Pool Me Wisely: On the Effect of Pooling in Transformer-Based Models

Sofiane Ennadir*
King AI Labs, Microsoft Gaming
sofiane.ennadir@king.com

Levente Zólyomi*†
NXAI GmbH
levente.zolyomi@nx-ai.com

Oleg Smirnov
King AI Labs, Microsoft Gaming
oleg.smirnov@microsoft.com

Tianze Wang†
Kreditz AB
tianze.wang@kreditz.com

John Pertoft
King AI Labs, Microsoft Gaming
john.pertoft@king.com

Filip Cornell†
Amazon
filipco@amazon.com

Lele Cao
King AI Labs, Microsoft Gaming
lelecao@microsoft.com

Abstract

Transformer models have become the dominant backbone for sequence modeling, leveraging self-attention to produce contextualized token representations. These are typically aggregated into fixed-size vectors via pooling operations for downstream tasks. While much of the literature has focused on attention mechanisms, the role of pooling remains underexplored despite its critical impact on model behavior. In this paper, we introduce a theoretical framework that rigorously characterizes the expressivity of Transformer-based models equipped with widely used pooling methods by deriving closed-form bounds on their representational capacity and the ability to distinguish similar inputs. Our analysis extends to different variations of attention formulations, demonstrating that these bounds hold across diverse architectural variants. We empirically evaluate pooling strategies across tasks requiring both *global* and *local* contextual understanding, spanning three major modalities: computer vision, natural language processing, and time-series analysis. Results reveal consistent trends in how pooling choices affect accuracy, sensitivity, and optimization behavior. Our findings unify theoretical and empirical perspectives, providing practical guidance for selecting or designing pooling mechanisms suited to specific tasks. This work positions pooling as a key architectural component in Transformer models and lays the foundation for more principled model design beyond attention alone.

1 Introduction

The profound impact of the Transformer [39] architectures across different modalities and in cross-modal modeling cannot be underestimated. These models have become the building stones of

*Equal contribution.
†Work conducted while at King AI Labs.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

large-scale systems capable of successfully addressing a multitude of downstream tasks in computer vision [1], natural language processing (NLP) [37, 14], and time series analysis [22, 12]. Since these models typically require substantial training data to perform effectively, pre-trained large-scale models trained with self-supervised objectives such as autoregressive, autoencoding, contrastive, or hybrid formulations have become the standard. These models are first trained to capture rich, contextualized representations and are later fine-tuned by attaching a classification or regression head specific to the downstream task, thereby serving as feature extractors.

Transformer-based models produce a sequence of token-level embeddings, which are typically aggregated into a single vector that captures task-relevant information. This operation, commonly referred to as *pooling*, has been widely studied in domains such as multi-modal learning [10] and Graph Neural Networks [23], where it is recognized as a key architectural component. The choice of pooling function directly affects the content of the final representation and thus the model's performance on downstream tasks. Recent studies have increasingly focused on understanding Transformer models from both empirical and theoretical perspectives. However, much of this research has concentrated on the backbone encoder, which processes the input into contextual embeddings, often neglecting the final pooling step that aggregates these into a single representation for prediction. Although some empirical work [35] has explored the effects of various pooling strategies, a systematic theoretical analysis is still largely absent.

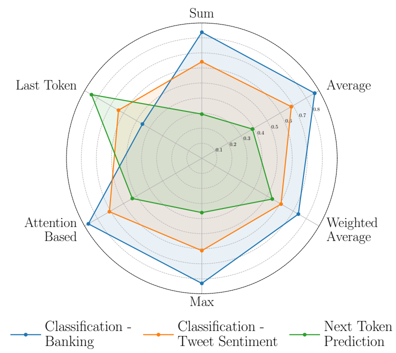


Figure 1: Performance of different pooling strategies using a GPT-2 pre-trained model.

In this work, we close this gap by providing an end-to-end analysis of Transformer-based models that explicitly incorporates the pooling stage. We introduce a theoretical framework to quantify Transformer expressivity, measuring the model's ability to distinguish dissimilar inputs while preserving similarity. Applying this framework, we analyze how common pooling functions influence expressivity by encoding different token-level properties. From these insights, we derive practical guidelines for choosing pooling methods based on a task's need for local versus global context. We then validate our theory with experiments in vision, natural language, and time-series domains using standard models and datasets. The results confirm that no single pooling strategy dominates all tasks, underscoring the importance of task-tailored pooling design. To our knowledge, this is the first theoretical examination of pooling mechanisms, offering a unified treatment of standard strategies found in the literature.

2 Related Work

A growing body of work has provided theoretical insights into Transformer-based models and their attention mechanisms. Prior research has explored a broad range of topics, including training dynamics [36], inductive biases [18], and in-context learning [41]. Much of this literature focuses on the core Transformer backbone, proposing architectural improvements and refined training procedures. However, in practical scenarios, particularly those involving large pretrained models, the backbone is typically frozen, and a lightweight classification or regression head is appended after the final pooling layer. This common design choice limits the direct applicability of many of the previously proposed modifications.

Prior studies have demonstrated that different pooling strategies can significantly influence downstream performance. For example, [6] showed that, in Vision Transformers (ViT), the choice between using a CLS token and average pooling leads to measurable differences in classification accuracy. Those findings were reaffirmed in a follow-up study [28], which examined the influence of the classification token across different Transformer layers. In the language domain, [19] conducted an empirical analysis of BERT-based [5] embedding models and decoder-only models from the GPT [33, 2] family, along with their standard pooling layers, and proposed a new technique to mitigate information dilution and recency bias commonly observed in the respective pooling operations. In addition, [35] systematically evaluated combinations of attention mechanisms and pooling methods in large language models, and introduced an attention-based pooling approach that aggregates representations from multiple hidden layers. However, the performance of these novel pooling mechanisms has been shown to vary considerably across different types of tasks.

Multiple studies have further examined the impact of pooling strategies across a variety of settings [38, 43]. This line of research concludes that the optimal pooling method generally depends on the specific downstream task, as well as other factors such as model size. Although these works provide extensive empirical evidence, to the best of our knowledge, none have proposed a principled theoretical framework to explain the behavior of pooling operations in a broader context.

More recently, increased attention has been directed toward studying the theoretical properties of Transformer architectures through the lens of Lipschitz continuity, with the goal of understanding its behavior and dynamics. For instance, [15] propose an L2-based attention mechanism that replaces the standard dot-product, providing both theoretical and empirical evidence of its Lipschitz continuity. In a similar direction, [4] introduce LipschitzNorm, a normalization technique designed to enforce Lipschitz continuity within the self-attention mechanism. Building on these efforts, [32] further redesign the full Transformer architecture to ensure that the model remains Lipschitz continuous throughout. However, these analyses typically do not extend to the final pooling operation, despite its widespread use in practical applications.

Our work extends both lines of inquiry by closing the gap between empirical findings and theoretical understanding, and contributing to a deeper comprehension of how pooling functions influence the performance of Transformer-based models.

3 Preliminaries

We start by reviewing the Transformer architecture and its key components, forming the basis for the concepts of our theoretical study.

Transformer Architecture. Let $X \in \mathcal{X} \subseteq \mathbb{R}^{n \times d}$ denote a sequence of n tokens, where each token $x_i \in \mathbb{R}^d$. The backbone of a transformer $h : \mathcal{X} \subseteq \mathbb{R}^{n \times d} \to \mathcal{Z} \subseteq \mathbb{R}^{n \times d}$, as introduced in [39], is the *self-attention* mechanism, which computes a weighted combination of all token representations. Specifically, given learnable query, key, and value parameter matrices W^Q, W^K, $W^V \in \mathbb{R}^{d \times (d/H)}$, the output of a single attention head AH for input X is defined as

$$\mathrm{AH}(X) = \mathrm{softmax}\left(\frac{(XW^Q)(XW^K)^\top}{\sqrt{d/H}}\right)(XW^V), \tag{1}$$

where H denotes the number of parallel attention heads and d/H is the dimension per head. In practice, multiple attention heads AH_i are computed in parallel, then concatenated and projected using a learnable weight matrix $W^O \in \mathbb{R}^{d \times d}$, yielding the Multi-Head Attention (MHA) operation:

$$\mathrm{MHA}(X) = \mathrm{concat}\big(\mathrm{AH}_1(X), \mathrm{AH}_2(X), \ldots, \mathrm{AH}_H(X)\big)W^O. \tag{2}$$

Attention Block. In addition to MHA, each Transformer attention block (AB) incorporates a residual connection, layer normalization [20] and a position-wise feed-forward network (FFN), and can be written in the following two steps:

$$X' = \mathrm{LN}\big(X + \mathrm{MHA}(X)\big); \quad \mathrm{AB}(X) = \mathrm{LN}\big(X' + \mathrm{FFN}(X')\big). \tag{3}$$

with $\mathrm{LN}(\cdot)$ denoting the layer normalization operation. $\mathrm{FFN}(\cdot)$ is a feed-forward network, formulated as $\mathrm{FFN}(X') = \sigma\big(X'W_{FFN}\big)$, with σ being a non-linear activation function. While different placements of normalization layers, commonly called Pre-LN and Post-LN, have been examined in prior work [21], this study focuses on the original Post-LN setup. We note that our main findings are transferable to other configurations.

Pooling. Given the output of a Transformer backbone $Z \in \mathcal{Z} \subseteq \mathbb{R}^{n \times d}$, a pooling function $g : \mathcal{Z} \to \mathcal{Y} \subseteq \mathbb{R}^d$ produces a fixed-size embedding for downstream tasks. Common choices are Average pooling, which computes the mean over tokens; Sum pooling, which sums the token embeddings; Max pooling, which takes the elementwise maximum; and Last-token pooling, which selects a designated token (for example the final or CLS token).These operations can be formally defined as:

$$g_{\text{Avg}}(Z) = \frac{1}{n}\sum_{i=1}^{n} Z[i,:]; \quad g_{\text{Sum}}(Z) = \sum_{i=1}^{n} Z[i,:]; \quad g_{\text{Max}}(Z) = \max_i Z[i,:]; \quad g_{\text{Last}}(Z) = Z[n,:].$$

Problem Setup. Let $f \colon \mathcal{X} \subseteq \mathbb{R}^{n\times d} \to \mathcal{Y} \subseteq \mathbb{R}^d$ be a Transformer-based model incorporating a final pooling layer. For our theoretical analysis, we model f as a single MHA block with H heads, followed by a one-layer FFN and the layer-normalization variant of [32], which is provably stable under small input perturbations. We assume all activation functions are 1-Lipschitz (e.g. ReLU, LeakyReLU, TanH) [40], and that the input space is bounded, $\mathcal{X} \subset [0, B]^{n\times d}$ This bound is realistic: in vision and time-series applications $B = 1$ after normalization, and in NLP the embedding process usually results in bounded input representations due to the initialization of the embedding matrix.

4 On the Expressivity of Transformer-Based Models

In this section, we introduce the notion of expressivity for Transformer-based models (TBMs). Building upon this definition, we develop a theoretical analysis of several attention mechanisms and commonly used pooling strategies.

4.1 Expressivity of TBMs

Inspired by work in graph representation learning, we define *expressivity* as the capacity of a model to distinguish between similar and dissimilar inputs [44, 27, 26]. Specifically, by being able to distinguish between such cases, a model is able to produce meaningful representation that could be used by a classification or regression head to produce the final downstream task. For instance, in natural language processing, two semantically similar sentences should yield representations that are closer in the output embedding space than those produced by two semantically disparate sentences. In this perspective, defining an accurate measure of semantic similarity that is applicable across diverse domains such as NLP and computer vision, is crucial and fundamental to evaluating expressivity. Let $\mathcal{X}$ and $\mathcal{Y}$ denote the input and output spaces, respectively, we consider both spaces to be measurable and equipped with a measures $|\cdot|_{\mathcal{X}}$ and $|\cdot|_{\mathcal{Y}}$. With a well-designed embedding function, semantically similar elements from the input space are mapped to proximate points in the output space. Consequently, the distances in $\mathcal{Y}$ are expected to accurately reflect the semantic relationships present in $\mathcal{X}$.

Let $f \colon \mathcal{X} \subseteq \mathbb{R}^{n\times d} \to \mathcal{Y} \subseteq \mathbb{R}^d$ be a TBM as defined in Section 3. For a given input $X \in \mathcal{X}$, we define its *neighborhood* with respect to the input distance metric and a threshold ϵ by:

$$\mathcal{B}(X, \epsilon) = \{\tilde{X} \in \mathcal{X} \, : \, |X - \tilde{X}|_{\mathcal{X}} \leq \epsilon\}.$$

Since the desired behavior is for inputs in close proximity to yield similar output representations, we consider the following measure:

$$\mathcal{E}_\epsilon[f] = \mathbb{P}_{X\sim\mathcal{D}_{\mathcal{X}}}\left[\tilde{X} \in \mathcal{B}(X, \epsilon) : d_{\mathcal{Y}}(f(\tilde{X}), f(X)) > \sigma\right], \tag{4}$$

where $\mathcal{D}_{\mathcal{X}}$ represents the underlying distribution over the input space $\mathcal{X}$, and $d_{\mathcal{Y}}$ is a distance metric on $\mathcal{Y}$. The quantity $\mathcal{E}_\epsilon[f]$ encapsulates the probability that two inputs, which are similar (within the same neighborhood) in the input space $\mathcal{X}$, are mapped to outputs that differ by more than a threshold σ. Intuitively, a small input distance should yield a correspondingly small output distance, while larger differences in the input should result in more pronounced variations in the output. A model that appropriately distinguishes these nuances is said to exhibit stronger expressive power, which is as motivated previously a vital attribute for achieving robust downstream performance. Definition 4.1 reflects such expressivity in the context of TBMs.

Definition 4.1. Let $f \colon \mathcal{X} \subseteq \mathbb{R}^{n\times d} \to \mathcal{Y} \subseteq \mathbb{R}^d$ be a TBM. The model f is said to be $(\epsilon, \sigma, \gamma)$-expressive if $\mathcal{E}_\epsilon[f] \leq \gamma$.

