
Accurate KV Cache Eviction via Anchor Direction Projection for Efficient LLM Inference

Zijie Geng¹, Jie Wang^{1*}, Ziqi Liu¹, Feng Ju¹, Yiming Li², Xing Li², Mingxuan Yuan²,
Jianye Hao^{2,3}, Defu Lian¹, Enhong Chen¹, Yongdong Zhang¹, Feng Wu¹

¹University of Science and Technology of China

²Noah's Ark Lab, Huawei

³Tianjin University

{zijiegeng, ziqiliu, fengju}@mail.ustc.edu.cn, jiewangx@ustc.edu.cn

Abstract

Key-Value (KV) cache eviction—which retains the KV pairs of the most important tokens while discarding less important ones—is a critical technique for optimizing both memory usage and inference latency in large language models (LLMs). However, existing approaches often rely on simple heuristics—such as attention weights—to measure token importance, overlooking the spatial relationships between token value states in the vector space. This often leads to suboptimal token selections and thus performance degradation. To tackle this problem, we propose a novel method, namely **AnDPro** (**A**nchor **D**irection **P**rojection), which introduces a projection-based scoring function to more accurately measure token importance. Specifically, AnDPro operates in the space of value vectors and leverages the projections of these vectors onto an “Anchor Direction”—the direction of the pre-eviction output—to measure token importance and guide more accurate token selection. Experiments on 16 datasets from the LongBench benchmark demonstrate that AnDPro can maintain 96.07% of the full cache accuracy using only 3.44% KV cache budget, reducing KV cache budget size by 46.0% without compromising quality compared to previous state-of-the-arts.

1 Introduction

Large language models (LLMs) have revolutionized human society by reshaping the way we process information [1–5]. Modern LLMs are primarily built upon the Transformer architecture, with self-attention as their core mechanism [6]. However, the self-attention mechanism inherently involves quadratic computational complexity, which grows dramatically as the sequence length increases [7]. To address this bottleneck, the Key-Value (KV) cache mechanism has been introduced to store and reuse intermediate results—i.e., the KV pairs in the self-attention mechanism—to reduce the computational complexity from quadratic to linear [8]. As the demand for processing longer sequences grows, however, the KV cache size expands proportionally, leading to substantial increases in both GPU memory consumption and I/O latency [9]. Therefore, effective KV cache management is critical for practical LLM scaling and deployment.

Recent studies have revealed that LLM self-attention exhibits inherent sparsity, with only a small subset of tokens making a substantial contribution to the model’s output [10–12]. This insight has sparked the development of a series of KV cache eviction strategies, which identify and retain the KV pairs of the most important tokens while discarding less important ones [13–15]. For instance, H2O [9] greedily eliminates KV pairs based on their cumulative attention weights during the generation phase. SnapKV [16] employs an observation window to generate queries and uses the obtained attention weights to measure the importance of prompt tokens during the prefilling phase. Ada-KV [17] further

*Corresponding author.

refines this paradigm by jointly identifying important tokens across attention heads within each Transformer layer. By selectively removing less critical tokens, these approaches can effectively optimize both memory and time efficiency without significantly compromising the performance. The aforementioned approaches essentially rely on an importance criterion—which we refer to as a “scoring function”—to determine token retention priority. Most of them use attention weights as the scoring function, assuming that tokens with higher attention weights should be retained preferentially. Although this assumption is intuitive, they overlook the spatial relationships between token value states in the vector space, which play a critical role in how these states interact to influence the model’s output. This often leads to suboptimal token selection decisions and thus performance degradation. Therefore, effective KV cache management requires advancing beyond attention-only heuristics by developing scoring functions that explicitly account for the vector space geometry.

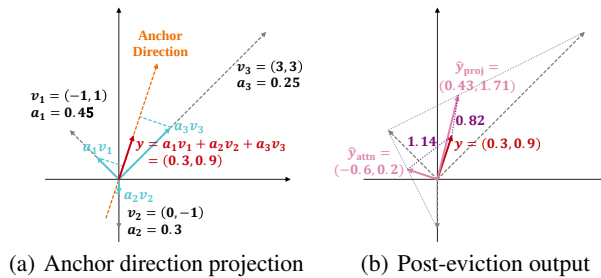


Figure 1: A diagram of the projection-based KV cache eviction. (a) Consider three tokens with value states v_1 , v_2 , v_3 , and attention weights a_1 , a_2 , a_3 . The **pre-eviction output** is denoted as y . Our approach computes the **projections of $a_i v_i$** onto the “**anchor direction**” (i.e., the direction of y). (b) With a budget of 2, the attention-based eviction approach selects v_1 and v_2 , resulting in a **post-eviction output** $\hat{y}_{\text{attn}}$ and an error of $\|y - \hat{y}_{\text{attn}}\|_2 = 1.14$. Our projection-based approach selects v_1 and v_3 , resulting in a **post-eviction output** $\hat{y}_{\text{proj}}$, with a smaller error of $\|y - \hat{y}_{\text{proj}}\|_2 = 0.82$.

To tackle this problem, we propose a novel method, namely **AnDPro (Anchor Direction Projection)**, which introduces a projection-based scoring function to more accurately measure token importance. Specifically, we formulate KV cache eviction as a combinatorial optimization problem and then transform it into a sparse optimization formulation through mild relaxations. The theoretical and intuitive analysis of this problem motivates us to introduce an “*anchor direction*” in the vector space, i.e., the direction of the pre-eviction output. Intuitively, this anchor direction guides the retention of the most important semantic information, leading us to use the projections of token value states onto this direction to measure token importance (see Figure 1). Experiments on 16 datasets from the LongBench benchmark demonstrate that AnDPro maintains 96.07% of the full cache accuracy using only 3.44% KV cache budget, reducing KV cache size by 46.0% without compromising model quality compared to previous state-of-the-arts.

2 Related Work

LLMs have been widely applied across various domains and tasks [18, 19]. Among the numerous research directions related to LLMs, our work is most closely aligned with LLM inference acceleration [20–25], particularly in the areas of KV cache management, eviction, and budget allocation.

KV Cache Management plays a crucial role in optimizing both memory and time efficiency in LLM inference. In a recent survey by Li et al. [13], KV cache management techniques are categorized into three levels: token-level, model-level, and system-level strategies. Our work is primarily related to token-level KV cache optimization, specifically KV cache eviction and KV cache budget allocation. Other token-level methods include KV cache merging [26–30], KV cache quantization [31–37], and KV cache low-rank decomposition [38–41].

KV Cache Eviction approaches selectively retain the most important KV pairs and discard the less critical ones. Some methods, such as StreamingLLM [14], FastGen [11], and recent studies [20, 21], are designed based on certain patterns and fixed rules, such as the sliding windows, for KV cache eviction. Other approaches, such as H2O [9], Scissorhands [10], and recent studies [42, 43, 22, 44], leverage attention scores as a criterion for selecting the most important KV pairs from the cache. SnapKV [16] introduces an observation window to generate queries, using accumulated attention scores to evaluate token importance in prefilling. Ada-KV [17] refines this framework by adaptively allocating the cache budget across attention heads based on the attention scores. More recently,

CriticalKV [45] combines attention scores with the norm of value vectors as a better scoring function. Our work builds upon this line of research, and draws inspiration from SnapKV and Ada-KV.

KV Cache Budget Allocation focuses on allocating the budget across different components of the model, such as layers and attention heads. Layer-wise allocation techniques include PyramidInfer [46] and DynamicKV [47]. Head-wise budget allocation methods include Ada-KV [17], HeadKV [48], and DuoAttention [49]. These methods are orthogonal to our work, as we focus on retaining important KV pairs for any layer- or head-level budget allocation. In this work, we implement our approach to achieve an adaptive budget allocation across attention heads.

3 Our Approach

In this section, we introduce our proposed method, **AnDPro** (**A**nchor **D**irection **P**rojection). Section 3.1 begins by formalizing the KV cache eviction problem to provide a theoretical foundation for deeper analysis. In Section 3.2, we present a theoretical exploration of the optimization problem, which motivates the general form of projection-based scoring function. Section 3.3 further derives a practical form of the scoring function through an intuitive analysis. Finally, in Section 3.4, we provide the implementation details of AnDPro. All the proofs can be found in Appendix A.

3.1 Problem Formulation

This section aims to answer the following question:

Q1: *How to formulate the KV cache eviction problem in a manner that enables in-depth analysis on the design of an appropriate scoring function for KV cache eviction?*

Our objective is to select a subset of tokens from the input prompt to retain, discarding the others, while approximating the original outputs as accurately as possible. In this section we consider one single attention head. Consider an input sequence $X = [x_1, \dots, x_n]$, where n represents the context length, i.e., the number of tokens in the prompt.¹ Let $\mathbf{x}_i \in \mathbb{R}^d$ denote the embedding vector of the i^{th} token, where d denotes the embedding dimension. Let $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V \in \mathbb{R}^{d \times d}$ be the parameter matrices for the query, key, and value projections, respectively. For the i^{th} token, its query $\mathbf{q}_i$, key $\mathbf{k}_i$, and value states $\mathbf{v}_i$ are given by:

$$\mathbf{q}_i = \mathbf{W}^Q \mathbf{x}_i, \quad \mathbf{k}_i = \mathbf{W}^K \mathbf{x}_i, \quad \mathbf{v}_i = \mathbf{W}^V \mathbf{x}_i. \quad (1)$$

Let $\mathbf{q}$ be a query given from an observation window, which will be further detailed in Section 3.4. The pre-eviction output of the attention head is given by:

$$\mathbf{y} = \sum_{i=1}^n a_i \mathbf{v}_i, \quad \text{where } a_i = \frac{\text{Softmax}(\mathbf{q}^\top \mathbf{k}_i)}{\sum_{1 \leq i \leq n} \text{Softmax}(\mathbf{q}^\top \mathbf{k}_i)}. \quad (2)$$

Let $S \subset [n]$ be the set of tokens selected to retain. The post-eviction output, i.e., the output derived when only retaining KV pairs corresponding to tokens $i \in S$, is then given by:

$$\hat{\mathbf{y}} = \sum_{i \in S} \hat{a}_i \mathbf{v}_i, \quad \text{where } \hat{a}_i = \frac{\text{Softmax}(\mathbf{q}^\top \mathbf{k}_i)}{\sum_{i \in S} \text{Softmax}(\mathbf{q}^\top \mathbf{k}_i)}. \quad (3)$$

Our objective is that the post-eviction output approximates the pre-eviction output as accurately as possible. It is intuitive to use the squared error as a metric to quantify the eviction loss:

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) \triangleq \frac{1}{2} \|\mathbf{y} - \hat{\mathbf{y}}\|_2^2. \quad (4)$$

Additionally, we impose a sparsity constraint $|S| \leq k$ on the number of retained tokens. The following proposition provides a more convenient expression of the post-eviction output to aid the further analysis.

Proposition 1. *Let $S \subset [n]$ be the set of retained tokens, and define the indicator function*

$$z_i \triangleq \begin{cases} 1, & i \in S, \\ 0, & i \notin S \end{cases} \quad (5)$$

¹Throughout this paper, all vectors are presented as column vectors. We use the index i and its corresponding token x_i interchangeably, given that no ambiguity arises.

to represent whether the i^{th} token is retained. Then, the post-eviction output can be expressed as:

$$\hat{\mathbf{y}} = \frac{\sum_{i=1}^n z_i a_i \mathbf{v}_i}{\sum_{i=1}^n z_i a_i}. \quad (6)$$

Based on Proposition 1, we can formulate the KV cache eviction problem as follows:

$$\min_{\mathbf{z} \in \{0,1\}^n} \frac{1}{2} \left\| \mathbf{y} - \frac{\sum_{i=1}^n z_i a_i \mathbf{v}_i}{\sum_{i=1}^n z_i a_i} \right\|_2^2, \quad \text{s.t.} \quad \sum_{i=1}^n z_i \leq k. \quad (\text{P0-C})$$

In the realm of sparse optimization, a prevalent approach for dealing with the sparsity constraint is to introduce an l_1 -norm regularization [50]. When dealing with binary variables, the l_1 -norm is equivalent to l_0 norm, which counts the number of non-zero elements. The regularized form of Problem (P0-C) is then expressed as:

$$\min_{\mathbf{z} \in \{0,1\}^n} \frac{1}{2} \left\| \mathbf{y} - \frac{\sum_{i=1}^n z_i a_i \mathbf{v}_i}{\sum_{i=1}^n z_i a_i} \right\|_2^2 + \lambda \sum_{i=1}^n z_i. \quad (\text{P0-R})$$

Although Problem (P0-R) is in a concise form, it remains $\mathcal{NP}$ -hard, making exact solution computationally infeasible. Given the context of LLM inference, where a solution needs to be found within milliseconds, we seek to identify high-quality solutions quickly and approximate the problem's optimal solution. To this end, we explore mild relaxations of (P0-R) to derive a scoring function that can facilitate fast filtering for efficient KV cache eviction. Notice that directly relaxing the binary variables z_i in (P0-R) to continuous values would lead to problem degeneration and results in trivial solutions. Specifically, when z_i are relaxed, they can be scaled arbitrarily without affecting the first term in the objective function. If the values of z_i are scaled sufficiently small, the second term in the objective function will diminish to zero, yielding trivial solutions.

We introduce new variables s and β_i , defined by

$$s \triangleq \sum_{i=1}^n z_i a_i \quad \text{and} \quad \beta_i \triangleq \frac{z_i}{s}. \quad (7)$$

Then, we obtain:

$$\frac{\sum_{i=1}^n z_i a_i \mathbf{v}_i}{\sum_{i=1}^n z_i a_i} = \frac{\sum_{i=1}^n \beta_i a_i \mathbf{v}_i}{\sum_{i=1}^n \beta_i a_i} \quad \text{and} \quad \sum_{i=1}^n \beta_i a_i = 1. \quad (8)$$

Consequently, Problem (P0-R) can be reformulated as:

$$\begin{aligned} \min_{\mathbf{z} \in \{0,1\}^n} \quad & \frac{1}{2} \left\| \mathbf{y} - \sum_{i=1}^n \beta_i a_i \mathbf{v}_i \right\|_2^2 + \lambda s \sum_{i=1}^n \beta_i, \\ \text{s.t.} \quad & \sum_{i=1}^n \beta_i a_i = 1, \quad s \beta_i = z_i, \quad i = 1, \dots, n. \end{aligned} \quad (9)$$

In this reformulation, we can select an arbitrarily small s , which would lead to extremely small z_i . As a result, the regularization term would approach zero, and smaller values of s would always yield better objective function values. Moreover, if we impose a lower bound on s , the optimal solutions will always set s take to this lower bound. Therefore, we fix s as a constant satisfying $0 < s \leq 1$, and perform a convex relaxation of the binary variables z_i . This leads to the following optimization problem:

$$\min_{\beta \in [0, \frac{1}{s}]^n} \frac{1}{2} \left\| \mathbf{y} - \sum_{i=1}^n \beta_i a_i \mathbf{v}_i \right\|_2^2 + \lambda s \sum_{i=1}^n \beta_i, \quad \text{s.t.} \quad \sum_{i=1}^n \beta_i a_i = 1. \quad (\text{P1})$$

A1: Through mild relaxations, we derive Problem (P1), based on which we further work on the design of the scoring function in the subsequent sections.

