PREDICTING OUTPUT PRESERVATION AND FIRST DIVERGENCE UNDER KV-CACHE COMPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Key-Value (KV) cache compression is usually evaluated by how much cache state is removed or distorted, but cache distortion alone does not determine whether the model output is preserved. Under deterministic greedy decoding, we study when compression preserves the dense-model token and when compressed generation first diverges. Our key observation is that output preservation depends not only on perturbation magnitude, but also on how the compression-induced change acts relative to the decision margin. Based on this observation, we derive an exact per-step characterization of output preservation through output crossing. We then introduce the Output Crossing Predictor (OCP), a first-order framework that predicts whether a candidate compression will cause output crossing before compressed logits are observed. OCP combines the local cache-read perturbation with the local sensitivity of the dense model. Repeated across decoding steps, the same prediction estimates the first divergence time. Experiments across multiple models and compression operators show that OCP predicts output crossing and first divergence, while the compression boundary becomes more conservative as generation length increases. These results shift KV-cache compression from measuring cache distortion to predicting output preservation and first divergence.

1 INTRODUCTION

KV-cache compression reduces the memory and decoding cost of long-context inference (Zhang et al., 2023; Li et al., 2024; He et al., 2024). Existing methods use token eviction (Zhang et al., 2023; Liu et al., 2023; Xiao et al., 2024), quantization (Liu et al., 2024b; Hooper et al., 2024; Duanmu et al., 2024), merging (Liu et al., 2024a; Zhang et al., 2024b), low-rank approximation (Chang et al., 2025; Saxena et al., 2024; Zhang et al., 2024a), or layer- and head-wise budget allocation (Cai et al., 2025; Feng et al., 2025; Fu et al., 2025). Representative eviction methods include H2O (Zhang et al., 2023), SnapKV (Li et al., 2024), and CAOTE (Goel et al., 2026). Existing methods mainly ask what cache information should be retained or how it should be represented under a limited memory budget. They often evaluate the result through compression ratio (Zhang et al., 2024a), reconstruction error (Saxena et al., 2024), or final task accuracy (He et al., 2024). These measures do not answer a more immediate question: will this particular compression decision change the model output? A heavily perturbed cache can still preserve the next token, while a much smaller perturbation can change it.

We study KV-cache compression through *output preservation*. Under greedy autoregressive decoding, *output preservation* means that compression leaves the dense-model token unchanged at a given decoding step. The *first divergence* is the earliest step at which the compressed generation departs from the dense generation. This leads to the central question of the paper:

Given a KV-cache compression intervention, when is the dense-model output preserved, and when does the compressed generation first diverge?

The key observation is that cache distortion does not directly determine output preservation. Compression may noticeably change the state read from the KV cache, but this does not mean that the next token will change. For example, suppose the dense model assigns logits of 10 and 9.8 to "yes" and "no", respectively. Compression could shift both logits down by 1 without changing the prediction. In contrast, shifts of only -0.15 and $+0.15$ would make "no" overtake "yes". The difference

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

lies in how the logit changes act relative to the decision margin. The output changes only when a competing token gains enough relative advantage to close that margin. Perturbation magnitude alone is therefore insufficient. Its direction relative to the current decision margin also matters. This view explains why internal distortion can grow smoothly while the generated output changes abruptly.

Based on this observation, we first characterize exactly when compression changes the next token. This happens when a competing token gains enough relative logit advantage to close the dense-model decision margin and overtake the original prediction. We refer to this event as *output crossing*. A natural question is whether we can anticipate such a crossing before the compressed logits are observed. To do this, we estimate how a candidate compression intervention changes the local cache read and how this change propagates to the output logits. We refer to the resulting prediction framework as the **Output Crossing Predictor (OCP)**. The same formulation applies to eviction (Zhang et al., 2023), value-side quantization (Tao et al., 2025), low-rank compression (Chang et al., 2025), and key-side perturbation (Wang et al., 2026).

Once output crossing can be predicted at a single decoding step, we extend the prediction across the whole generation. The first predicted output crossing gives the predicted first divergence time. We also use OCP to adjust the compression level at each step. When output crossing is predicted, a milder compression setting can be used. Otherwise, stronger compression can be applied. In this way, the OCP framework works with existing KV-cache compression methods rather than replacing them. It can assess candidate compression decisions before obtaining their compressed logits and help determine how aggressively to compress at each step. Overall, our approach shifts KV-cache compression from measuring how much the cache changes to predicting when a compression decision is likely to change the model output.

2 Problem Setup

We compare two decoding runs of the same model: one uses the dense KV cache, and the other uses a compressed KV cache. Let the index t denote the current decoding step. Let C_t be the dense KV cache available at step t, and let $\widetilde{C}_t^{(\rho)}$ be its compressed version, where $\rho \in [0, 1]$ denotes the compression level, with $\rho = 0$ recovering the dense cache and larger ρ corresponding to stronger compression. While the two generations share the same prefix, let $y_t^\star$ and $\widetilde{y}_t^{(\rho)}$ denote the tokens selected by greedy decoding under the dense and compressed caches, respectively. We assume a unique greedy-decoding winner for both the dense and compressed branches at the analyzed steps.

At each decoding step t, compression preserves the output when $\widetilde{y}_t^{(\rho)} = y_t^\star$. The first divergence time τ_ρ is then

$$\tau_\rho \triangleq \inf \left\{ t : \widetilde{y}_t^{(\rho)} \neq y_t^\star \right\}. \tag{1}$$

Before τ_ρ, compression may change the cache or hidden states, but the generated sequence remains unchanged. Our goal is to characterize output preservation at each decoding step and use this characterization to predict first divergence.

Consider token eviction within a single attention head as an example. The head output is $o_t = \sum_i a_{t,i} \boldsymbol{v}_i$, where $a_{t,i}$ is the attention weight on cached entry i, and $\boldsymbol{v}_i \in \mathbb{R}^d$ denotes its value vector (Vaswani et al., 2017). Let $o_t(C_t)$ denote its output under the dense cache and $o_t(\widetilde{C}_t^{(\rho)})$ its output under the compressed cache. The perturbation in head output is

$$\Delta o_t^{(\rho)} = o_t(\widetilde{C}_t^{(\rho)}) - o_t(C_t). \tag{2}$$

This perturbation describes how compression changes the state read from the cache. Its magnitude tells us how much the state changes, while its direction tells us how the state changes. Different compression operators produce different forms of $\Delta o_t^{(\rho)}$. Appendix A provides a detailed comparison with recent related work.

3 Output Preservation Under KV-Cache Compression

In this section, we first characterize output crossing at a single decoding step and then predict it before the compressed logits are observed. Finally, we extend the per-step prediction to first divergence over the full generation.

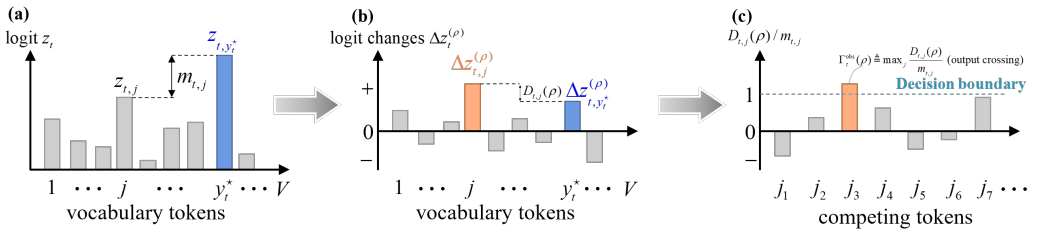

Figure 1: **Output preservation under KV-cache compression.** (a) The decision margin measures how much relative logit gain a competing token needs to overtake the dense-model token. (b) Compression changes the output logits and produces compression pressure on the competing token. (c) Comparing compression pressure with the decision margin across competing tokens determines output preservation and output crossing.

3.1 When Does Compression Preserve the Next Token?

Compression can change the state read from the cache without changing the next token. To understand when the output changes, we examine the output logits. Let $z_t \in \mathbb{R}^V$ denote the dense-model logits at step t, where V is the vocabulary size. The dense model selects the token $y_t^\star$ with the largest logit. For any competing token $j \neq y_t^\star$, we define its decision margin as

$$m_{t,j} \triangleq z_{t,y_t^\star} - z_{t,j} > 0. \tag{3}$$

The margin $m_{t,j}$ tells us how much relative logit gain token j needs to overtake the dense-model token, as illustrated in Fig. 1(a).

Compression changes the logits of both the dense-model token and the competing tokens. For a competing token j, what matters is its logit change relative to the dense-model token. Let $\widetilde{z}_t^{(\rho)}$ denote the output logits under compression level ρ, and let $\Delta z_t^{(\rho)} = \widetilde{z}_t^{(\rho)} - z_t$ denote the compression-induced change in the output logits. We define this relative logit gain as

$$D_{t,j}(\rho) \triangleq \Delta z_{t,j}^{(\rho)} - \Delta z_{t,y_t^\star}^{(\rho)}. \tag{4}$$

The quantity $D_{t,j}(\rho)$ measures how much token j gains relative to the dense-model token after compression, as illustrated in Fig. 1(b). We call it the *compression pressure* on token j. When this relative gain exceeds the decision margin $m_{t,j}$, an output crossing occurs.

Although $D_{t,j}(\rho)$ tells us how compression changes token j relative to the dense-model token, this value alone does not tell us whether token j will overtake it. We also need to know how large the original decision margin $m_{t,j}$ is. We therefore compare the two quantities through their ratio $D_{t,j}(\rho)/m_{t,j}$. This ratio tells us how much of the original margin has been closed by compression. We then define

$$\Gamma_t^{\text{obs}}(\rho) \triangleq \max_{j \neq y_t^\star} \frac{D_{t,j}(\rho)}{m_{t,j}}, \qquad \Gamma_t^{\text{obs}}(\rho) < 1 \iff \widetilde{y}_t^{(\rho)} = y_t^\star, \tag{5}$$

where $\Gamma_t^{\text{obs}}(\rho)$ gives an exact characterization of output preservation at step t, as illustrated in Fig. 1(c). Thus,

- $\Gamma_t^{\text{obs}}(\rho) < 1$: the dense-model output is preserved.
- $\Gamma_t^{\text{obs}}(\rho) = 1$: the output reaches the decision boundary.
- $\Gamma_t^{\text{obs}}(\rho) > 1$: an output crossing occurs.