Definition 4.1 depends on several hyperparameters. The threshold ϵ specifies when two inputs are considered semantically similar and is inherently application-specific. For instance, a minor perturbation in an image may be negligible, while the same in a financial time series could be meaningful. The parameter σ defines the acceptable variation in the output space for representations to be considered similar. By setting ϵ based on domain knowledge, the interaction between ϵ and σ allows us to capture the model's expressive capacity in a way that reflects the semantic structure of the data. This formulation highlights the model's adaptability and expressivity in application-specific contexts.

4.2 Expressivity of Pooling Strategies

Based on Definition 4.1, and given fixed values of ϵ and σ, our objective is to quantify the corresponding expressivity parameter γ for different pooling strategies. This analysis allows us to assess how the choice of pooling influences the model's ability to distinguish semantically meaningful variations in the input. Throughout the remainder of this paper, $\|\cdot\|$ denotes the operator norm.

Theorem 4.2. *Let $f\colon \mathcal{X} \subseteq \mathbb{R}^{n\times d} \to \mathcal{Y} \subseteq \mathbb{R}^d$ be a TBM following the framework introduced in Section 3. In respect to Definition 4.1, we have:*

- *If f employs Average pooling, then f is (ϵ,σ,γ)-expressive with $\gamma = \frac{\epsilon}{\sigma\sqrt{n}}\left(\frac{d}{d-1}\right)^2 C_1C_2$*
- *If f employs Sum pooling, then f is (ϵ,σ,γ)-expressive with $\gamma = \frac{\sqrt{n}\,\epsilon}{\sigma}\left(\frac{d}{d-1}\right)^2 C_1C_2$*
- *If f employs Last-token pooling, then f is (ϵ,σ,γ)-expressive with $\gamma = \frac{\epsilon}{\sigma}\left(\frac{d}{d-1}\right)^2 C_1C_2$*
- *If f employs Max pooling, then f is (ϵ,σ,γ)-expressive with $\gamma = \frac{\epsilon\sqrt{\min(n,d)}}{\sigma}\left(\frac{d}{d-1}\right)^2 C_1C_2$,*

with: $C_1 = 1+\|W_O\|\sqrt{H}\,\max_h\left[\left\|W^{V,h}\right\|\left(4\,\frac{n}{\sqrt{d/H}}\,B^2\left\|W^{Q,h}\right\|\left\|W^{K,h}\right\|+1\right)\right],\quad C_2 = 1+\left\|W_{FFN}\right\|.$

Theorem 4.2 shows that expressivity bounds across pooling strategies depend on shared architectural parameters, including the number of attention heads H, embedding dimension d, and sequence length n, captured through constants C_1 and C_2. These constants reflect the norms of key model components, such as attention weights, projection layers, and feed-forward networks. Each pooling function introduces distinct scaling effects, shaping how these elements combine to influence the model's ability to separate similar from dissimilar inputs.

For Average pooling, the bound scales with $1/\sqrt{n}$, smoothing the output by evenly distributing token contributions. This favors tasks where *global* structure matters more than individual token details. In contrast, Sum pooling scales with $\sqrt{n}$, amplifying token-level variation. This is useful for tasks where *localized* information is essential. Using a single token (e.g., the last or CLS token) leads to a scaling of 1, preserving variations without change. This suits scenarios where a specific token encodes the most relevant context, such as in sentiment analysis. Max pooling introduces a bound that scales as $\sqrt{\min(n,d)}$. When d is large relative to n, it behaves similarly to Sum pooling, capturing fine-grained differences. When d is smaller, it emphasizes broader context. This flexibility enables Max pooling to shift between local and global focus based on model size and sequence length.

Overall, the theoretical results emphasize that pooling is a key factor in how Transformer models aggregate local token information into a global representation. Theorem 4.2 formalizes how this choice affects model expressivity across different settings.

On the generalization to multi-layer TBMs. We note that the current theoretical analysis focuses on a single-layer Transformer-based model; nonetheless, the results naturally extend to the multi-layer case. Specifically, a Transformer model with L layers, denoted as $f^{(L)}$, can be expressed as a composition of L single-layer functions: $f^{(L)}(x) = f^{(L-1)} \circ f^{(L-2)} \circ \cdots \circ f^{(1)}(x)$. Under this formulation, and following standard results from Lipschitz continuity, the overall expressivity bound bound γ for each pooling becomes a multiplicative composition of the bounds for each individual layer. As a result, our theoretical study remains applicable to deeper architectures, as confirmed by experiments involving exclusively multi-layer models.

4.3 Expressivity of Alternative Attention Mechanisms

Recent studies have proposed alternative formulations of the scaled dot-product self-attention mechanism to improve model behavior and facilitate theoretical analysis. For example, L2 Multi-Head Attention (L2-MHA) [15] employs an L2-kernel attention function, while LipsFormer [32] replaces the dot-product with a scaled cosine similarity. The theoretical results from the previous section are general and extend to these variants. In the following, we consider the same problem setup, with the only change being the use of an alternative attention mechanism in place of the standard formulation.

Lemma 4.3. *Let $f\colon \mathcal{X} \subseteq \mathbb{R}^{n\times d} \to \mathcal{Y} \subseteq \mathbb{R}^d$ be a L2-MHA-based TBM [15]. In respect to Definition 4.1, the following holds:*

- *If f employs Average pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon}{\sigma\sqrt{n}} \left(\frac{d}{d-1}\right)^2 C_1 C_2$*
- *If f employs Sum pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\sqrt{n}\,\epsilon}{\sigma} \left(\frac{d}{d-1}\right)^2 C_1 C_2$*
- *If f employs Last-token pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon}{\sigma} \left(\frac{d}{d-1}\right)^2 C_1 C_2$*
- *If f employs Max pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon\sqrt{\min(n,d)}}{\sigma} \left(\frac{d}{d-1}\right)^2 C_1 C_2$,*

$$\text{with } C_1 = 1 + \frac{\sqrt{n}}{\sqrt{d/H}} \left(4W_O \left(\frac{n}{e}\right) + 1\right) \left(\sqrt{\sum_h \|W^{Q,h}\|^2 \|W^{V,h}\|^2}\right) \|W^O\|, \quad C_2 = 1 + \|W_{FFN}\|.$$

Lemma 4.3 analyzes an L2-based attention mechanism [15] in which the query and key matrices are tied. This constraint influences the constant C_1 in the expressivity bound, reflecting the interaction among shared parameters, sequence length n, number of heads H, and embedding dimension d. Compared to the standard dot-product formulation, this structure alters how the L2-kernel shapes the bound, resulting in slightly different expressivity dependencies.

The pooling-related terms in Lemma 4.3 are consistent with those derived under standard self-attention, and the same trade-offs between local and global context remain applicable. Similar behavior is observed in Swin [24] and LipsFormer [32], which employs scaled cosine similarity and normalizes the key, query, and value matrices to maintain Lipschitz-continuity. Lemma 4.4 provides the corresponding bound for a single-layer model with H attention heads and window size w.

In both cases, the bounds reveal comparable structure, reinforcing that pooling remains a critical component in controlling the balance between local preservation and global aggregation in TBMs.

Lemma 4.4. *Let $f\colon \mathcal{X} \to \mathcal{Y}$ to be a function based on the LipsFormer [32] framework, with corresponding hyper-parameters $\nabla, \nu, \tau > 0$ and window size w. In respect to Definition 4.1, we have:*

- *If f is based on Average pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon}{\sigma \times \sqrt{n}} \times \left(\frac{d}{d-1}\right)^2 C_1 C_2$*
- *If f is based on Sum pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\sqrt{n} \times \epsilon}{\sigma} \times \left(\frac{d}{d-1}\right)^2 C_1 C_2$*
- *If f is based on Last-token pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon}{\sigma} \times \left(\frac{d}{d-1}\right)^2 C_1 C_2$*
- *If f is based on Max pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon\sqrt{\min(n,d)}}{\sigma} \times \left(\frac{d}{d-1}\right)^2 C_1 C_2$,*

where

$$C_1 = 1 + \|W_O\|\sqrt{H} \max_h \Big\{ 2w(w-1)\nu\tau\nabla^{-\frac{1}{2}} \|W_h^K\| + 2(w-1)\nu\tau\nabla^{-\frac{1}{2}} \|W_h^Q\| + 2w\nu\nabla^{-\frac{1}{2}} \|W_h^V\| \Big\},$$
$$C_2 = \big(1 + \|W_{FFN}\|\big)$$

5 Experimental Validation

Our theoretical analysis shows that the pooling strategy influences the classifier's expressivity bound via a leading multiplicative factor, which can be contractive ($1/\sqrt{n}$), non-expansive (1), or expansive ($\sqrt{n}$ or $\sqrt{\min(n,d)}$). Contractive methods like Average pooling enhance stability by smoothing small variations, whereas expansive methods such as Last-token and Sum pooling increase expressivity, but can be more sensitive to minor perturbations. Therefore, we posit that pooling should be selected based on task requirements: tasks emphasizing global context (e.g., image inpainting or text classification) benefit from contractive pooling, while those relying on local detail (e.g., next-token prediction) may perform better with expansive alternatives. In this section, we validate these theoretical insights through empirical evaluation, to demonstrate the applicability of our findings and provide practical guidance for choosing pooling strategies in different domains.

Experimental Setup. We evaluate how pooling choice affects downstream performance across three domains where TBMs have shown strong results: (a) computer vision, (b) natural language processing, and (c) time series analysis. For each modality, we select a diverse set of established benchmarks with tasks requiring *global* and *local* contexts. Across all settings, we examine commonly used pooling methods: (i) Last-token pooling (or CLS/EOS, depending on the task), (ii) Average (Avg) pooling, (iii) Sum pooling, and (iv) Max pooling. We also include two learnable strategies: (v) Attention (Attn) pooling, which uses a learnable latent dictionary attended by the model output [19, 35], and (vi) Weighted Average (W-Avg) pooling, which learns scalar weights over token positions. Further details on training and evaluation protocols are provided in Appendix D. Our code and implementation to reproduce the results is available in the following link: https://github.com/king/transformer-pooling.

5.1 Expressivity Analysis

We begin by empirically analyzing the expressivity of the pooling strategies under study, in accordance with the theoretical bounds introduced earlier. Using the framework in Section 4.1, we define local neighborhoods by injecting Gaussian noise into input samples, scaled to a chosen ϵ. We then compute the average distance between the resulting pooled outputs, yielding an empirical estimate of γ.

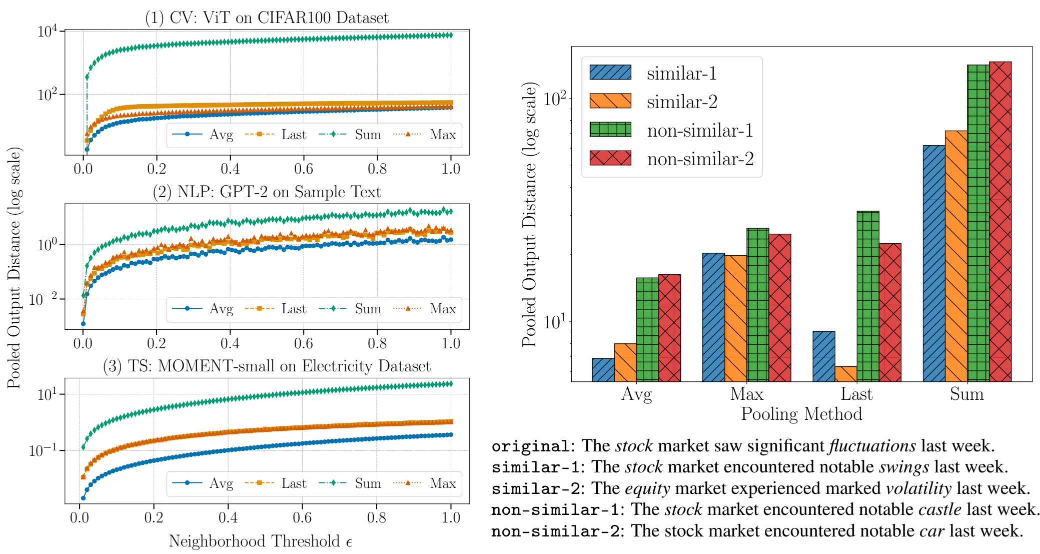


Figure 2: Empirical analysis of the expressivity power across modalities and pooling strategies. Left: Mean pooled-output distance γ versus perturbation ϵ across modalities highlighting the behavior of various methods. Right: pooled-output distances for similar and dissimilar inputs, exemplifying expressivity of different strategies.

Figure 2 (left part) presents results across different ϵ values and modalities, confirming the theoretical contrast between contractive and expansive pooling. Sum pooling shows high sensitivity to even small perturbations; as ϵ increases, its γ grows rapidly. In contrast, Average pooling remains stable. To further illustrate this, Figure 2 (right part) shows how replacing a word with either a synonym or a semantically different term in NLP settings affects the pooled output across different strategies. This supports our hypothesis that expansive pooling better captures subtle variations, as required in tasks like sentiment analysis.

5.2 Effect on the Downstream Performance

In line with our theoretical analysis, we empirically evaluate tasks with varying dependence on local versus global context to assess how different pooling strategies perform. This allows us to validate the extent to which each method's empirical behavior aligns with its expected theoretical properties.

Table 1: Mean and standard deviation of test metrics for computer vision tasks. Best performance per dataset and model is indicated in **bold**. Best performance among non-learnable pooling methods is underlined.