Remark 1. We would like to re-emphasize that our goal is not to achieve an exact solution to the original problem, which is complex enough so that an exact solution is intractable. Instead, we aim to leverage this analysis to provide insightful motivations for the design of scoring functions.

3.2 Projection-Based Scoring Function: General Form

Although Problem (P1) offers a concise formulation, solving it directly during the LLM inference process remains infeasible. Inspired by the “screening rules” in sparse optimization [51], we now investigate the design of a closed-form scoring function that can identify the active variables in Problem (P1), i.e., those with non-zero coefficients in the optimal solution. In this way, we can retain the active variables without explicitly solving the entire optimization problem. Thus, this section aims to answer the following question:

Q2: *Is it possible to find a closed-form scoring function that identifies the active variables for Problem (P1)? If so, what is the ideal mathematical form of such a scoring function?*

Let β^* denote an optimal solution to Problem (P1). In this section, we will show that under certain conditions, it is possible to determine that $\beta_i^* = 0$ for some $i \in [n]$, implying that the corresponding tokens can be removed from the cache. To further investigate the problem, we introduce a new set of variables, $\mathbf{u} \triangleq \mathbf{y} - \sum_{i=1}^n \beta_i a_i \mathbf{v}_i$. We can then reformulate Problem (P1) as:

$$\begin{aligned} \min_{\beta \in [0, \frac{1}{s}]^n} \quad & \frac{1}{2} \|\mathbf{u}\|_2^2 + \lambda s \sum_{i=1}^n \beta_i, \\ \text{s.t.} \quad & \mathbf{u} = \mathbf{y} - \sum_{i=1}^n \beta_i a_i \mathbf{v}_i, \quad \sum_{i=1}^n \beta_i a_i = 1 \end{aligned} \quad (\text{P-Primal})$$

We can now derive the dual problem of (P-Primal), as presented in the following theorem.

Theorem 1. *The dual problem to Problem (P-Primal) is given by:*

$$\sup_{\boldsymbol{\eta}, \zeta} -\frac{1}{2} \|\boldsymbol{\eta}\|^2 + \boldsymbol{\eta}^\top \mathbf{y} - \zeta + \frac{1}{s} \sum_{i=1}^n \rho_i(\boldsymbol{\eta}, \zeta), \quad (\text{P-Dual})$$

where $\boldsymbol{\eta}^* \in \mathbb{R}^n$ and $\zeta \in \mathbb{R}$ are dual variables, and

$$\rho_i(\boldsymbol{\eta}, \zeta) \triangleq \min(0, \lambda s - a_i (\boldsymbol{\eta}^\top \mathbf{v}_i - \zeta)). \quad (10)$$

Problem (P-Primal) is convex, and its constraints are affine. According to Slater’s condition [52], as long as (P-Primal) is feasible, strong duality holds. The reason why we care about (P-Dual) lies in the utilization of the optimal conditions, which can provide valuable insights into the optimal solution β^* . Specifically, by applying the Karush-Kuhn-Tucker (KKT) optimal conditions, we can derive the following theorem:

Theorem 2. *Let β^* be an optimal solution to Problem (P-Primal), and $(\boldsymbol{\eta}^*, \zeta^*)$ be an optimal solution to Problem (P-Dual). Then we have:*

$$\begin{aligned} a_i (\boldsymbol{\eta}^{*\top} \mathbf{v}_i - \zeta^*) < \lambda s & \Rightarrow \beta_i^* = 0, \\ a_i (\boldsymbol{\eta}^{*\top} \mathbf{v}_i - \zeta^*) > \lambda s & \Rightarrow \beta_i^* = \frac{1}{s}. \end{aligned} \quad (11)$$

The following corollary concludes this section.

Corollary 1. *There exist $\boldsymbol{\theta} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ such that:*

$$\begin{aligned} a_i (\boldsymbol{\theta}^\top \mathbf{v}_i + b) < 1 & \Rightarrow \beta_i^* = 0, \\ a_i (\boldsymbol{\theta}^\top \mathbf{v}_i + b) > 1 & \Rightarrow \beta_i^* = \frac{1}{s}. \end{aligned} \quad (12)$$

Corollary 1 reveals the existence of a scoring function of the form:

$$s_i \triangleq a_i (\boldsymbol{\theta}^\top \mathbf{v}_i + b). \quad (13)$$

As illustrated in Figure 2(a), this scoring function can precisely identify the active and inactive variables in (P1). Tokens with low scores will be inactive in (P1). Therefore, these tokens can be safely evicted from the cache.

A2: *We can define a scoring function in the form of Equation (13).*

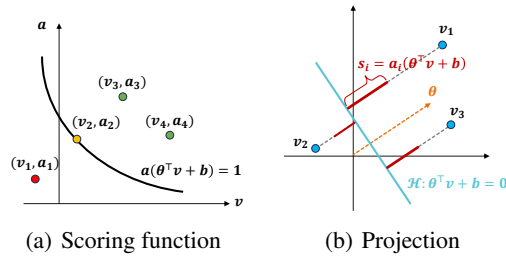


Figure 2: Illustration of the scoring function. (a) The scoring function s_i , as a function of both $\mathbf{v}_i$ and a_i , can distinguish between activate (green plots) and inactive (red plots) samples. (b) The computation of $s_i = a_i (\boldsymbol{\theta}^\top \mathbf{v}_i + b)$, which represents the signed distance of $\mathbf{v}_i$ to the hyperplane $\mathcal{H}: \boldsymbol{\theta}^\top \mathbf{v} + b = 0$, scaled by a_i .

Remark 2. In Theorem 2, we have shown that η^* and ζ^* correspond to the parameters θ and b that we are interested in. However, it is infeasible to directly derive η^* and ζ^* without explicitly solving Problem (P-Dual). Therefore, we turn to an intuitive analysis on θ and b to derive a more practical form in the subsequent section.

3.3 Anchor Direction Projection: Practical Form

The theoretical analysis in Section 3.2 motivates us to define a scoring function in the form of Equation 13. Despite the theoretical soundness, the determination of the parameters θ and b remains challenging, because the dual variables η^* and ζ^* are inaccessible before solving Problem (P-Dual). Therefore, this section aims to answer:

Q3: How can we determine generally good parameters θ and b in practice?

We begin by analyzing a special case when $\theta = \mathbf{0}$. In this case, the scoring function is simplified to $s_i = a_i b$, where b is a constant. Setting $b = 1$ gives us the most commonly used score, i.e., the attention weights. From this perspective, existing methods that relied solely on attention weights can be viewed as a special instance of our framework.

For the more general case when $\theta \neq \mathbf{0}$, without loss of generality, we can assume that $\|\theta\|_2 = 1$. With this assumption, the equation $\mathcal{H} : \theta^\top x + b = 0$ represents a hyperplane in $\mathbb{R}^n$, and $\theta^\top v_i + b$ represents the signed distance (a distance metric that can assume negative values) of the point v_i from $\mathcal{H}$. The factor a_i scales these signed distances. This computation is illustrated in Figure 2(b).

When applying this scoring function for top- k token selection, the tokens with relatively larger signed distances from $\mathcal{H}$, scaled by a_i , are retained, while the other tokens are discarded. This procedure essentially preserves the projection of the output $\hat{y}$ onto the direction of θ . We refer to the direction of θ as an “anchor direction”, as it acts as a reference to guide the selection of retained tokens. A natural and intuitive way is to set the direction of y as the anchor direction. This choice helps strengthen the semantic information along the direction of the pre-eviction output, ensuring that the most significant semantic features are preserved.

In this case, the proposed scoring function is defined as $s_i = a_i(\mathbf{y}^\top v_i + b)$, where b acts as a bias term to balance the relative importance of projections and attention weights. As $b \rightarrow \infty$, the scoring function gradually converges to the traditional attention-based score. The results suggest that a carefully chosen b leads to a slight performance gain, while setting $b = 0$ also results in stable performance. Therefore, we set $b = 0$ and thus $s_i = a_i \mathbf{y}^\top v_i$ in the main experiments. We conduct experiments on the choices of θ and b in Section 4.5.

A3: We can simply set $\theta = y$ and $b = 0$.

3.4 Implementation Details

In this section, we supplement some implementation details of AnDPro. Following SnapKV [16], we retain an observation window $\mathcal{W}$ to accumulate attention weights from the last several prompt tokens, which generates queries $\{q^t\}_{t \in \mathcal{W}}$. Specifically, We retain the tokens within this window as well as the first token. At each time step in the observation window, the attention for the i^{th} token and the corresponding pre-eviction output are computed as:

$$a_i^t = \text{Softmax}_{1 \leq i \leq t}(q^{t^\top} k_i) \quad \text{and} \quad \mathbf{y}^t = \sum_{i=1}^n a_i^t v_i. \quad (14)$$

The scoring function for each token is then computed as:

$$s_i \triangleq \sum_{t \in \mathcal{W}} s_i^t = \sum_{t \in \mathcal{W}} a_i^t \mathbf{y}^{t^\top} v_i. \quad (15)$$

We perform top- k selection across all attention heads within each layer, based on their scores. This allows for flexible budget allocation among heads, significantly enhancing generation quality. Additionally, we introduce a useful technique of merging tokens into chunks. Notice that both SnapKV and Ada-KV employ max/mean pooling operations, ensuring that adjacent tokens have similar scores. This leads to a successive selection of tokens, significantly improving the performance.

Table 1: **Comparison based on Mistral-7B-Instruct-v0.2 among 16 datasets from LongBench [53].** The results for H2O, Streaming, SnapKV, and Pyramid are from Feng et al. [17], and the results for Ada-KV and CriticalKV are reproduced based on their papers [17, 45]. We mark the best results in **bold red**, and the second best results in underlined blue.

	Single-Doc. QA			Multi-Doc. QA			Summarization			Few-shot Learning			Synthetic		Code		Ave. Score
	NrivQA	Qasper	MF-en	HopoutQA	2WikiMQA	Musique	GovReport	QMSum	MultiNews	TREC	TriviaQA	SAMSum	PCount	PRe	Lcc	RB-p	
Full Cache	26.63	32.99	49.34	42.77	27.35	18.77	32.87	24.24	27.10	71.00	86.23	42.96	2.75	86.98	55.33	52.87	42.51
Budget=128																	
H2O	21.19	21.66	38.60	30.63	20.65	12.19	<u>20.65</u>	22.42	21.81	39.00	82.52	40.68	2.98	79.56	49.13	46.76	34.40
Streaming	16.61	14.74	31.40	28.05	21.36	12.08	18.44	18.91	19.26	43.50	74.22	29.00	2.75	31.65	41.27	38.84	27.63
Snapkv	19.17	21.40	42.93	36.76	22.44	15.86	19.16	21.84	21.55	47.50	84.15	40.24	2.30	68.26	50.69	47.13	35.09
Pyramid	20.16	21.77	43.55	36.78	23.12	14.39	19.53	22.03	21.47	51.00	84.62	40.24	2.79	70.77	50.57	46.53	35.58
Ada-KV	<u>21.79</u>	23.03	<u>47.07</u>	38.70	22.85	15.92	19.94	23.05	21.84	63.00	85.36	40.01	<u>3.20</u>	72.13	51.36	49.29	37.41
CriticalKV	21.55	<u>23.27</u>	46.67	<u>38.95</u>	24.95	<u>16.85</u>	20.83	<u>23.14</u>	22.16	<u>65.00</u>	<u>85.53</u>	40.02	3.48	<u>77.81</u>	<u>53.58</u>	<u>50.98</u>	<u>38.42</u>
AnDPro	24.84	25.14	47.33	39.79	<u>23.99</u>	17.34	20.40	23.45	<u>21.94</u>	67.50	85.59	<u>40.62</u>	3.19	76.94	53.77	51.55	38.96
Budget=256																	
H2O	21.54	22.92	42.56	31.07	22.53	13.76	22.52	22.40	23.09	40.50	84.20	40.77	3.41	86.10	50.98	48.17	36.03
Streaming	17.93	16.01	33.36	30.71	21.30	10.08	20.66	19.47	22.89	53.50	73.59	29.22	3.00	27.77	42.30	39.87	28.85
SnapKV	22.37	23.74	48.13	38.56	22.43	15.66	21.91	23.13	23.15	61.50	85.45	41.42	3.09	84.54	53.22	50.24	38.66
Pyramid	20.09	24.00	47.33	38.24	22.48	16.02	21.40	22.45	22.63	63.00	84.93	40.98	<u>3.40</u>	82.48	52.78	49.36	38.22
Ada-KV	23.02	25.70	<u>49.23</u>	40.18	24.76	17.43	<u>22.17</u>	23.21	<u>23.48</u>	67.50	85.58	41.67	2.99	84.65	<u>55.07</u>	<u>52.17</u>	39.93
CriticalKV	<u>23.70</u>	<u>26.66</u>	49.00	<u>40.05</u>	<u>25.12</u>	<u>17.86</u>	21.99	<u>23.35</u>	23.45	<u>68.00</u>	<u>86.19</u>	<u>42.07</u>	2.92	<u>85.81</u>	<u>53.81</u>	50.42	<u>40.03</u>
AnDPro	25.71	28.17	49.31	40.04	25.81	18.68	21.66	23.64	23.63	70.50	86.59	42.60	2.86	85.62	55.74	52.87	40.84
Budget=512																	
H2O	21.72	26.03	44.81	32.33	23.16	14.86	23.65	22.84	24.70	42.00	85.22	41.57	3.40	86.45	53.04	49.68	37.22
Streaming	18.76	17.17	37.09	30.21	21.64	9.93	24.44	20.00	25.57	62.00	72.36	29.95	2.48	18.17	43.70	40.13	29.60
SnapKV	<u>24.60</u>	<u>27.81</u>	48.98	39.46	25.25	16.98	<u>23.70</u>	22.96	24.37	67.00	85.88	41.26	2.78	86.56	54.81	51.71	40.26
Pyramid	23.23	27.94	48.87	<u>40.50</u>	24.36	16.74	23.22	23.16	24.37	67.00	85.73	41.74	<u>3.16</u>	85.67	54.16	50.34	40.01
Ada-KV	23.88	<u>29.05</u>	48.79	40.44	25.30	17.66	23.47	23.47	24.48	69.50	<u>86.39</u>	41.38	2.94	87.56	<u>56.58</u>	<u>52.88</u>	40.86
CriticalKV	23.83	28.67	<u>49.13</u>	40.18	<u>26.18</u>	<u>18.19</u>	23.52	<u>23.57</u>	24.28	<u>70.00</u>	<u>86.39</u>	<u>42.16</u>	2.82	<u>87.55</u>	56.51	52.64	<u>40.98</u>
AnDPro	24.87	30.99	49.58	41.34	27.01	18.44	23.40	23.76	<u>25.01</u>	71.00	86.74	43.65	2.83	86.23	57.46	53.61	41.62
Budget=1024																	
H2O	23.90	28.62	46.46	37.03	24.74	15.04	25.30	23.11	25.92	46.00	85.93	41.80	3.24	86.57	54.46	51.01	38.70
Streaming	19.42	21.69	41.75	32.40	22.18	11.18	27.13	21.09	26.59	67.00	71.79	30.11	2.88	16.57	44.82	39.76	31.02
SnapKV	25.47	29.57	<u>49.33</u>	40.90	25.53	19.01	<u>25.94</u>	23.89	26.21	69.50	<u>86.48</u>	42.10	<u>2.98</u>	88.56	55.57	51.92	41.44
Pyramid	24.21	29.86	48.93	40.75	25.05	18.77	25.73	<u>24.06</u>	25.65	68.50	86.31	42.25	2.97	87.17	54.75	52.10	41.07
Ada-KV	25.60	31.15	49.02	41.47	<u>27.07</u>	18.77	25.40	23.84	25.92	70.50	86.55	43.07	2.54	87.14	57.27	53.85	41.82
CriticalKV	26.03	<u>31.50</u>	48.39	<u>41.73</u>	<u>27.10</u>	<u>18.84</u>	25.30	24.10	25.99	71.00	86.30	<u>43.16</u>	2.66	<u>87.23</u>	56.69	53.20	<u>41.83</u>
AnDPro	<u>25.78</u>	32.54	49.35	42.20	26.94	18.63	25.33	24.01	<u>26.29</u>	71.00	86.30	43.46	2.68	86.98	<u>57.18</u>	<u>53.77</u>	42.03

Inspired by this, we merge tokens into chunks, each containing a predefined number of tokens, to maintain semantic coherence. The chunk size is treated as a hyperparameter. The value state of a merged chunk is the weighted sum of the original tokens’ values, and its attention score is the sum of the individual attention weights. The pseudocode of AnDPro is provided in Appendix B.1.