Appendix B.1 derives this exact criterion and clarifies the decision-boundary case. Since $\Gamma_t^{\text{obs}}(\rho)$ uses the compressed logits, it can only be obtained after the candidate compression has been evaluated.

3.2 Can We Predict Output Crossing Before It Happens?

Our exact criterion helps us understand output preservation, but it is available only after compression. To compute $\Gamma_t^{\text{obs}}(\rho)$, we first need the compressed logits. This is too late if we want to evaluate

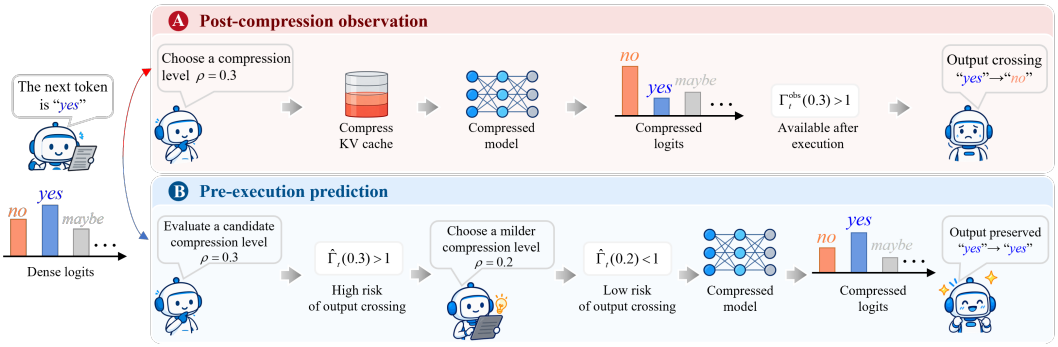

Figure 2: **Post-compression observation versus pre-execution prediction.** (a) Compression is executed first, and output crossing is identified only afterward. (b) The predictor evaluates a candidate compression decision before compressed-logit evaluation, allowing a milder compression level to be chosen when needed.

a candidate compression decision before its compressed logits are obtained. Before compressed-logit evaluation, however, we can obtain the local cache-read perturbations associated with that candidate compression from the current dense decoding state. The question is whether these perturbations can be used to predict output crossing in advance.

To predict output crossing, we need to know how each local cache-read perturbation affects the output logits. We index each cache-read location by $k = (\ell, h)$, corresponding to attention head h in transformer layer ℓ. A small change at one cache-read location can increase some token logits and decrease others. We describe this local sensitivity by $\mathcal{K}_{t,k} \triangleq \frac{\partial z_t}{\partial o_t^{(k)}} \in \mathbb{R}^{V \times d}$. Each row of $\mathcal{K}_{t,k}$ describes how the logit of one vocabulary token responds to a small change in the cache-read state at location k. The derivative is taken through the full downstream computation graph at the dense operating point, so it includes first-order effects propagated through subsequent transformer layers, including the resulting first-order changes in queries at subsequent transformer layers within the same decoding step. Thus, $\mathcal{K}_{t,k}$ connects the local cache-read perturbation $\Delta o_t^{(k)}(\rho)$ to its first-order effect on the output logits.

The next step is to estimate how much compression moves each competing token relative to the dense-model token. Using these local sensitivities, we obtain the first-order estimate

$$\widehat{D}_{t,j}(\rho) = \sum_k (\mathbf{e}_j - \mathbf{e}_{y_t^\star})^\top \mathcal{K}_{t,k} \Delta o_t^{(k)}(\rho), \tag{6}$$

where $\mathbf{e}_y \in \mathbb{R}^V$ selects the logit of token y. The term $\mathbf{e}_j^\top \mathcal{K}_{t,k} \Delta o_t^{(k)}(\rho)$ estimates the logit change of competing token j, while $\mathbf{e}_{y_t^\star}^\top \mathcal{K}_{t,k} \Delta o_t^{(k)}(\rho)$ estimates the change of the dense-model token. Their difference gives the predicted compression pressure from location k. Summing over all locations gives $\widehat{D}_{t,j}(\rho)$. We then compare this predicted compression pressure with the same decision margin $m_{t,j}$:

$$\widehat{\Gamma}_t(\rho) \triangleq \max_{j \neq y_t^\star} \frac{\widehat{D}_{t,j}(\rho)}{m_{t,j}}. \tag{7}$$

This has the same structure as $\Gamma_t^{\mathrm{obs}}(\rho)$ in Section 3.1, except that the observed compression pressure $D_{t,j}(\rho)$ is replaced by its first-order estimate $\widehat{D}_{t,j}(\rho)$. Therefore,

- $\widehat{\Gamma}_t(\rho) < 1$: the predictor expects the dense-model output to be preserved.
- $\widehat{\Gamma}_t(\rho) = 1$: the predicted output reaches the decision boundary.
- $\widehat{\Gamma}_t(\rho) > 1$: the predictor expects an output crossing.

We call $\widehat{\Gamma}_t(\rho)$ the predicted output crossing ratio. It is the decision quantity used by OCP. Unlike the exact ratio $\Gamma_t^{\mathrm{obs}}(\rho)$, which is available only after the compressed logits are obtained, $\widehat{\Gamma}_t(\rho)$ estimates

whether output crossing will occur before compressed-logit evaluation. This allows a candidate compression decision to be assessed before its compressed logits are observed. If output crossing is predicted, a milder compression level can be chosen. Otherwise, stronger compression can be used. Fig. 2 illustrates this difference between post-compression observation and pre-execution prediction. Since the estimate uses first-order propagation at the dense operating point, nonlinear effects away from this operating point are not included. Implementation details and the information used for pre-execution prediction are provided in Appendix C.3.1.

The OCP framework applies to different KV-cache compression operators through their local cache-read perturbations $\Delta o_t^{(k)}(\rho)$. Appendix B.2 details how the evaluated compression operators enter this common representation, including the set-dependent local perturbation induced by token eviction.

3.3 From Output Crossing to First Divergence

We can now predict output crossing at each decoding step using the predicted ratio $\widehat{\Gamma}_t(\rho)$. However, a generation contains many decoding steps, and a per-step prediction does not tell us when the compressed generation first departs from the dense generation. As decoding proceeds, we can track this prediction from one step to the next. The actual first divergence time τ_ρ marks the first step at which the two generations differ. In the same way, the first step at which output crossing is predicted marks the predicted first divergence. We call this step the predicted first divergence time, denoted by $\widehat{\tau}_\rho \triangleq \inf\left\{t : \widehat{\Gamma}_t(\rho) > 1\right\}$. This also makes generation length important. A compressed generation remains aligned with the dense generation only if output crossing does not occur at any earlier step. As the generation becomes longer, compression must therefore preserve the output over more decoding steps. A compression level that works well for a short generation may not behave the same way over a longer one.

We use $q_t(\rho)$ to denote how much perturbation step t can tolerate in the current direction before output crossing occurs. A small $q_t(\rho)$ therefore indicates a fragile decoding step. This raises a natural question: how should the compression level change with generation length to maintain the same level of output preservation? A longer generation contains more decoding steps and therefore more chances to encounter a fragile step. Once output crossing occurs at any decoding step, the compressed generation has already diverged from the dense generation. To maintain the same level of output preservation, compression must become more conservative as generation length increases. We use $\rho^\star(T)$ to denote this length-dependent compression boundary. It is the compression level that maintains a fixed reference level of output preservation over a generation of length T.

We describe these two factors with p and γ. The exponent p describes how the cache-read perturbation grows with compression level, while γ describes how frequently very fragile decoding steps occur. Under a stationary small-hazard approximation, these two effects together determine how the compression boundary changes with generation length:

$$\rho^\star(T) \propto T^\beta, \qquad \beta \approx -\frac{1}{p\gamma}. \tag{8}$$

The negative exponent means that the compression boundary becomes more conservative as the generation length increases. The derivation and the formal definitions of $q_t(\rho)$, $\rho^\star(T)$, p, and γ are given in Appendix B.3. This scaling follows from the stated assumptions rather than holding universally.

These results provide two complementary signals for adjusting KV-cache compression. The length-dependent boundary tells us how the compression level should change with generation length, rather than using the same level for short and long generations. The OCP framework evaluates candidate compression decisions at each decoding step through the predicted ratio. Together, they provide a basis for adjusting compression across generation lengths and decoding steps before output crossing occurs.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

We evaluate Qwen2.5-7B-Instruct (Yang et al., 2024a), Llama-3.1-8B-Instruct (Grattafiori et al., 2024), and Mistral-7B-Instruct-v0.3 (Mistral AI, 2024) on GSM8K (Cobbe et al., 2021), four LongBench (Bai et al., 2024) tasks, and RULER (Hsieh et al., 2024). Token eviction is the primary compression intervention, while value-side quantization, low-rank compression, and key-side perturbation are additionally used to test cross-operator generalization. Appendix C.1 and C.2 provide the detailed experimental configurations, evaluation protocols, and aggregation rules. All primary experiments use deterministic greedy decoding with identical tokenization and stopping rules. We use single-step counterfactual evaluation for output crossing analysis and sustained shared-prefix evaluation for first divergence analysis. The OCP framework does not use compressed logits or future generated tokens.

4.2 OUTPUT PRESERVATION BEYOND CACHE DISTORTION

We examine whether cache distortion is sufficient to determine output preservation. Fig. 3(a) plots the cache-read perturbation magnitude $\|\Delta O_t\|$ against the exact observed ratio Γ_t^{obs}. Larger perturbations are more often associated with output crossing, but output preservation and output crossing still coexist at similar perturbation magnitudes. Fig. 3(b) provides a stricter comparison. Each matched pair comes from the same prompt and decoding step and therefore has the same dense decoding state and decision margins. Although the two interventions differ in cache-read perturbation magnitude by at most 5%, one preserves the dense-model token while the other produces output crossing. These results confirm that cache distortion alone does not determine output preservation.