Model		**Classification** (Accuracy)					**Inpainting** (MSE)			**Segmentation** (Accuracy)	
	Pooling	CIFAR-10	CIFAR-100	ImageNet-100	CUB-200-2011	MiniPlaces	CelebA	OxfordFlower-102	Oxford-IIIT Pet	PascalVOC-Cls	PascalVOC-Det
ViT-small	Last (CLS)	$\underline{90.35 \pm 0.03}$	$\underline{76.42 \pm 0.08}$	$\underline{87.84 \pm 0.03}$	$\underline{\mathbf{77.45 \pm 0.13}}$	54.51 ± 0.38	0.246 ± 0.001	0.275 ± 0.001	0.268 ± 0.005	70.68 ± 0.35	29.11 ± 0.31
	Avg	90.14 ± 0.12	76.26 ± 0.33	86.85 ± 0.19	71.48 ± 0.18	$\underline{56.94 \pm 0.66}$	$\underline{0.239 \pm 0.001}$	$\underline{\mathbf{0.264 \pm 0.004}}$	$\underline{\mathbf{0.260 \pm 0.008}}$	$\underline{\mathbf{71.88 \pm 0.08}}$	$\underline{\mathbf{35.10 \pm 0.25}}$
	Sum	89.45 ± 0.32	75.72 ± 0.29	82.42 ± 0.12	68.37 ± 0.41	54.78 ± 0.71	0.285 ± 0.021	0.454 ± 0.012	0.282 ± 0.007	68.92 ± 0.97	25.21 ± 0.07
	Max	84.96 ± 0.25	69.42 ± 0.33	83.21 ± 0.23	56.81 ± 1.57	51.30 ± 0.40	0.267 ± 0.003	0.343 ± 0.008	0.275 ± 0.003	69.43 ± 1.83	21.57 ± 0.87
	W-Avg	$\mathbf{90.88 \pm 0.11}$	$\mathbf{78.41 \pm 0.02}$	$\mathbf{87.29 \pm 0.12}$	72.97 ± 0.31	$\mathbf{56.95 \pm 0.11}$	0.247 ± 0.001	0.286 ± 0.003	0.267 ± 0.007	71.83 ± 0.02	33.63 ± 1.65
	Attn	89.81 ± 0.22	74.61 ± 0.66	87.84 ± 0.06	64.78 ± 1.29	53.08 ± 1.17	$\mathbf{0.192 \pm 0.002}$	0.289 ± 0.007	0.273 ± 0.011	69.85 ± 0.84	24.22 ± 0.39
LipsFormer	Last	87.70 ± 0.03	66.53 ± 0.08	86.18 ± 0.17	48.77 ± 0.38	48.84 ± 1.03	0.225 ± 0.002	0.342 ± 0.003	0.260 ± 0.003	70.04 ± 0.01	34.22 ± 0.21
	Avg	$\underline{\mathbf{93.26 \pm 0.03}}$	$\underline{\mathbf{78.05 \pm 0.01}}$	$\underline{89.74 \pm 0.08}$	$\underline{\mathbf{72.23 \pm 0.29}}$	$\underline{65.04 \pm 0.12}$	$\underline{0.216 \pm 0.001}$	$\underline{0.306 \pm 0.005}$	$\underline{\mathbf{0.237 \pm 0.003}}$	$\underline{76.51 \pm 0.26}$	$\underline{34.59 \pm 0.34}$
	Sum	90.82 ± 0.65	72.70 ± 0.13	87.44 ± 0.28	65.96 ± 0.13	57.38 ± 0.08	0.268 ± 0.004	0.364 ± 0.088	0.308 ± 0.080	73.62 ± 0.05	24.27 ± 0.28
	Max	90.75 ± 0.35	70.73 ± 0.15	87.68 ± 0.32	59.46 ± 0.19	56.53 ± 0.18	0.234 ± 0.001	0.369 ± 0.043	0.247 ± 0.002	71.43 ± 1.62	22.64 ± 3.84
	W-Avg	93.28 ± 0.09	78.00 ± 0.12	89.48 ± 0.12	72.23 ± 0.07	$\mathbf{65.21 \pm 0.08}$	0.223 ± 0.001	0.325 ± 0.002	0.251 ± 0.002	76.28 ± 0.15	34.39 ± 0.40
	Attn	92.36 ± 0.09	76.10 ± 0.03	$\mathbf{89.92 \pm 0.16}$	67.01 ± 0.06	63.76 ± 0.05	$\mathbf{0.148 \pm 0.003}$	$\mathbf{0.287 \pm 0.004}$	0.279 ± 0.023	$\mathbf{77.37 \pm 0.64}$	$\mathbf{36.07 \pm 0.61}$

Computer Vision. Results for image-based tasks are shown in Table 1. As predicted by the analysis, Average pooling outperforms other fixed pooling methods in inpainting, segmentation, and in the MiniPlaces classification dataset, which benefits from modeling global structure. In contrast, Last-token (CLS) pooling generally yields the best results on classification tasks, particularly those involving large-scale or fine-grained datasets where local information is more critical. Max and Sum pooling consistently perform worse across all tasks. These trends also hold for alternative attention mechanisms, such as LipsFormer [32], which employs scaled cosine similarity attention (see Lemma 4.4). In this Swin-based model, which does not include a CLS token, Average pooling again achieves the best performance among fixed strategies, further emphasizing its strength in capturing localized context when a dedicated classification token is absent.

Among learnable pooling methods, Weighted Average pooling performs competitively across tasks, likely due to its capacity to adaptively weight tokens based on task-specific context. Attention-based pooling often underperforms, except in high-resource settings such as ImageNet-100, where sufficient supervision allows it to learn effective attention patterns. Its reduced reliability in low-resource or fine-grained tasks may be attributed to the additional complexity introduced by its parameterization.

NLP. The impact of pooling across downstream NLP tasks and models is summarized in Table 2. The results support our theoretical analysis: no single pooling method is optimal across all tasks, and the best strategy depends on the task's contextual requirements. For tasks requiring global context, such as classification and semantic similarity, global pooling methods like Average or Sum significantly outperform Last-token pooling. Conversely, Last-token pooling yields superior performance in next-token prediction tasks.

These trends hold across models, although the performance gap between local and global pooling narrows for larger architectures (e.g., Mistral-7B [14] and Llama [37]). In such models, Weighted Average pooling matches or exceeds the best-performing non-learnable methods, due to its ability to adaptively approximate effective pooling strategies. In smaller models, particularly the GPT-2 [33] family, Attention pooling performs well on global-context tasks, often outperforming fixed global methods like Average or Sum. However, this advantage does not consistently generalize to larger models or all task types. Additional results on larger models are provided in Table 8 (Appendix E). We additionally analyze the effect

Time Series. As shown in Table 3, for time series classification tasks, Last-token and Max pooling generally yield the worst results, as they focus on local features and fail to capture the broader context required for accurate classification. In contrast, Attention-based pooling consistently achieves the highest performance, due to its ability to assign task-specific weights to different input segments during joint training. Sum pooling outperforms Average pooling in several cases, that can be attributed to the larger norm of summed representations, which results in stronger gradients and faster learning under fixed training hyperparameters.

Forecasting shows that Last-token pooling yields the best results overall. This is consistent with its ability to retain fine-grained temporal details, which are critical for predicting future values based on recent history. In imputation tasks, results indicate that Attention-based pooling again performs best, while Sum pooling performs the worst. This supports the hypothesis that given sufficient data and training time, Attention pooling can adaptively focus on the most relevant parts of the sequence,

Table 2: Mean and standard deviation of test metrics for NLP tasks. Best performance per dataset and model is indicated in **bold**. Best performance among non-learnable pooling methods is underlined. (-) indicates non-applicable, as the model uses bidirectional attention mechanism.

	Dataset	Pooling	BERT	L2-GPT-2	GPT-2	Qwen 2.5	Mistral-7B	Llama3-8B
Similarity	STSB (Spearman)	Last	0.587 ± 0.009	0.375 ± 0.006	0.602 ± 0.005	0.286 ± 0.005	0.514 ± 0.001	0.017 ± 0.085
		Avg	0.713 ± 0.008	0.659 ± 0.004	$\underline{0.671 \pm 0.004}$	$\underline{0.620 \pm 0.005}$	$\underline{0.635 \pm 0.005}$	0.624 ± 0.004
		Sum	$\underline{0.714 \pm 0.009}$	$\underline{0.660 \pm 0.004}$	0.670 ± 0.003	0.619 ± 0.005	0.634 ± 0.007	$\underline{0.626 \pm 0.005}$
		Max	0.695 ± 0.013	0.648 ± 0.002	0.653 ± 0.002	0.560 ± 0.011	0.449 ± 0.017	0.487 ± 0.003
		W-Avg	$\mathbf{0.727 \pm 0.002}$	0.562 ± 0.002	0.568 ± 0.003	$\mathbf{0.671 \pm 0.002}$	$\mathbf{0.653 \pm 0.004}$	$\mathbf{0.673 \pm 0.001}$
		Attn	0.703 ± 0.013	$\mathbf{0.678 \pm 0.016}$	$\mathbf{0.677 \pm 0.010}$	0.616 ± 0.014	0.452 ± 0.088	0.496 ± 0.037
	Hellaswag (F1)	Last	0.307 ± 0.001	0.231 ± 0.025	0.264 ± 0.024	0.344 ± 0.001	$\underline{0.770 \pm 0.002}$	0.678 ± 0.002
		Avg	0.315 ± 0.000	$\underline{\mathbf{0.297 \pm 0.001}}$	$\underline{\mathbf{0.305 \pm 0.000}}$	$\underline{0.432 \pm 0.000}$	0.769 ± 0.000	$\underline{0.734 \pm 0.000}$
		Sum	$\underline{0.316 \pm 0.001}$	$\underline{\mathbf{0.297 \pm 0.001}}$	$\underline{\mathbf{0.305 \pm 0.005}}$	0.431 ± 0.001	0.769 ± 0.000	$\underline{0.734 \pm 0.001}$
		Max	0.298 ± 0.003	0.291 ± 0.002	0.293 ± 0.002	0.364 ± 0.003	0.709 ± 0.001	0.682 ± 0.005
		W-Avg	$\mathbf{0.318 \pm 0.001}$	0.284 ± 0.001	0.278 ± 0.001	$\mathbf{0.452 \pm 0.000}$	$\mathbf{0.801 \pm 0.000}$	$\mathbf{0.763 \pm 0.000}$
		Attn	0.295 ± 0.004	0.264 ± 0.012	0.260 ± 0.038	0.410 ± 0.009	0.737 ± 0.018	0.459 ± 0.271
Classification	Banking (Accuracy)	Last	77.175 ± 0.112	17.014 ± 0.323	45.528 ± 0.244	23.243 ± 0.378	74.847 ± 1.584	45.107 ± 0.403
		Avg	$\underline{85.142 \pm 0.002}$	$\underline{86.882 \pm 0.076}$	$\underline{86.497 \pm 0.103}$	$\underline{83.486 \pm 0.402}$	$\underline{88.183 \pm 0.402}$	$\underline{87.558 \pm 0.323}$
		Sum	83.863 ± 0.806	84.130 ± 0.807	83.724 ± 1.047	79.164 ± 0.776	86.442 ± 0.703	82.984 ± 1.107
		Max	80.785 ± 0.008	83.091 ± 0.315	83.023 ± 0.226	74.007 ± 1.106	74.890 ± 2.476	75.324 ± 1.602
		W-Avg	84.987 ± 0.153	67.253 ± 0.275	73.989 ± 0.223	$\mathbf{85.513 \pm 0.221}$	$\mathbf{89.271 \pm 0.426}$	$\mathbf{88.928 \pm 0.296}$
		Attn	$\mathbf{86.558 \pm 0.559}$	$\mathbf{87.340 \pm 0.302}$	$\mathbf{86.968 \pm 0.604}$	83.792 ± 1.813	73.352 ± 1.731	51.143 ± 34.316
	Tweet (Accuracy)	Last	67.621 ± 0.083	48.383 ± 0.204	63.899 ± 0.143	51.738 ± 0.713	56.693 ± 0.492	60.446 ± 0.538
		Avg	$\underline{69.348 \pm 0.128}$	$\underline{67.593 \pm 0.103}$	$\underline{68.573 \pm 0.208}$	$\underline{68.961 \pm 0.330}$	$\underline{67.231 \pm 0.218}$	$\underline{67.328 \pm 0.223}$
		Sum	65.381 ± 0.273	63.149 ± 0.195	64.122 ± 0.317	59.625 ± 2.702	64.151 ± 1.623	64.231 ± 2.029
		Max	67.070 ± 0.177	61.392 ± 0.254	61.262 ± 0.178	62.971 ± 0.433	64.897 ± 2.402	64.804 ± 1.089
		W-Avg	$\mathbf{69.560 \pm 0.121}$	62.458 ± 0.123	60.718 ± 0.224	66.031 ± 0.403	$\mathbf{67.293 \pm 0.228}$	$\mathbf{67.476 \pm 0.185}$
		Attn	69.455 ± 0.879	$\mathbf{69.131 \pm 0.529}$	$\mathbf{70.844 \pm 0.821}$	$\mathbf{70.627 \pm 0.663}$	46.584 ± 6.903	55.890 ± 13.232
Next Token	Tiny Stories (Top-10 Acc)	Last	—	$\underline{\mathbf{82.170 \pm 0.225}}$	$\underline{\mathbf{84.569 \pm 0.001}}$	86.634 ± 0.428	$\underline{\mathbf{89.948 \pm 0.092}}$	$\underline{\mathbf{90.608 \pm 0.227}}$
		Avg	—	37.718 ± 0.229	38.826 ± 0.435	40.654 ± 0.327	61.047 ± 0.228	56.012 ± 0.337
		Sum	—	24.852 ± 0.428	29.362 ± 0.893	28.911 ± 0.630	50.481 ± 0.625	42.731 ± 1.318
		Max	—	35.450 ± 0.332	36.022 ± 0.257	37.229 ± 0.253	9.900 ± 0.108	32.199 ± 0.252
		W-Avg	—	61.310 ± 0.018	53.998 ± 0.348	$\mathbf{86.859 \pm 0.212}$	89.160 ± 0.118	87.389 ± 0.019
		Attn	—	50.364 ± 0.287	53.299 ± 0.338	55.970 ± 0.302	14.210 ± 5.239	11.138 ± 8.959

Table 3: Mean and standard deviation test metrics in time series tasks. Best performance per dataset and model in **bold**. Best performance among non-learnable pooling methods is underlined.