4 Experiments

4.1 Experimental Setup

Datasets We conduct comprehensive experiments to demonstrate the effectiveness of our approach on two commonly-used benchmarks, Longbench [53] and Needle-in-a-Haystack [54]. Specifically, LongBench is a long-sequence benchmark containing 16 datasets covering multi-task domains. The Needle-in-a-Haystack dataset was specifically used to assess the context retrieval capabilities of different budget strategies, particularly in tasks involving long-context retrieval. More detailed information about these datasets are in Appendix B.2. Code is available at <https://github.com/MIRALab-USTC/LLM-AnDPro>.

Baselines We evaluate our approach using two open-source large language models: Mistral-7B-Instruct-v0.2 [55] and Llama-3.1-8B-Instruct [56]. We compare AnDPro against several strong baselines. These include representative prior works such as H2O [9], StreamingLLM [14], and SnapKV [16], as well as latest SOTA methods, including Ada-KV [17] and CriticalKV [45].

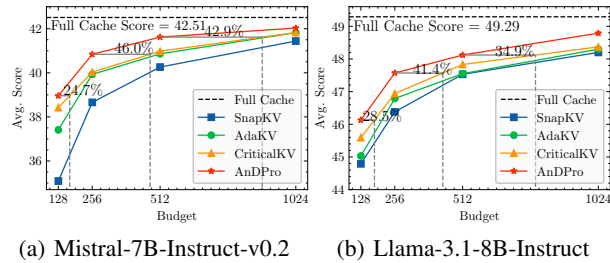


Figure 3: **Average scores on the LongBench dataset.** The numbers annotated on the gray lines represent the proportion of budget saved by our method compared to the current SOTA to achieve the same accuracy.

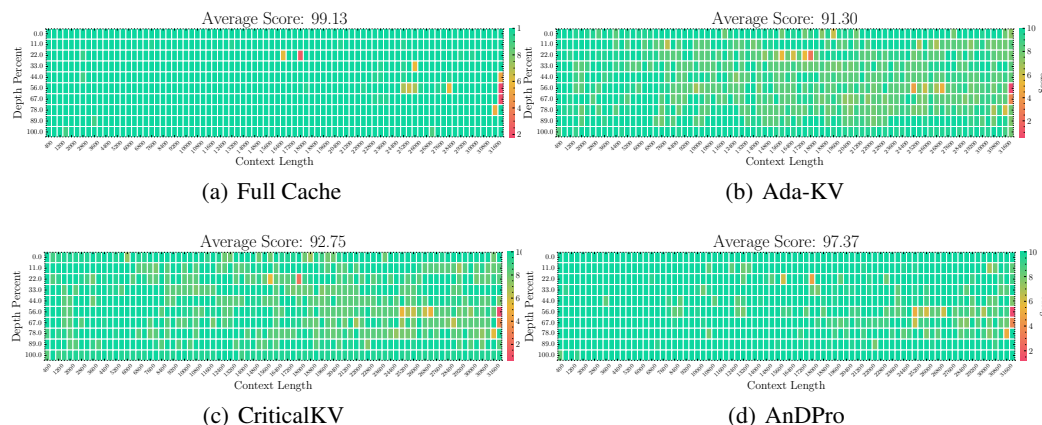


Figure 4: **Results of Mistral-7b-Instruct-v0.2 on the Needle-in-a-Haystack test.** The x-axis represents the document’s context length, and the y-axis indicates the depth of the needle insertion. Each square in the heatmap demonstrates the correctness of retrieving a proper sentence inserted in the context for a given context length and insertion depth. The Average Score is calculated by averaging the sum of score with different insert depth across various context lengths.

HyperParameters and Experimental Environment In our main experiments, we evaluate the performance of different methods across a range of budget sizes, specifically setting budgets to $B \in \{128, 256, 512, 1024\}$ to test their effectiveness under different memory constraints. All experiments are run on a single A800 – 80G GPU. In the main experiments, we set the observation window size to 32 and the chunk size to 4. Our implementation is built upon the Ada-KV framework, and the hyperparameters are aligned with those used in the baseline. More details can be found in Appendix B.3. We will release our code once the paper is accepted for publication.

4.2 Evaluations on LongBench Test

Table 1 presents the performance scores of various methods under different budget allocations for the Mistral model. Due to space limitations, results for the Llama model are provided in Appendix C.1. Results on the Chinese datasets in LongBench are in Appendix C.2. Overall, AnDPro consistently delivers state-of-the-art (SOTA) performance across both models and a range of budget configurations.

We further visualize the performance scores in Figure 3. Notably, for the Mistral model with a budget of 256 ($k = 256$), AnDPro achieves an average score of 40.84, which is 96.07% of the full cache accuracy, while utilizing only 3.44% (256/7425) of the total KV cache budget. In comparison, the previous SOTA method, CriticalKV, requires a budget of 474 to achieve similar performance. AnDPro achieves a 46.0% reduction in budget size compared with CriticalKV. Moreover, on multiple datasets, such as MF-en, SAMSum, and Lcc, AnDPro achieves comparable or even better results compared with full cache.

4.3 Evaluations on Needle-in-a-Haystack Test

We evaluate the long-context retrieval capability of different KV methods using the Needle-in-a-Haystack test. This test involves retrieving a specific statement (the “needle”) placed within a long context window (the “haystack”). In our experiment, we set the budget size to $k = 128$ and compare AnDPro against Ada-KV and CriticalKV. As shown in Figure 4, AnDPro significantly outperforms the baselines, achieving a score of 97.37, which is very close to the performance of the full cache. Additional results of other baselines are in Appendix C.3.

4.4 Memory & Latency

We evaluate the computational efficiency of various KV eviction strategies by measuring the peak memory usage and decoding latency across different context lengths, ranging from 8K to 256K.

We conduct the experiments on the Mistral-7b-Instruct-v0.2 model with a fixed budget size $k = 128$. To measure decoding latency, we use different methods to generate 1K tokens and compute the per-token decoding latency. The results are in Figure 5. The peak memory usage and decoding latency are roughly the same for these eviction methods, significantly optimized compared with the full cache model. This demonstrates that AnD-Pro achieves a similar memory and time efficiency with much better accuracy compared with previous SOTA eviction methods. Further analysis of the inference time composition of different methods is in Appendix C.4, demonstrating that the additional computation introduced by AnDPro—specifically the Update KV Phase—incurs negligible time overhead.

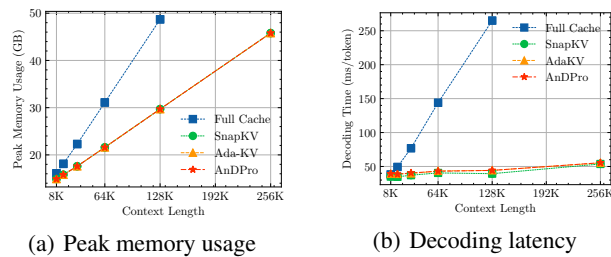


Figure 5: **Memory and latency of different methods.**

4.5 Analysis

Ablation Study We conduct ablation studies in Appendix C.5 to isolate the impact of our core contribution, i.e., the projection-based scoring function compared to attention-based scores. We then analyze the effects of three design choices in our algorithm: (1) merging tokens into chunks, (2) preserving the first token, and (3) cross-head budget allocation. The results confirm that the projection-based scoring function consistently outperforms its attention-based counterparts, and that each component contributes positively to performance in our framework.

Different Choices of θ and b We explore different choices of anchor direction θ and the bias term b . The results are in Appendix C.6, demonstrating that our setting $\theta = \mathbf{y}$ and $b = 0$ achieves a superior and robust performance.

Eviction Loss and Score Visualization We analyze the eviction loss between pre- and post-eviction outputs. Results in Appendix C.7 demonstrate that our method can effectively reduce this loss. Furthermore, we visualize the distribution of token scores in Appendix C.8.

Value Vector Analysis We investigate the role of value vectors in the eviction process. Using PCA, we project value vectors and model outputs into two-dimensional space. The results are in Appendix C.9. We also provide statistical histograms of value vector magnitudes in Figure 15 in Appendix C.9. These results help understand the reason why projection-based score works intuitively.

Case Study We provide some case studies in Appendix C.10, showing that AnDPro can more accurately preserve important tokens compared to attention-based scoring functions.

Longer context we evaluate performance on extremely long contexts ranging from 64k to 384k tokens. The results are in Appendix C.11, demonstrating that AnDPro achieves the best performance even on much longer contexts.

5 Conclusion

In this work, we model KV cache eviction as a combinatorial optimization problem, and relaxes it to a sparse optimization problem. Motivated by the theoretical and intuitive analysis, we propose **AnDPro** (**A**nchor **D**irection **P**rojection), which leverages projection-based scoring functions for more accurate KV cache eviction. Extensive experiments on LongBench and Needle-in-a-Haystack benchmarks demonstrate the effectiveness of our approach.

Acknowledgements

The authors would like to thank all the anonymous reviewers for their insightful comments. This work was supported in part by National Key R&D Program of China under contract 2022ZD0119801, National Nature Science Foundations of China grants U23A20388 and 62021001.

References

- [1] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.
- [2] Muhammad Usman Hadi, Rizwan Qureshi, Abbas Shah, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Jia Wu, Seyedali Mirjalili, et al. A survey on large language models: Applications, challenges, limitations, and practical usage. *Authorea Preprints*, 2023.
- [3] Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large language models: A survey. *arXiv preprint arXiv:2402.06196*, 2024.
- [4] Xize Liang, Chao Chen, Shuang Qiu, Jie Wang, Yue Wu, Zhihang Fu, Hanzhu Chen, Feng Wu, and Jieping Ye. ROPO: Robust preference optimization for large language models. In *Forty-second International Conference on Machine Learning*, 2025. URL <https://openreview.net/forum?id=5WEmyTooVV>.
- [5] Zehao Wang, Lin Yang, Jie Wang, Kehan Wang, Hanzhu Chen, Bin Wang, Jianye Hao, Defu Lian, Bin Li, and Enhong Chen. Logictree: Improving complex reasoning of LLMs via instantiated multi-step synthetic logical data. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.
- [6] 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), 2017.
- [7] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *ArXiv*, abs/1904.10509, 2019. URL <https://api.semanticscholar.org/CorpusID:129945531>.
- [8] Shi Luohe, Hongyi Zhang, Yao Yao, Zuchao Li, and hai zhao. Keep the cost down: A review on methods to optimize LLM’s KV-cache consumption. In *First Conference on Language Modeling*, 2024. URL <https://openreview.net/forum?id=8tKjqmM5z>.
- [9] Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36:34661–34710, 2023.
- [10] Zichang Liu, Aditya Desai, Fangshuo Liao, Weitao Wang, Victor Xie, Zhaozhuo Xu, Anastasios Kyrillidis, and Anshumali Shrivastava. Scissorhands: Exploiting the persistence of importance hypothesis for llm kv cache compression at test time. *Advances in Neural Information Processing Systems*, 36, 2024.
- [11] Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. Model tells you what to discard: Adaptive KV cache compression for LLMs. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=uNrFpDPMYo>.
- [12] Mirko Farina, Usman Ahmad, Ahmad Taha, Hussein Younes, Yusuf Mesbah, Xiao Yu, and Witold Pedrycz. Sparsity in transformers: A systematic literature review. *Neurocomputing*, page 127468, 2024.

- [13] Haoyang Li, Yiming Li, Anxin Tian, Tianhao Tang, Zhanchao Xu, Xuejia Chen, Nicole Hu, Wei Dong, Qing Li, and Lei Chen. A survey on large language model acceleration based on kv cache management. *arXiv preprint arXiv:2412.19442*, 2024.
- [14] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.
- [15] Qingyue Yang, Jie Wang, Xing Li, Zhihai Wang, Chen Chen, Lei Chen, Xianzhi Yu, Wulong Liu, Jianye Hao, Mingxuan Yuan, et al. Attentionpredictor: Temporal pattern matters for efficient llm inference. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.
- [16] Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. Snapkv: Llm knows what you are looking for before generation. *arXiv preprint arXiv:2404.14469*, 2024.
- [17] Yuan Feng, Junlin Lv, Yukun Cao, Xike Xie, and S Kevin Zhou. Ada-kv: Optimizing kv cache eviction by adaptive budget allocation for efficient llm inference. *arXiv preprint arXiv:2407.11550*, 2024.
- [18] Pengkun Wang, Zhe Zhao, HaiBin Wen, Fanfu Wang, Binwu Wang, Qingfu Zhang, and Yang Wang. Llm-autoda: Large language model-driven automatic data augmentation for long-tailed problems. *Advances in Neural Information Processing Systems*, 37:64915–64941, 2024.
- [19] Haoyang Liu, Jie Wang, Yuyang Cai, Xiongwei Han, Yufei Kuang, and Hao Jianye. Optitree: Hierarchical thoughts generation with tree search for LLM optimization modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.
- [20] Chi Han, Qifan Wang, Hao Peng, Wenhan Xiong, Yu Chen, Heng Ji, and Sinong Wang. Lm-infinite: Zero-shot extreme length generalization for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3991–4008, 2024.
- [21] Huiqiang Jiang, Yucheng Li, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H Abdi, Dongsheng Li, Chin-Yew Lin, et al. Minference 1.0: Accelerating pre-filling for long-context llms via dynamic sparse attention. *arXiv preprint arXiv:2407.02490*, 2024.
- [22] Muhammad Adnan, Akhil Arunkumar, Gaurav Jain, Prashant Nair, Ilya Soloveychik, and Purushotham Kamath. Keyformer: Kv cache reduction through key tokens selection for efficient generative inference. *Proceedings of Machine Learning and Systems*, 6:114–127, 2024.
- [23] Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.
- [24] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.
- [25] Zhihai Wang, Jie Wang, Jilai Pan, Xilin Xia, Huiling Zhen, Mingxuan Yuan, Jianye Hao, and Feng Wu. Accelerating large language model reasoning via speculative search. In *Forty-second International Conference on Machine Learning*, 2025. URL <https://openreview.net/forum?id=oq0t5BX1lT>.
- [26] Jang-Hyun Kim, Junyoung Yeom, Sangdoo Yun, and Hyun Oh Song. Compressed context memory for online language model interaction. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=64kSvC4iPg>.
- [27] Zhongwei Wan, Xinjian Wu, Yu Zhang, Yi Xin, Chaofan Tao, Zhihong Zhu, Xin Wang, Siqi Luo, Jing Xiong, and Mi Zhang. D2o: Dynamic discriminative operations for efficient generative inference of large language models. *CoRR*, abs/2406.13035, 2024. URL <https://doi.org/10.48550/arXiv.2406.13035>.