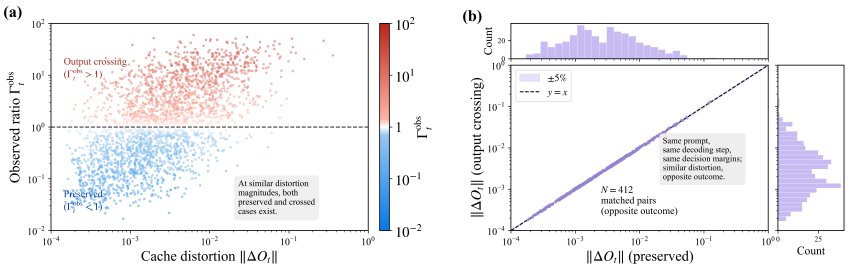

Figure 3: **Cache distortion does not determine output preservation.** (a) Cache distortion vs. output boundary for Qwen2.5-7B-Instruct on RULER. (b) Matched-distortion pairs with output preservation and output crossing for Llama-3.1-8B-Instruct on GSM8K.

4.3 PRE-EXECUTION OUTPUT CROSSING PREDICTION

The exact observed ratio Γ_t^{obs} determines output preservation only after the compressed logits are obtained. We therefore examine whether the predicted output crossing ratio $\widehat{\Gamma}_t$ can identify the same boundary before the compressed logits are obtained. Fig. 4(a) compares $\widehat{\Gamma}_t$ with Γ_t^{obs} near the output boundary. Most interventions fall on the same side of the threshold at 1 for both ratios, while the predicted ratio also tracks the observed ratio within each region. We define the false-safe rate (FSR) as the fraction of observed output crossings predicted as preservation, and the false-alarm rate (FAR) as the fraction of preserved outputs predicted as crossing. Fig. 4(b) compares $\widehat{\Gamma}_t$ with cache distortion and distortion normalized by the decision margin at a fixed FAR of 10%. The OCP framework yields a consistently lower FSR across all three models, showing that it misses fewer output crossings at the same false-alarm rate.

4.4 ABLATION OF OCP

We study which components of the OCP formulation are important for output crossing prediction. As shown in Table 1, normalizing cache distortion by the smallest decision margin, $\|\Delta O_t\|/m_t^{\min}$, reduces the overall median FSR from 6.71% to 4.21%, showing that the same perturbation magnitude can have different effects under different decision margins. Replacing perturbation magnitude

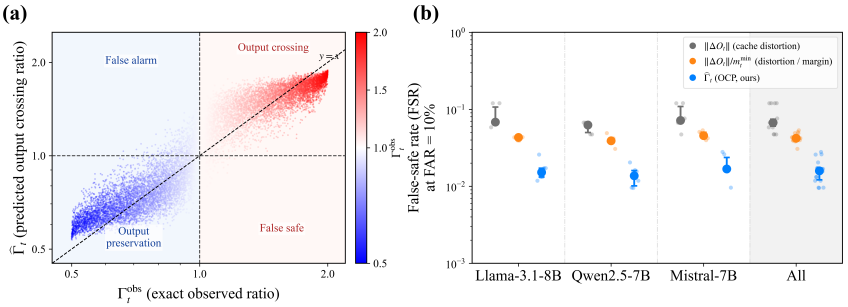

Figure 4: **Pre-execution output crossing prediction.** (a) Predicted output crossing ratio vs. exact observed ratio. (b) Output crossing prediction at a fixed false-alarm rate.

with the predicted compression pressure for the competing token with the smallest decision margin, $\widehat{D}_{t,j_t^{\min}}/m_t^{\min}$, further reduces the FSR to 2.60%, indicating that the direction of the compression-induced change provides information that magnitude alone cannot capture. When this direction is shuffled while the perturbation magnitudes are preserved, the FSR rises to 4.09%. Finally, the full predicted output crossing ratio $\widehat{\Gamma}_t$ achieves the lowest FSR of 1.59%. The same ordering is observed for all three models. These results show that the prediction gain comes from both the direction of the compression-induced perturbation and its comparison with token-specific decision margins.

Table 1: **Ablation of OCP.** All entries report the FSR at a fixed FAR=10%. Lower is better. Each model column reports the median across the evaluated tasks, while *All* reports the median across all models and tasks.

Index	Prediction score	Llama-3.1-8B	Qwen2.5-7B	Mistral-7B	All
1	$\|\Delta O_t\|$	6.83%	6.26%	7.17%	6.71%
2	$\|\Delta O_t\|/m_t^{\min}$	4.32%	3.90%	4.55%	4.21%
3	$\widehat{D}_{t,j_t^{\min}}/m_t^{\min}$	2.74%	2.43%	2.95%	2.60%
4	Direction-shuffled $\widehat{\Gamma}_t$	4.05%	3.79%	4.21%	4.09%
5	$\widehat{\Gamma}_t$ (Ours, OCP)	**1.52%**	**1.37%**	**1.68%**	**1.59%**

4.5 First Divergence Prediction

We next examine whether repeated per-step output crossing prediction can locate first divergence during generation. Fig. 5(a) compares the predicted first divergence time $\widehat{\tau}_\rho$ with the first divergence time τ_ρ. Among generations for which both first divergence times are observable, covering 73.0% of the evaluated generations, the exact-hit rate is 38.6%, and the median absolute error is 4 decoding steps. Additional observability statistics and per-model results are provided in Appendix D.2. Fig. 5(b) further shows the signed prediction error $\widehat{\tau}_\rho - \tau_\rho$. The errors are concentrated around zero for all three models, while both early and late predictions become less frequent as the error magnitude increases. These results show that per-step output crossing prediction can locate when the compressed generation first diverges from the dense generation.

4.6 Generalization Across Compression Operators

We test whether the same OCP formulation extends beyond token eviction. Table 2 shows a consistent advantage for OCP under value-side quantization, low-rank compression, and key-side perturbation, as well as token eviction. Relative to $\|\Delta O_t\|/m_t^{\min}$, the OCP framework reduces the median FSR by 2.62–2.85 percentage points for token eviction, value-side quantization, and key-side perturbation, and by 2.36 percentage points for low-rank compression. Although low-rank compression remains the most challenging operator, the OCP formulation still achieves the lowest FSR for every operator. Per-model results across the four compression operators are provided in Appendix D.4. These results show that the predicted output crossing ratio remains effective when the source of the local cache-read perturbation changes, rather than being specific to token eviction.

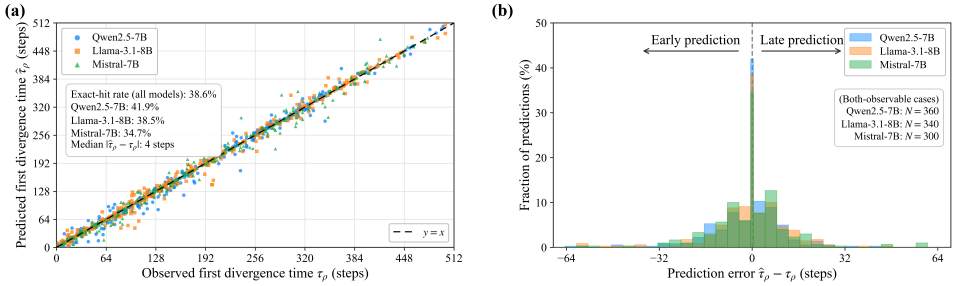

Figure 5: **First divergence prediction.** (a) Predicted vs. observed first divergence time. (b) Signed first divergence error.

Table 2: **Generalization across compression operators.** FSR is reported at FAR $= 10\%$, with lower values indicating better prediction. Results are medians across all models and tasks. ΔFSR denotes the absolute FSR reduction of OCP relative to $\|\Delta O_t\|/m_t^{\min}$, measured in percentage points (pp), with larger values indicating greater improvement.

Compression operator	$\|\Delta O_t\|$	$\|\Delta O_t\|/m_t^{\min}$	$\widehat{\Gamma}_t$ (Ours, OCP)	ΔFSR $\uparrow$
Token eviction	6.71%	4.21%	**1.59%**	+2.62 pp
Value-side quantization	7.36%	4.88%	**2.03%**	+2.85 pp
Low-rank compression	8.14%	5.27%	**2.91%**	+2.36 pp
Key-side perturbation	7.02%	4.53%	**1.84%**	+2.69 pp

4.7 Generation-Length Dependence

A compression level that preserves the dense-model output over a short generation need not remain sufficient for a longer one. Longer generations contain more decoding steps and therefore more opportunities to encounter very fragile decoding steps, suggesting that the compression boundary should become more conservative as generation length increases. We study this dependence through the length-dependent compression boundary $\rho^\star(T)$ defined in Section 3.3. Fig. 6(a) shows that $\rho^\star(T)$ decreases consistently as generation length increases. Fig. 6(b) further shows that this dependence follows the conditional scaling in Eq. 8. The median empirical exponents are -0.52, -0.41, and -0.28 for Qwen2.5-7B, Llama-3.1-8B, and Mistral-7B, respectively. The corresponding fits achieve R^2_{scale} values between 0.92 and 0.96, supporting the conditional power-law approximation over the evaluated range.

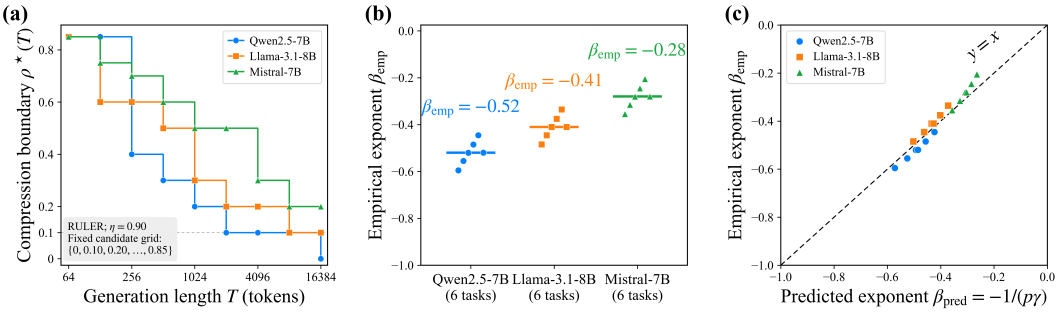

Figure 6: **Generation-length dependence of the compression boundary.** (a) Compression boundary vs. generation length. (b) Empirical scaling exponent. (c) Predicted vs. empirical scaling exponent.

We further evaluate whether the observed scaling agrees with the relation in Section 3.3. The parameters p and γ are estimated independently and give the predicted exponent $\beta_{\text{pred}} = -1/(p\gamma)$. As shown in Fig. 6(c), the predicted exponents broadly follow the empirical exponents across models and tasks. Table 3 makes this comparison explicit. The predicted exponents are -0.49, -0.43, and -0.31, compared with empirical values of -0.52, -0.41, and -0.28, with absolute differences of

0.02–0.03. Additional robustness results across reference preservation levels are provided in Appendix D.3. The compression boundary therefore becomes more conservative as generation length increases.