		Classification (Accuracy)				**Forecasting** (MSE)			**Imputation** (MSE)		
Model	**Pooling**	ECG200	Electric Devices	FordA	SmallKitchen Appliances	ETTh1	Electricity	Traffic	ETTh1	Electricity	Traffic
MOMENT-small	Last	72.29 ± 0.59	60.45 ± 0.48	76.39 ± 0.15	64.06 ± 0.75	$\underline{\mathbf{0.082 \pm 0.000}}$	$\underline{\mathbf{0.400 \pm 0.001}}$	$\underline{0.273 \pm 0.001}$	0.081 ± 0.002	0.753 ± 0.010	1.709 ± 0.016
	Avg	65.19 ± 0.00	61.40 ± 0.54	88.12 ± 0.14	62.40 ± 1.01	0.105 ± 0.000	0.790 ± 0.000	1.724 ± 0.001	$\underline{0.080 \pm 0.002}$	0.774 ± 0.013	1.583 ± 0.016
	Sum	$\underline{\mathbf{80.35 \pm 1.82}}$	60.62 ± 2.85	$\underline{92.93 \pm 0.43}$	$\underline{67.13 \pm 2.95}$	0.103 ± 0.001	0.826 ± 0.014	1.422 ± 0.050	0.106 ± 0.008	1.072 ± 0.143	2.130 ± 0.238
	Max	65.19 ± 0.00	$\underline{61.78 \pm 1.24}$	90.42 ± 0.43	63.94 ± 1.76	0.106 ± 0.001	0.800 ± 0.001	1.759 ± 0.002	0.082 ± 0.002	$\underline{0.720 \pm 0.013}$	$\underline{1.211 \pm 0.051}$
	W-Avg	65.19 ± 0.00	62.54 ± 0.67	87.62 ± 0.64	62.40 ± 1.01	0.105 ± 0.000	0.539 ± 0.008	0.954 ± 0.014	0.080 ± 0.002	0.774 ± 0.013	1.582 ± 0.016
	Attn	78.84 ± 3.04	$\mathbf{62.95 \pm 2.04}$	$\mathbf{93.60 \pm 0.37}$	$\mathbf{67.63 \pm 2.95}$	0.106 ± 0.001	0.475 ± 0.059	$\mathbf{0.258 \pm 0.002}$	$\mathbf{0.076 \pm 0.003}$	$\mathbf{0.379 \pm 0.028}$	$\mathbf{0.265 \pm 0.015}$
MOMENT-base	Last	71.01 ± 1.94	63.05 ± 0.62	83.46 ± 0.40	63.25 ± 0.54	$\underline{\mathbf{0.081 \pm 0.000}}$	$\underline{\mathbf{0.397 \pm 0.000}}$	$\underline{\mathbf{0.265 \pm 0.000}}$	0.082 ± 0.001	0.760 ± 0.011	1.668 ± 0.020
	Avg	65.19 ± 0.00	63.71 ± 0.40	89.75 ± 0.37	63.05 ± 1.60	0.105 ± 0.000	0.785 ± 0.001	1.719 ± 0.001	$\underline{0.082 \pm 0.001}$	0.769 ± 0.016	1.424 ± 0.020
	Sum	$\underline{\mathbf{83.30 \pm 1.51}}$	$\underline{64.30 \pm 0.98}$	$\underline{92.51 \pm 0.13}$	$\underline{66.55 \pm 1.48}$	0.101 ± 0.002	0.805 ± 0.009	1.029 ± 0.023	0.103 ± 0.002	1.006 ± 0.161	1.301 ± 0.089
	Max	65.78 ± 1.32	62.31 ± 1.14	90.23 ± 0.80	64.77 ± 2.12	0.106 ± 0.000	0.796 ± 0.001	1.747 ± 0.003	0.082 ± 0.002	$\underline{0.707 \pm 0.014}$	$\underline{1.000 \pm 0.113}$
	W-Avg	65.19 ± 0.00	$\mathbf{64.48 \pm 0.69}$	89.93 ± 0.36	63.20 ± 1.74	0.105 ± 0.000	0.511 ± 0.006	0.857 ± 0.014	0.082 ± 0.001	0.769 ± 0.015	1.425 ± 0.020
	Attn	82.57 ± 1.19	64.15 ± 1.37	$\mathbf{92.74 \pm 0.35}$	$\mathbf{66.76 \pm 1.39}$	0.096 ± 0.009	0.507 ± 0.148	0.306 ± 0.037	$\mathbf{0.072 \pm 0.003}$	$\mathbf{0.370 \pm 0.013}$	$\mathbf{0.273 \pm 0.004}$
MOMENT-large	Last	72.67 ± 0.95	61.10 ± 0.53	79.61 ± 0.31	$\underline{65.45 \pm 1.71}$	$\underline{\mathbf{0.080 \pm 0.000}}$	$\underline{\mathbf{0.379 \pm 0.000}}$	$\underline{\mathbf{0.272 \pm 0.001}}$	0.082 ± 0.001	0.752 ± 0.014	1.699 ± 0.017
	Avg	65.19 ± 0.00	61.72 ± 0.93	85.98 ± 0.53	60.42 ± 0.91	0.105 ± 0.000	0.778 ± 0.000	1.711 ± 0.001	$\underline{0.081 \pm 0.002}$	0.753 ± 0.018	1.508 ± 0.011
	Sum	$\underline{75.97 \pm 2.82}$	$\underline{\mathbf{63.45 \pm 0.76}}$	$\underline{92.85 \pm 0.42}$	64.92 ± 7.21	0.101 ± 0.001	0.688 ± 0.007	0.782 ± 0.003	0.095 ± 0.008	0.887 ± 0.075	$\underline{1.150 \pm 0.065}$
	Max	65.19 ± 0.00	60.08 ± 0.54	87.14 ± 0.39	62.27 ± 0.64	0.104 ± 0.001	0.785 ± 0.001	1.743 ± 0.000	0.081 ± 0.001	$\underline{0.705 \pm 0.008}$	1.158 ± 0.025
	W-Avg	65.19 ± 0.00	61.48 ± 0.90	86.07 ± 0.36	60.54 ± 0.53	0.105 ± 0.001	0.534 ± 0.003	0.951 ± 0.005	0.081 ± 0.002	0.753 ± 0.018	1.503 ± 0.011
	Attn	$\mathbf{78.73 \pm 3.09}$	62.83 ± 1.38	$\mathbf{93.02 \pm 0.26}$	$\mathbf{65.64 \pm 3.54}$	0.097 ± 0.004	0.684 ± 0.003	0.423 ± 0.028	$\mathbf{0.072 \pm 0.003}$	$\mathbf{0.295 \pm 0.006}$	$\mathbf{0.231 \pm 0.009}$

whereas Sum pooling may amplify irrelevant noise and obscure important local patterns required for accurate imputation. Additional results for all datasets are provided in Appendix E.

5.3 Positional Weighting in Weighted Average Pooling

We investigate the role of learnable weights in the Weighted Average pooling layer through an ablation-style comparison, while keeping all other model components and training settings constant.

Figure 3 depicts results for the Mistral [14] backbone on representative tasks. In each case, the trainable variant converges to weight distributions that closely resemble the expected optimal pooling strategy for each task: near-uniform weights on text classification, a strong focus on the final token for next-token prediction, and intermediate patterns for tasks that require both local detail and global context. Consistent with our theoretical analysis, it shows that the benefits of Weighted Average

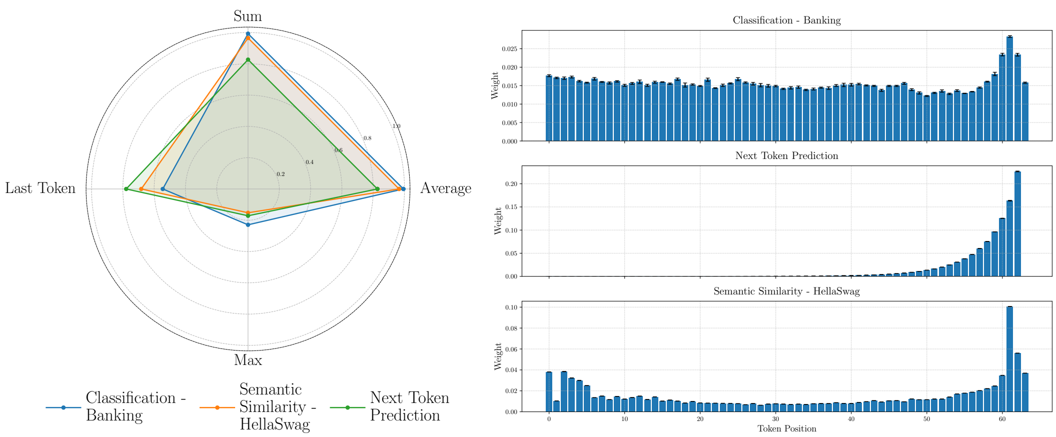


Figure 3: **Left:** Cosine similarity between W-Avg pooling and other pooling methods, showing task-dependent alignment. **Right:** The distribution of the learned weights in the W-avg pooling, illustrating the adaptability of the pooling mechanism.

pooling arise from its ability to mimic the best-performing fixed pooling strategy which depends on the task's contextual demands. Full results for all models and datasets are provided in Appendix E.

6 Conclusion

We presented an end-to-end study of pooling in Transformer models by introducing a formal expressivity framework and deriving closed-form bounds for standard pooling mechanisms. Our theory extends to alternative attention variants and shows that pooling is the key factor balancing local detail and global aggregation. Extensive experiments in vision, language, and time-series tasks validate these bounds: contractive pooling excels on global-context tasks, expansive pooling captures fine-grained distinctions, and learnable methods converge to the best-balanced strategy when given sufficient training data and time. No single pooling method dominates universally, highlighting the need for task-specific pooling design. In data-scarce regimes, our guidelines enable principled selection of pooling methods based on task demands and inductive biases. These contributions bridge theory and practice, enhance understanding of Transformer expressivity, and inform the design of adaptive pooling schemes for diverse downstream applications.

Limitations and future work. While our theoretical and empirical findings provide a solid foundation for understanding pooling in Transformer architectures, several limitations remain. Our evaluation, though broad, is limited to frozen backbones, leaving the effect of jointly adapting pooling and the backbone under end-to-end training less explored. The expressivity bounds we establish are necessary but not sufficient for optimal performance; bridging the gap between theoretical capacity and practical effectiveness remains an open challenge. In future work, we aim to develop hybrid pooling methods that dynamically balance global smoothing with token-level sensitivity while adapting to the TBM's inherent smoothing behavior. In addition to the analysis provided in Appendix E.3, we also plan to investigate further how pooling impacts robustness under perturbations, derive scaling laws that govern pooling performance as datasets grow, and refine our theoretical framework with tighter bounds that identify when specific strategies are provably optimal.

Acknowledgments and Disclosure of Funding

This work was partially supported by Wallenberg Autonomous Systems Program (WASP). L.Z. gratefully acknowledges NXAI GmbH for supporting his participation in NeurIPS 2025.

References

[1] Muhammad Awais, Muzammal Naseer, Salman Khan, Rao Muhammad Anwer, Hisham Cholakkal, Mubarak Shah, Ming-Hsuan Yang, and Fahad Shahbaz Khan. Foundation models

defining a new era in vision: a survey and outlook. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[3] Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, page 1. Association for Computational Linguistics, 2017.

[4] George Dasoulas, Kevin Scaman, and Aladin Virmaux. Lipschitz normalization for self-attention layers with application to graph neural networks. In *International Conference on Machine Learning*, pages 2456–2466. PMLR, 2021.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[7] Ronen Eldan and Yuanzhi Li. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*, 2023.

[8] Sofiane ENNADIR, Johannes F. Lutzeyer, Michalis Vazirgiannis, and El houcine Bergou. If you want to be robust, be wary of initialization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[9] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111:98–136, 2015.

[10] Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. In *Conference on Empirical Methods in Natural Language Processing*, pages 457–468. ACL, 2016.

[11] Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`, 2019.

[12] Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models. *arXiv preprint arXiv:2402.03885*, 2024.

[13] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[14] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

[15] Hyunjik Kim, George Papamakarios, and Andriy Mnih. The Lipschitz constant of self-attention. In *International Conference on Machine Learning*, pages 5562–5571. PMLR, 2021.

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[17] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[18] Itay Lavie, Guy Gur-Ari, and Zohar Ringel. Towards understanding inductive bias in transformers: A view from infinity. In *Forty-first International Conference on Machine Learning*, 2024.

[19] Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. NV-embed: Improved techniques for training LLMs as generalist embedding models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[20] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *ArXiv e-prints*, pages arXiv–1607, 2016.

[21] Pengxiang Li, Lu Yin, and Shiwei Liu. Mix-ln: Unleashing the power of deeper layers by combining pre-ln and post-ln. In *International Conference on Learning Representations*, 2025.

[22] Yuxuan Liang, Haomin Wen, Yuqi Nie, Yushan Jiang, Ming Jin, Dongjin Song, Shirui Pan, and Qingsong Wen. Foundation models for time series analysis: A tutorial and survey. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pages 6555–6565, 2024.

[23] Chuang Liu, Yibing Zhan, Jia Wu, Chang Li, Bo Du, Wenbin Hu, Tongliang Liu, and Dacheng Tao. Graph pooling for graph neural networks: progress, challenges, and opportunities. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, IJCAI '23, 2023.

[24] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[25] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, 15(2018):11, 2018.

[26] Christopher Morris, Gaurav Rattan, and Petra Mutzel. Weisfeiler and leman go sparse: Towards scalable higher-order graph embeddings. *Advances in Neural Information Processing Systems*, 33:21824–21840, 2020.

[27] Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 4602–4609, 2019.

[28] Muhammad Muzammal Naseer, Kanchana Ranasinghe, Salman H Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Intriguing properties of vision transformers. *Advances in Neural Information Processing Systems*, 34:23296–23308, 2021.

[29] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pages 722–729. IEEE, 2008.

[30] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012.

[31] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[32] Xianbiao Qi, Jianan Wang, Yihao Chen, Yukai Shi, and Lei Zhang. Lipsformer: Introducing Lipschitz continuity to vision transformers. In *The Eleventh International Conference on Learning Representations*, 2023.

[33] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[34] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[35] Yixuan Tang and Yi Yang. Pooling and attention: What are effective designs for llm-based embedding models? *arXiv preprint arXiv:2409.02727*, 2024.

[36] Yuandong Tian, Yiping Wang, Beidi Chen, and Simon S Du. Scan and snap: Understanding training dynamics and token composition in 1-layer transformer. *Advances in neural information processing systems*, 36:71911–71947, 2023.

[37] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[38] Hayato Tsukagoshi, Ryohei Sasano, and Koichi Takeda. Defsent: Sentence embeddings using definition sentences. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 411–418, 2021.

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[40] Aladin Virmaux and Kevin Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[41] Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pages 35151–35174. PMLR, 2023.