- [28] Akide Liu, Jing Liu, Zizheng Pan, Yefei He, Gholamreza Haffari, and Bohan Zhuang. Mini-cache: Kv cache compression in depth dimension for large language models. *arXiv preprint arXiv:2405.14366*, 2024.
- [29] Yifei Yang, Zouying Cao, Qiguang Chen, Libo Qin, Dongjie Yang, Hai Zhao, and Zhi Chen. Kvsharer: Efficient inference via layer-wise dissimilar kv cache sharing. *arXiv preprint arXiv:2410.18517*, 2024.
- [30] Jian Yuan, Ziwei He, Haoli Bai, Jingwen Leng, and Bo Jiang. Weightedkv: Attention scores weighted key-value cache merging for large language models. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2025.
- [31] Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in Neural Information Processing Systems*, 35:27168–27183, 2022.
- [32] Hailin Zhang, Xiaodong Ji, Yilin Chen, Fangcheng Fu, Xupeng Miao, Xiaonan Nie, Weipeng Chen, and Bin Cui. Pqcache: Product quantization-based kvcache for long context llm inference. *CoRR*, 2024.
- [33] Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. Kvquant: Towards 10 million context length llm inference with kv cache quantization. *arXiv preprint arXiv:2401.18079*, 2024.
- [34] Haojie Duanmu, Zhihang Yuan, Xiuhong Li, Jiangfei Duan, Xingcheng Zhang, and Dahua Lin. Skvq: Sliding-window key and value cache quantization for large language models. *arXiv preprint arXiv:2405.06219*, 2024.
- [35] Mengzhao Chen, Yi Liu, Jiahao Wang, Yi Bin, Wenqi Shao, and Ping Luo. Prefixquant: Static quantization beats dynamic through prefixed outliers in llms. *arXiv preprint arXiv:2410.05265*, 2024.
- [36] Yujun Lin, Haotian Tang, Shang Yang, Zhekai Zhang, Guangxuan Xiao, Chuang Gan, and Song Han. Qserve: W4a8kv4 quantization and system co-design for efficient llm serving. *CoRR*, abs/2405.04532, 2024. URL <https://doi.org/10.48550/arXiv.2405.04532>.
- [37] Haocheng Xi, Changhao Li, Jianfei Chen, and Jun Zhu. Training transformers with 4-bit integers. *Advances in Neural Information Processing Systems*, 36:49146–49168, 2023.
- [38] Hao Yu, Zelan Yang, Shen Li, Yong Li, and Jianxin Wu. Effectively compress kv heads for llm. *arXiv preprint arXiv:2406.07056*, 2024.
- [39] Hanshi Sun, Li-Wen Chang, Wenlei Bao, Size Zheng, Ningxin Zheng, Xin Liu, Harry Dong, Yuejie Chi, and Beidi Chen. Shadowkv: Kv cache in shadows for high-throughput long-context llm inference. *arXiv preprint arXiv:2410.21465*, 2024.
- [40] Peiyu Liu, Ze-Feng Gao, Wayne Xin Zhao, Yipeng Ma, Tao Wang, and Ji-Rong Wen. Unlocking data-free low-bit quantization with matrix decomposition for kv cache compression. *arXiv preprint arXiv:2405.12591*, 2024.
- [41] Harry Dong, Xinyu Yang, Zhenyu Zhang, Zhangyang Wang, Yuejie Chi, and Beidi Chen. Get more with LESS: Synthesizing recurrence with KV cache compression for efficient LLM inference. In *Forty-first International Conference on Machine Learning*, 2024. URL <https://openreview.net/forum?id=uHdHVKFMW>.
- [42] Junqi Zhao, Zhijin Fang, Shu Li, Shaohui Yang, and Shichao He. Buzz: Beehive-structured sparse kv cache with segmented heavy hitters for efficient llm inference. *arXiv preprint arXiv:2410.23079*, 2024.
- [43] Yilong Chen, Guoxia Wang, Junyuan Shang, Shiyao Cui, Zhenyu Zhang, Tingwen Liu, Shuo-huan Wang, Yu Sun, Dianhai Yu, and Hua Wu. Nacl: A general and effective kv cache eviction framework for llm at inference time. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7913–7926, 2024.

- [44] Insu Han, Michael Kapralov, Ekaterina Kochetkova, Kshiteej Sheth, and Amir Zandieh. Balancekv: Kv cache compression through discrepancy theory. *arXiv preprint arXiv:2502.07861*, 2025.
- [45] Yuan Feng, Junlin Lv, Yukun Cao, Xike Xie, and S Kevin Zhou. Identify critical KV cache in LLM inference from an output perturbation perspective, 2025. URL <https://openreview.net/forum?id=1RTDMGYCpy>.
- [46] Dongjie Yang, XiaoDong Han, Yan Gao, Yao Hu, Shilin Zhang, and Hai Zhao. Pyramid-infer: Pyramid kv cache compression for high-throughput llm inference. *arXiv preprint arXiv:2405.12532*, 2024.
- [47] Xiabin Zhou, Wenbin Wang, Minyan Zeng, Jiaxian Guo, Xuebo Liu, Li Shen, Min Zhang, and Liang Ding. Dynamickv: Task-aware adaptive kv cache compression for long context llms. *arXiv preprint arXiv:2412.14838*, 2024.
- [48] Yu Fu, Zefan Cai, Abedelkadir Asi, Wayne Xiong, Yue Dong, and Wen Xiao. Not all heads matter: A head-level kv cache compression method with integrated retrieval and reasoning. *arXiv preprint arXiv:2410.19258*, 2024.
- [49] Guangxuan Xiao, Jiaming Tang, Jingwei Zuo, Junxian Guo, Shang Yang, Haotian Tang, Yao Fu, and Song Han. Duoattention: Efficient long-context llm inference with retrieval and streaming heads. *arXiv preprint arXiv:2410.10819*, 2024.
- [50] Jonas Rånstam and Jonathan A Cook. Lasso regression. *Journal of British Surgery*, 105(10): 1348–1348, 2018.
- [51] Jie Wang, Jiayu Zhou, Peter Wonka, and Jieping Ye. Lasso screening rules via dual polytope projection. *Advances in neural information processing systems*, 26, 2013.
- [52] Stephen Boyd. Convex optimization. *Cambridge UP*, 2004.
- [53] Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*, 2023.
- [54] Greg Kamradt. Needle in a haystack-pressure testing llms. *Github Repository*, page 28, 2023.
- [55] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
- [56] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

A Proofs

Proposition 1. Let $S \subset [n]$ be the set of retained tokens, and define the indicator function

$$z_i \triangleq \begin{cases} 1, & i \in S, \\ 0, & i \notin S \end{cases} \quad (5)$$

to represent whether the i^{th} token is retained. Then, the post-eviction output can be expressed as:

$$\hat{\mathbf{y}} = \frac{\sum_{i=1}^n z_i a_i \mathbf{v}_i}{\sum_{i=1}^n z_i a_i}. \quad (6)$$

Proof. We have

$$\begin{aligned} \hat{\mathbf{y}} &= \sum_{i \in S} \text{Softmax}_{i \in S}(\mathbf{q}_n^\top \mathbf{k}_i) \mathbf{v}_i = \sum_{i \in S} \frac{\exp(\mathbf{q}_n^\top \mathbf{k}_i)}{\sum_{j \in S} \exp(\mathbf{q}_n^\top \mathbf{k}_j)} \mathbf{v}_i \\ &= \frac{\sum_{i=1}^n z_i \frac{\exp(\mathbf{q}_n^\top \mathbf{k}_i)}{\sum_{k=1}^n \exp(\mathbf{q}_n^\top \mathbf{k}_k)} \mathbf{v}_i}{\sum_{j=1}^n z_j \frac{\exp(\mathbf{q}_n^\top \mathbf{k}_j)}{\sum_{k=1}^n \exp(\mathbf{q}_n^\top \mathbf{k}_k)}} = \frac{\sum_{i=1}^n z_i a_i \mathbf{v}_i}{\sum_{i=1}^n z_i a_i}. \end{aligned} \quad (16)$$

□

Theorem 1. The dual problem to Problem (P-Primal) is given by:

$$\sup_{\boldsymbol{\eta}, \zeta} -\frac{1}{2} \|\boldsymbol{\eta}\|^2 + \boldsymbol{\eta}^\top \mathbf{y} - \zeta + \frac{1}{s} \sum_{i=1}^n \rho_i(\boldsymbol{\eta}, \zeta), \quad (\text{P-Dual})$$

where $\boldsymbol{\eta}^* \in \mathbb{R}^n$ and $\zeta \in \mathbb{R}$ are dual variables, and

$$\rho_i(\boldsymbol{\eta}, \zeta) \triangleq \min(0, \lambda s - a_i (\boldsymbol{\eta}^\top \mathbf{v}_i - \zeta)). \quad (10)$$

Proof. Recall that Problem (P-Primal) takes the form of:

$$\begin{aligned} \min_{\boldsymbol{\beta} \in [0, \frac{1}{s}]^n} \quad & \frac{1}{2} \|\mathbf{u}\|_2^2 + \lambda s \sum_{i=1}^n \beta_i \\ \text{s.t.} \quad & \mathbf{u} = \mathbf{y} - \sum_{i=1}^n \beta_i a_i \mathbf{v}_i \\ & \sum_{i=1}^n \beta_i a_i = 1 \end{aligned}$$

By introducing the new variables $\boldsymbol{\eta} \in \mathbb{R}^n$ and $\zeta \in \mathbb{R}$, we get its Lagrangian:

$$L(\boldsymbol{\beta}, \mathbf{u}, \boldsymbol{\eta}, \zeta) = \frac{1}{2} \|\mathbf{u}\|_2^2 + \lambda s \sum_{i=1}^n \beta_i + \boldsymbol{\eta}^\top \left(\mathbf{y} - \sum_{i=1}^n \beta_i a_i \mathbf{v}_i - \mathbf{u} \right) + \zeta \left(\sum_{i=1}^n \beta_i a_i - 1 \right). \quad (17)$$

The primal variables are $\boldsymbol{\beta} \in [0, \frac{1}{s}]^n$ and $\mathbf{u} \in \mathbb{R}^n$. The dual function $g(\boldsymbol{\eta}, \zeta)$ is then:

$$\begin{aligned} g(\boldsymbol{\eta}, \zeta) &= \inf_{\boldsymbol{\beta}, \mathbf{u}} L(\boldsymbol{\beta}, \mathbf{u}, \boldsymbol{\eta}, \zeta) \\ &= \boldsymbol{\eta}^\top \mathbf{y} - \zeta + \inf_{\mathbf{u}} \left(\frac{1}{2} \|\mathbf{u}\|^2 - \boldsymbol{\eta}^\top \mathbf{u} \right) + \inf_{\boldsymbol{\beta} \in [0, \frac{1}{s}]^n} \left(\lambda s \sum_{i=1}^n \beta_i - \sum_{i=1}^n \beta_i a_i \boldsymbol{\eta}^\top \mathbf{v}_i + \zeta \sum_{i=1}^n \beta_i a_i \right) \\ &= \boldsymbol{\eta}^\top \mathbf{y} - \zeta + \inf_{\mathbf{u}} \left(\frac{1}{2} \|\mathbf{u}\|^2 - \boldsymbol{\eta}^\top \mathbf{u} \right) + \sum_{i=1}^n \inf_{\beta_i \in [0, \frac{1}{s}]} \beta_i (\lambda s - a_i (\boldsymbol{\eta}^\top \mathbf{v}_i - \zeta)). \end{aligned} \quad (18)$$

It is easy to derive that

$$\inf_{\mathbf{u}} \left(\frac{1}{2} \|\mathbf{u}\|^2 - \boldsymbol{\eta}^\top \mathbf{u} \right) = -\frac{1}{2} \|\boldsymbol{\eta}\|^2, \quad (19)$$

with the optimal solution $\mathbf{u}^* = \boldsymbol{\eta}$, and

$$\inf_{\beta_i \in [0, \frac{1}{s}]} \beta_i (\lambda s - a_i (\boldsymbol{\eta}^\top \mathbf{v}_i - \zeta)) = \frac{1}{s} \min (0, \lambda s - a_i (\boldsymbol{\eta}^\top \mathbf{v}_i - \zeta)). \quad (20)$$

with the optimal solutions

$$\beta_i^* = \begin{cases} 0, & \lambda s - a_i (\boldsymbol{\eta}^\top \mathbf{v}_i - \zeta) > 0, \\ \in \left[0, \frac{1}{s}\right], & \lambda s - a_i (\boldsymbol{\eta}^\top \mathbf{v}_i - \zeta) = 0, \\ \frac{1}{s}, & \lambda s - a_i (\boldsymbol{\eta}^\top \mathbf{v}_i - \zeta) < 0. \end{cases} \quad (21)$$

Therefore, we have:

$$g(\boldsymbol{\eta}, \zeta) = -\frac{1}{2} \|\boldsymbol{\eta}\|^2 + \boldsymbol{\eta}^\top \mathbf{y} - \zeta + \frac{1}{s} \sum_{i=1}^n \min (0, \lambda s - a_i (\boldsymbol{\eta}^\top \mathbf{v}_i - \zeta)). \quad (22)$$

Combining the above, we get the dual problem:

$$\sup_{\boldsymbol{\eta}, \zeta} g(\boldsymbol{\eta}, \zeta) = -\frac{1}{2} \|\boldsymbol{\eta}\|^2 + \boldsymbol{\eta}^\top \mathbf{y} - \zeta + \frac{1}{s} \sum_{i=1}^n \min (0, \lambda s - a_i (\boldsymbol{\eta}^\top \mathbf{v}_i - \zeta)). \quad (23)$$

□

Theorem 2. Let β^* be an optimal solution to Problem (P-Primal), and $(\boldsymbol{\eta}^*, \zeta^*)$ be an optimal solution to Problem (P-Dual). Then we have:

$$\begin{aligned} a_i (\boldsymbol{\eta}^{*\top} \mathbf{v}_i - \zeta^*) < \lambda s & \Rightarrow \beta_i^* = 0, \\ a_i (\boldsymbol{\eta}^{*\top} \mathbf{v}_i - \zeta^*) > \lambda s & \Rightarrow \beta_i^* = \frac{1}{s}. \end{aligned} \quad (11)$$

Proof. By the strong duality, it is easy to show that $(\boldsymbol{\eta}^*, \zeta^*)$ is a geometric multiplier. Therefore, from the optimal condition, we have

$$(\beta^*, \mathbf{u}^*) \in \inf_{\beta, \mathbf{u}} L(\beta, \mathbf{u}, \boldsymbol{\eta}^*, \zeta^*). \quad (24)$$

We then complete the proof according to Equation 21. □

Corollary 1. There exist $\boldsymbol{\theta} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ such that:

$$\begin{aligned} a_i (\boldsymbol{\theta}^\top \mathbf{v}_i + b) < 1 & \Rightarrow \beta_i^* = 0, \\ a_i (\boldsymbol{\theta}^\top \mathbf{v}_i + b) > 1 & \Rightarrow \beta_i^* = \frac{1}{s}. \end{aligned} \quad (12)$$

Proof. We can take

$$\boldsymbol{\theta} \triangleq \frac{\boldsymbol{\eta}^*}{\lambda s}, \quad b \triangleq -\frac{\zeta^*}{\lambda s}, \quad (25)$$

where $\boldsymbol{\eta}^*$ and ζ^* are the optimal solution to Problem (P-Dual). This completes the proof. □

B Implementation Details

B.1 Algorithm

A simplified pseudocode of AnDPro is provided in Algorithm 1. We omit some unimportant details (such as preserving the first token) for simplicity.