Table 3: **Scaling mechanism.** The predicted exponent is $\beta_{\text{pred}} = -1/(p\gamma)$, where p and γ are estimated independently. The empirical exponent β_{emp} is obtained from the observed scaling of the compression boundary with generation length. R^2_{scale} measures the empirical scaling fit, and $|\Delta\beta| = |\beta_{\text{pred}} - \beta_{\text{emp}}|$.

Model	p	γ	β_{pred}	β_{emp}	$R^2_{\text{scale}} \uparrow$	$\lvert\Delta\beta\rvert \downarrow$
Qwen2.5-7B	1.56	1.31	-0.49	-0.52	0.94	0.03
Llama-3.1-8B	1.48	1.57	-0.43	-0.41	0.96	0.02
Mistral-7B	1.37	2.39	-0.31	-0.28	0.92	0.03

4.8 OCP-Guided Compression

We integrate OCP with four representative compression methods on Llama-3.1-8B-Instruct using reference-conditioned evaluation under matched average KV-memory usage. Step-wise compression decisions are made relative to the dense reference trajectory rather than a standalone persistent compressed-cache state. Appendices C.3.2 and C.3.3 provide the selection rule and resource accounting. Table 4 reports the median first divergence time on GSM8K, LongBench, and RULER. The LongBench result is the median across the four evaluated tasks. For each benchmark, $\Delta\tau_{\text{med}}$ is the increase in median first divergence time of the OCP-guided variant over its corresponding baseline. The last column reports the mean of these increases across the three benchmarks. As shown

Table 4: **OCP-guided compression.** All experiments use $\rho_{\max} = 0.85$. Each entry reports the median first divergence time in decoding steps. OCP-guided variants are matched to their corresponding baselines in average KV-memory usage, with less than 2% mismatch. All results are obtained under our evaluation protocol.

Compression operator	Method	Median first divergence time ↑			$\Delta\tau_{\text{med}} \uparrow$
		GSM8K	LongBench	RULER	
Eviction	H2O (Zhang et al., 2023)	214	168	151	–
	H2O + OCP	258	204	184	+37.7
Value-side quantization	AsymKV (Tao et al., 2025)	251	196	177	–
	AsymKV + OCP	287	225	204	+30.7
Low-rank compression	Palu (Chang et al., 2025)	229	181	164	–
	Palu + OCP	257	203	185	+23.7
Key-side perturbation	SpectrumQuant (Wang et al., 2026)	263	207	188	–
	SpectrumQuant + OCP	299	235	215	+30.3

in Table 4, the OCP framework delays first divergence for all four compression methods, with mean increases of 23.7–37.7 decoding steps across the three benchmarks. These results show that the OCP framework's prediction advantage translates into later first divergence when used with existing compression methods. Runtime overhead is 6.8–8.6% (Appendix D.4).

5 Conclusion

This work studies KV-cache compression through output preservation and first divergence. We show that cache distortion alone does not determine whether the dense-model token is preserved, because compression also depends on how the induced change acts relative to the decision margin. This leads to an exact characterization through output crossing and to the Output Crossing Predictor (OCP), which uses the local cache-read perturbation and local sensitivity of the dense model to predict output crossing before compressed logits are observed. Extending the same prediction across decoding provides an estimate of first divergence time. Our results further show that the same formulation applies across different compression operators and that the compression boundary becomes more conservative as generation length increases, providing a common framework for reasoning about output preservation and first divergence under KV-cache compression.

AI USE STATEMENT

Generative AI tools were used for language editing, rewriting, presentation, and occasional assistance with experimental organization. We also used generative AI tools to assist with code implementation, debugging, experimental scripting, and parts of the experiment-execution workflow. The research questions, theoretical development, experimental design, interpretation of results, and scientific conclusions were determined by the authors. All AI-assisted code and text were reviewed and validated by the authors. The authors take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Appendix B provides the formal derivations and assumptions underlying the theoretical results. Appendix C specifies the models, datasets, compression interventions, evaluation protocols, aggregation rules, OCP computation, and resource accounting used in the experiments, and Appendix D provides additional robustness and per-model results. These details are provided to support reproduction of the theoretical and empirical results reported in the paper.

REFERENCES

Muhammad Adnan, Akhil Arunkumar, Gaurav Jain, Prashant J 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.

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. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 3119–3137, 2024.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Yucheng Li, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Junjie Hu, and Wen Xiao. PyramidKV: Dynamic KV cache compression based on pyramidal information funneling. In *Second Conference on Language Modeling*, 2025.

Chi-Chih Chang, Wei-Cheng Lin, Chien-Yu Lin, Chong-Yan Chen, Yu-Fang Hu, Pei-Shuo Wang, Ning-Chi Huang, Luis Ceze, Mohamed Abdelfattah, and Kai-Chiang Wu. Palu: Kv-cache compression with low-rank projection. In *International Conference on Learning Representations*, volume 2025, pp. 50222–50249, 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

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.

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. *Advances in Neural Information Processing Systems*, 38:113152–113188, 2025.

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. In *International Conference on Learning Representations*, 2025.

Raghavv Goel, Junyoung Park, Mukul Gagrani, Dalton Jones, Matthew J Morse, Matthew Harper Langston, Christopher Lott, and Mingu Lee. Caote: Optimizing kv cache memory through attention output error-based token eviction. In *ICLR 2026 Workshop on Memory for LLM-Based Agentic Systems*, 2026.

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

Yefei He, Luoming Zhang, Weijia Wu, Jing Liu, Hong Zhou, and Bohan Zhuang. Zipcache: Accurate and efficient KV cache quantization with salient token identification. *Advances in Neural Information Processing Systems*, 37:68287–68307, 2024.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W Mahoney, Yakun S Shao, Kurt Keutzer, and Amir Gholami. Kvquant: Towards 10 million context length LLM inference with KV cache quantization. *Advances in Neural Information Processing Systems*, 37:1270–1303, 2024.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

Hao Kang, Qingru Zhang, Souvik Kundu, Geonhwa Jeong, Zaoxing Liu, Tushar Krishna, and Tuo Zhao. Gear: An efficient error reduction framework for KV cache compression in LLM inference. In *ENLSP*, pp. 305–321, 2024.

Wonbeom Lee, Jungi Lee, Junghwan Seo, and Jaewoong Sim. InfiniGen: Efficient generative inference of large language models with dynamic KV cache management. In *18th USENIX symposium on operating systems design and implementation (OSDI 24)*, pp. 155–172, 2024.

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. *Advances in Neural Information Processing Systems*, 37:22947–22970, 2024.

Akide Liu, Jing Liu, Zizheng Pan, Yefei He, Gholamreza Haffari, and Bohan Zhuang. Minicache: KV cache compression in depth dimension for large language models. *Advances in Neural Information Processing Systems*, 37:139997–140031, 2024a.

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*, 36:52342–52364, 2023.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. KIVI: A tuning-free asymmetric 2bit quantization for KV cache. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32332–32344. PMLR, 2024b.

Mistral AI. Model card for Mistral-7B-Instruct-v0.3. Hugging Face model card, 2024.

Utkarsh Saxena, Gobinda Saha, Sakshi Choudhary, and Kaushik Roy. Eigen attention: Attention in low-rank space for KV cache compression. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 15332–15344, 2024.

Hanlin Tang, Yang Lin, Jing Lin, Qingsen Han, Danning Ke, Shikuan Hong, Yiwu Yao, and Gongyi Wang. Razorattention: Efficient KV cache compression through retrieval heads. In *International Conference on Learning Representations*, volume 2025, pp. 16632–16646, 2025.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. QUEST: Query-aware sparsity for efficient long-context LLM inference. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 47901–47911. PMLR, 2024.

Qian Tao, Wenyuan Yu, and Jingren Zhou. Asymkv: Enabling 1-bit quantization of KV cache with layer-wise asymmetric quantization configurations. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 2316–2328, 2025.

Yuxuan Tian, Zihan Wang, Yebo Peng, Aomufei Yuan, Zhiming Wang, Bairen Yi, Xin Liu, Yong Cui, and Tong Yang. Keepkv: Achieving periodic lossless KV cache compression for efficient LLM inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 33259–33267, 2026.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Xuefei Wang, Haoyu Tang, Tianyuan Liang, Zhibin Wang, Yupeng Hu, and Weili Guan. Resonating with rope: Spectral quantization for high-fidelity key cache compression. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 37328–37348, 2026.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, volume 2024, pp. 21875–21895, 2024.

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. In *International Conference on Learning Representations*, 2025.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.

June Yong Yang, Byeongwook Kim, Jeongin Bae, Beomseok Kwon, Gunho Park, Eunho Yang, Se Jung Kwon, and Dongsoo Lee. No token left behind: Reliable KV cache compression via importance-aware mixed precision quantization. *arXiv preprint arXiv:2402.18096*, 2024b.

Rongzhi Zhang, Kuang Wang, Liyuan Liu, Shuohang Wang, Hao Cheng, Chao Zhang, and Yelong Shen. Lorc: Low-rank compression for LLMs KV cache with a progressive compression strategy. *arXiv preprint arXiv:2410.03111*, 2024a.

Yuxin Zhang, Yuxuan Du, Gen Luo, Yunshan Zhong, Zhenyu Zhang, Shiwei Liu, and Rongrong Ji. Cam: Cache merging for memory-efficient LLMs inference. In *Forty-first international conference on machine learning*, 2024b.

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.

A Related Work

We position our work along three closely related directions, focusing on how existing methods compress the KV cache, preserve information under approximate inference, and adapt compression decisions during generation. These perspectives clarify how OCP differs in predicting output crossing for a candidate compression intervention before the compressed logits are observed.