[42] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

[43] Jinming Xing, Dongwen Luo, Chang Xue, and Ruilin Xing. Comparative analysis of pooling mechanisms in llms: A sentiment analysis perspective. *arXiv preprint arXiv:2411.14654*, 2024.

[44] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

[45] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[46] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, 2019.

[47] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017.

Supplementary Material:

A Proof of Theorem 4.2

Theorem. *Let $f\colon \mathcal{X} \subseteq \mathbb{R}^{n\times d} \to \mathcal{Y} \subseteq \mathbb{R}^d$ be a TBM following the framework introduced in Section 3. In respect to Definition 4.1, we have:*

- *If f employs Average pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon}{\sigma\sqrt{n}}\left(\frac{d}{d-1}\right)^2 C_1 C_2$*
- *If f employs Sum pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\sqrt{n}\,\epsilon}{\sigma}\left(\frac{d}{d-1}\right)^2 C_1 C_2$*
- *If f employs Last-token pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon}{\sigma}\left(\frac{d}{d-1}\right)^2 C_1 C_2$*
- *If f employs Max pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon\sqrt{\min(n,d)}}{\sigma}\left(\frac{d}{d-1}\right)^2 C_1 C_2$,*

where

$$C_1 = 1 + \|W_O\|\sqrt{H}\,\max_h\left[\left\|W^{V,h}\right\|\left(4\,\frac{n}{\sqrt{d/H}}\,B^2\left\|W^{Q,h}\right\|\left\|W^{K,h}\right\| + 1\right)\right], \quad C_2 = 1 + \left\|W_{FFN}\right\|.$$

Proof. Let the input $X \in \mathcal{X}$ consist of n tokens $x_i \in \mathbb{R}^d$. We consider a Transformer model f using scaled dot-product attention as defined in Equation 1, and formulated as:

$$\begin{aligned}\mathrm{AH}(X) &= \mathrm{softmax}\left(\frac{(XW^Q)(XW^K)^\top}{\sqrt{D/H}}\right)(XW^V)\\ &= PXW^V = h(X)W^V,\end{aligned}$$

where W^Q, W^K, W^V are learnable projection matrices. The attention matrix P is computed from the softmax of pairwise scores:

$$f(X) = PX = \mathrm{softmax}(XA^\top X^\top)X,$$

with

$$A = \frac{W^K W^{Q^\top}}{\sqrt{d/H}} \in \mathbb{R}^{d\times d}.$$

Each row of the output $f(X)$ can be expressed as:

$$f(X) = \begin{bmatrix} h_1(X)^\top \\ \vdots \\ h_n(X)^\top \end{bmatrix} \in \mathbb{R}^{n\times d}, \quad \text{with} \quad h_i(X) = \sum_{j=1}^n P_{ij}x_j,$$

where $P_i^\top = \mathrm{softmax}(XAx_i)$.

To analyze the Jacobian of h, we derive its partial derivatives:

$$J_{ij} = X^\top P^{(i)} E_{ji} X A^\top + \delta_{ij}\left(X^\top P^{(i)} X A\right) + P_{ij} I_d,$$

where:

- $P^{(i)} = \mathrm{diag}(P_{i:}) - P_{i:}^\top P_{i:}$ is the Jacobian of the softmax,
- E_{ji} is an $n \times n$ matrix with a 1 at position (j, i) and zeros elsewhere.

From this, we have:

$$\text{If } i \neq j: \quad J_{ij} = X^\top P^{(i)} E_{ji} X A^\top + P_{ij} I_d, \tag{5}$$

$$\text{If } i = j: \quad J_{ii} = X^\top P^{(i)} E_{ii} X A^\top + X^\top P^{(i)} X A + P_{ii} I_d. \tag{6}$$

Assuming the input space is bounded, i.e., $\|x_i\|_2 \leq B$ for all i, we get:

$$\|X\|_F^2 = \sum_i \|x_i\|_2^2 \leq nB^2 \quad \Rightarrow \quad \|X\| \leq \|X\|_F \leq \sqrt{n}B.$$

Since $P_{i:}$ is a probability distribution and $\sigma_{max}(diag(p)) \leq 1$, we have $\|P^{(i)}\| \leq 2$.

Case 1: $i \neq j$

$$\begin{aligned}\|J_{ij}\| &\leq \|X^\top P^{(i)} E_{ji} X A^\top\| + \|P_{ij} I_d\| \\ &\leq \|X\|^2 \|P^{(i)}\| \|A\| + 1 \leq 2nB^2 \|A\| + 1.\end{aligned}$$

Case 2: $i = j$

$$\begin{aligned}\|J_{ii}\| &\leq \|X^\top P^{(i)} E_{ii} X A^\top\| + \|X^\top P^{(i)} X A\| + \|P_{ii} I_d\| \\ &\leq 2nB^2 \|A\| + 2nB^2 \|A\| + 1 = 4nB^2 \|A\| + 1.\end{aligned}$$

Thus, the Jacobian of h is bounded:

$$\|J_{ij}\|_{op} \leq \begin{cases} 2nB^2 \|A\| + 1, & \text{if } i \neq j, \\ 4nB^2 \|A\| + 1, & \text{if } i = j. \end{cases}$$

Therefore, the function h is bounded with constant:

$$\mathcal{L}_h \leq 4nB^2 \|A\| + 1.$$

Attention Head Bound. Including the value projection:

$$\mathcal{L}_{\text{head}} \leq \|W^{V,h}\| \left[4 \frac{n}{\sqrt{d/H}} B^2 \|W^{Q,h}\| \|W^{K,h}\| + 1 \right].$$

Multi-Head Attention Bound. Since f is represented by H attention head, their concatenated output as explained in Equation 2 satisfies:

$$\begin{aligned}\mathcal{L}_{\text{MH}} &\leq \|W_O\| \sqrt{H} \max_h \mathcal{L}_{\text{head}} \\ &\leq \|W_O\| \sqrt{H} \max_h \left\{ \|W^{V,h}\| \left[4 \frac{n}{\sqrt{d/H}} B^2 \|W^{Q,h}\| \|W^{K,h}\| + 1 \right] \right\}.\end{aligned}$$

Full Transformer Block. Incorporating FFN and layer norm (with $\gamma = \beta = 1$):

$$\begin{aligned}\mathcal{L}_f &\leq L_{LN}^2 (1 + \mathcal{L}_{\text{MH}})(1 + \|W_{\text{FFN}}\|) \\ &\leq \left(\frac{d}{d-1} \right)^2 (1 + \mathcal{L}_{\text{MH}})(1 + \|W_{\text{FFN}}\|) \\ &\leq \left(\frac{d}{d-1} \right)^2 C_1 C_2,\end{aligned}$$

where:

$$\begin{aligned}C_1 &= 1 + \|W_O\| \sqrt{H} \max_h \left\{ \|W^{V,h}\| \left[4 \frac{n}{\sqrt{d/H}} B^2 \|W^{Q,h}\| \|W^{K,h}\| + 1 \right] \right\}, \\ C_2 &= 1 + \|W_{\text{FFN}}\|.\end{aligned}$$

Impact of Pooling Strategies

Given a final representation $z = g(f(X))$ using pooling function g, we evaluate its effect on the bound.

Average pooling. We recall that this pooling method can be written as a linear layer with weights W_{Avg}, therefore:

$$W_{\text{Avg}} = \frac{1}{n}\mathbf{1}_n \quad \Rightarrow \quad \|W_{\text{Avg}}\| = \frac{1}{\sqrt{n}}.$$

$$\|f(X) - f(\tilde{X})\| \leq \frac{1}{\sqrt{n}} \left(\frac{d}{d-1}\right)^2 C_1 C_2 \epsilon.$$

Sum pooling. Similarly to Average pooling:

$$W_{\text{Sum}} = \mathbf{1}_n \quad \Rightarrow \quad \|W_{\text{Sum}}\| = \sqrt{n}.$$

$$\|f(X) - f(\tilde{X})\| \leq \sqrt{n} \left(\frac{d}{d-1}\right)^2 C_1 C_2 \epsilon.$$

Last-Token pooling. Considering the last token as the output as the pooling operation:

$$W_{\text{Last}} = [0, \dots, 0, 1]^\top \quad \Rightarrow \quad \|W_{\text{Last}}\| = 1.$$

$$\|f(X) - f(\tilde{X})\| \leq \left(\frac{d}{d-1}\right)^2 C_1 C_2 \epsilon.$$

Note that the same treatment can be applied to CLS or any other chosen token.

Max pooling. Using norm bounds:

$$\begin{aligned}
\|f(X) - f(\tilde{X})\|^2 &= \sum_{j=1}^{d} \mid (f(X))_j - (f(\tilde{X}))_j \mid^2 \\
&\leq \sum_{j=1}^{d} \max_i \mid X_{i,j} - \tilde{X}_{i,j} \mid^2 \\
&= \|X - \tilde{X}\|_F^2
\end{aligned}$$

For spectral norm:

$$\begin{aligned}
\|f(X) - f(\tilde{X})\| &\leq \|X - \tilde{X}\|_F \\
&\leq \sqrt{\text{rank}(X - \tilde{X})}\|X - \tilde{X}\| \\
&\leq \sqrt{\min(n, d)}\|X - \tilde{X}\|
\end{aligned}$$

From those two results, we have:

$$\begin{aligned}
\|f(X) - f(\tilde{X})\| &\leq \sqrt{\min(n, d)}\|X - \tilde{X}\| \\
&\leq \sqrt{\min(n, d)}\big(\frac{d}{d-1}\big)^2 C_1 C_2 \times \epsilon
\end{aligned}$$

Applying the Markov inequality concludes the proof.

$\square$

B Proof of Lemma 4.3

Lemma. *Let $f\colon \mathcal{X} \subseteq \mathbb{R}^{n\times d} \to \mathcal{Y} \subseteq \mathbb{R}^d$ be a L2-MHA-based TBM [15]. In respect to Definition 4.1, the following holds:*

- *If f employs Average pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon}{\sigma\sqrt{n}}\left(\frac{d}{d-1}\right)^2 C_1C_2$*
- *If f employs Sum pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\sqrt{n}\,\epsilon}{\sigma}\left(\frac{d}{d-1}\right)^2 C_1C_2$*
- *If f employs Last-token pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon}{\sigma}\left(\frac{d}{d-1}\right)^2 C_1C_2$*
- *If f employs Max pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon\sqrt{\min(n,d)}}{\sigma}\left(\frac{d}{d-1}\right)^2 C_1C_2$,*

where

$$C_1 = 1 + \frac{\sqrt{n}}{\sqrt{d/H}}\left(4W_O\left(\frac{n}{e}\right)+1\right)\left(\sqrt{\sum_h \|W^{Q,h}\|^2\|W^{V,h}\|^2}\right)\|W^O\|, \quad C_2 = 1 + \|W_{FFN}\|.$$

Proof. Let the input $X \in \mathcal{X}$ be composed of n tokens $x_i \in \mathbb{R}^d$. In this proof, we consider the Transformer-based model f built using the L2 Multi-Head Attention (L2-MHA) mechanism, where the attention weights are computed as:

$$P_{ij} \propto \exp\left(-\frac{\|\mathbf{x}_i W^Q - \mathbf{x}_j W^K\|^2}{\sqrt{d/H}}\right),$$

with W^Q, W^K being learnable projections.

From Theorem 3.2 of [15], the L2-MHA operator is bounded by:

$$\mathrm{Lip}_2(F) \le \frac{\sqrt{n}}{\sqrt{d/H}}\left(4W_0\left(\frac{n}{e}\right)+1\right)\left(\sqrt{\sum_h \|W^{Q,h}\|^2\|W^{V,h}\|^2}\right)\|W^O\|,$$

where $W_0(\cdot)$ denotes the Lambert W-function.

Following the Transformer architecture defined in Section 3, we account for the additional effects of LayerNorm (LN) and the Feed-Forward Network (FFN). As in previous derivations, we obtain:

$$\mathcal{L}_f \le \mathcal{L}_{\text{LN}}^2(1+\mathcal{L}_{\text{MHA}})(1+\mathcal{L}_{\text{FFN}}),$$

where we now substitute the bound for L2-MHA:

$$\begin{aligned}\mathcal{L}_f &\le \left(\frac{d}{d-1}\right)^2\left(1 + \frac{\sqrt{n}}{\sqrt{d/H}}\left(4W_0\left(\frac{n}{e}\right)+1\right)\left(\sqrt{\sum_h \|W^{Q,h}\|^2\|W^{V,h}\|^2}\right)\|W^O\|\right)(1+\|W_{\text{FFN}}\|)\\ &\le \left(\frac{d}{d-1}\right)^2 C_1C_2,\end{aligned}$$

with constants defined as:

$$\begin{aligned}C_1 &= 1 + \frac{\sqrt{n}}{\sqrt{d/H}}\left(4W_0\left(\frac{n}{e}\right)+1\right)\left(\sqrt{\sum_h \|W^{Q,h}\|^2\|W^{V,h}\|^2}\right)\|W^O\|,\\ C_2 &= 1 + \|W_{\text{FFN}}\|.\end{aligned}$$

Following the same steps as in the Theorem 4.2 proof, we get the following:

For Average pooling:

$$\mathcal{L}_{\text{Avg}} \le \frac{1}{\sqrt{n}} \times \left(\frac{d}{d-1}\right)^2 C_1C_2.$$

For Sum pooling:

$$\mathcal{L}_{\text{Sum}} \leq \sqrt{n} \times \left(\frac{d}{d-1}\right)^2 C_1 C_2.$$

For Last-token pooling:

$$\mathcal{L}_{\text{Last}} \leq \left(\frac{d}{d-1}\right)^2 C_1 C_2.$$

For Max pooling:

$$\mathcal{L}_{\text{Max}} \leq \sqrt{\min(n,d)} \left(\frac{d}{d-1}\right)^2 C_1 C_2.$$

□

C Proof of Lemma 4.4

Lemma. *Let $f \colon \mathcal{X} \to \mathcal{Y}$ to be a function based on the LipsFormer [32] framework, with corresponding hyper-parameters $\nabla, \nu, \tau > 0$ and window size w. In respect to Definition 4.1, we have:*

- *If f is based on Average pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon}{\sigma \times \sqrt{n}} \times \left(\frac{d}{d-1}\right)^2 C_1 C_2$*
- *If f is based on Sum pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\sqrt{n} \times \epsilon}{\sigma} \times \left(\frac{d}{d-1}\right)^2 C_1 C_2$*
- *If f is based on Last-token pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon}{\sigma} \times \left(\frac{d}{d-1}\right)^2 C_1 C_2$*
- *If f is based on Max pooling, then f is $(\epsilon, \sigma, \gamma)$-expressive with $\gamma = \frac{\epsilon\sqrt{\min(n,d)}}{\sigma} \times \left(\frac{d}{d-1}\right)^2 C_1 C_2$,*

where

$C_1 = 1 + \|W_O\| \sqrt{H} \max_h \Big\{ 2w(w-1)\nu\tau\nabla^{-\frac{1}{2}} \|W_h^K\| + 2(w-1)\nu\tau\nabla^{-\frac{1}{2}} \|W_h^Q\| + 2w\nu\nabla^{-\frac{1}{2}} \|W_h^V\| \Big\}$,
$C_2 = \big(1 + \|W_{FFN}\|\big)$

Proof. Before delving in the specific analysis of the Lipsformodel model, we start the proof by providing some preliminary elements about the Swin Transformer [24] which is different from the original Transformer-based Model defined in Section 3.

