information from sentences needed to determine review sentiment.

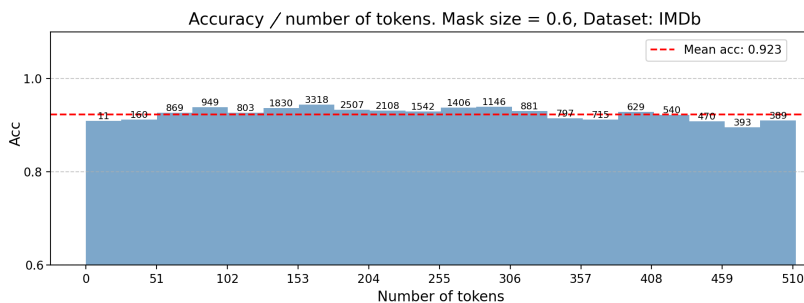


Fig. 13. Dependence of the classification accuracy on the number of tokens for dataset IMDb

In order to study different FCNs generating logits, we compare the following structures:

1. Linear (no hidden layers): consists of input and output layers with 768 and 1 parameters, respectively.
2. Medium: the number of hidden layers is 2; the number of parameters per hidden layer is 64.
3. Large: the number of hidden layers is 2; the number of parameters per hidden layer: 768.

The comparison shows that the medium version of the neural network is the optimal choice: it is less prone to overfitting than the Large version and outperforms the Linear model in accuracy. The comparison results are shown in Fig. 14, where the ROC AUC metric on the test set is measured for different values of the mask size. The analysis of training dynamics in Fig. 14 confirms the stability and effectiveness of the proposed learning procedure.

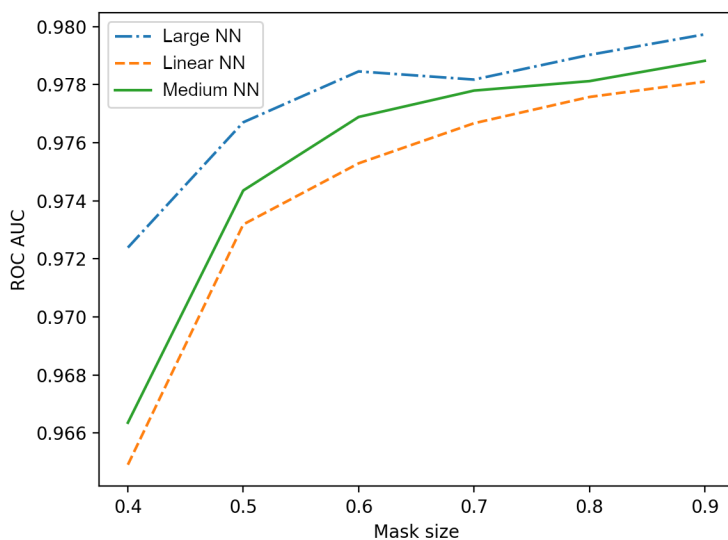


Fig. 14. Comparison of different FCNs using the dataset IMDB with the number of replaced layers equal to 6

The numerical experiments provides compelling evidence of the efficacy and advantages of our proposed differentiable mask learning approach. The consistent outperformance of the trainable mask over the random mask across all datasets and mask sizes underscores a critical finding: the reduction in computational cost achieved through token-selection need

not come at the expense of model accuracy, provided the selection is learned and optimized for the task.

6. Conclusion and Future Work. In this work, we proposed a differentiable token-selection mechanism for optimizing the self-attention computation in transformer models. The method introduces a lightweight auxiliary scoring network that learns to identify and select the most informative tokens via a Gumbel-Softmax-based relaxation, enabling hard token pruning while preserving differentiable training of the scoring module. By reducing the effective sequence length processed by attention layers, the proposed approach reduces the effective attention cost at inference time in the setting considered without modifying the underlying attention formulation.