Algorithm 1 AnDPro Algorithm in One Layer

Require: A set H of all heads, A set $\mathcal{W}$ of the observation window, Number of tokens L outside the observation window, Queries $\{\mathbf{q}^{h,t}\}_{h \in H, t \in \mathcal{W}}$, Keys $\{\mathbf{k}_i^h\}_{h \in H, i \in [L]}$, Values $\{\mathbf{v}_i^h\}_{h \in H, i \in [L]}$, Chunk size C , Budget size B for each head

Ensure: Retained KV cache $\{S^h\}_{h \in H}$

- 1: $\mathbf{a}_i^{h,t} \mid_{h \in H, t \in \mathcal{W}, i \in [L]} \leftarrow \text{Softmax}_{i \in [L]}(\mathbf{q}^{h,t} \mathbf{v}_i^h)$
 - 2: $\hat{\mathbf{v}}_i^{h,t} \mid_{h \in H, t \in \mathcal{W}, i \in [L/C]} \leftarrow \sum_{j \in \text{Chunk}_i} \mathbf{a}_j^{h,t} \mathbf{v}_j^h$
 - 3: $\hat{\mathbf{a}}_i^{h,t} \mid_{h \in H, t \in \mathcal{W}, i \in [L/C]} \leftarrow \sum_{j \in \text{Chunk}_i} \mathbf{a}_j^{h,t}$
 - 4: $\hat{\mathcal{W}} \leftarrow$ observation window of chunks
 - 5: $\mathbf{y}^{h,t} \mid_{h \in H, t \in \hat{\mathcal{W}}} \leftarrow \sum_{i \in [L/C]} \hat{\mathbf{v}}_i^{h,t}$
 - 6: $\mathbf{s}_i^{h,t} \mid_{h \in H, t \in \hat{\mathcal{W}}, i \in [L/C]} \leftarrow \langle \mathbf{y}^{h,t}, \hat{\mathbf{v}}_i^{h,t} \rangle$
 - 7: $\mathbf{s}_i^h \mid_{h \in H, i \in [L/C]} \leftarrow \sum_{t \in \hat{\mathcal{W}}} \mathbf{s}_i^{h,t}$
 - 8: $\mathbf{s} \leftarrow \text{Concat}_{h \in H, i \in [L/C]}(\mathbf{s}_i^h)$
 - 9: $\{S^h\}_{h \in H} \leftarrow \text{TopK}(\mathbf{s}, k = |H| \times B)$
 - 10: $S^h \mid_{h \in H} \leftarrow S^h \cup \mathcal{W}$
 - 11: **return** $\{S^h\}_{h \in H}$
-

B.2 Datasets

Longbench is a comprehensive benchmark that consists of 16 datasets and serves as a robust evaluation tool due to its wide variety of prompts with different domains, types, and lengths. It is designed for long-sequence tasks spanning multiple domains, including single-document QA, multi-document QA, summarization, few-shot learning, synthetic tasks, and code generation. Table 2 provides detailed information of the 16 datasets in LongBench. These datasets have average input lengths ranging from 1, 235 to 18, 409 tokens, with an average token length of 7, 425. For our evaluation, we followed the recommended evaluation procedures for each dataset, with scores ranging from 0 to 100.

Table 2: Details of 16 datasets from the LongBench benchmark.

Task	Task Type	Eval Metric	Avg Len	Language	Sample Num
NarrativeQA	Single-Doc. QA	F1	18,409	EN	200
Qasper	Single-Doc. QA	F1	3,619	EN	200
MultiFieldQA-en	Single-Doc. QA	F1	4,559	EN	150
HotpotQA	Multi-Doc. QA	F1	9,151	EN	200
2WikiMultihopQA	Multi-Doc. QA	F1	4,887	EN	200
MuSiQue	Multi-Doc. QA	F1	11,214	EN	200
GovReport	Summarization	Rouge-L	8,734	EN	200
QMSum	Summarization	Rouge-L	10,614	EN	200
MultiNews	Summarization	Rouge-L	2,113	EN	200
TREC	Few-shot Learning	Accuracy	5,177	EN	200
TriviaQA	Few-shot Learning	F1	8,209	EN	200
SAMSum	Few-shot Learning	Rouge-L	6,258	EN	200
PassageCount	Synthetic	Accuracy	11,141	EN	200
PassageRetrieval-en	Synthetic	Accuracy	9,289	EN	200
LCC	Code Edit	Sim	1,235	Python/C#/Java	500
RepoBench-P	Code Edit	Sim	4,206	Python/Java	500

B.3 Code and Hyperparameter

Our code framework is adapted from Ada-KV [17], specifically leveraging its concise and user-friendly custom classes to facilitate more efficient and convenient budget filtering and allocation. The primary updates include a cross-head filtering strategy. We employ max pooling with a kernel size of 7 and set the size of observation window to 32. The code is released at <https://github.com/MIRALab-USTC/LLM-AnDPro>.

C Additional Results

C.1 Main results of Llama on LongBench

Table 3 shows the scores of different methods based on the Llama model in 16 datasets. Overall, the results are consistent with those of Mistral, and AnDPro also leads to improved quality after cache eviction.

Table 3: **Comparison based on Llama-3.1-8B-Instruct among 16 datasets from LongBench.** We run all the baselines to derive the results based on their released code. We mark the best results in **bold red**, and we mark the second best results in underlined blue.

	Single-Doc. QA			Multi-Doc. QA			Summarization			Few-shotLearning			Synthetic		Code		Ave. Score
	NrivQA	Qasper	MF-en	HopoorQA	2WikiMQA	Musique	GovReport	QMSum	MultiNews	TREC	TriviaQA	SAMSum	PCount	Pre	Lcc	RB-P	
Full Cache	30.12	46.60	56.41	58.10	49.01	32.52	34.04	25.37	27.06	73.00	91.90	43.46	7.06	100.00	62.13	51.80	49.29
Budget=128																	
SnapKV	25.54	31.40	50.99	55.17	45.50	27.88	20.77	23.41	21.05	47.50	90.46	40.76	8.00	99.50	56.88	46.50	43.21
PyramidKV	27.83	32.98	51.45	56.72	44.67	30.54	<u>21.89</u>	22.90	21.67	63.00	90.45	40.06	8.00	100.00	55.37	43.79	44.46
Ada-KV	25.06	34.50	52.27	56.29	47.57	29.06	21.17	23.36	21.51	62.00	<u>92.17</u>	40.84	8.00	99.50	58.45	48.72	45.03
CriticalKV	26.56	<u>35.71</u>	<u>52.90</u>	<u>57.21</u>	47.53	<u>29.57</u>	21.77	<u>23.64</u>	<u>21.79</u>	<u>63.50</u>	91.44	41.62	7.75	100.00	<u>59.09</u>	<u>49.32</u>	<u>45.59</u>
AnDPro	<u>26.66</u>	38.22	54.54	57.47	47.57	29.42	22.09	23.67	21.81	67.00	92.35	<u>40.90</u>	7.70	100.00	59.15	50.06	46.13
Budget=256																	
SnapKV	26.92	38.19	52.62	57.03	46.75	29.63	23.07	23.92	22.82	58.50	91.83	40.81	<u>7.75</u>	99.50	59.95	48.63	45.50
PyramidKV	28.46	39.36	54.30	56.77	45.50	30.74	23.84	23.88	22.91	69.00	91.09	40.74	7.88	99.50	56.94	45.18	46.01
Ada-KV	27.04	39.24	<u>54.41</u>	<u>57.36</u>	47.53	<u>31.35</u>	23.06	23.81	<u>23.43</u>	69.00	92.50	41.23	7.67	100.00	61.45	<u>49.52</u>	46.79
CriticalKV	26.09	<u>40.60</u>	<u>54.23</u>	57.12	47.21	<u>30.72</u>	<u>23.68</u>	24.37	<u>23.34</u>	<u>70.00</u>	92.75	42.11	7.67	100.00	<u>61.70</u>	49.46	<u>46.94</u>
AnDPro	<u>27.28</u>	42.09	54.72	57.98	<u>47.52</u>	31.98	23.57	<u>24.27</u>	23.58	72.00	<u>92.54</u>	<u>41.55</u>	7.67	100.00	62.57	51.77	47.57
Budget=512																	
SnapKV	<u>28.40</u>	41.52	55.40	57.66	47.48	30.95	24.92	<u>24.46</u>	24.54	68.50	<u>92.33</u>	41.98	7.33	99.50	61.25	50.39	47.29
PyramidKV	29.15	43.04	55.55	57.71	47.67	31.27	26.05	24.08	24.54	70.50	92.67	41.52	7.33	100.00	59.96	47.51	47.41
Ada-KV	26.92	42.68	55.20	57.47	48.06	31.27	24.95	24.17	24.40	71.00	92.14	42.35	7.33	100.00	<u>62.26</u>	50.62	47.55
CriticalKV	26.75	<u>43.75</u>	<u>55.45</u>	<u>58.25</u>	<u>48.11</u>	32.03	<u>24.96</u>	24.44	<u>24.50</u>	<u>72.50</u>	92.14	41.92	7.12	100.00	62.25	<u>51.17</u>	<u>47.83</u>
AnDPro	28.03	44.25	55.61	58.29	48.30	<u>31.99</u>	24.86	24.55	24.83	73.00	91.97	<u>42.08</u>	7.33	100.00	62.68	52.12	48.12
Budget=1024																	
SnapKV	28.32	45.37	56.81	<u>58.16</u>	48.28	31.89	26.91	24.58	25.82	70.50	<u>92.22</u>	<u>43.08</u>	6.96	100.00	62.46	51.50	48.30
PyramidKV	<u>29.32</u>	44.57	56.14	58.39	48.36	31.93	<u>28.20</u>	23.78	26.00	70.50	92.41	42.46	6.88	100.00	61.33	49.12	48.09
Ada-KV	28.52	46.01	55.72	57.85	48.10	32.02	26.98	<u>24.72</u>	<u>25.96</u>	72.00	91.97	42.55	<u>6.96</u>	100.00	61.98	<u>51.52</u>	48.30
CriticalKV	28.15	<u>46.19</u>	56.00	57.99	<u>48.47</u>	32.68	<u>27.20</u>	24.69	25.74	<u>72.50</u>	91.97	42.33	6.92	100.00	61.79	51.36	<u>48.37</u>
AnDPro	29.40	46.83	<u>56.72</u>	58.01	48.68	<u>32.64</u>	26.69	24.96	25.92	73.00	91.97	43.55	7.38	100.00	<u>62.43</u>	52.39	48.79

C.2 Results on the Chinese Datasets in LongBench

Table 4 shows the scores of different methods based on the Llama model on 5 Chinese datasets from LongBench. Although Llama model don't support Chinese language, AnDPro leads to improved quality after cache eviction.

Table 4: Comparison based on Llama-3.1-8B-Instruct on 5 Chinese datasets from LongBench. The results demonstrate that AnDPro still achieves the best overall performance on Chinese datasets.

	MF-zh	dureader	vcsum	lsht	Pre-zh	Avg.
Full Cache	58.69	33.78	17.32	46.00	97.67	50.69
Budget=128						
CriticalKV	42.57	24.32	<u>12.51</u>	<u>29.50</u>	93.75	40.53
Ada-KV	<u>52.32</u>	<u>24.33</u>	12.15	25.75	<u>97.25</u>	<u>42.36</u>
AnDPro	55.40	24.77	13.04	39.50	97.75	46.09
Budget=256						
CriticalKV	48.12	24.74	<u>13.52</u>	<u>24.00</u>	94.00	40.88
Ada-KV	<u>54.86</u>	<u>24.88</u>	13.40	23.52	<u>98.00</u>	42.93
AnDPro	56.84	25.71	13.92	42.50	98.50	47.49
Budget=512						
CriticalKV	55.51	<u>26.21</u>	14.47	<u>29.25</u>	79.66	41.02
Ada-KV	<u>57.83</u>	25.75	14.82	22.50	<u>97.67</u>	<u>43.71</u>
AnDPro	57.99	27.63	<u>14.65</u>	45.00	98.50	48.75
Budget=1024						
CriticalKV	57.07	<u>27.40</u>	<u>15.47</u>	<u>23.00</u>	91.50	42.89
Ada-KV	<u>58.54</u>	27.19	<u>15.47</u>	22.00	<u>98.00</u>	<u>44.24</u>
AnDPro	58.57	29.07	15.85	45.50	98.17	49.43

C.3 Results of Needle-in-a-Haystack test

Figure 6 presents the Needle-in-a-Haystack (NIAH) scores across all methods. AnDPro significantly outperforms all baselines, achieving a high score of 97.37, which is very close to the performance of the full cache.