A Swin Transformer block's input is similar to the one from a TBM, specifically, the input $X \in \mathbb{R}^{n \times d}$, can be viewed as n tokens (or patches) each of dimension d. Rather than applying global self-attention to all n tokens, the model partitions X into small "local windows" of size w, thereby reducing complexity. Between successive Swin stages, there is a "patch merging" step, which consists of a linear downsampling that reduces the number of tokens while increasing their dimension.

Let $\mathcal{W}$ denote the total number of windows, $X_\ell \in \mathbb{R}^{w \times d}$ be the slice of input corresponding to window ℓ and W^Q, W^K, W^V are the query/key/value projection matrices, within each block, a window-based self-attention is computed as follows:

$$\text{LocalAttn}(X) = \bigoplus_{\ell=1}^{\mathcal{W}} \text{softmax}\Big(\tfrac{(X_\ell W^Q)(X_\ell W^K)^\top}{\sqrt{d/H}}\Big)(X_\ell W^V)$$

In our analysis, we rather focus on the LipsFormer [32] model, which is an adaptation of the previous equation. Specifically, the model modifies the Swin Transformer architecture by replacing the standard dot-product attention with a *scaled cosine similarity attention* mechanism. Given an input $X \in \mathbb{R}^{w \times d}$ (e.g., the tokens in a window), define the following:

$$q_i = \frac{X_i W^Q}{\|X_i W^Q\|_2 + \nabla}, \quad k_j = \frac{X_j W^K}{\|X_j W^K\|_2 + \nabla}, \quad v_j = \frac{X_j W^V}{\|X_j W^V\|_2 + \nabla},$$

where $X_i \in \mathbb{R}^d$ is the i-th token row of X, and $\nabla > 0$ is a small constant to avoid division by 0. Then accordingly write the usual attention matrices as:

$$Q = \big[q_1^\top, \ldots, q_w^\top\big], \quad K = \big[\,k_1^\top, \ldots;\ k_w^\top\big], \quad V = \big[\,v_1^\top, \ldots, v_w^\top\big],$$

which are all in $\mathbb{R}^{w \times d}$. The scaled cosine similarity attention (SCSA) can be then formulated as:

$$\mathrm{SCSA}(X) = \nu\, \mathrm{softmax}\Big[\tau\big(QK^\top\big)\Big]V,$$

where $\tau, \nu > 0$ are scalars that scale the argument of the softmax and the final output. As can be seen in the formulation, the key difference from standard attention is that each query/key vector is row-normalized to unit ℓ_2 length (up to ∇).

Similar to a TBM, H attention heads are used with each one using separate projection matrices W_h^Q, W_h^K, W_h^V, forming Q, K, V for each head, then concatenates the outputs and multiplies by W_O :

$$\mathrm{MultiHead\,SCSA}(X) = \big[\mathrm{SCSA}_1(X), \mathrm{SCSA}_2(X), \ldots, \mathrm{SCSA}_H(X)\big].$$

Let's now derive the upper-bounds of this model. Similar to the previous proofs, let's consider that our model f is built using the scaled cosine similarity attention, with H attention heads and one layer. From Appendix H.2 in the original paper [32], we have the following for a single head of attention:

$$\mathcal{L}_{\mathrm{SCSA}} \leq 2n(n-1)\nu\tau\nabla^{-\frac{1}{2}}\|W^K\| + 2(n-1)\nu\tau\nabla^{-\frac{1}{2}}\|W^Q\| + 2n\nu\nabla^{-\frac{1}{2}}\,\|W^V\|,$$

with n being the number of tokens within a local window, and $\nu, \tau > 0$ and $\nabla > 0$ the chosen hyper-parameters of in SCSA.

When considering the multi-head attention framework, we get:

$$\mathcal{L}_{\mathrm{MH-SCSA}} \leq \|W_O\|\sqrt{H} \max_{1 \leq h \leq H}\Big[\mathcal{L}_{\mathrm{SCSA_h}}\big].$$

Similar to previous proofs and since we consider the same the Feed-Forward and Layer Normalization aspect, we directly get the following result:

$$\mathcal{L}_f \leq \left(\tfrac{d}{d-1}\right)^2 \Big[1 + \|W_O\|\sqrt{H} \max_{h=1,\ldots,H}\Big\{2w(w-1)\nu\tau\nabla^{-\frac{1}{2}}\|W_h^K\| + 2(w-1)\nu\tau\nabla^{-\frac{1}{2}}\|W_h^Q\| + 2w\nu\nabla^{-\frac{1}{2}}\|W_h^V\|\Big\}\Big]\Big[1 + \|W_{\mathrm{FFN}}\|\Big]$$

which could be written as:

$$\mathcal{L}_f \leq \left(\frac{d}{d-1}\right)^2 C_1 C_2,$$

with $C_1 = 1 + \|W_O\|\sqrt{H}\max_h\Big\{2w(w-1)\nu\tau\nabla^{-\frac{1}{2}}\|W_h^K\| + 2(w-1)\nu\tau\nabla^{-\frac{1}{2}}\|W_h^Q\| + 2w\nu\nabla^{-\frac{1}{2}}\|W_h^V\|\Big\}$
$C_2 = (1 + \|W_{FFN}\|)$,

For the pooling operation, similar analogy that was used in the case of dot-product attention can be used, and we find therefore the final results:

For Average pooling:

$$\mathcal{L}_{\text{avg}} \leq \frac{1}{\sqrt{n}} \times \left(\frac{d}{d-1}\right)^2 C_1 C_2.$$

For Sum pooling:

$$\mathcal{L}_{\text{sum}} \leq \sqrt{n} \times \left(\frac{d}{d-1}\right)^2 C_1 C_2.$$

For Last-token pooling:

$$\mathcal{L}_{\text{last}} \leq \left(\frac{d}{d-1}\right)^2 C_1 C_2.$$

For Max pooling:

$$\mathcal{L}_{\text{last}} \leq \sqrt{\min(n, d)} \left(\frac{d}{d-1}\right)^2 C_1 C_2.$$

By applying the Markov inequality we get the desired result.

□

D Experimental Details

We start by noting that the necessary code to reproduce the results is publicaly available in the following link: https://github.com/king/transformer-pooling.

In what follows, we provide experimental details and hyper-parameters choices.

D.1 Computer Vision.

All computer vision experiments used a frozen Transformer backbone (ViT-base [6], ViT-small, or LipsFormer [32] built on Swin Transformer [24] architecture), with a randomly initialized heads fine-tuned on each task. We optimized with Adam [16] at a learning rate of 1×10^{-3}. All the tasks were trained for 10 epochs; all of which yielded stable convergence. Images were resized and either padded or center-cropped to the model's input resolution. No data augmentation was applied during training.

We evaluated tasks that require both local and global context. For classification (CIFAR 10/100 [17], ImageNet-100 [34], MiniPlaces [47], Caltech-UCSD Birds (CUB) [42]), the head's output dimension matched the number of classes and was trained with cross-entropy loss. For inpainting (CelebA [25], Oxford Flowers [29], Oxford-IIIT Pet [30]), the head predicted pixel values in masked regions and was trained with mean squared error. For segmentation (Pascal-VOC [9]), a linear per-pixel classifier trained with cross-entropy loss was used and report mean pixel accuracy. All the experiments were run using a single NVIDIA L4 GPU and took 25 GPU hours to obtain all results.

D.2 NLP

In all experiments involving LLMs, the Transformer backbone was kept frozen, and only a randomly initialized linear head and parametric pooling parameters were fine-tuned. We used the provided pretrained (non instruction-tuned) checkpoints for Llama3 [13], Mistral 7B [14], Qwen 2.5 [45] and BERT [5], and we pre-trained GPT-2 and L2-GPT-2 (see details below). Optimization was performed using the Adam [16] optimizer with a learning rate of 1×10^{-3}. Ten epochs of fine-tuning consistently yielded stable convergence across tasks. Each experiment was repeated five times with fixed random seeds to improve robustness and ensure reproducibility.

All pooling methods, including those with learnable components, were trained using the same configuration. Experiments were conducted on an instance with 2×NVIDIA L4 GPUs using PyTorch [31] with the Distributed Data Parallel framework and a batch size of 32 per GPU. Running all experiments took 1832 GPU hours on L4 GPUs.

Each dataset's training split was used for fine-tuning. Hyperparameters were selected based on validation performance (where available), and final results were reported on the held-out test set. To maintain consistent input dimensions, all sequences were padded or truncated to a predefined maximum length. Tokenization was done using each model's default tokenizer, and [PAD] tokens were used for padding.

For classification tasks, we used a linear head with output dimensionality matching the number of classes, trained by minimizing the cross-entropy loss. In semantic similarity tasks, the pooled embeddings were linearly projected without changing dimensionality. Cosine similarity between embedding pairs was used as the main metric. For STS Benchmark (STSB) [3], similarity scores (rescaled from $[1, 5]$ to $[0, 1]$) were predicted by minimizing mean squared error. In the HellaSwag [46] task, the goal was to match a given context to its correct ending. The context and four candidates were encoded with the same LLM and pooling method, projected linearly, and compared via cosine similarity. Cross-entropy loss was applied over the similarity scores, encouraging correct pairings. For next-token prediction, we used the TinyStories [7] corpus under a standard autoregressive setting. The training set comprised 4000 batches randomly sampled from the corpus. A randomly initialized language modeling head was trained to predict the next token based on preceding context. We used a held out test-set and randomly sampled tokens to predict.

GPT-2 Pretraining. To obtain a checkpoint for L2-GPT-2 (a GPT-2 model with L2-MHA), we followed the standard pretraining procedure described in [33], modifying the attention mechanism to use the L2 kernel with tied query and key matrices. This change slightly reduced the number of parameters (from 123M to 116M). The model was pretrained on the OpenWebText [11] corpus for $60\,000$ iterations using a batch size of 12, block size of 1024, and 40 gradient accumulation steps. Training was conducted on $8\times$NVIDIA L4 GPUs and took about 960 GPU hours.

For a fair comparison, we also pretrained a baseline GPT-2 checkpoint using identical settings, differing only in the use of the standard dot-product attention mechanism.

D.3 Time Series

For time series analysis, we used MOMENT [12], a family of Transformer-based foundation models for time series. We evaluate three pretrained checkpoints (AutonLab/MOMENT-1-{small, base, large}) trained on the Time Series Pile dataset [12].

During training, we kept the model backbone frozen and fine-tuned only the linear head and pooling operator (Weighted Average and Attention-based) for classification, forecasting, and imputation tasks. Optimization was performed using Adam [16] with a learning rate of 1×10^{-3} with a batch size of 64 on a single NVIDIA L4 GPU.

For classification, we run optimization for 20 epochs across six datasets, six pooling methods, three model sizes, and five random seeds, yielding 540 experiment trials and a total of approximately 22 GPU hours. For forecasting, we trained the prediction head for 10 epochs using a forecasting horizon of 96 future time steps across seven datasets, six pooling methods, three model sizes, and five random seeds, resulting in 630 experiment trials and approximately 90 GPU hours. For imputation, we trained the prediction head for 10 epochs across seven datasets, six pooling methods, three model sizes, and five random seeds, adding another 630 experiment trials and approximately 96 GPU hours.

E Additional Results

Computer Vision. In addition to ViT-small model, we extend our analysis to evaluate whether the theoretical insights hold in larger architectures with more attention heads and blocks. Table 4 reports the results for the same pooling benchmarks using ViT-base as the backbone.

Consistent with previous findings, Weighted Average pooling maintains strong performance across tasks, reflecting its ability to adapt to context and produce stable, generalizable representations. Similar patterns emerge for Attention-based pooling, which performs best in the inpainting task but does not outperform Weighted Average pooling in other settings. This suggests that Attention-based pooling may require more data and computational resources to reach its full potential.

Among flat pooling strategies, CLS pooling continues to yield the best results for classification tasks. Notably, the performance gap between CLS and Average pooling narrows, indicating that larger models can offset suboptimal pooling through increased representational capacity.

Table 4: Mean and standard deviation of test metrics for computer vision tasks. Best performance per dataset and model is indicated in **bold**. Best performance among non-learnable pooling methods is underlined.

