Practical Test-Time Adaptation for Large Language Models via Attention-Head Alignment

Anonymous Author(s)
Anonymous Institution
anonymous@example.com

Abstract

Test-time adaptation is important for improving the robustness of large language models under distribution shift; however, most existing methods rely on impractical deployment assumptions, such as coherent test-time distributions, access to multiple same-domain samples, external corpora, or shared adaptation state across independent queries. We propose a per-sample test-time adaptation method that selects outputs according to distributional alignment measured via attention-head activation patterns. Using a small calibration set drawn from arbitrary distributions, we identify attention heads whose activation shifts correlate with output quality. At inference, structured prefix exploration generates candidate outputs, and the most distributionally aligned candidate is selected without modifying model parameters. Experiments across multiple backbones and benchmarks show consistent improvements over vanilla decoding and strong robustness compared to prior test-time learning methods, which often degrade under single-sample adaptation.

1 Introduction

Large language models (LLMs) have emerged as a general-purpose interface for language understanding and generation, demonstrating strong few-shot generalization, instruction-following ability, and reasoning capabilities across a wide range of tasks [10, 1, 13, 55]. Prompting strategies such as chain-of-thought reasoning further improve multi-step reasoning and problem solving by eliciting intermediate computations [50]. However, the effectiveness of these advances can be limited when deployment inputs differ distributionally from the data seen during pretraining or instruction tuning. Such distribution shifts commonly arise in real-world use, where LLMs are exposed to *domain-specific* input distributions [2, 25], including specialized terminology (e.g., legal clauses [3] or medical abbreviations [21]), uncommon input or output formats (e.g., tabular reports, structured logs, or code-like templates), localized conventions [7], and rapidly evolving knowledge [32]. Under these conditions, models that perform well on broad benchmarks can become brittle, exhibiting degraded reliability, factuality, or task performance [2].

Several inference-time strategies attempt to mitigate this brittleness. Domain-specific prompting can guide responses toward a target area [43], but it requires reliably identifying the query domain and selecting an appropriate pre-designed prompt, which is difficult when the domain is unknown or instance-specific. LLM-as-judge reranking methods can select among multiple outputs [54, 27], but they require a sufficiently domain-aware evaluator and multiple model invocations, creating a substantial cost and deployment bottleneck [28]. These limitations motivate the study of *test-time adaptation* (TTA), which aims to adjust model behavior at inference time to better align with the distributional characteristics of the test-time query. Existing TTA methods for LLMs mainly follow two directions. Retrieval-augmented generation (RAG) improves factual grounding by conditioning generation on externally retrieved documents [31, 9, 5]. Test-time learning (TTL) instead adapts the model during inference, either with supervision [22] or without explicit labels [48, 25]. Supervised TTL typically retrieves task-relevant data and performs gradient-based updates, while unsupervised TTL relies on objectives such as entropy or perplexity minimization.

Preprint.

Table 1: Comparison of LLM-oriented test-time adaptation paradigms under test-time distribution shift. Our method is designed for per-sample LLM adaptation without requiring external corpora, coherent test-time assumptions, or risky parameter updates.

Method	Requires External Knowledge Corpus	Assumes Coherent Test-Time Distribution	Updates Model Parameters at Test Time	Designed for Single Test Instance
Fine-tuning	✓	✓	✗	✗
Test-Time Adaptation (TTA) Methods				
RAG [18]	✓	✗	✗	✓
TENT [48]	✗	✓	✓	✗
TTT-NN [22]	✓	✗	✓	✓
TLM [25]	✗	✓	✓	✗
Ours (Per-sample Head-Based TTA)	✗	✗	✗	✓

While these approaches demonstrate promising results, they face significant practical limitations in realistic LLM deployment settings. First, **reliance on large, highly relevant external corpora is often impractical**. RAG and supervised TTL critically depend on the availability of up-to-date, high-quality retrieval databases that closely match the test-time query distribution. In many domain-shifted scenarios, such corpora are scarce or nonexistent, and retrieval pipelines introduce additional system complexity, latency, and sensitivity to retrieval errors [20]. Second, **the assumption of a coherent target distribution at test time is frequently violated**. Many unsupervised TTL methods require batch-level statistics or shared updates across multiple test samples to stabilize optimization and avoid degenerate solutions [48]. However, in modern LLM deployments, queries often arrive sequentially and may each correspond to distinct, instance-specific distribution shifts (e.g., per-user, per-document, or per-task variation), rendering such assumptions unrealistic. Third, **updating model parameters at inference time is inherently risky**. Even lightweight parameter updates can introduce instability, interference across successive inputs, or unintended forgetting, especially under limited and non-stationary data [33, 24, 16, 25]. These risks are exacerbated in the *single-sample* regime, where no opportunity exists to amortize adaptation over multiple examples from a shared distribution. As a result, existing TTL methods fundamentally struggle when adaptation must be performed per instance rather than per domain. The limitations of these approaches are summarized in Table 1.