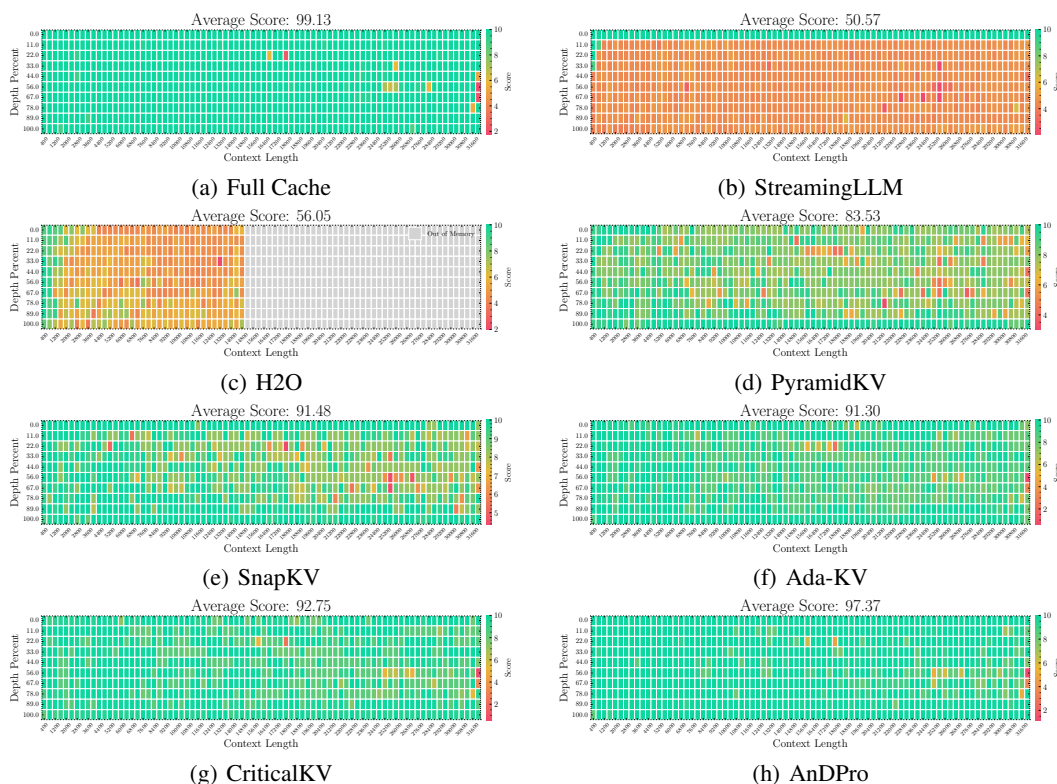


Figure 6: **Results of Mistral-7b-Instruct-v0.2 on the Needle-in-a-Haystack test.** The x-axis represents the document's context length, and the y-axis indicates the depth of the needle insertion. Each square in the heatmap demonstrates the correctness of retrieving a proper sentence inserted in the context for a given context length and insertion depth. The Average Score is calculated by averaging the sum of score with different insert depth across various context lengths.

C.4 Runtime Analysis

We visualize The detailed comparison of time consumption across different phases in the inference process in Figure 7. The results demonstrate that the additional computation introduced by AnDPro—specifically the Update KV Phase—incurs negligible time overhead.

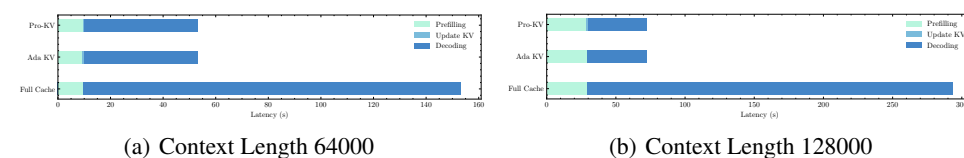


Figure 7: The detailed comparison of time consumption across different phases in the inference process under varying input lengths. The figure demonstrates the time costs of three key phases when processing input sequences of 64,000 and 128,000 tokens with an output length of 1,000 tokens.

To provide a more detailed quantitative breakdown, the specific runtime measurements are presented below (unit: seconds/milliseconds):

Table 5: Runtime analysis of different methods on sequences of length 64K and 128K.

64K	Prefilling (s)	KV update (s)	Layer-wise KV update (ms)
Full	9.45	-	-
SnapKV	9.50	0.10	3.05
Ada-KV	9.88	0.43	13.28
AnDPro	10.04	0.58	18.13
128K	Prefilling (s)	KV update (s)	Layer-wise KV update (ms)
Full	28.78	-	-
SnapKV	29.00	0.11	3.42
Ada-KV	29.47	0.59	18.31
AnDPro	29.78	0.88	27.41

Although AnDPro introduces slightly higher update latency compared to SnapKV and AdaKV, the additional overhead is negligible relative to the total prefilling time.

We further conduct experiments using the Mistral-7B-Instruct-v0.2 model on the Needle-in-a-Haystack (NIAH) benchmark. We measure the decoding latency (in seconds) for an output length of 1K under various input lengths (from 8K to 256K tokens) with a budget size of 128. The results are in Table 6.

Table 6: AnDPro maintains comparable runtime efficiency to AdaKV and remains close to SnapKV across all input lengths.

Input Length	8K	16K	32K	64K	128K	256K
SnapKV	35.10	34.72	37.15	40.49	39.40	53.69
AdaKV	39.22	38.99	39.64	43.39	44.23	55.30
AnDPro	39.23	38.67	40.44	42.98	44.20	55.51

C.5 Ablation Study

C.5.1 Ablation Study on Projection-Based Scoring Function

We conduct comprehensive ablation studies to isolate the effectiveness of the projection-based scoring function—which is the core contribution of this paper—and distinguish it from the other components in our method. Specifically, we conduct 5 sets of experiments to compare the projection-based scoring function (ProjScore) with the attention-based scoring function (AttnScore) with different combinations of other components. Specifically, we consider five experimental groups A–E:

- **A:** CrossHead + Chunk + FirstToken with ProjScore vs. AttnScore
- **B:** Chunk + FirstToken with ProjScore vs. Pooling + FirstToken with AttnScore
- **C:** CrossHead + FirstToken with ProjScore vs. AttnScore
- **D:** CrossHead + Chunk with ProjScore vs. CrossHead + Pooling with AttnScore
- **E:** Isolated ProjScore vs. AttnScore comparison

We conduct these ablation studies on the three datasets from the Single-Document QA task with the Mistral model. The results are in Table 7 and visualized in Figure 8. The results demonstrate that the projection-based scoring function consistently outperforms the attention-based scoring function, across all different configurations of other components.

We further validate this conclusion on two other based models Qwen and Llama. Table 8 and Table 9 show that AnDPro has good performance on other models, confirming both its effectiveness and robustness.

C.5.2 Ablation Study on Other Adopted Techniques

We further conduct ablation studies to investigate the contributions of three different components in our algorithm: (1) merging tokens into chunks, (2) preserving the first token, and (3) cross-head budget allocation. The results are in Figure 9. It visually compares the performance improvements contributed by each component to the project-based method, demonstrating that all components are indispensable for achieving the final high-performance results.

Table 7: Ablation results of Mistral-7b-Instruct-v0.2 on LongBench (single QA Doc). The comparisons demonstrate that ProjScore consistently outperforms AttnScore across different settings of adopted techniques, isolating the novel contribution of AnDPro.

	Budget	Nrtvqa	Qasper	MF-en	Avg.	Nrtvqa	Qasper	MF-en	Avg.
A		CrossHead + Chunk + FirstToken + ProjScore				CrossHead + Chunk + FirstToken + AttnScore			
	128	24.84	25.14	47.33	32.44	22.19	23.67	48.33	31.40
	256	25.71	28.17	49.31	34.40	23.85	28.12	49.29	33.75
	512	24.87	30.99	49.58	35.15	25.25	30.21	48.94	34.80
	1024	25.87	32.54	49.35	35.92	25.94	32.45	49.02	35.80
B		Chunk + FirstToken + ProjScore				Pooling + FirstToken + AttnScore			
	128	20.25	20.92	46.89	29.35	19.17	21.40	42.93	27.83
	256	22.81	25.25	48.30	32.12	22.37	23.74	48.13	31.41
	512	24.72	27.61	49.38	33.90	24.60	27.81	48.98	33.80
	1024	26.09	30.75	49.20	35.35	25.47	29.57	49.33	34.79
C		CrossHead + FirstToken + ProjScore				CrossHead + FirstToken + AttnScore			
	128	22.78	24.98	47.39	31.72	21.12	21.16	38.46	26.91
	256	24.32	27.41	48.49	33.41	21.50	23.50	40.38	28.46
	512	25.64	29.03	49.09	34.59	23.93	24.67	43.08	30.56
	1024	26.30	30.90	49.08	35.43	24.94	27.82	46.60	33.12
D		CrossHead + Chunk + ProjScore				CrossHead + Pooling + AttnScore			
	128	24.90	24.47	48.57	32.65	21.64	23.77	46.48	30.63
	256	24.92	28.25	49.21	34.13	23.29	26.32	48.99	32.87
	512	26.18	30.20	49.41	35.26	24.73	28.59	48.84	34.05
	1024	26.14	31.57	49.60	35.77	25.60	31.27	48.49	35.12
E		ProjScore				AttnScore			
	128	20.27	22.02	43.67	28.65	21.50	20.85	39.11	27.15
	256	22.33	24.95	48.51	31.93	22.48	22.65	43.64	29.59
	512	24.91	26.80	49.01	33.57	23.07	25.91	44.84	31.27
	1024	25.20	30.02	49.25	34.82	25.02	29.48	47.25	33.92

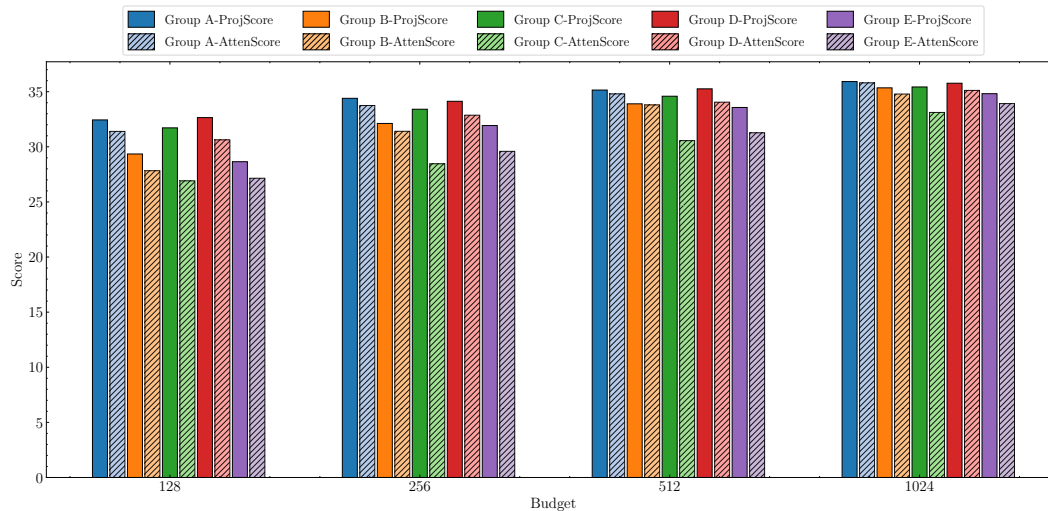


Figure 8: Ablation results of Mistral-7b-Instruct-v0.2 on LongBench (single QA Doc). The comparisons demonstrate that ProjScore consistently outperforms AttnScore across different settings of adopted techniques, isolating the novel contribution of AnDPro.

Our discoveries are as follows. (1) To achieve a better semantic succession, we merge a certain number of adjacent tokens as one chunk. We test different chunk sizes $C \in \{1, 2, 4, 8, 16\}$ on the three datasets from the Single-Doc. QA task. The experiments are conducted on Mistral-7b-Instruct-v0.2. Results are in Figure 10. As a hyperparameter directly related to contextual semantics, we found that when the budget is small, smaller chunks will lead to better performance, and vice versa. In the main experiments, we set the chunk size to 4. (2) Following SnapKV and Ada-KV, we retain the first token by default. Results show that this leads to a very slight performance gain. (3) We follow Ada-KV to allocate budget across all heads within each layer according to their scores. The results show that this operator is critical to the overall performance gain. The integration of the

Table 8: **Ablation results of Qwen2-7B-Instruct on LongBench (single QA Doc).** The comparisons demonstrate that ProjScore consistently outperforms AttnScore across different settings of adopted techniques, isolating the novel contribution of AnDPro.

	Budget	Nrtvqa	Qasper	MF-en	Avg.	Nrtvqa	Qasper	MF-en	Avg.
A		CrossHead + Chunk + FirstToken + ProjScore				CrossHead + Chunk + FirstToken + AttnScore			
	128	23.95	35.67	44.72	34.78	21.69	35.07	45.74	34.17
	256	22.52	39.90	46.20	36.21	24.25	39.16	46.87	36.76
	512	21.52	43.93	46.74	37.40	22.96	43.32	46.02	37.43
	1024	22.07	44.95	46.69	37.90	23.44	44.38	47.37	38.40
B		Chunk + FirstToken + ProjScore				Pooling + FirstToken + AttnScore			
	128	20.49	33.72	42.26	32.16	20.43	31.45	42.80	31.56
	256	21.25	37.88	43.43	34.19	22.28	36.70	43.95	34.31
	512	23.88	40.25	46.89	37.01	22.80	41.12	44.38	36.10
	1024	23.40	42.24	46.51	37.38	24.58	43.55	46.95	38.36
C		CrossHead + FirstToken + ProjScore				CrossHead + FirstToken + AttnScore			
	128	21.84	35.04	44.64	33.84	17.07	32.11	39.11	29.43
	256	22.30	38.72	44.31	35.11	23.36	36.89	43.24	34.50
	512	25.02	43.50	47.55	38.69	22.43	40.82	42.02	35.09
	1024	23.93	43.18	46.78	37.96	23.19	40.62	46.28	36.70
D		CrossHead + Chunk + ProjScore				CrossHead + Pooling + AttnScore			
	128	22.16	34.32	47.45	34.64	20.54	32.82	40.26	31.21
	256	25.12	39.42	45.01	36.52	22.86	35.62	43.93	34.14
	512	24.30	42.95	46.68	37.98	23.44	41.50	44.64	36.53
	1024	24.56	45.52	45.71	38.60	23.75	42.82	47.91	38.16
E		ProjScore				AttnScore			
	128	21.32	33.44	41.42	32.06	19.40	30.84	34.30	28.18
	256	21.67	36.45	45.76	34.63	21.34	31.67	38.96	30.66
	512	24.02	41.40	44.30	36.57	21.68	34.55	39.60	31.94
	1024	23.37	43.13	47.61	38.04	23.31	39.16	42.60	35.02

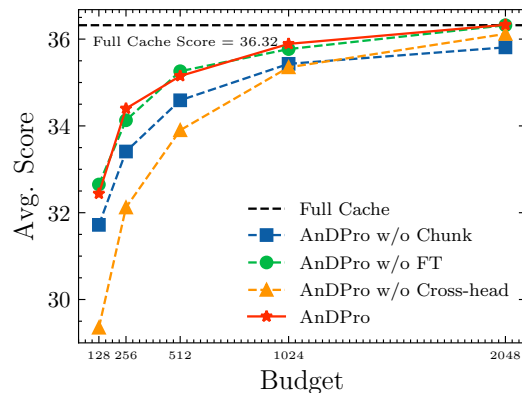


Figure 9: **Ablation Study on the adopted techniques.**

aforementioned techniques results in the high performance of AnDPro. Notably, AnDPro outperforms full cache with a budget size $k = 2048$.