A key advantage of the proposed method is that it is formulated as a plug-in token-selection module for an already trained transformer backbone. In the experimental setting considered here, the method is added to a pretrained model and optimized through an additional lightweight training stage for the scoring network, without redesigning the backbone architecture from scratch. The resulting attention mechanism achieves a controllable trade-off between computational efficiency and predictive performance through a single tunable parameter governing the number of selected tokens.

The experimental results indicate that the proposed method effectively reduces computational overhead while maintaining competitive performance in the considered setting. Across all datasets, the learned token-selection consistently outperformed random masking strategies, achieving an 8–12% reduction in inference time while preserving most of the baseline quality. The benefits were particularly pronounced for long sequences, where the method preserved high accuracy even when attending to only half of the input tokens. Moreover, the learned masks showed qualitatively interpretable behavior in the presented examples, often prioritizing semantically meaningful and sentiment-bearing tokens.

The present study has several limitations. First, the empirical evaluation is limited to DistilBERT and sentiment-classification benchmarks. Second, the paper focuses on comparison with the original backbone and with a random-mask baseline, because the main goal is to isolate the effect of the proposed plug-in module rather than to benchmark alternative transformer architectures. Third, repeated-run variance estimates are not included in the current experimental protocol. These limitations should be kept in mind when interpreting the scope of the reported results.

Overall, the results indicate that adaptive, learned token-selection provides a practical mechanism for reducing the effective attention cost at inference time in the considered experimental setting. Broader validation

on larger backbones and additional task families remains future work. Future work will also explore the interaction of the proposed method with alternative efficient-attention formulations, adaptive selection of the number of retained tokens, and richer context-aware scoring mechanisms.

References

1. Luong T., Pham H., Manning C. Effective approaches to attention-based neural machine translation. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 2015. pp. 1412–1421. DOI: 10.18653/v1/d15-1166.
2. Vaswani A., Shazeer N., Parmar N., et al. Attention is all you need. *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*. 2017. pp. 6000–6010.
3. Fournier Q., Caron G.M., Aloise D. A Practical Survey on Faster and Lighter Transformers. *ACM Computing Surveys*. 2023. vol. 55. no. 14s. pp. 1–40. DOI: 10.1145/3586074.
4. Han Z., Gao C., Liu J., et al. Parameter-Efficient Fine-Tuning for Large Models: A Comprehensive Survey. *arXiv preprint arXiv:2403.14608*. 2024.
5. Huang H., Liang Z., Fang Z., et al. A Concise Review of Long Context in Large Language Models. *Proceedings of the International Conference on Algorithms, Software Engineering, and Network Security (ASENS'24)*. 2024. pp. 563–566. DOI: 10.1145/3677182.3677282.
6. Shao M., Basit A., Karri R., Shafique M. Survey of Different Large Language Model Architectures: Trends, Benchmarks, and Challenges. *IEEE Access*. 2024. vol. 12. pp. 188664–188706. DOI: 10.1109/ACCESS.2024.3482107.
7. Sun W., Hu J., Zhou Y., et al. Speed Always Wins: A Survey on Efficient Architectures for Large Language Models. *arXiv preprint arXiv:2508.09834*. 2025.
8. Wang X., Salmani M., Omid P., et al. Beyond the limits: a survey of techniques to extend the context length in large language models. *Proceedings of the Thirty Third International Joint Conference on Artificial Intelligence (IJCAI'24)*. 2024. pp. 8299–8307. DOI: 10.24963/ijcai.2024/917.
9. Zhou Z., Ning X., Hong K., et al. A Survey on Efficient Inference for Large Language Models. *arXiv preprint arXiv:2404.14294*. 2024.
10. Beltagy I., Peters M.E., Cohan A. Longformer: The Long-Document Transformer. *arXiv preprint arXiv:2004.05150*. 2020.
11. Child R., Gray S., Radford A., Sutskever I. Generating Long Sequences with Sparse Transformers. *arXiv preprint arXiv:1904.10509*. 2019.
12. Wang S., Li B.Z., Khabsa M., et al. Linformer: Self-attention with Linear Complexity. *arXiv preprint arXiv:2006.04768*. 2020.
13. Choromanski K.M., Likhoshesterov V.L., Dohan D., et al. Rethinking Attention with Performers. *Proceedings of the International Conference on Learning Representations (ICLR'21)*. 2021. pp. 1–38.
14. Dai Z., Yang Z., Yang Y., et al. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019. pp. 2978–2988. DOI: 10.18653/v1/P19-1285.
15. Rae J.W., Potapenko A., Jayakumar S.M., et al. Compressive Transformers for Long-Range Sequence Modelling. *Proceedings of the International Conference on Learning Representations (ICLR'20)*. 2020. pp. 1–19.
16. Zaheer M., Guruganesh G., Dubey A., et al. Big Bird: transformers for longer sequences. *Proceedings of the 34th International Conference on Neural Information Processing System (NIPS'20)*. 2020. vol. 21. pp. 17283–17297.