Model	Pooling	Classification (Accuracy) CIFAR-10	CIFAR-100	ImageNet-100	CUB-200-2011	MiniPlaces	Inpainting (MSE) CelebA	OxfordFlower-102	Oxford-IIIT Pet	Segmentation (Accuracy) PascalVOC-Cls	PascalVOC-Det
ViT-base	Last (CLS)	$\underline{92.34 \pm 0.05}$	79.57 ± 0.13	$\underline{\mathbf{90.67 \pm 0.17}}$	$\underline{\mathbf{78.87 \pm 0.53}}$	58.86 ± 0.13	0.240 ± 0.002	$\underline{0.314 \pm 0.003}$	$\underline{\mathbf{0.256 \pm 0.003}}$	$\underline{\mathbf{72.49 \pm 0.68}}$	$\underline{\mathbf{28.01 \pm 0.94}}$
	Avg	92.25 ± 0.34	$\underline{79.67 \pm 0.40}$	90.50 ± 0.02	73.36 ± 0.50	$\underline{\mathbf{59.81 \pm 0.33}}$	$\underline{0.237 \pm 0.001}$	0.319 ± 0.005	0.266 ± 0.003	72.19 ± 0.73	26.59 ± 1.45
	Sum	91.75 ± 0.59	78.94 ± 0.10	86.98 ± 0.04	72.19 ± 0.07	59.07 ± 0.44	0.312 ± 0.004	0.678 ± 0.091	0.831 ± 0.083	71.73 ± 0.91	25.09 ± 0.17
	Max	91.39 ± 0.88	74.80 ± 0.64	90.18 ± 0.15	59.51 ± 0.89	56.61 ± 0.68	0.255 ± 0.001	0.401 ± 0.045	0.281 ± 0.009	70.21 ± 1.29	22.38 ± 0.67
	W-Avg	$\mathbf{92.55 \pm 0.17}$	$\mathbf{80.62 \pm 0.13}$	90.48 ± 0.07	74.81 ± 0.25	59.80 ± 0.12	0.236 ± 0.001	0.328 ± 0.002	0.270 ± 0.002	71.82 ± 0.71	26.62 ± 0.20
	Attn	91.81 ± 0.22	76.84 ± 0.66	90.39 ± 0.21	68.62 ± 1.89	57.62 ± 0.27	$\mathbf{0.162 \pm 0.003}$	$\mathbf{0.303 \pm 0.004}$	0.323 ± 0.048	71.89 ± 0.23	25.14 ± 1.04

NLP. Beyond the theoretical expressivity bounds shown in Figure 2, we further examine how these bounds manifest empirically in NLP settings. To this end, we construct a set of sentence variants: a base sentence (the original), two semantically similar versions created by replacing adjectives, and two dissimilar versions using unrelated words. Figure 4 shows the resulting changes in pooled representations across different pooling strategies.

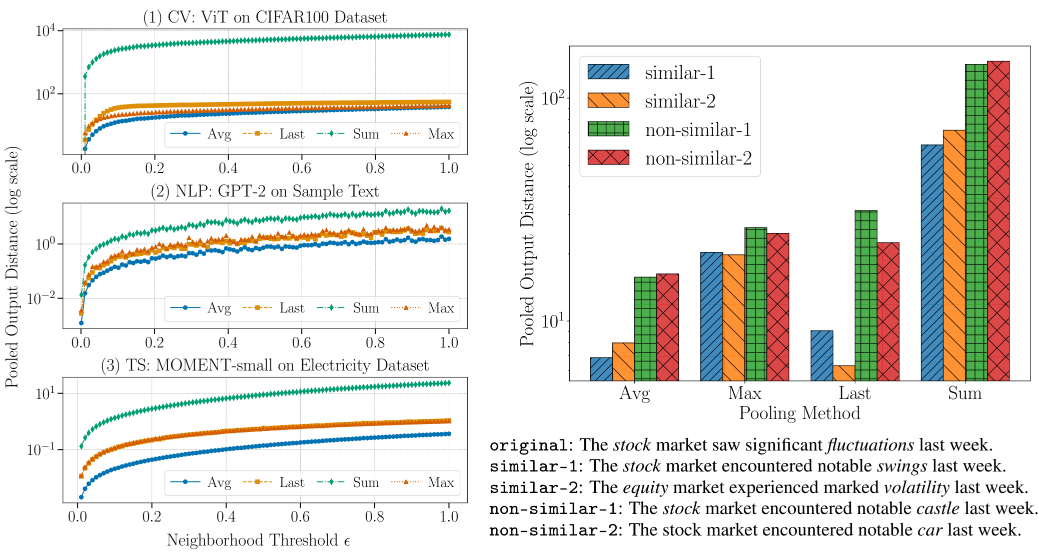


Figure 4: Empirical analysis of the expressivity power across modalities and pooling strategies. Left: Mean pooled-output distance γ versus perturbation ϵ across modalities highlighting the behavior of various methods. Right: pooled-output distances for similar and dissimilar inputs, exemplifying expressivity of different strategies.

Time Series. In addition to the results presented in Table 3, we report extended empirical evaluations of pooling operators on a broader range of time-series datasets. Tables 5, 6, and 7 provide results for classification, forecasting, and imputation tasks, respectively. Overall, the findings on these additional datasets are consistent with the trends discussed in Section 5.2, further supporting our analysis.

Weighted Average Pooling. In Section 5.3, we presented an analysis of the learned weights in the Weighted Average pooling method using the Mistral-7B model. Figure 5 extends this analysis to additional models and datasets. We observe that the learned weight distributions for a given dataset remain consistent across models, with smaller models (e.g., GPT-2 family) placing more emphasis on later tokens, while larger models exhibit more uniform weighting. The average cosine similarity between Weighted Average pooling and non-learnable pooling methods follows a similar trend: as model size increases, similarity to Max pooling decreases, suggesting reduced reliance on token-level extremes in larger architectures.

Table 5: Mean and standard deviation of test accuracy for time series classification tasks. Best performance per dataset and model in **bold**. Best performance among non-learnable pooling methods is underlined.

Model	Pooling Operator	ECG200	Electric Devices	FordA	FordB	SmallKitchen Appliances	SwedishLeaf
MOMENT-small	Last	72.29 ± 0.59	60.45 ± 0.48	76.39 ± 0.15	62.07 ± 0.44	64.06 ± 0.75	69.25 ± 0.74
	Avg	65.19 ± 0.00	61.40 ± 0.54	88.12 ± 0.14	67.05 ± 0.13	62.40 ± 1.01	55.53 ± 9.55
	Sum	**80.35 ± 1.82**	60.62 ± 2.85	92.93 ± 0.43	78.57 ± 0.72	67.13 ± 2.95	**79.42 ± 1.33**
	Max	65.19 ± 0.00	61.78 ± 1.24	90.42 ± 0.43	72.28 ± 1.00	63.94 ± 1.76	58.59 ± 6.53
	W-Avg	65.19 ± 0.00	62.54 ± 0.67	87.62 ± 0.64	67.36 ± 0.44	62.40 ± 1.01	58.81 ± 8.41
	Attn	78.84 ± 3.04	**62.95 ± 2.04**	**93.60 ± 0.37**	**79.61 ± 0.86**	**67.63 ± 2.95**	75.93 ± 2.07
MOMENT-base	Last	71.01 ± 1.94	63.05 ± 0.62	83.46 ± 0.40	64.60 ± 0.37	63.25 ± 0.54	72.03 ± 1.09
	Avg	65.19 ± 0.00	63.71 ± 0.40	89.75 ± 0.37	70.63 ± 0.54	63.05 ± 1.60	62.92 ± 5.93
	Sum	**83.30 ± 1.51**	64.30 ± 0.98	92.51 ± 0.13	79.02 ± 0.98	66.55 ± 1.48	**84.05 ± 0.94**
	Max	65.78 ± 1.32	62.31 ± 1.14	90.23 ± 0.80	70.91 ± 2.14	64.77 ± 2.12	66.24 ± 4.44
	W-Avg	65.19 ± 0.00	**64.48 ± 0.69**	89.93 ± 0.36	70.98 ± 0.62	63.20 ± 1.74	64.87 ± 5.16
	Attn	82.57 ± 1.19	64.15 ± 1.37	**92.74 ± 0.35**	**79.94 ± 0.59**	**66.76 ± 1.39**	78.66 ± 2.25
MOMENT-large	Last	72.67 ± 0.95	61.10 ± 0.53	79.61 ± 0.31	63.63 ± 0.64	65.45 ± 1.71	76.75 ± 1.47
	Avg	65.19 ± 0.00	61.72 ± 0.93	85.98 ± 0.53	67.56 ± 1.18	60.42 ± 0.91	59.39 ± 6.89
	Sum	75.97 ± 2.82	**63.45 ± 0.76**	92.85 ± 0.42	78.08 ± 0.48	64.92 ± 7.21	**82.23 ± 0.88**
	Max	65.19 ± 0.00	60.08 ± 0.54	87.14 ± 0.39	69.52 ± 0.74	62.27 ± 0.64	64.50 ± 3.75
	W-Avg	65.19 ± 0.00	61.48 ± 0.90	86.07 ± 0.36	67.65 ± 1.09	60.54 ± 0.53	60.84 ± 6.34
	Attn	**78.73 ± 3.09**	62.83 ± 1.38	**93.02 ± 0.26**	**79.95 ± 1.02**	**65.64 ± 3.54**	80.66 ± 1.04

Table 6: Mean and standard deviation of test MSE for time series forecasting tasks. Best performance per dataset and model in **bold**. Best performance among non-learnable pooling methods is underlined.

Model	Pooling Operator	ETTh1	ETTh2	ETTm1	ETTm2	electricity	traffic	weather
MOMENT-small	Last	**0.082 ± 0.000**	**0.193 ± 0.000**	**0.040 ± 0.000**	0.107 ± 0.000	**0.400 ± 0.001**	0.273 ± 0.001	0.002 ± 0.000
	Avg	0.105 ± 0.000	0.305 ± 0.001	0.079 ± 0.000	0.234 ± 0.000	0.790 ± 0.000	1.724 ± 0.001	**0.002 ± 0.000**
	Sum	0.103 ± 0.001	0.300 ± 0.005	0.053 ± 0.000	0.141 ± 0.000	0.826 ± 0.014	1.422 ± 0.050	0.005 ± 0.001
	Max	0.106 ± 0.001	0.311 ± 0.002	0.082 ± 0.001	0.230 ± 0.001	0.800 ± 0.001	1.759 ± 0.002	0.002 ± 0.000
	W-Avg	0.105 ± 0.000	0.287 ± 0.002	0.058 ± 0.005	0.184 ± 0.000	0.539 ± 0.008	0.954 ± 0.014	0.002 ± 0.000
	Attn	0.106 ± 0.001	0.313 ± 0.003	0.041 ± 0.001	**0.097 ± 0.000**	0.475 ± 0.059	**0.258 ± 0.002**	0.003 ± 0.000
MOMENT-base	Last	**0.081 ± 0.000**	**0.193 ± 0.000**	**0.040 ± 0.000**	0.105 ± 0.000	**0.397 ± 0.000**	**0.265 ± 0.000**	0.002 ± 0.000
	Avg	0.105 ± 0.000	0.304 ± 0.002	0.070 ± 0.000	0.226 ± 0.000	0.785 ± 0.001	1.719 ± 0.001	**0.002 ± 0.000**
	Sum	0.101 ± 0.002	0.283 ± 0.002	0.052 ± 0.000	0.136 ± 0.001	0.805 ± 0.009	1.029 ± 0.023	0.004 ± 0.000
	Max	0.106 ± 0.000	0.309 ± 0.002	0.075 ± 0.001	0.225 ± 0.000	0.796 ± 0.001	1.747 ± 0.003	0.002 ± 0.000
	W-Avg	0.105 ± 0.000	0.280 ± 0.002	0.058 ± 0.000	0.178 ± 0.001	0.511 ± 0.006	0.857 ± 0.014	0.002 ± 0.000
	Attn	0.096 ± 0.009	0.291 ± 0.012	0.043 ± 0.000	**0.097 ± 0.001**	0.507 ± 0.148	0.306 ± 0.037	0.003 ± 0.000
MOMENT-large	Last	**0.080 ± 0.000**	**0.195 ± 0.000**	0.039 ± 0.000	**0.103 ± 0.000**	**0.379 ± 0.000**	**0.272 ± 0.001**	0.002 ± 0.000
	Avg	0.105 ± 0.000	0.306 ± 0.000	0.073 ± 0.000	0.207 ± 0.000	0.778 ± 0.000	1.711 ± 0.001	**0.002 ± 0.000**
	Sum	0.101 ± 0.001	0.269 ± 0.002	0.049 ± 0.000	0.126 ± 0.001	0.688 ± 0.007	0.782 ± 0.003	0.003 ± 0.000
	Max	0.104 ± 0.001	0.306 ± 0.002	0.073 ± 0.000	0.206 ± 0.001	0.785 ± 0.001	1.743 ± 0.000	0.002 ± 0.000
	W-Avg	0.105 ± 0.001	0.283 ± 0.000	0.053 ± 0.000	0.170 ± 0.003	0.534 ± 0.003	0.951 ± 0.005	0.002 ± 0.000
	Attn	0.097 ± 0.004	0.306 ± 0.000	**0.039 ± 0.000**	0.106 ± 0.003	0.684 ± 0.003	0.423 ± 0.028	0.003 ± 0.000

E.1 Additional NLP-related Results on Larger Models

Table E.1 below reports results across tasks for larger models under a consistent experimental setup to our previous experiments. With larger models, trends across pooling strategies remain visible, but the absolute differences between pooling methods diminish, aligning with the theoretical interpretation.

E.2 Additional Results on Sequence Length Changes for NLP

To further examine pooling behavior, we conducted experiments with the Mistral-7B model on HellaSwag, varying the maximum input sequence length from 16 to 128 tokens. Inputs were truncated or padded as required, with padding tokens excluded from pooling operations to maintain consistency with our setup.

Table 7: Mean and standard deviation of test MSE for time series imputation tasks. Best performance per dataset and model in **bold**. Best performance among non-learnable pooling methods is underlined.