Table 9: **Ablation results of Meta-Llama-3.1-8B-Instruct on LongBench (single QA Doc).** The comparisons demonstrate that ProjScore consistently outperforms AttnScore across different settings of adopted techniques, isolating the novel contribution of AnDPro.

	Budget	Nrtvqa	Qasper	MF-en	Avg.	Nrtvqa	Qasper	MF-en	Avg.
A		CrossHead + Chunk + FirstToken + ProjScore				CrossHead + Chunk + FirstToken + AttnScore			
	128	26.66	38.22	54.54	39.81	27.19	38.01	53.13	39.44
	256	27.28	42.09	54.72	41.36	26.36	41.94	54.77	41.02
	512	28.03	44.25	55.61	42.63	29.12	44.86	54.86	42.95
	1024	29.40	46.83	56.72	44.32	28.14	46.77	55.81	43.57
B		Chunk + FirstToken + ProjScore				Pooling + FirstToken + AttnScore			
	128	24.87	32.33	52.51	36.57	25.54	31.40	50.99	35.98
	256	27.04	38.52	54.21	39.92	26.92	38.19	52.62	39.24
	512	27.24	42.18	54.25	41.22	28.40	41.52	55.40	41.77
	1024	28.57	45.80	54.89	43.09	28.32	45.37	56.81	43.50
C		CrossHead + FirstToken + ProjScore				CrossHead + FirstToken + AttnScore			
	128	28.51	38.05	52.92	39.83	26.78	34.67	52.32	37.92
	256	26.72	41.48	55.23	41.14	27.05	40.23	54.29	40.52
	512	29.79	43.50	55.11	42.80	28.96	43.11	54.61	42.23
	1024	31.48	45.54	56.75	44.59	29.98	46.06	55.32	43.79
D		CrossHead + Chunk + ProjScore				CrossHead + Pooling + AttnScore			
	128	30.13	37.46	54.01	40.53	25.06	34.50	52.27	37.28
	256	29.70	42.56	55.59	42.62	27.04	39.24	54.41	40.23
	512	31.08	44.48	55.62	43.73	26.92	42.68	55.20	41.60
	1024	31.11	46.05	56.39	44.52	28.52	46.01	55.72	43.42
E		ProjScore				AttnScore			
	128	25.19	32.36	51.41	36.32	28.10	28.56	42.69	33.12
	256	26.07	37.74	52.05	38.62	29.56	31.00	44.19	34.92
	512	28.51	41.88	55.28	41.89	28.91	36.00	51.55	38.82
	1024	27.75	45.54	56.22	43.17	29.51	40.70	54.31	41.51

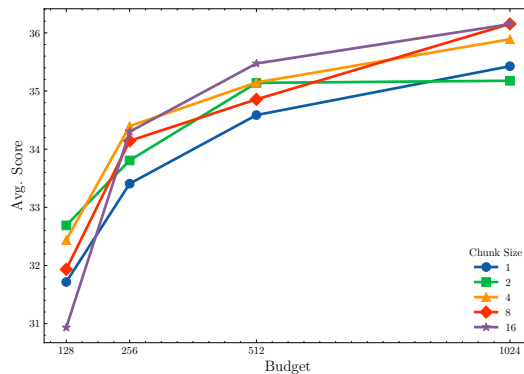


Figure 10: Comparison of experimental results with different chunk sizes on Single-Doc. QA

C.6 Different Choices of θ and b

Recall that our projection-based scoring function is defined in the form $s_i = a_i(\mathbf{y}^\top \mathbf{v}_i + b)$. In this section, we investigate the effect of different choices of θ and b . We conduct these experiments using Mistral-7B-Instruct-v2.0 on the datasets from the Single-Document QA task with the Mistral model.

To investigate the effect of the anchor direction θ , we parameterize the anchor direction as $\theta = \sum_{i=1}^n \tilde{a}_i \mathbf{v}_i$ where $\tilde{a}_i = \text{Softmax}(q^\top \mathbf{k}_i / \alpha)$ with a temperature coefficient α . This generalization covers several cases: $\alpha = 1$ corresponds exactly to our proposed anchor direction $\theta = \mathbf{y}$; $\alpha \rightarrow \infty$ corresponds to the average of all value vectors; an $\alpha \rightarrow 0$ corresponds to the value vector with the highest attention weight. The results are in Table 10, demonstrating that setting the anchor direction as $\theta = \mathbf{y}$ achieves a robust performance.

Table 10: Results of different choices of the anchor direction $\theta = \sum_{i=1}^n \tilde{a}_i \mathbf{v}_i$, where $\tilde{a}_i = \text{Softmax}(q^\top \mathbf{k}_i / \alpha)$ with a temperature coefficient α . The used model is Mistral-7B-Instruct-v0.2 and the dataset is LongBench (single QA Doc). The results demonstrate that setting the anchor direction as $\theta = \mathbf{y}$ achieves a robust performance.

	Nrtvqa	Qasper	MF-en	Avg.	Nrtvqa	Qasper	MF-en	Avg.
	Budget=128				Budget=256			
$\alpha = 0$ (highest attention)	15.79	14.67	26.22	18.89	16.01	15.05	28.52	19.86
$\alpha = 0.5$	23.10	25.68	47.69	32.16	23.15	27.12	48.82	33.03
$\alpha = 1$ ($\theta = \mathbf{y}$)	24.84	25.14	47.33	32.44	25.71	28.17	49.31	34.40
$\alpha = 1.5$	22.81	23.47	47.99	31.42	24.62	28.08	48.35	33.68
$\alpha = \infty$ (average)	15.83	15.75	27.96	19.85	16.62	16.06	28.96	20.55
	Budget=512				Budget=1024			
$\alpha = 0$ (highest attention)	16.24	14.56	29.83	20.21	16.96	15.57	34.68	22.40
$\alpha = 0.5$	24.87	29.80	49.33	34.67	25.89	30.57	48.93	35.13
$\alpha = 1$ ($\theta = \mathbf{y}$)	24.87	30.99	49.58	35.15	25.87	32.54	49.35	35.92
$\alpha = 1.5$	25.13	30.34	48.14	34.54	25.81	31.57	49.64	35.67
$\alpha = \infty$ (average)	16.81	17.62	33.50	22.64	17.11	21.56	38.68	25.78

We then investigate the impact of b , and the results are in Figure 11. We find that though a careful choice of b promise lead to a better results, simply setting $b = 0$ is enough to derive a stable performance. Therefore, we set $b = 0$ in our main experiments, without the need for labor-intensive hyperparameter tuning. Moreover, when b approaches infinity (e.g., $b = 1000$), the scoring function converges to attention-based ones, and the performance approaches the performance of Ada-KV.

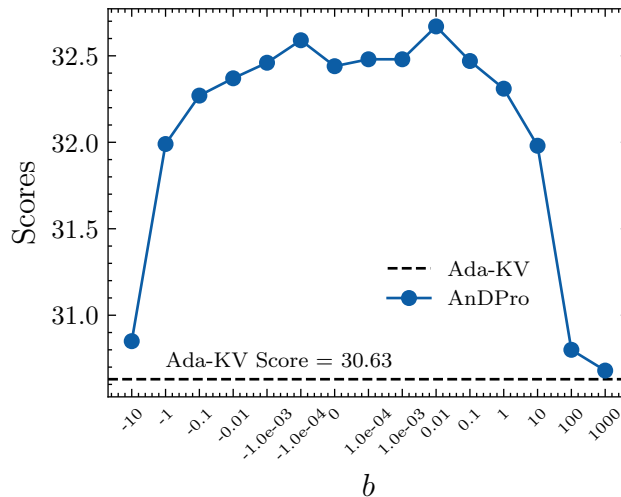


Figure 11: Different choices of b

C.7 Eviction Losses and Cosine Similarity

Figure 12 illustrates the eviction loss and cosine similarity derived from 2 methods under different budget allocations, with the eviction loss quantified as $\frac{\|y - \hat{y}\|^2}{\|y\|^2}$. The results show that AnDPro consistently yields a lower eviction loss, and a larger cosine similarity, which can to some extent explain the high performance of AnDPro.

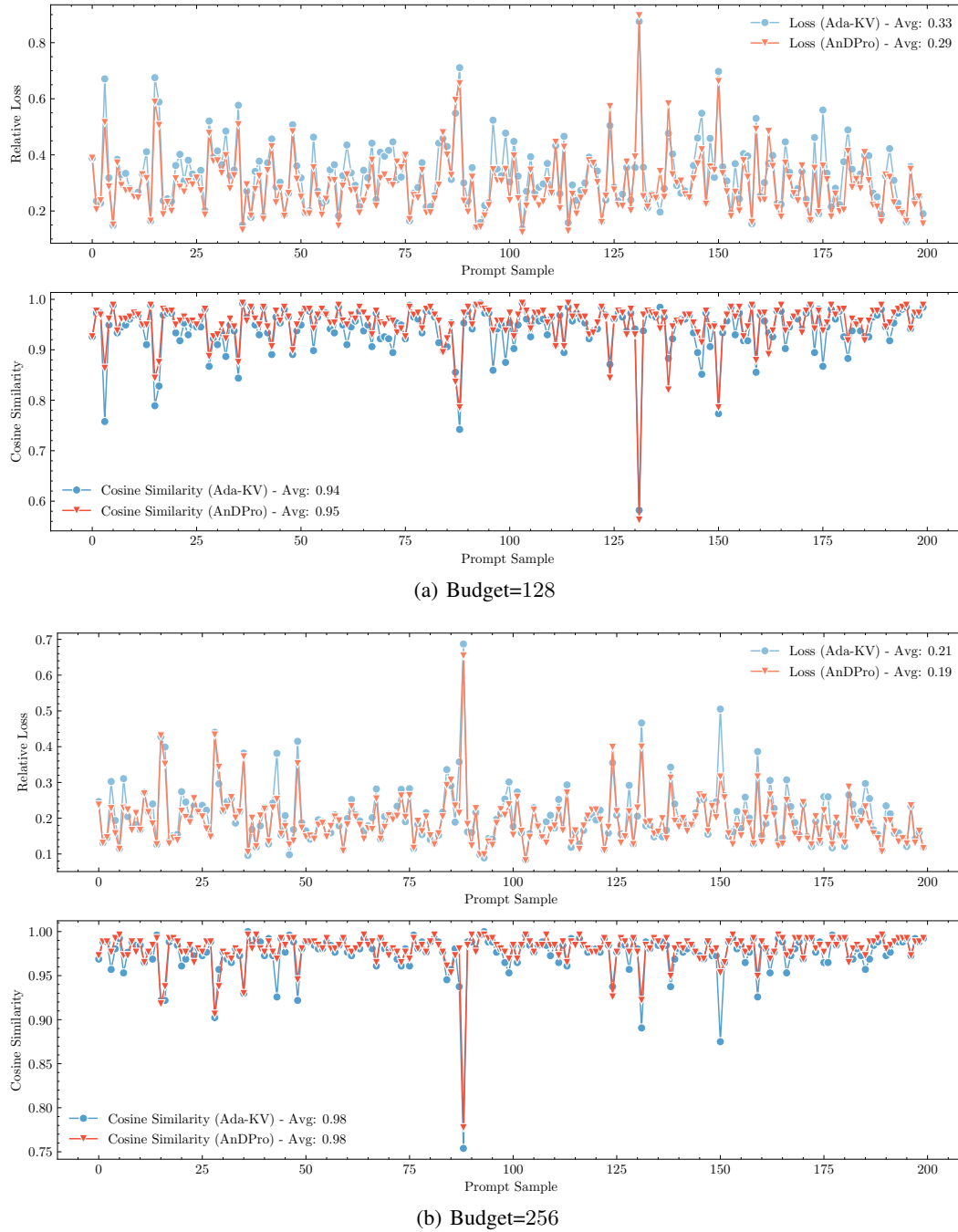


Figure 12: Comparison of eviction loss and cosine similarity. Experiments are conducted using Mistral7B-instruct-v0.2 on 200 samples from the Qasper dataset.

C.8 Token Visualization

Figure 13 illustrates the distribution of our defined scoring functions across various layers (using Mistral7B-instruct-v0.2 on a sample from the Qasper Dataset), demonstrating that the distribution exhibits a sparsity feature like that of attention weights.

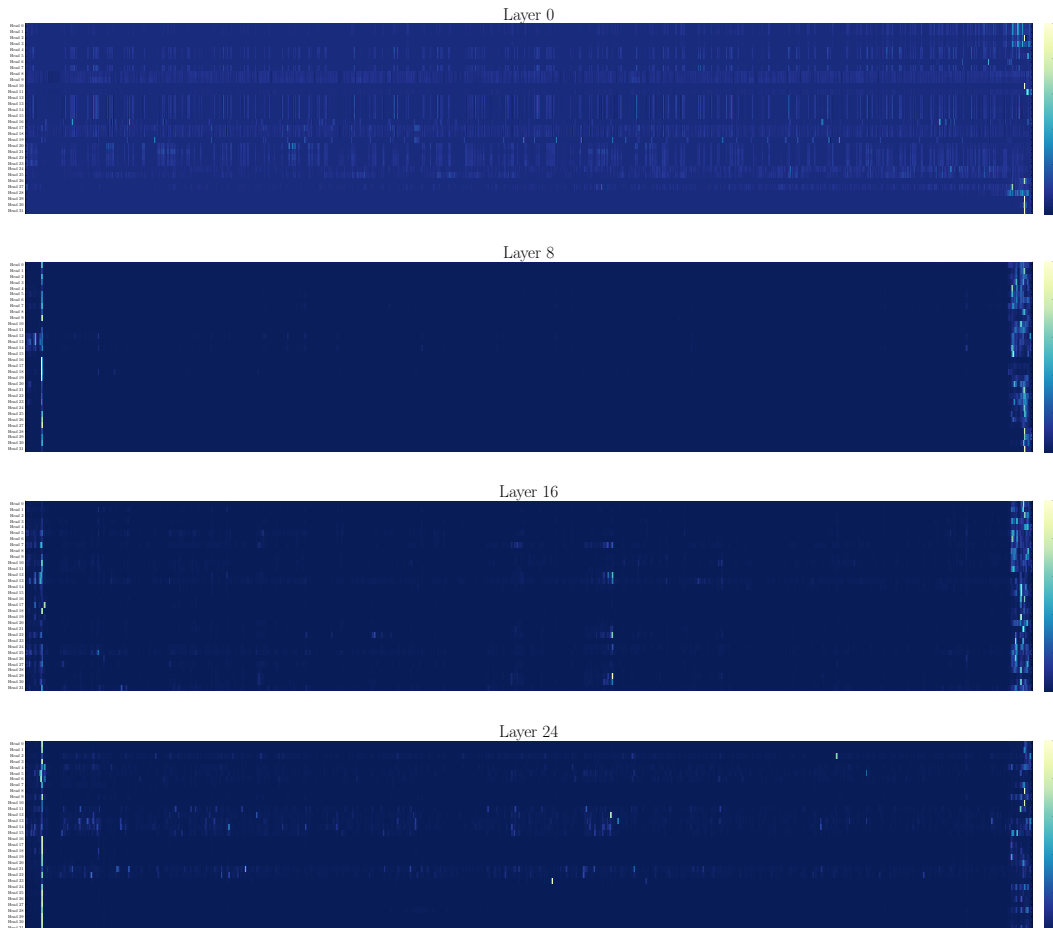


Figure 13: Visualization of Token Scores for Different Layers.