17. Kitaev N., Kaiser L., Levskaya A. Reformer: The Efficient Transformer. Proceedings of the International Conference on Learning Representations (ICLR'20). 2020. pp. 1–12.
18. Kim S., Shen S., Thorsley D., et al. Learned Token Pruning for Transformers. Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'22). 2022. pp. 784–794. DOI: 10.1145/3534678.3539260.
19. Ravula A., Alberti C., Ainslie J., et al. ETC: Encoding Long and Structured Inputs in Transformers. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). 2020. 268–284.
20. Dao T., Fu D.Y., Ermon S., et al. FLASHATTENTION: fast and memory-efficient exact attention with IO-awareness. Proceedings of the 36th International Conference on Neural Information Processing System (NIPS'22). 2022. pp. 16344–16359.
21. Fu D.Y., Arora S., Grogan J., et al. Monarch MIXER: a simple sub-quadratic GEMM-based architecture. Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS'23). 2023. pp. 77546–77603.
22. Gu A., Dao T. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. Proceedings of the 1st Conference on Language Modeling. 2024. pp. 1–32.
23. Liu H., Abbeel P. Blockwise Parallel Transformer for Long Context Models. Proceedings of the 36th International Conference on Neural Large Information Processing Systems (NIPS'23). 2023. pp. 8828–8844. DOI: 10.52202/075280-0386.
24. Peng B., Alcaide E., Anthony Q., et al. RWKV: Reinventing RNNs for the Transformer Era. Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2023. pp. 1–30.
25. Guo Y., Ding L., Yuan Y., Wang G. Macformer: Transformer with Random Maclaurin Feature Attention. arXiv preprint arXiv:2408.11656. 2024.
26. Vladymyrov M., von Oswald J., Sandler M., Ge R. Linear Transformers are Versatile In-Context Learners. Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS'25). 2025. pp. 48784–48809. DOI: 10.52202/079017-1546.
27. Katharopoulos A., Vyas A., Pappas N., Fleuret F. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. Proceedings of the 37th International Conference on Machine Learning (PMLR). 2020. vol. 119. pp. 5156–5165.
28. Qin Z., Sun W., Deng H., et al. CosFormer: Rethinking Softmax In Attention. Proceedings of the International Conference on Learning Representations (ICLR'22). 2022. pp. 1–15.
29. Liu J., Pan Z., He H., et al. EcoFormer: Energy-Saving Attention with Linear Complexity. Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS'22). 2022. pp. 10295–10308.
30. Sanh V., Debut L., Chaumond J., Wolf T. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and Lighter. arXiv preprint arXiv:1910.01108. 2019.
31. Maas A.L., Daly R.E., Pham P.T., et al. Learning word vectors for sentiment analysis. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. 2011. vol. 1. pp. 142–150.
32. Socher R., Perelygin A., Wu J., et al. Recursive deep models for semantic compositionality over a sentiment treebank. Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2013. pp. 1631–1642.
33. Pang B., Lee L. Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05). 2005. pp. 115–124. DOI: 10.3115/1219840.1219855.
34. Zhang X., Zhao J., Lecun Y. Character-level convolutional networks for text classification. Proceedings of the 29th International Conference on Neural Information Processing Systems (NIPS'15). 2015. pp. 649–657.