Model	Pooling Operator	ETTh1	ETTh2	ETTm1	ETTm2	electricity	traffic	weather
MOMENT-small	Last	0.081 ± 0.002	0.241 ± 0.002	0.051 ± 0.000	0.181 ± 0.001	0.753 ± 0.010	1.709 ± 0.016	0.002 ± 0.000
	Avg	0.080 ± 0.002	0.233 ± 0.002	**0.050 ± 0.000**	0.178 ± 0.001	0.774 ± 0.013	1.583 ± 0.016	**0.002 ± 0.000**
	Sum	0.106 ± 0.008	0.309 ± 0.041	0.054 ± 0.001	0.186 ± 0.010	1.072 ± 0.143	2.130 ± 0.238	0.037 ± 0.038
	Max	0.082 ± 0.002	0.229 ± 0.004	0.051 ± 0.000	0.174 ± 0.003	0.720 ± 0.013	1.211 ± 0.051	0.003 ± 0.000
	W-Avg	0.080 ± 0.002	0.233 ± 0.002	0.050 ± 0.000	0.178 ± 0.001	0.774 ± 0.013	1.582 ± 0.016	0.002 ± 0.000
	Attn	**0.076 ± 0.003**	**0.088 ± 0.007**	0.051 ± 0.001	**0.152 ± 0.001**	**0.379 ± 0.028**	**0.265 ± 0.015**	0.003 ± 0.001
MOMENT-base	Last	0.082 ± 0.001	0.243 ± 0.005	0.051 ± 0.000	0.181 ± 0.002	0.760 ± 0.011	1.668 ± 0.020	0.002 ± 0.000
	Avg	0.082 ± 0.001	0.233 ± 0.005	0.051 ± 0.000	0.178 ± 0.002	0.769 ± 0.016	1.424 ± 0.020	**0.002 ± 0.000**
	Sum	0.103 ± 0.002	0.255 ± 0.022	0.054 ± 0.001	0.186 ± 0.006	1.006 ± 0.161	1.301 ± 0.089	0.015 ± 0.009
	Max	0.082 ± 0.002	0.219 ± 0.004	0.051 ± 0.000	0.173 ± 0.002	0.707 ± 0.014	1.000 ± 0.113	0.002 ± 0.000
	W-Avg	0.082 ± 0.001	0.233 ± 0.005	0.051 ± 0.000	0.178 ± 0.002	0.769 ± 0.015	1.425 ± 0.020	0.002 ± 0.000
	Attn	**0.072 ± 0.003**	**0.082 ± 0.002**	**0.050 ± 0.000**	**0.151 ± 0.002**	**0.370 ± 0.013**	**0.273 ± 0.004**	0.004 ± 0.003
MOMENT-large	Last	0.082 ± 0.001	0.242 ± 0.005	0.051 ± 0.001	0.181 ± 0.001	0.752 ± 0.014	1.699 ± 0.017	0.002 ± 0.000
	Avg	0.081 ± 0.002	0.238 ± 0.007	0.050 ± 0.000	0.177 ± 0.001	0.753 ± 0.018	1.508 ± 0.011	**0.002 ± 0.000**
	Sum	0.095 ± 0.008	0.254 ± 0.020	0.053 ± 0.001	0.177 ± 0.006	0.887 ± 0.075	1.150 ± 0.065	0.014 ± 0.002
	Max	0.081 ± 0.001	0.230 ± 0.003	0.050 ± 0.001	0.170 ± 0.002	0.705 ± 0.008	1.158 ± 0.025	0.003 ± 0.000
	W-Avg	0.081 ± 0.002	0.238 ± 0.007	0.050 ± 0.000	0.176 ± 0.001	0.753 ± 0.018	1.503 ± 0.011	0.002 ± 0.000
	Attn	**0.072 ± 0.003**	**0.091 ± 0.006**	**0.050 ± 0.000**	**0.147 ± 0.003**	**0.295 ± 0.006**	**0.231 ± 0.009**	0.004 ± 0.001

Table 8: Mean and standard deviation of test metrics for NLP tasks. STSB and HellaSwag are grouped under Sentiment Analysis. Values are mean ± std.

Model	**Pooling**	Sentiment (STSB)	Sentiment (HellaSwag)	Banking (Accuracy)	Tweet (Accuracy)	Next Token (Accuracy)
Qwen2.5–14B	Last	0.288 ± 0.003	0.692 ± 0.011	34.497 ± 0.178	57.008 ± 0.004	**91.039 ± 0.002**
	Avg	0.581 ± 0.000	0.773 ± 0.000	85.390 ± 0.001	**69.930 ± 0.008**	53.893 ± 0.250
	Sum	0.579 ± 0.002	0.776 ± 0.002	82.711 ± 0.008	61.218 ± 1.194	47.053 ± 0.341
	Max	0.488 ± 0.005	0.727 ± 0.001	77.825 ± 0.006	55.390 ± 0.728	25.578 ± 0.332
	W-Avg	**0.589 ± 0.001**	**0.798 ± 0.002**	**86.867 ± 0.002**	69.770 ± 0.007	90.653 ± 0.020
	Attn	0.259 ± 0.011	0.712 ± 0.013	66.387 ± 1.939	56.410 ± 3.001	40.573 ± 0.892
Qwen2.5–32B	Last	0.303 ± 0.001	0.723 ± 0.009	34.383 ± 0.268	55.886 ± 0.006	**89.886 ± 0.005**
	Avg	0.603 ± 0.000	0.781 ± 0.001	87.760 ± 0.002	68.328 ± 0.012	48.536 ± 0.334
	Sum	0.604 ± 0.001	0.782 ± 0.005	85.227 ± 0.013	63.665 ± 0.865	39.204 ± 0.627
	Max	0.488 ± 0.005	0.729 ± 0.008	83.929 ± 0.008	65.268 ± 0.596	31.273 ± 0.586
	W-Avg	**0.627 ± 0.003**	**0.812 ± 0.003**	**89.903 ± 0.004**	**69.464 ± 0.009**	89.022 ± 0.019
	Attn	0.355 ± 0.016	0.730 ± 0.011	68.929 ± 1.054	58.828 ± 2.695	29.750 ± 1.028
Mistral3.1–24B	Last	0.503 ± 0.002	0.745 ± 0.008	75.487 ± 0.087	53.701 ± 0.005	**88.972 ± 0.007**
	Avg	0.631 ± 0.001	0.784 ± 0.001	87.403 ± 0.006	66.871 ± 0.009	51.723 ± 0.812
	Sum	0.622 ± 0.004	0.783 ± 0.004	87.597 ± 0.015	62.791 ± 0.976	42.306 ± 0.732
	Max	0.488 ± 0.003	0.733 ± 0.007	79.675 ± 0.010	61.655 ± 0.473	27.804 ± 0.923
	W-Avg	**0.682 ± 0.002**	**0.816 ± 0.003**	**88.711 ± 0.017**	**66.200 ± 0.005**	87.849 ± 0.024
	Attn	0.392 ± 0.043	0.697 ± 0.009	72.922 ± 1.012	31.294 ± 4.452	19.911 ± 2.023

The results, summarized in Table 9, compare pooling strategies across different sequence lengths. Naturally, shorter contexts lead to performance drops due to truncation of semantically important content. To isolate the contribution of pooling itself, comparisons should be made column-wise (i.e., at fixed sequence lengths).

We find that pooling sensitivity is most pronounced at shorter lengths, especially for Last-token and Attention-based pooling. For longer contexts (64 or 128 tokens), performance stabilizes across pooling methods, and the relative differences align more closely with theoretical expectations.

E.3 Pooling and Adversarial Robustness

Beyond shaping model expressivity for downstream tasks, the choice of pooling operation also impacts the model's adversarial robustness. Our theoretical framework provides insights in this direction by interpreting neighborhood changes, introduced in Section 4.1, as adversarial rather than semantic perturbations intentionally crafted to mislead the model. The analysis suggests that certain pooling operations, such as *Average*, may naturally smooth out adversarial noise, while others like

Table 9: Mean and standard deviation of metrics for Mistral-7B on HellaSwag across different input sequence lengths. Values are mean ± std. Best performance per column is in **bold**.

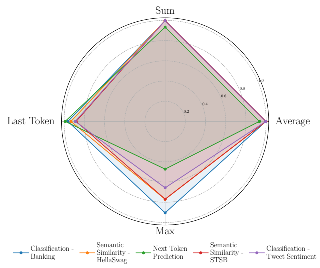
Pooling	**16**	**32**	**64**	**128**
Last	0.454 ± 0.003	0.621 ± 0.002	0.770 ± 0.002	0.781 ± 0.001
Avg	0.503 ± 0.001	0.702 ± 0.000	0.769 ± 0.000	0.771 ± 0.001
Sum	0.504 ± 0.001	0.702 ± 0.001	0.769 ± 0.000	0.771 ± 0.001
Max	0.435 ± 0.003	0.616 ± 0.003	0.709 ± 0.001	0.700 ± 0.008
W-Avg	$\mathbf{0.523 \pm 0.002}$	$\mathbf{0.724 \pm 0.000}$	$\mathbf{0.801 \pm 0.000}$	$\mathbf{0.802 \pm 0.003}$
Attn	0.220 ± 0.169	0.278 ± 0.251	0.737 ± 0.018	0.764 ± 0.025

Table 10: Attack Success rate for different considered Pooling strategies using the FGSM adversarial attack on the CIFAR-10 and CIFAR-100.

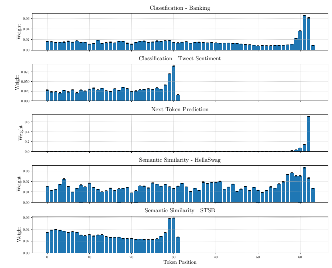
Dataset	Attack Budget	CLS	Avg	Sum	Max	W-Avg	Attention-Based
CIFAR-10	$\epsilon = 3/255$	18.56	18.94	10.92	20.32	16.93	13.9
	$\epsilon = 8/255$	29.38	28.16	10.92	25.97	25.59	14.44
CIFAR-100	$\epsilon = 3/255$	34.29	32.44	20.67	31.84	31.57	29.54
	$\epsilon = 8/255$	43.99	41.42	20.68	38.07	40.49	32.94

Max can either amplify or ignore the perturbation depending on whether the adversarial signal falls within the selected region.

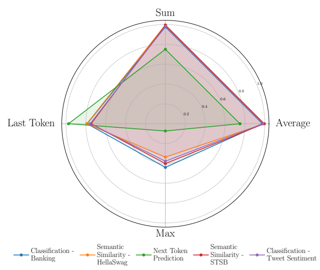
To empirically validate these insights, we apply the Fast Gradient Sign Method (FGSM) to a pre-trained ViT model evaluated on CIFAR-10 and CIFAR-100. We use the same attack budget as the one usually used in the literature ($\epsilon = 3/255$ and $\epsilon = 2/255$) and we used the same number of epochs and the same initialization for all the poolings to ensure fairness of the comparison [8]. Table 10 reports the attack success rates for various pooling strategies under FGSM. As expected, the success rates vary across pooling methods, confirming that pooling choices can meaningfully influence robustness. Therefore, in domains where robustness is critical, such as healthcare or finance, the pooling strategy should be selected not only for its expressivity but also for its impact on adversarial resilience.



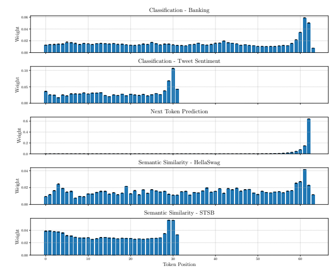
(a) GPT-2



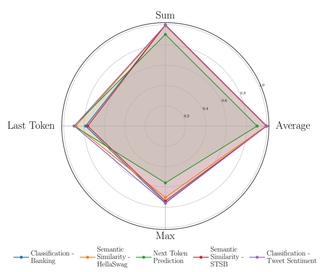
(b) GPT-2



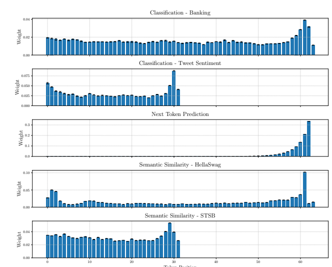
(c) L2-GPT-2



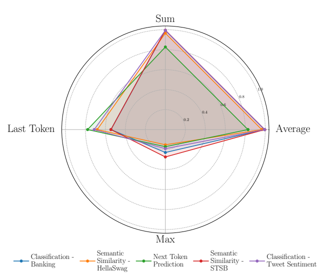
(d) L2-GPT-2



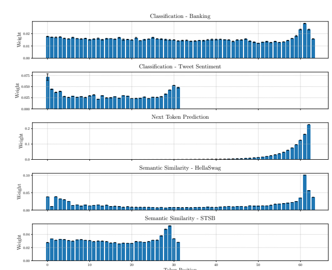
(e) Qwen2.5



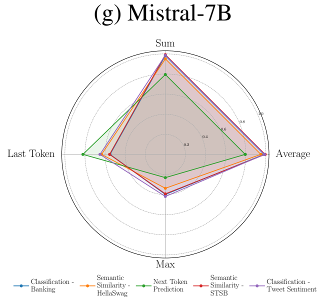
(f) Qwen2.5



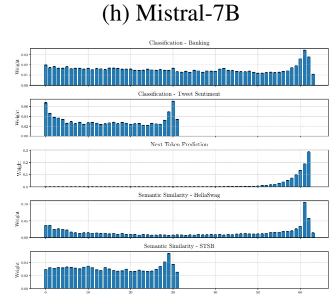
(g) Mistral-7B



(h) Mistral-7B



(i) Llama



(j) Llama

Figure 5: Left: Cosine similarity of weighted average pooling with other pooling methods. Right: Learned weight distributions.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and the introduction are consistent with the provided theoretical results in Section 4 and additionally empirically validated in our experimental results in Section 5.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations of our work with respect to both the theoretical and empirical results are discussed in Section 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: All the theoretical claims are explained in the corresponding proofs (Appendices A, B and C). In addition, we clearly state our problem setup detailing the assumptions in Section 3, which are referenced in each proof and theorem.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: In addition to providing a detailed experimental setup in Appendix D, we provide the source code to reproduce our results in the supplementary materials section.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: In all our experiments we used publicly available datasets that can be found on, e.g., HuggingFace or other open source platforms. We provide the source code in the supplementary material for reproducibility.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Appendix D contains the experimental setup for all modalities, including training parameters and data splits. The code contains all hyperparameters for every model and experiment.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report standard deviation of all results in the respective tables in Section 5.2 obtained via repeated experiments with 5 random seeds. Further details in Appendix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the type of GPUs and number of GPU hours used for our experiments in Appendix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our work adheres to the NeurIPS Code of Ethics. We use public datasets and publicly available models and report on the limitations of our work.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

Justification: Our work has no further societal impacts apart from known impacts of LLMs and other Transformer-based models. We are not aware of any applications of our insights that would result in negative societal impacts.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our paper does not release any data or models that have a high risk for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets, models and repositories were cited appropriately.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The source code released along with our paper is properly documented and contains the license terms.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our work did not involve crowdsourcing nor research with human subjects. All experiments are performed on publicly available datasets.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

Justification: Our work did not involve crowdsourcing nor research with human subjects. All experiments are performed on publicly available datasets.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: No LLM was used in this work for the core methods.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.