C.9 Value Vectors Visualization & Magnitude Histogram

We conduct two experiments to intuitively explain why projection-based methods outperform attention-based approaches. First, we provide the PCA visualization of value vectors in Figure 14. It reveals that projection methods preserve geometrically critical tokens along the original output direction, minimizing post-eviction semantic drift. In contrast, attention-based selection disrupts the latent spatial distribution, causing divergent outputs. Second, complementary analysis in Figure 15 demonstrates significant variance of the vector vector magnitude. This indicates that the value vectors contain rich semantic information, which is ignored in attention-based approaches. This further underscores the necessity of projection-based strategies.

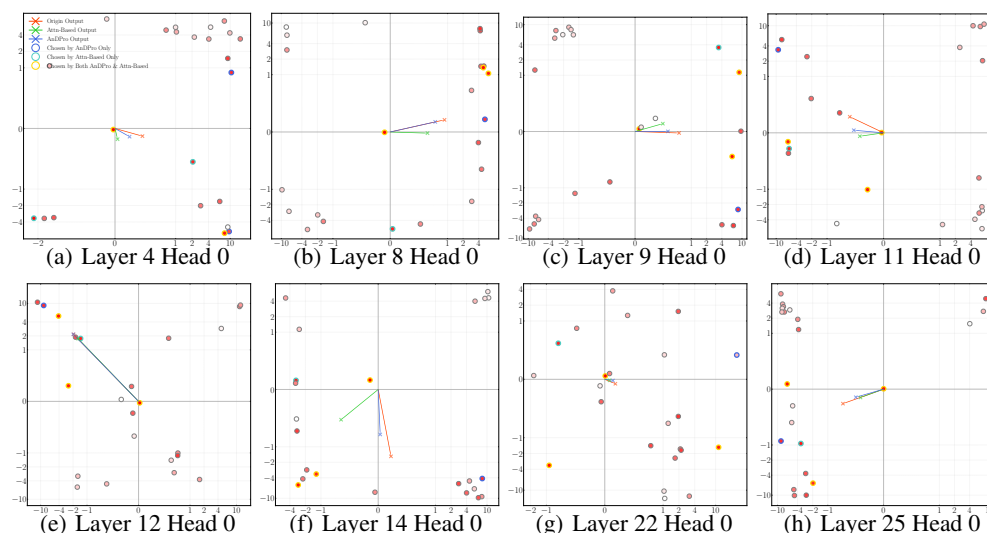


Figure 14: The visualization of token value vectors and outputs of different approaches using PCA dimensionality reduction. We use an easy case with 23 tokens. Each dot represents a token value vector, with color intensity (red hues) indicating attention scores. Tokens selected by different methods are highlighted by colored outer rings. Arrow vectors depict output directions. We visualize the results from the Head 0 across several layers. The results demonstrate that the projection-based approach can achieve smaller distances between the pre- can post-eviction outputs by maintaining the semantical directions of the original outputs.

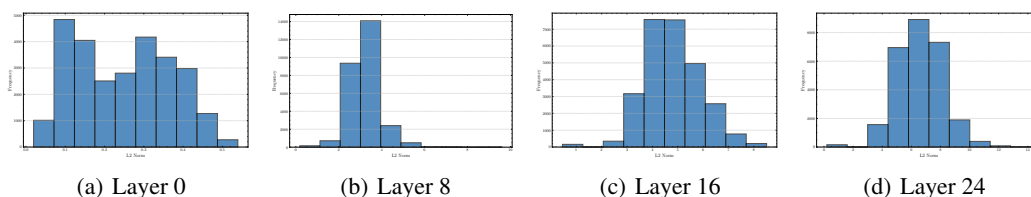
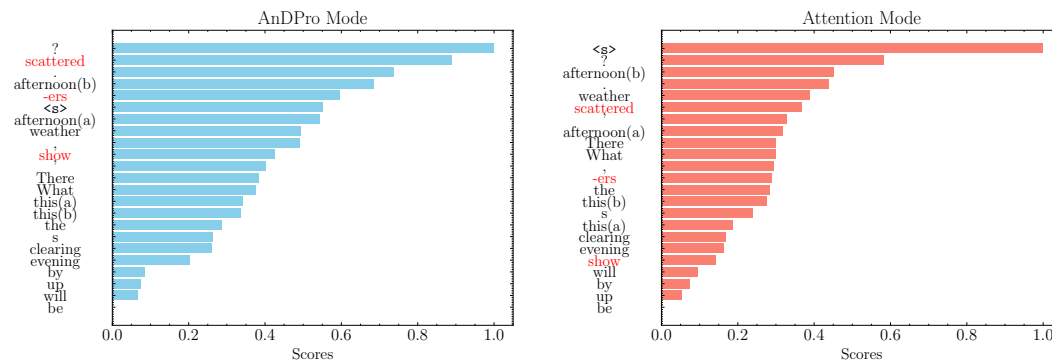


Figure 15: The distribution of the value vector magnitudes from several different layers in a test example. This reveals the significant variability and underscores the important semantic information captured by projection but overlooked by pure attention methods.

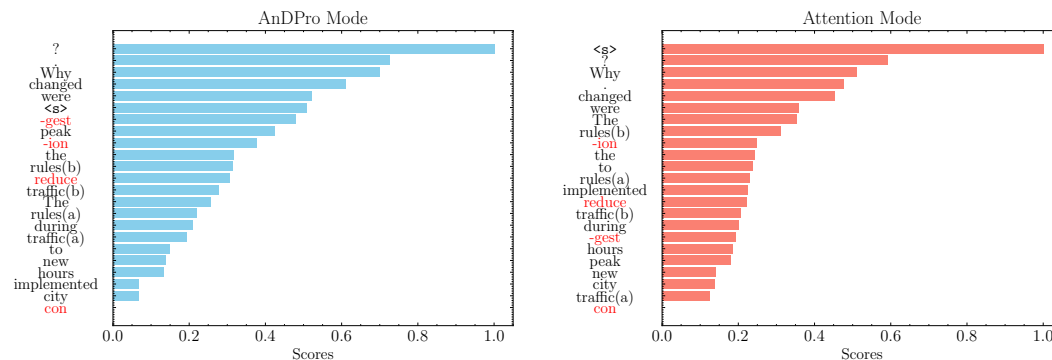
C.10 Case Study

We conduct case study on some simple test cases. We calculate the total scores of each token in the prefilling Phase based on the AnDPro and attention weights. Some results are presented in Figure 16. These token-level scores are scaled to the range of 0 to 1 and sorted in descending order. Specifically, a higher score indicates that the token is considered more important.

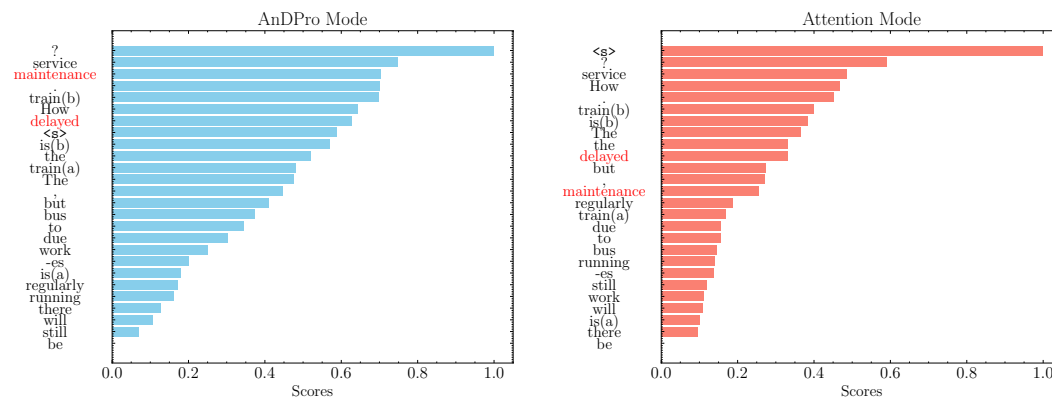
By observing the key tokens marked in red (identified by humans), AnDPro places better emphasis on the key tokens compared to the attention-based approach, thereby intuitively demonstrating the effectiveness of our method.



Prompt 1: There will be scattered showers this afternoon, clearing up by evening. What's the weather this afternoon?



Prompt 2: The city implemented new traffic rules to reduce congestion during peak hours. Why were the traffic rules changed?



Prompt 3: The train is delayed due to maintenance work, but there will still be buses running regularly. How is the train service?

Figure 16: **Case study.** We visualize the importance orders given by different approaches, AnDPro, and attention-based approach. Red tokens are key tokens identified by humans.

C.11 Long Context

We have conducted additional experiments on very long sequences using the Needle-in-a-Haystack benchmark, extending the test sequence length from the original 32K tokens up to 384K tokens. Specifically, we use the Llama-3.1-8B-Instruct model (pretrain context length 128K) under a budget size of 128. Results are in Figure 17.

The results demonstrate that AnDPro consistently achieves superior performance even on long sequences. Notably, when sequence length exceeds 256K tokens, Full cache fails due to memory constraints, while AnDPro remains effective. Although all methods fail beyond 512K tokens, AnDPro maintains top performance within feasible length ranges.

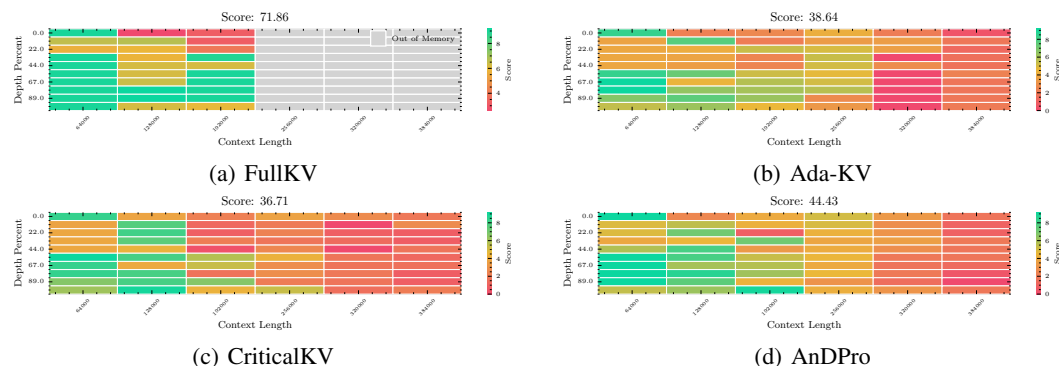


Figure 17: Results of Llama-3.1-8B-instruct on the Needle-in-a-Haystack test with extremely long sequences (up to 384K). The x-axis represents the document’s context length, and the y-axis indicates the depth of the needle insertion. Each square in the heatmap demonstrates the correctness of retrieving a proper sentence inserted in the context for a given context length and insertion depth. The Average Score is calculated by averaging the sum of score with different insert depth across various context lengths. Gray squares indicate out-of-memory errors due to excessive context length.

C.12 Large-Sized LLMs

To evaluate the generalization and applicability of AnDPro on larger models, we conduct additional experiments using the Qwen2.5-32B-Instruct model on the LongBench single-document QA benchmark. The results are in Table 11.

Table 11: Results of Qwen2.5-32B-Instruct on the LongBench single-document QA benchmark.

	Budget= 128	Budget= 256	Budget= 512	Budget= 1024
Ada-KV	34.66	38.02	41.57	43.82
AnDPro	38.63	40.91	43.62	45.81

C.13 Long Decoding Tasks

To further demonstrate the broader applicability of our method, we conduct additional experiments on long decoding tasks, particularly in LLM reasoning settings. For a better comparison with the baselines, we additionally implemented the decoding versions of AnDPro and SnapKV, while Ada-KV does not have a decoding version. Specifically, we evaluated our approach on the AIME24 dataset using DeepSeek-R1-Distill-Qwen-14B.

Table 12 reports the reasoning performance (accuracy, %) under different KV cache budget settings. Table 13 reports the throughput (tokens/s) and maximum supported batch size (OOM threshold) during 10K-token generation.

Table 12: Performance (accuracy, %) of different methods on the AIME24 dataset under two KV cache budget settings (2K and 4K).

	Budget= 2K	Budget= 4K
SnapKV	50.00	53.33
AnDPro	60.00	63.33

Table 13: Throughput (tokens/s) and maximum supported batch size (before out-of-memory occurs) for different methods during 10K-token generation on a single GPU with 80GB of VRAM.

	Generation Length	Max Batch Size (OOM Threshold)	Throughput (token/s)
Full	10K	28	181.09
SnapKV	10K	80	415.48
AnDPro	10K	80	413.15

D Limitation and Broader Impact

Limitation We have limited this work to the setting of uniform budget allocation across layers. It remains to be investigated how to perform fine-grained budget allocation between layers, building upon existing studies that have demonstrated the varying importance of different layers.

Broader Impact Our methodology pioneers a novel paradigm for KV Cache Compression by incorporating directional characteristics of value vectors (beyond attention weight exclusivity), which we believe will catalyze transformative breakthroughs in this field. Crucially, the proposed technique achieves substantial inference acceleration with only marginal performance degradation, establishing its deployment potential for edge computing scenarios and long-context sequence processing.

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: Our main contribution is to propose AnDPro, which is clearly described in the abstract and introduction.

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: We discuss the limitations of the work in Appendix D

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: The proofs of all theorems can be found in Appendix A.

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: We provide necessary details to reproduce the results. We release our code at <https://github.com/MIRALab-USTC/LLM-AnDPro>.

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: We use open-sources datasets and will release our code once the paper is accepted for publication.

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, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We follow the standard benchmarks and provide necessary details for all experiments.

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 follow previous relevant works in the field to report experimental statistical information.

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 have provided the information in Appendix 4.1.

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: The research conducted in this paper conform, in every respect, with the NeurIPS Code of Ethics.

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: [Yes]

Justification: We have included broader impacts in Appendix D.

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: This paper poses no such risks.

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: The creators or original owners of assets used in the paper are properly credited. The license and terms of use are properly respected.

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: [NA]

Justification: This paper does not release new assets.

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: The paper does not involve crowdsourcing nor research with human subjects.

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: The paper does not involve crowdsourcing nor research with human subjects.

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: [\[Yes\]](#)

Justification: We have provided a detailed description of the use of LLMs in our method.

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.