A.1 KV-Cache Compression

Existing KV-cache compression methods mainly reduce memory by discarding cached states or by representing them more compactly. For token eviction, Zhang et al. (2023) retained heavy hitters identified from accumulated attention scores, while Goel et al. (2026) selected tokens according to the error induced in the attention output. Quantization methods instead reduce the precision of cached keys and values. Liu et al. (2024b) introduced asymmetric low-bit KV-cache quantization, and Tao et al. (2025) further used layer-wise asymmetric configurations for more aggressive compression. Zhang et al. (2024b) reduced KV-cache memory by merging cached states. Low-rank methods provide another representation-level alternative: Chang et al. (2025) compressed the KV cache through low-rank projection. Wang et al. (2026) instead exploited spectral structure for high-fidelity key-cache compression. These methods primarily determine what cache information to retain or how to represent it under a memory budget. Our work addresses a different question: for a given candidate compression intervention, does the induced local cache-read perturbation preserve the dense-model output or cause output crossing? The OCP framework therefore complements existing compression operators rather than introducing another compression rule.

A.2 Output Preservation under Approximate Inference

Several studies have examined how approximation error in the KV cache affects subsequent computation and generation. Kang et al. (2024) reduced quantization error through low-rank and sparse error correction, while Yang et al. (2024b) used importance-aware mixed precision to preserve information that would otherwise be lost under cache compression. Goel et al. (2026) moved closer to the model output by measuring the attention-output error induced by token eviction, and Tian et al. (2026) studied lossless KV-cache compression and the accumulation of compression error across decoding steps. These approaches establish that controlling internal approximation error can improve the fidelity of compressed inference, but approximation error alone does not determine whether the dense-model output is preserved. Our work characterizes this distinction at the decoding decision itself: output preservation depends on whether the compression-induced change closes a dense-model decision margin, while output crossing occurs when a competing token gains sufficient relative logit advantage to overtake the dense-model token. This output-level formulation separates cache distortion from its effect on the decoding decision and provides the basis for analyzing first divergence over autoregressive generation.

A.3 Prediction and Adaptive KV-Cache Compression

Recent work has increasingly adapted KV-cache compression to the information available during inference. Li et al. (2024) used attention patterns observed before generation to identify prompt positions expected to remain important, while Feng et al. (2025) adapted eviction budgets across attention heads according to their importance. Tang et al. (2024) used the current query to identify critical KV pages, and Lee et al. (2024) dynamically identified cache entries required for subsequent computation. Adnan et al. (2024) selected key tokens for retention, whereas Tang et al. (2025) and Xiao et al. (2025) exploited differences among attention heads to determine where cache information should be preserved. These approaches adapt compression by predicting or identifying which cached information is likely to remain useful, or by allocating the available cache budget accordingly. The OCP framework predicts a different quantity: for a given candidate compression intervention, it estimates whether the induced local cache-read perturbation will cause output crossing before the compressed logits are observed. OCP therefore does not replace the underlying compression operator or predict which cache entries should be retained; it provides a pre-execution signal for evaluating candidate compression decisions at the output level.

B Additional Theoretical Details

This section provides the formal details underlying the analysis in Section 3. We derive the exact output crossing boundary, show how the evaluated compression operators enter the OCP formulation through local cache-read perturbations, and derive the conditional scaling of the length-dependent compression boundary.

B.1 Exact Output-Crossing Boundary

We derive the exact criterion for output preservation in Section 3.1. Let $y_t^\star$ be the unique dense-model token and

$$m_{t,j} = z_{t,y_t^\star} - z_{t,j} > 0, \qquad j \neq y_t^\star, \tag{9}$$

the decision margin for competing token j. After applying the candidate compression intervention, the compression pressure on token j is

$$D_{t,j}(\rho) = \left(\widetilde{z}_{t,j}^{(\rho)} - z_{t,j}\right) - \left(\widetilde{z}_{t,y_t^\star}^{(\rho)} - z_{t,y_t^\star}\right). \tag{10}$$

The compressed logit difference between the dense-model token and competing token j is therefore

$$\widetilde{z}_{t,y_t^\star}^{(\rho)} - \widetilde{z}_{t,j}^{(\rho)} = m_{t,j} - D_{t,j}(\rho). \tag{11}$$

Hence, the dense-model token remains strictly ahead of every competing token if and only if

$$D_{t,j}(\rho) < m_{t,j}, \qquad \forall j \neq y_t^\star. \tag{12}$$

Since every $m_{t,j}$ is positive under the unique-argmax assumption, this condition is equivalent to

$$\Gamma_t^{\mathrm{obs}}(\rho) = \max_{j \neq y_t^\star} \frac{D_{t,j}(\rho)}{m_{t,j}} < 1. \tag{13}$$

The strict inequalities identify the three cases used in Section 3.1. If $\Gamma_t^{\mathrm{obs}}(\rho) < 1$, every competing token remains below the dense-model token and the dense-model output is preserved. If $\Gamma_t^{\mathrm{obs}}(\rho) > 1$, at least one competing token gains more relative logit than its decision margin and an output crossing occurs. If $\Gamma_t^{\mathrm{obs}}(\rho) = 1$, at least one competing token exactly closes its decision margin, placing the compressed logits on the decision boundary; the resulting token then depends on the fixed tie-breaking rule. Therefore, under the unique-argmax assumption, $\Gamma_t^{\mathrm{obs}}(\rho) < 1$ is necessary and sufficient for output preservation, $\Gamma_t^{\mathrm{obs}}(\rho) > 1$ is necessary and sufficient for output crossing, and $\Gamma_t^{\mathrm{obs}}(\rho) = 1$ is the decision boundary.

B.2 Local Cache-Read Perturbations under Compression

The OCP formulation represents the evaluated compression operators through the local cache-read perturbations they induce. For cache-read location k, we use the common representation

$$\Delta o_t^{(k)}(\rho) = o_t^{(k)}\left(\widetilde{C}_t^{(\rho)}\right) - o_t^{(k)}(C_t). \tag{14}$$

For this local perturbation, the query entering cache-read location k is held at its dense-state value, so $\Delta o_t^{(k)}(\rho)$ isolates the direct change in the cache read induced by the candidate cache intervention at that location. Subsequent first-order propagation from this local change to the output logits is captured by $\mathcal{K}_{t,k}$ in Eq. 6. The four compression operators evaluated in Section 4.6 differ in how they modify the KV cache, but each produces a candidate local cache-read perturbation of this form:

$$\begin{aligned}
\text{Token eviction:} \quad & C_t \to \widetilde{C}_{t,\mathrm{evict}}^{(\rho)} \to \Delta o_t^{(k)}(\rho), \\
\text{Value-side quantization:} \quad & V_t \to \widetilde{V}_{t,\mathrm{quant}}^{(\rho)} \to \Delta o_t^{(k)}(\rho), \\
\text{Low-rank compression:} \quad & V_t \to \widetilde{V}_{t,\mathrm{rank}}^{(\rho)} \to \Delta o_t^{(k)}(\rho), \\
\text{Key-side perturbation:} \quad & K_t \to \widetilde{K}_{t,\mathrm{key}}^{(\rho)} \to \Delta o_t^{(k)}(\rho).
\end{aligned} \tag{15}$$

The operator-specific configurations that produce these cache changes are given in Appendix C.1. For token eviction, the local cache-read perturbation can depend on the entire retained or removed

set because attention weights are renormalized after multiple KV entries are removed. The resulting perturbation is therefore set-dependent rather than a simple sum of independent single-entry perturbations. This set dependence changes how $\Delta o_t^{(k)}(\rho)$ is produced, but not the common representation used by the OCP formulation.

Once the candidate local cache-read perturbation is obtained, the subsequent OCP computation has the same form across the evaluated compression operators. The dense-model local sensitivity propagates the perturbation to the predicted compression pressure,

$$\widehat{D}_{t,j}(\rho) = \sum_k (e_j - e_{y_t^\star})^\top \mathcal{K}_{t,k} \Delta o_t^{(k)}(\rho), \tag{16}$$

and the predicted output crossing ratio is then computed as in Section 3.2. Thus, the compression operator determines the source and structure of the local cache-read perturbation, while the OCP formulation applied after that perturbation is unchanged. This common interface is what allows the same OCP framework to be evaluated across token eviction, value-side quantization, low-rank compression, and key-side perturbation without introducing an operator-specific predictor.

B.3 First Divergence and Length-Dependent Compression Boundary

We formalize the length-dependent compression boundary in Section 3.3 using the perturbation tolerance of individual decoding steps. Let

$$\sigma_t(\rho) = \|\Delta O_t(\rho)\|_2 \tag{17}$$

denote the cache-read perturbation magnitude at step t. When $\sigma_t(\rho) = 0$, the local first-order perturbation vanishes, and output preservation holds under this characterization; the directional quantities below are therefore defined for $\sigma_t(\rho) > 0$. For $\sigma_t(\rho) > 0$, the directional tolerance $q_t(\rho)$ is the smallest perturbation magnitude, along the normalized candidate perturbation direction $u_t(\rho) = \Delta O_t(\rho)/\|\Delta O_t(\rho)\|_2$, required under the first-order propagation at the dense operating point to close one of the dense-model decision margins; specifically, $q_t(\rho) = \min_{j:d_{t,j}(\rho)>0} m_{t,j}/d_{t,j}(\rho)$, where $d_{t,j}(\rho) = (e_j - e_{y_t^\star})^\top J_t u_t(\rho)$, with $q_t(\rho) = +\infty$ if no competing token has positive directional sensitivity. Thus, $\sigma_t(\rho) < q_t(\rho)$, $\sigma_t(\rho) = q_t(\rho)$, and $\sigma_t(\rho) > q_t(\rho)$ correspond respectively to output preservation, the decision boundary, and output crossing under this local first-order characterization. For a generation length T, let E_T denote the eligibility event that the dense trajectory provides at least T valid decoding states. For a fixed reference preservation level $\eta \in (0, 1)$, we define

$$\rho^\star(T; \eta) = \sup\{\rho : \Pr(\tau_\rho > T \mid E_T) \geq \eta\}. \tag{18}$$

The main text suppresses the dependence on η and writes $\rho^\star(T)$. We use $\eta = 0.90$ in the main experiment and evaluate $\eta \in \{0.85, 0.90, 0.95\}$ in Appendix D.3.