Konstantinov Andrey — Ph.D., Senior Lecturer, Higher School of Artificial Intelligence Technologies, Federal State Autonomous Educational Institution of Higher Education «Peter the Great St. Petersburg Polytechnic University» (SPbPU). Research interests: machine learning, survival analysis, explainable intelligence. The number of publications — 50. konstantinov_av@spbstu.ru; 29, Polytechnicheskaya st., 195251, St. Petersburg, Russia; office phone: +7(911)954-5565.

Utkin Lev — Ph.D., Dr.Sci., Professor, Higher School of Artificial Intelligence Technologies, Federal State Autonomous Educational Institution of Higher Education «Peter the Great St. Petersburg Polytechnic University» (SPbPU). Research interests: machine learning, survival analysis, theory of generalized probabilities, decision-making. The number of publications — 350. lev.utkin@gmail.com; 29, Polytechnicheskaya st., 195251, St. Petersburg, Russia; office phone: +7(921)344-6390.

Khomets Semen — Master's Student, Higher School of Artificial Intelligence Technologies, Federal State Autonomous Educational Institution of Higher Education «Peter the Great St. Petersburg Polytechnic University» (SPbPU). Research interests: machine learning. The number of publications — 2. homets_sp@spbstu.ru; 29, Polytechnicheskaya st., 195251, St. Petersburg, Russia; office phone: +7(928)135-4848.

Acknowledgements. The work was supported by the Ministry of Science and Higher Education of the Russian Federation within the framework of the state assignment FSEG-2024-0027.

А.В. КОНСТАНТИНОВ, Л.В. УТКИН, С.П. ХОМЕЦ
**ПОДХОД НА ОСНОВЕ ОПЕРАЦИИ GUMBEL-SOFTMAX ДЛЯ
ОТБОРА ЗНАЧИМЫХ ТОКЕНОВ В МЕХАНИЗМЕ
САМОВНИМАНИЯ**

Константинов А.В., Уткин Л. В., Хомец С.П. Подход на основе операции Gumbel-Softmax для отбора значимых токенов в механизме самовнимания.

Аннотация. Квадратичная вычислительная сложность механизма самовнимания в трансформерах остается существенным узким местом для эффективной обработки длинных последовательностей. В данной статье представлен метод снижения вычислительных затрат на использование механизма внимания с помощью дифференцируемого обучения масок, обеспечивающий динамический, зависящий от входных данных отбор токенов и при этом сохраняющий способность модели улавливать глобальную контекстную информацию. Предложенный подход не накладывает каких-либо конкретных архитектурных ограничений и может быть интегрирован в существующие модели трансформеров с минимальными изменениями. В отличие от существующих методов разреженного или приближенного внимания, предложенный подход обучается специфичным для задачи маскам отбора токенов полностью дифференцируемым образом, без наложения предопределенных шаблонов разреженности или архитектурных ограничений. Наш подход использует механизм на основе Gumbel-Softmax для обучения масок выбора токенов в сквозном режиме, предлагая решение, которое сокращает время вывода, сохраняя при этом точность модели. Эксперименты на четырех эталонных наборах данных, предназначенных для анализа настроения, показывают, что предложенный метод обеспечивает сокращение времени вывода на 8-12% при относительном снижении точности менее чем на 1% по сравнению с базовыми моделями. Обучаемые маски поддаются интерпретации, последовательно отдают приоритет токенам, содержащим эмоциональную окраску, и поддерживают эффективный, масштабируемый вывод с помощью трансформеров.

Ключевые слова: трансформер, самовнимание, Gumbel-Softmax, выбор токенов, дифференцируемая маска.