The change of $\rho^\star(T)$ with generation length depends on the two factors described in Section 3.3. Over the evaluated compression range, we assume that the cache-read perturbation magnitude admits a local scale representation

$$\sigma_t(\rho) = \rho^p s_t(\rho), \tag{19}$$

where p describes the leading-order growth of the perturbation scale with compression level and the normalized magnitude $s_t(\rho)$ has a non-degenerate local distribution whose relevant lower-order moments remain finite and do not introduce additional leading-order power-law dependence on ρ. Very fragile decoding steps correspond to the lower tail of the directional tolerance. In the evaluated local regime, we approximate

$$\Pr(q_t(\rho) \leq x) \approx a(\rho) x^\gamma, \qquad x \to 0^+, \tag{20}$$

where γ describes how frequently very fragile decoding steps occur. Assuming that, over the evaluated local regime, the tail prefactor $a(\rho)$ and the joint lower-tail behavior of $s_t(\rho)$ and $q_t(\rho)$ do not introduce additional leading-order power-law dependence on ρ, the resulting per-step output crossing hazard, conditional on the shared prefix, satisfies

$$h(\rho) = \Pr(\sigma_t(\rho) > q_t(\rho) \mid \text{shared prefix}) \propto \rho^{p\gamma}, \tag{21}$$

where $h(\rho)$ is the local first-order crossing hazard induced by the OCP characterization. We use it as a conditional approximation to the per-step hazard underlying observed first divergence; the

resulting scaling prediction is evaluated independently against the empirical boundary defined from observed τ_ρ below. Under the stationary small-hazard approximation over the eligible trajectories,

$$\Pr(\tau_\rho > T \mid E_T) \approx \exp[-Th(\rho)]. \tag{22}$$

At the length-dependent compression boundary, this preservation probability is approximately η, which gives

$$\rho^\star(T;\eta) \propto \left(\frac{-\log \eta}{T}\right)^{1/(p\gamma)}. \tag{23}$$

For fixed η,

$$\rho^\star(T) \propto T^{-1/(p\gamma)}, \qquad \beta_{\text{pred}} = -\frac{1}{p\gamma}. \tag{24}$$

This scaling is conditional on the local perturbation scaling, the lower-tail approximation and the stated regularity conditions, and the stationary small-hazard regime rather than being a universal law.

The predicted and empirical exponents are obtained through separate estimation paths. We estimate the perturbation-scale exponent p from the log–log slope of the median cache-read perturbation magnitude against the nonzero candidate compression levels, using the median as a robust estimator of the characteristic scale in Eq. 19. We estimate γ independently from the lower tail of the directional tolerance distribution defined above, using the lowest 10% of the $q_t(\rho)$ values as the main fitting range and 7.5% and 12.5% as sensitivity checks. Neither fit uses the observed length-dependent compression boundary or β_{emp}. In the experiments, the empirical boundary is estimated from the observed first divergence times over the eligible trajectories on the fixed candidate grid as

$$\widehat{\rho}^\star(T;\eta) = \max\left\{\rho_m : \Pr(\tau_{\rho_m} > T \mid E_T) \geq \eta\right\}. \tag{25}$$

The empirical exponent β_{emp} is then estimated separately from the log–log relation between the positive values of $\widehat{\rho}^\star(T;\eta)$ and T; zero-boundary points are retained in the boundary analysis but excluded from the logarithmic fit. Therefore, $\beta_{\text{pred}} = -1/(p\gamma)$ is determined independently of the empirical generation-length scaling, so the comparison in Section 4.7 directly tests the conditional scaling relation rather than fitting it to the observed boundary.

C Experimental Setup and Implementation Details

This section provides the experimental and implementation details needed to reproduce the evaluation in Section 4. We specify the model, dataset, and compression configurations, formalize the evaluation protocols and aggregation rules, and describe the practical computation, information usage, and resource accounting of the OCP framework.

C.1 Models, Datasets, and Compression Interventions

Models and decoding setup. We evaluate Qwen2.5-7B-Instruct (Yang et al., 2024a), Llama-3.1-8B-Instruct (Grattafiori et al., 2024), and Mistral-7B-Instruct-v0.3 (Mistral AI, 2024) using deterministic greedy decoding with batch size 1 and BF16 precision on one NVIDIA A100 80GB GPU. Each model uses its native tokenizer and official chat template. Generation terminates at EOS or the applicable task-specific generation limit, subject to an overall cap of 16,384 generated tokens. Dense and compressed runs use identical prompts, tokenization, stopping rules, and generation limits, ensuring the same decoding configuration for comparison.

Datasets and evaluation samples. We use 180 GSM8K (Cobbe et al., 2021) prompts, 320 LongBench (Bai et al., 2024) examples, and 180 RULER (Hsieh et al., 2024) examples per model, giving a raw pool of 680 prompts or examples per model. The LongBench evaluation contains four tasks: NarrativeQA, HotpotQA, GovReport, and TREC, with 80 examples from each task. GSM8K is used for first divergence and OCP-guided compression, LongBench for output crossing, first divergence, and OCP-guided compression, and RULER for single-step output crossing and cross-operator generalization. For single-step counterfactual evaluation, we collect 4,000 decoding states from each benchmark, yielding 12,000 states per model; multiple candidate compression interventions are evaluated from each state under the protocol in Appendix C.2. The generation-length experiments use $T \in \{64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384\}$, with a reference preservation level of

$\eta = 0.90$ in the main experiment and $\eta \in \{0.85, 0.90, 0.95\}$ in the robustness analysis. For each generation length T, we include only trajectories that provide at least T valid dense decoding states; decoding steps beyond EOS or the applicable generation limit are not treated as output preservation.

Compression interventions. We use token eviction as the primary compression intervention and additionally evaluate controlled value-side quantization, low-rank compression, and key-side perturbation for the cross-operator analysis in Section 4.6. These four interventions use the fixed candidate grid

$$\rho \in \{0, 0.10, 0.20, 0.30, 0.40, 0.50, 0.60, 0.70, 0.75, 0.80, 0.85\}, \tag{26}$$

where $\rho = 0$ corresponds to the dense cache and larger ρ represents a stronger compression intervention. For token eviction, we use an H2O-style accumulated attention score (Zhang et al., 2023) to rank KV entries. Let s_i denote the accumulated attention score of KV entry i. The ranking is fixed for a given decoding state and does not depend on ρ. Entries are sorted by s_i in descending order. With N denoting the number of cached KV entries in this decoding state, the top

$$N_{\text{keep}}(\rho) = \max\{1, \lceil (1-\rho)N \rceil\} \tag{27}$$

KV entries are retained. Value-side quantization uses per-token symmetric uniform quantization with

$$b(\rho) = \begin{cases} \text{BF16}, & \rho = 0, \\ \max\{2, \text{round}(16(1-\rho))\} \text{ bits}, & \rho > 0, \end{cases} \tag{28}$$

and the resulting codes are bit-packed; KV-memory accounting includes both the packed codes and quantization metadata. Low-rank compression applies truncated SVD to the value-cache matrix independently for each layer and attention head, retaining

$$r_{\text{keep}}(\rho) = \max\{1, \lceil (1-\rho) r_{\text{full}} \rceil\}. \tag{29}$$

For key-side perturbation,

$$\widetilde{k}_i^{(\rho)} = k_i + \rho \frac{\|k_i\|_2}{\|\epsilon_i\|_2} \epsilon_i, \tag{30}$$

where ϵ_i is sampled from a standard Gaussian distribution and its normalized direction is fixed for each key across all candidate compression levels using a fixed random seed. Hence,

$$\frac{\|\widetilde{k}_i^{(\rho)} - k_i\|_2}{\|k_i\|_2} = \rho. \tag{31}$$

These definitions give ρ a common monotonic interpretation across the four interventions while preserving their different sources of local cache-read perturbation.

C.2 Evaluation Protocols

C.2.1 Single-Step Counterfactual Evaluation

For output crossing analysis, we adopt single-step counterfactual evaluation to isolate the effect of a candidate compression intervention at a given decoding step. For each prompt and decoding step t, we fix the prefix generated by the dense model. This fixes the dense decoding state, dense-model token $y_t^\star$, output logits z_t, and decision margins $m_{t,j}$, which are shared across all candidate interventions evaluated at that step. Each candidate compression intervention is evaluated from this same dense decoding state, and the compressed generation is not propagated to subsequent decoding steps.

The compressed output at the current step determines whether the dense-model token is preserved or an output crossing occurs. Specifically, output preservation corresponds to $\widetilde{y}_t^{(\rho)} = y_t^\star$ and $\Gamma_t^{\text{obs}}(\rho) < 1$, while output crossing corresponds to $\widetilde{y}_t^{(\rho)} \neq y_t^\star$ and $\Gamma_t^{\text{obs}}(\rho) > 1$. We use this protocol for the output crossing experiments in Sections 4.2–4.4 and for the cross-operator evaluation in Section 4.6. For the matched-distortion analysis in Section 4.2, candidate intervention pairs are formed only from the same prompt and decoding step, ensuring identical dense decoding states and decision margins. The main experiment requires their cache-read perturbation magnitudes to differ by at most 5%; Appendix D.1 further evaluates matching tolerances of 2.5% and 10%.

C.2.2 SUSTAINED SHARED-PREFIX EVALUATION

For first divergence analysis, we use sustained shared-prefix evaluation to compare the dense and compressed generations while they remain aligned. Unlike single-step counterfactual evaluation, this protocol runs autoregressively across decoding steps. Starting from the same prompt, the two generations share the same prefix until the first decoding step at which the compressed token differs from the dense-model token. This step defines the first divergence time τ_ρ. In parallel, the OCP framework is evaluated at each decoding step along the dense reference trajectory, including after the observed first divergence when needed, and the first step satisfying $\widehat{\Gamma}_t(\rho) > 1$ defines the predicted first divergence time $\widehat{\tau}_\rho$. The protocol therefore extends per-step output crossing prediction across decoding while maintaining a shared prefix before first divergence.

Both first divergence times are evaluated within a finite generation horizon. When both τ_ρ and $\widehat{\tau}_\rho$ occur within this horizon, their prediction error can be measured directly; these generations form the both-observable subset used for the exact-hit rate and median absolute error in Section 4.5. Generations for which only one or neither first divergence time is observable are retained as separate observability cases, as reported in Appendix D.2. The same sustained shared-prefix evaluation is used for the generation-length analysis in Section 4.7 and the OCP-guided compression evaluation in Section 4.8.

C.2.3 METRICS AND AGGREGATION

We evaluate output crossing prediction using the false-safe rate (FSR) and false-alarm rate (FAR). FSR is the fraction of observed output crossings predicted as output preservation, whereas FAR is the fraction of preserved outputs predicted as output crossing. The main experiments compare prediction scores at FAR$= 10\%$, while Appendix D.1 extends the comparison over the full FSR–FAR curve. For each reporting condition and prediction score, the operating threshold is selected using a held-out calibration split under the empirical constraint $\mathrm{FAR} \leq 10\%$ and is then frozen before evaluation on the disjoint test split. Test labels are not used for threshold selection. Condition-level FSR values are computed using these frozen thresholds and aggregated only afterward according to the rules below. As an additional diagnostic, applying OCP's natural decision boundary $\widehat{\Gamma}_t = 1$ directly to the pooled disjoint test records, without FAR-targeted threshold calibration, yields an FSR of 2.74% and FAR of 7.02%. The fixed-FAR results in the main experiments instead provide a common operating point for comparing prediction scores with different numerical scales. For cross-operator comparison, ΔFSR denotes the absolute FSR reduction of the OCP framework relative to margin-normalized distortion at the same FAR, measured in percentage points (pp).

For first divergence prediction, we report the exact-hit rate and median absolute error over the both-observable subset defined above. The exact-hit rate is the fraction satisfying $\widehat{\tau}_\rho = \tau_\rho$, and the median absolute error is computed from $|\widehat{\tau}_\rho - \tau_\rho|$ over the same generations.

Unless stated otherwise, results are aggregated by the median. Per-model output crossing results use the median across the evaluated tasks, while aggregate results use the median across all models and tasks. For OCP-guided compression, LongBench is summarized by the median across the four evaluated tasks. For each benchmark, $\Delta\tau_{\mathrm{med}}$ is the increase in median first divergence time of the OCP-guided variant over its corresponding baseline; the reported aggregate value is the mean of these increases across GSM8K, LongBench, and RULER.

C.3 OCP IMPLEMENTATION AND INFORMATION USAGE

C.3.1 PRACTICAL COMPUTATION AND INFORMATION USAGE

We compute the predicted output crossing ratio directly from the dense decoding state and a candidate compression intervention. Algorithm 1 summarizes the procedure. The candidate intervention first determines the local cache-read perturbations $\Delta o_t^{(k)}(\rho)$ at the affected cache-read locations from the current dense decoding state. These perturbations are concatenated into a joint direction $\Delta O_t(\rho)$, and a Jacobian–vector product at the dense operating point gives the predicted logit change $\widehat{\Delta z}_t(\rho)$. The predicted compression pressure $\widehat{D}_{t,j}(\rho)$ and predicted output crossing ratio $\widehat{\Gamma}_t(\rho)$ then follow from the same decision margins used in Section 3.2. Here, J_t is the joint form of the per-location sensitivities $\mathcal{K}_{t,k}$ defined in Section 3.2, so the practical computation implements the

Algorithm 1 Pre-execution computation of the predicted output crossing ratio.

Require: Dense decoding state at step t; candidate compression intervention at level ρ
Ensure: Predicted output crossing ratio $\widehat{\Gamma}_t(\rho)$

Obtain dense logits z_t and dense-model token $y_t^\star \leftarrow \arg\max_j z_{t,j}$
for each competing token $j \neq y_t^\star$ **do**
    $m_{t,j} \leftarrow z_{t,y_t^\star} - z_{t,j}$
end for
for each affected cache-read location k **do**
    Compute the candidate local cache-read perturbation

$$\Delta o_t^{(k)}(\rho) \leftarrow o_t^{(k)}(\widetilde{C}_t^{(\rho)}) - o_t^{(k)}(C_t)$$

end for
Concatenate the local perturbations:

$$\Delta O_t(\rho) \leftarrow \mathrm{concat}_k\, \Delta o_t^{(k)}(\rho)$$

Evaluate the Jacobian–vector product at the dense operating point:

$$\Delta \widehat{z}_t(\rho) \leftarrow J_t\, \Delta O_t(\rho)$$

for each competing token $j \neq y_t^\star$ **do**
    $\widehat{D}_{t,j}(\rho) \leftarrow \widehat{\Delta z}_{t,j}(\rho) - \widehat{\Delta z}_{t,y_t^\star}(\rho)$
    $r_{t,j}(\rho) \leftarrow \widehat{D}_{t,j}(\rho)/m_{t,j}$
end for
$\widehat{\Gamma}_t(\rho) \leftarrow \max_{j \neq y_t^\star} r_{t,j}(\rho)$
return $\widehat{\Gamma}_t(\rho)$

same first-order propagation without explicitly constructing each full Jacobian. The Jacobian–vector product directly produces the vocabulary-dimensional predicted logit perturbation $\widehat{\Delta z}_t \in \mathbb{R}^V$, after which the competitor ratios are computed vectorially over the full vocabulary; no top-M competitor truncation or vocabulary approximation is used. When multiple candidate compression levels are evaluated at the same decoding step, they share the same dense operating point and local sensitivity, and their perturbation vectors are propagated jointly using a batched multi-vector Jacobian–vector product rather than separate full model evaluations. Here, "pre-execution" refers specifically to prediction before compressed-logit evaluation; constructing the candidate local cache-read perturbations and propagating them through the batched multi-vector Jacobian–vector product are part of the OCP computation.

Table 5 makes the information boundary of this computation explicit. The OCP framework relies on the dense cache, dense logits, dense-model token, decision margins, candidate compression level, local cache-read perturbation, and joint dense-model sensitivity, all available before compressed-logit evaluation. The compressed logits $\widetilde{z}_t^{(\rho)}$ do not enter the computation of $\widehat{\Gamma}_t(\rho)$; they are used only after compressed execution to construct the observed ratio $\Gamma_t^{\mathrm{obs}}(\rho)$ and the ground-truth output crossing label. Future generated tokens and future compressed-branch states are likewise not used for prediction. Therefore, the OCP framework predicts output crossing from dense-model information and the candidate local cache-read perturbation before the compressed logits are evaluated, preserving the distinction between pre-execution prediction and post-compression observation.

C.3.2 OCP-Guided Compression

We use the output crossing prediction in Section 3.2 to guide the compression level of existing compression methods. The OCP-guided evaluation in Section 4.8 uses the actual implementations of H2O, AsymKV, Palu, and SpectrumQuant, rather than the controlled interventions used for the cross-operator analysis in Section 4.6. When output preservation is predicted, stronger compression can be retained; when output crossing is predicted, a milder setting is selected. To implement this rule, we use a fixed discrete set of candidate compression levels,

$$0 = \rho_0 < \rho_1 < \cdots < \rho_M \leq \rho_{\max}, \qquad \rho_{\max} = 0.85. \tag{32}$$

Table 5: **Information used for pre-execution output crossing prediction.** Quantities above the midrule are available to the OCP framework before compressed-logit evaluation, whereas compressed logits, the observed ratio, and future decoding information are not used for prediction.

Quantity	Source	Available pre-execution	Used by OCP prediction
Dense cache C_t	Dense decoding state	Yes	Yes
Dense logits z_t	Dense decoding state	Yes	Yes
Dense-model token $y_t^\star$	Dense logits	Yes	Yes
Decision margins $m_{t,j}$	Dense logits	Yes	Yes
Candidate compression level ρ	Candidate intervention	Yes	Yes
Local cache-read perturbation $\Delta o_t^{(k)}(\rho)$	Candidate intervention	Yes	Yes
Joint dense-model sensitivity J_t	Dense decoding state	Yes	Yes
Compressed logits $\widetilde{z}_t^{(\rho)}$	Compressed-logit evaluation	No	No
Observed ratio $\Gamma_t^{\mathrm{obs}}(\rho)$	Compressed logits	No	No
Future generated tokens	Future decoding	No	No
Future compressed-branch states	Future decoding	No	No

The candidate levels are fixed before decoding, and each compression operator maps the common parameter ρ to its own operator-specific configuration. At each decoding step, all candidate compression interventions are defined with respect to the current reference cache state C_t, rather than by modifying the compressed cache state produced at the previous step. After selecting ρ_t^{OCP}, the corresponding compression intervention is instantiated from this same reference state for the current-step compressed evaluation. The dense reference trajectory is maintained separately across decoding steps; thus, the guided evaluation measures step-wise compression decisions relative to the reference trajectory rather than irreversible transitions of a standalone compressed-cache state. In this way, the OCP framework evaluates candidate compression interventions without changing the underlying compression operator.

At decoding step t, the predicted output crossing ratio $\widehat{\Gamma}_t(\rho_m)$ is computed for each nonzero candidate level. The OCP-guided variant then selects the strongest candidate predicted to preserve the dense-model output,

$$\rho_t^{\mathrm{OCP}} = \max\left\{\rho_m : \widehat{\Gamma}_t(\rho_m) < 1,\ 1 \le m \le M\right\}. \tag{33}$$

If no nonzero candidate satisfies this condition, we set $\rho_t^{\mathrm{OCP}} = 0$, corresponding to no candidate compression, or equivalently the most conservative available setting. Because the candidate levels are bounded by $\rho_{\max} = 0.85$, stronger candidates are neither evaluated nor selected. The same selection rule is applied to all four compression methods in Section 4.8. Thus, the OCP framework guides the compression level through predicted output crossing while leaving the original compression operator unchanged.

C.3.3 Resource Accounting

We match each OCP-guided variant to its corresponding baseline using actual KV-memory usage measured in bytes. After determining the average KV-memory usage of the OCP-guided variant, we select the fixed baseline compression configuration with the closest average KV-memory usage, without using first divergence outcomes for matching. Let $M_{\text{OCP-guided}}$ and M_{baseline} denote the average number of KV-cache bytes occupied over a complete generation under the same evaluation protocol. The KV-memory mismatch is defined as

$$\text{KV-memory mismatch} = \frac{|M_{\text{OCP-guided}} - M_{\text{baseline}}|}{M_{\text{baseline}}} \times 100\%. \tag{34}$$

The KV-memory matching compares the compressed-cache footprint of the baseline and OCP-guided variants; the separately maintained dense reference state required by the evaluation protocol is not included in this matched compressed-cache footprint. This byte-level accounting provides

a common resource measure across different compression operators rather than relying on token counts or operator-specific normalized quantities. The resulting mismatch is used to verify the matched average KV-memory usage in Section 4.8.

We measure the runtime overhead of each OCP-guided variant relative to its corresponding baseline using end-to-end generation runtime over the same complete generation horizon. The OCP-guided runtime includes candidate perturbation construction, the batched multi-vector Jacobian–vector product, and candidate selection. Let $T_{\mathrm{OCP-guided}}$ and T_{baseline} denote the runtimes measured under this common evaluation protocol. The runtime overhead is

$$\text{Runtime overhead} = \frac{T_{\text{OCP-guided}} - T_{\text{baseline}}}{T_{\text{baseline}}} \times 100\%. \tag{35}$$

Together, these quantities provide the resource-matching and runtime-cost measurements reported in Appendix D.4.