Литература

1. Luong T., Pham H., Manning C. Effective approaches to attention-based neural machine translation // Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. 2015. pp. 1412–1421. DOI: 10.18653/v1/d15-1166.
2. Vaswani A., Shazeer N., Parmar N., et al. Attention is all you need // Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). 2017. pp. 6000–6010.
3. Fournier Q., Caron G.M., Aloise D. A Practical Survey on Faster and Lighter Transformers // ACM Computing Surveys. 2023. vol. 55. no. 14s. pp. 1–40. DOI: 10.1145/3586074.
4. Han Z., Gao C., Liu J., et al. Parameter-Efficient Fine-Tuning for Large Models: A Comprehensive Survey // arXiv preprint arXiv:2403.14608. 2024.
5. Huang H., Liang Z., Fang Z., et al. A Concise Review of Long Context in Large Language Models // Proceedings of the International Conference on Algorithms, Software Engineering, and Network Security (ASENS'24). 2024. pp. 563–566. DOI: 10.1145/3677182.3677282.

6. Shao M., Basit A., Karri R., Shafique M. Survey of Different Large Language Model Architectures: Trends, Benchmarks, and Challenges // IEEE Access. 2024. vol. 12. pp. 188664–188706. DOI: 10.1109/ACCESS.2024.3482107.
7. Sun W., Hu J., Zhou Y., et al. Speed Always Wins: A Survey on Efficient Architectures for Large Language Models // arXiv preprint arXiv:2508.09834. 2025.
8. Wang X., Salmani M., Omid P., et al. Beyond the limits: a survey of techniques to extend the context length in large language models // Proceedings of the Thirty Third International Joint Conference on Artificial Intelligence (IJCAI'24). 2024. pp. 8299–8307. DOI: 10.24963/ijcai.2024/917.
9. Zhou Z., Ning X., Hong K., et al. A Survey on Efficient Inference for Large Language Models // arXiv preprint arXiv:2404.14294. 2024.
10. Beltagy I., Peters M.E., Cohan A. Longformer: The Long-Document Transformer // arXiv preprint arXiv:2004.05150. 2020.
11. Child R., Gray S., Radford A., Sutskever I. Generating Long Sequences with Sparse Transformers // arXiv preprint arXiv:1904.10509. 2019.
12. Wang S., Li B.Z., Khabza M., et al. Linformer: Self-attention with Linear Complexity // arXiv preprint arXiv:2006.04768. 2020.
13. Choromanski K.M., Likhoshesterov V.L., Dohan D., et al. Rethinking Attention with Performers // Proceedings of the International Conference on Learning Representations (ICLR'21). 2021. pp. 1–38.
14. Dai Z., Yang Z., Yang Y., et al. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context // Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019. pp. 2978–2988. DOI: 10.18653/v1/P19-1285.
15. Rae J.W., Potapenko A., Jayakumar S.M., et al. Compressive Transformers for Long-Range Sequence Modelling // Proceedings of the International Conference on Learning Representations (ICLR'20). 2020. pp. 1–19.
16. Zaheer M., Guruganesh G., Dubey A., et al. Big Bird: transformers for longer sequences // Proceedings of the 34th International Conference on Neural Information Processing System (NIPS'20). 2020. vol. 21. pp. 17283–17297.
17. Kitaev N., Kaiser L., Levskaya A. Reformer: The Efficient Transformer // Proceedings of the International Conference on Learning Representations (ICLR'20). 2020. pp. 1–12.
18. Kim S., Shen S., Thorsley D., et al. Learned Token Pruning for Transformers // Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'22). 2022. pp. 784–794. DOI: 10.1145/3534678.3539260.
19. Ravula A., Alberti C., Ainslie J., et al. ETC: Encoding Long and Structured Inputs in Transformers // Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). 2020. 268–284.
20. Dao T., Fu D.Y., Ermon S., et al. FLASHATTENTION: fast and memory-efficient exact attention with IO-awareness // Proceedings of the 36th International Conference on Neural Information Processing System (NIPS'22). 2022. pp. 16344–16359.
21. Fu D.Y., Arora S., Grogan J., et al. Monarch MIXER: a simple sub-quadratic GEMM-based architecture // Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS'23). 2023. pp. 77546–77603.
22. Gu A., Dao T. Mamba: Linear-Time Sequence Modeling with Selective State Spaces // Proceedings of the 1st Conference on Language Modeling. 2024. pp. 1–32.
23. Liu H., Abbeel P. Blockwise Parallel Transformer for Long Context Models // Proceedings of the 36th International Conference on Neural Large Information Processing Systems (NIPS'23). 2023. pp. 8828–8844. DOI: 10.52202/075280-0386.