D Additional Experimental Results

This section provides additional evidence for the experiments in Section 4, focusing on robustness, observability, cross-operator generalization, and the practical use of the OCP framework. The results further examine the evaluation conditions and aggregate findings reported in the main experiments.

D.1 Robustness of Output Crossing Prediction

We further test whether the matched-distortion finding in Section 4.2 depends on the $\pm 5\%$ tolerance used to identify interventions with similar cache distortion. We vary this matching tolerance across 2.5%, 5%, and 10% to assess whether the coexistence of output preservation and output crossing depends on this choice. As shown in Fig. 7(a), pooling matched pairs across the evaluated models and tasks yields 1,842, 4,781, and 11,602 pairs under the three tolerances, respectively. In all cases, output preservation and output crossing coexist at similar cache distortion magnitudes. The matched-distortion finding therefore remains under both stricter and looser matching tolerances rather than being specific to the $\pm 5\%$ criterion used in the main experiment.

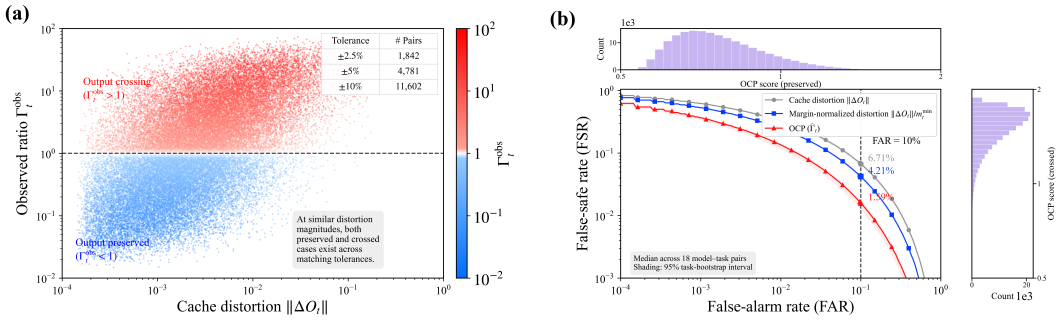

Figure 7: **Robustness of output crossing prediction.** (a) Matched-distortion analysis across different matching tolerances. Output preservation and output crossing coexist at similar cache distortion magnitudes. (b) FSR–FAR curves comparing cache distortion, margin-normalized distortion, and the OCP framework. The OCP framework maintains a lower FSR across the evaluated FAR range.

The prediction comparison in Section 4.3 uses FAR$= 10\%$ as the operating point for comparing FSR. Fig. 7(b) extends this evaluation across the full FSR–FAR curve. Across the evaluated FAR range, the OCP framework consistently achieves lower FSR than cache distortion and margin-normalized distortion. At FAR$= 10\%$, the corresponding FSR values are 6.71%, 4.21%, and 1.59%, respectively, reproducing the operating-point results in the main experiment. The prediction advantage of the OCP framework therefore extends beyond the single FAR$= 10\%$ operating point.

We further examine whether the first-order prediction degrades as the compression intervention becomes stronger. Using the same held-out calibration protocol as in Appendix C.2.3, we stratify the

Table 6: **Detailed results for first divergence prediction.** "Both observable" means that both the observed and predicted first divergence times occur within the evaluation horizon. "Observed only" means that only the observed first divergence time occurs within the horizon, while "Predicted only" means that only the predicted first divergence time occurs; "Neither observable" means that neither occurs within the horizon. The both-observable rate is the proportion of all evaluated generations for which both first divergence times are observable. Exact hit and median absolute error (MedAE) are computed on these generations.

Evaluation item	Model		
	Qwen2.5-7B	Llama-3.1-8B	Mistral-7B
Total generations (count)	480	460	430
Both observable (count)	360	340	300
Observed only (count)	47	51	55
Predicted only (count)	38	41	44
Neither observable (count)	35	28	31
Both-observable rate	75.0%	73.9%	69.8%
Exact-hit rate (%) ↑	41.9%	38.5%	34.7%
MedAE (steps) ↓	3	4	5

single-step evaluation by compression level ρ. The OCP FSR increases gradually from 1.05% at $\rho = 0.10$ to 2.18% at $\rho = 0.85$, consistent with the increasing difficulty of first-order prediction farther from the dense operating point. However, OCP remains better than margin-normalized distortion at every evaluated compression level: the corresponding FSR reduction ranges from 2.05 to 3.02 percentage points over $\rho \in [0.10, 0.85]$. Thus, stronger interventions increase absolute prediction difficulty, but the directional advantage of OCP persists throughout the evaluated compression range.

D.2 Additional Results for First Divergence Prediction

Evaluating the error in first divergence time requires both the observed and predicted first divergence times to occur within the evaluation horizon. Table 6 therefore reports how often this condition is satisfied and explicitly accounts for the remaining generations. The two first divergence times are both observable for 75.0%, 73.9%, and 69.8% of the generations for Qwen2.5-7B, Llama-3.1-8B, and Mistral-7B, respectively, giving a pooled both-observable rate of 73.0%. The remaining generations are accounted for by the observed-only, predicted-only, and neither-observable cases and are excluded from the error statistics. Across all 1,370 evaluated generations, 153 (11.2%) are observed-only cases, 123 (9.0%) are predicted-only cases, and 94 (6.9%) have neither divergence time observable within the evaluation horizon. Within the both-observable subset, the exact-hit rates are 41.9%, 38.5%, and 34.7% for Qwen2.5-7B, Llama-3.1-8B, and Mistral-7B, respectively. The corresponding median absolute errors are 3, 4, and 5 decoding steps. The similar both-observable rates and consistent prediction pattern across models show that the first divergence results in Section 4.5 are supported by a substantial observable subset across all three models rather than by a small subset or a single model.

D.3 Robustness of Generation-Length Scaling

We estimate the parameters p and γ independently of the empirical scaling fit and keep them fixed while varying the reference level of output preservation. Therefore, the predicted exponent $\beta_{\mathrm{pred}} = -1/(p\gamma)$ is determined independently of β_{emp}. The agreement between the predicted and empirical exponents therefore does not result from fitting p and γ to the observed scaling. This separation also allows us to assess whether the empirical scaling depends on the particular reference level used to define the length-dependent compression boundary.

Table 7 reports the empirical exponents at reference preservation levels of 0.85, 0.90, and 0.95. As the reference level increases, the empirical exponents change smoothly from -0.50 to -0.54 for Qwen2.5-7B, from -0.39 to -0.43 for Llama-3.1-8B, and from -0.27 to -0.30 for Mistral-7B. Across all evaluated reference levels, the empirical exponents remain close to the predicted

Table 7: **Generation-length scaling across different reference preservation levels.** The predicted and empirical exponents remain close across all evaluated reference levels.

Model	β_{pred}	β_{emp}		
		$\eta = 0.85$	$\eta = 0.90$	$\eta = 0.95$
Qwen2.5-7B	-0.49	-0.50	-0.52	-0.54
Llama-3.1-8B	-0.43	-0.39	-0.41	-0.43
Mistral-7B	-0.31	-0.27	-0.28	-0.30

Table 8: **Per-model generalization across compression operators.** ΔFSR denotes the absolute FSR reduction of the OCP framework relative to margin-normalized distortion at FAR$= 10\%$, measured in percentage points (pp). Each model column reports the median across the evaluated tasks, while All reports the median across all models and tasks.

Compression operator	ΔFSR (pp) $\uparrow$			
	Qwen2.5-7B	Llama-3.1-8B	Mistral-7B	All
Token eviction	+2.74	+2.61	+2.47	**+2.62**
Value-side quantization	+3.02	+2.87	+2.66	**+2.85**
Low-rank compression	+2.51	+2.34	+2.17	**+2.36**
Key-side perturbation	+2.84	+2.72	+2.53	**+2.69**

exponents, with absolute differences no larger than 0.05. These results show that the conditional scaling remains stable across the evaluated reference levels and is not specific to the reference level of 0.90 used in Fig. 6(a).

D.4 Additional Cross-Operator and OCP-Guided Compression Results

As reported in Section 4.6, the cross-operator results are summarized by medians across all models and tasks. This aggregate summary leaves open whether the observed gains also hold at the individual-model level. Table 8 therefore reports the absolute FSR reduction of the OCP framework relative to margin-normalized distortion for each model at FAR$= 10\%$. The OCP framework reduces FSR for every evaluated compression operator across all three models, with ΔFSR ranging from 2.17 to 3.02 percentage points. The same ordering in FSR reduction is preserved across models: value-side quantization shows the largest reduction, followed by key-side perturbation and token eviction, while low-rank compression remains the most challenging operator. These results show that the cross-operator advantage reported in Section 4.6 is consistently observed for each of the three models rather than being driven by aggregation across models and tasks.

Table 9: **Additional results for OCP-guided compression on Llama-3.1-8B-Instruct.** The increase in median first divergence time is reported together with the KV-memory mismatch and runtime overhead relative to the corresponding baseline under matched average KV-memory usage.

Compression method	$\Delta\tau_{\text{med}}$ (steps) $\uparrow$	KV-memory mismatch (%) $\downarrow$	Runtime overhead (%) $\downarrow$
H2O + OCP	+37.7	1.2	6.8
AsymKV + OCP	+30.7	0.9	7.4
Palu + OCP	+23.7	1.5	8.6
SpectrumQuant + OCP	+30.3	1.1	7.9

We further assess OCP-guided compression together with the resources required to obtain the reported first-divergence improvement. Table 9 therefore reports the KV-memory mismatch and runtime overhead relative to the corresponding baselines, together with the increase in median first divergence time. Across the four compression methods, the KV-memory mismatch ranges from

0.9% to 1.5%, confirming that the later first divergence is obtained under closely matched average KV-memory usage. Under this matched comparison, the OCP-guided variants increase the median first divergence time by 23.7–37.7 decoding steps, with runtime overheads of 6.8%–8.6%. These results show that the later first divergence reported in Section 4.8 persists under matched average KV-memory usage, with runtime overhead below 9% for all four methods under the reference-conditioned evaluation protocol.