24. Peng B., Alcaide E., Anthony Q., et al. RWKV: Reinventing RNNs for the Transformer Era // *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 2023. pp. 1–30.
25. Guo Y., Ding L., Yuan Y., Wang G. Macformer: Transformer with Random Maclaurin Feature Attention // *arXiv preprint arXiv:2408.11656*. 2024.
26. Vladymyrov M., von Oswald J., Sandler M., Ge R. Linear Transformers are Versatile In-Context Learners // *Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS'25)*. 2025. pp. 48784–48809. DOI: 10.52202/079017-1546.
27. Katharopoulos A., Vyas A., Pappas N., Fleuret F. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention // *Proceedings of the 37th International Conference on Machine Learning (PMLR)*. 2020. vol. 119. pp. 5156–5165.
28. Qin Z., Sun W., Deng H., et al. CosFormer: Rethinking Softmax In Attention // *Proceedings of the International Conference on Learning Representations (ICLR'22)*. 2022. pp. 1–15.
29. Liu J., Pan Z., He H., et al. EcoFormer: Energy-Saving Attention with Linear Complexity // *Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS'22)*. 2022. pp. 10295–10308.
30. Sanh V., Debut L., Chaumond J., Wolf T. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and Lighter // *arXiv preprint arXiv:1910.01108*. 2019.
31. Maas A.L., Daly R.E., Pham P.T., et al. Learning word vectors for sentiment analysis // *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. 2011. vol. 1. pp. 142–150.
32. Socher R., Perelygin A., Wu J., et al. Recursive deep models for semantic compositionality over a sentiment treebank // *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. 2013. pp. 1631–1642.
33. Pang B., Lee L. Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales // *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. 2005. pp. 115–124. DOI: 10.3115/1219840.1219855.
34. Zhang X., Zhao J., Lecun Y. Character-level convolutional networks for text classification // *Proceedings of the 29th International Conference on Neural Information Processing Systems (NIPS'15)*. 2015. pp. 649–657.

Константинов Андрей Владимирович — канд. физ.-мат. наук, старший преподаватель, высшая школа технологий искусственного интеллекта, Федеральное государственное автономное образовательное учреждение высшего образования «Санкт-Петербургский политехнический университет Петра Великого» (СПбПУ). Область научных интересов: машинное обучение, анализ выживаемости, объяснительный интеллект. Число научных публикаций — 50. konstantinov_av@spbstu.ru; ул. Политехническая, 29, 195251, Санкт-Петербург, Россия; р.т.: +7(911)954-5565.

Уткин Лев Владимирович — д-р техн. наук, профессор, высшая школа технологий искусственного интеллекта, Федеральное государственное автономное образовательное учреждение высшего образования «Санкт-Петербургский политехнический университет Петра Великого» (СПбПУ). Область научных интересов: машинное обучение, анализ выживаемости, теория обобщенных вероятностей, принятие решений. Число научных публикаций — 350. lev.utkin@gmail.com; ул. Политехническая, 29, 195251, Санкт-Петербург, Россия; р.т.: +7(921)344-6390.

Хомец Семен Павлович — магистрант, высшая школа технологий искусственного интеллекта, Федеральное государственное автономное образовательное учреждение высшего образования «Санкт-Петербургский политехнический университет

Петра Великого» (СПбПУ). Область научных интересов: машинное обучение. Число научных публикаций — 2. homets_sp@spbstu.ru; ул. Политехническая, 29, 195251, Санкт-Петербург, Россия; р.т.: +7(928)135-4848.

Поддержка исследований. Работа выполнена при поддержке Министерства науки и высшего образования РФ в рамках государственного задания FSEG-2024-0027.