To address these limitations, we propose a *per-sample test-time adaptation* method for LLMs that operates on a *single* test instance, requires no external retrieval corpus, no access to additional samples from a coherent target distribution, and no updates to model parameters. Our approach is grounded in two empirical findings. First, **LLMs possess an intrinsic capacity to uncover higher-quality generations through limited exploration** (see **Observation 1**). In particular, lightweight exploration over top-n next-token predictions can substantially improve generation quality. We find that allowing the model to briefly branch—by selecting among the top-n predicted tokens for only a few initial decoding steps—often reveals candidate generations that are more *distributionally aligned* with the input prompt, in the sense that they exhibit higher similarity to the target output than those produced by greedy decoding. Even short explorations (e.g., depth 3–4 tokens with $n = 3$) can expose alternative generations that better match the latent intent of the input. Second, **distributionally aligned generations induce consistent activation patterns in specific attention heads** (see **Observation 2**). Building on prior work demonstrating functional specialization of attention heads [35, 56], we find that candidate generations that are better aligned with the prompt exhibit more similar attention-head activation signatures relative to the prompt itself. Importantly, this alignment signal correlates more strongly with downstream task performance than commonly used uncertainty measures such as perplexity. Third, **we find that attention-head alignment patterns learned from one distribution generalize reliably across different domains** (see **Observation 3**). This suggests that a subset of attention heads exhibits consistent alignment behavior under similar distributional conditions, independent of the specific domain. As a result, the corresponding activation signatures capture *distribution-agnostic* properties of prompt–generation compatibility and can be learned from arbitrary distributional data.

Building on these observations, we propose a lightweight test-time adaptation framework that operates entirely without model updates. At inference, it performs a small structured exploration to produce candidate generations, and selects the one most distributionally aligned with the prompt based on attention-head activation patterns. The required alignment signals are learned from a small calibration set drawn from arbitrary distributions, enabling a sample-efficient, distribution-agnostic, and practically deployable per-sample adaptation method for LLMs.

2 Related Work and Background

Test-Time Adaptation Methods for LLMs. Real-world LLM deployment frequently involves distribution shifts between pretraining, instruction-tuning data, and incoming queries, motivating methods that adapt model behavior at test time. We use *test-time adaptation* to refer to methods that adapt a model's behavior under test-time distribution shift without access to the original training pipeline and without assuming explicit domain knowledge for each query. This definition excludes *prompting-based methods*, such as domain-specific prompting or LLM-as-judge selection. Although such methods can improve responses in some settings, they typically require strong auxiliary assumptions: domain-specific prompting requires identifying the correct domain and prompt template for each query, while LLM-as-judge reranking requires a sufficiently domain-aware evaluator and multiple model invocations.

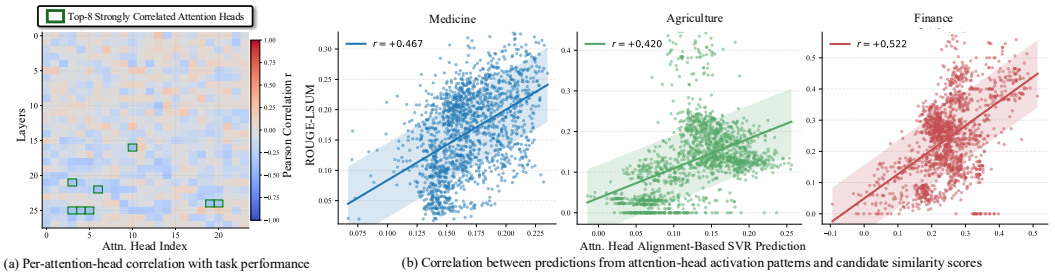
Existing LLM-oriented TTA methods mainly fall into two families: (i) *retrieval-conditioned adaptation*, where the model is guided by external information, and (ii) *parameter-update adaptation*, where the model or a lightweight adapter is updated online. A prominent retrieval-based family is RAG, which conditions decoding on documents retrieved from an external corpus to improve factual grounding and domain coverage [31, 9, 5]. While effective, such approaches can be bottlenecked by corpus relevance, retrieval quality, system complexity, and latency, particularly when queries are niche or rapidly evolving [20]. In contrast, parameter-update methods adapt the model itself during inference. Unsupervised TTA typically optimizes self-supervised criteria such as entropy minimization on unlabeled test inputs [48, 39, 11, 25], whereas supervised or semi-supervised test-time training leverages retrieved or cached examples, often from training data or a curated memory, to perform gradient-based updates [22, 26].

In the vision domain, several TTA methods have been developed for single-sample test streams [40, 19, 52]. However, these methods are typically built around vision-specific mechanisms such as normalization-layer adaptation, entropy minimization over class probabilities, or image augmentations, and therefore do not directly transfer to autoregressive LLM generation. Moreover, entropy- and perplexity-based objectives have been observed to be brittle for LLM adaptation under distribution shift [25]. These limitations leave an important gap: many LLM deployment settings provide neither a reliably relevant retrieval corpus nor a coherent stream of same-domain test samples, while online parameter updates may introduce instability or interference across sequential queries. Our method targets this gap by enabling *per-sample* test-time adaptation for autoregressive decoding without weight updates and without assuming batch-level test-time stationarity, using lightweight exploration and an internal alignment signal derived from attention-head activations.

Multi-Head Attention and Functionality. Transformer multi-head attention represents token interactions through multiple parallel heads [45]. Prior analyses show that heads are not interchangeable: different heads specialize in behaviors such as positional patterns, syntactic dependencies, coreference, delimiter tracking, and long-range routing, with such roles often stable across runs or checkpoints [14, 46, 29]. Pruning and ablation studies further show that many heads are redundant, while a small subset of critical or outlier heads can be disproportionately important, suggesting that head-level signals provide compact information about model computation [37, 46]. Recent mechanistic studies likewise indicate that head activations correlate with latent computational roles and can probe context or reasoning state [35, 56]. These findings motivate our *head-activation alignment* as a lightweight internal signal to discriminate candidates under distribution shift.

3 Problem Formulation

Let f_Θ denote a LLM with fixed parameters Θ, trained on data drawn from a train distribution $\mathcal{P}_{\text{train}}$. After deployment, the model is queried with test-time inputs that may not follow a single coherent distribution. Instead, we consider a realistic setting in which each test instance x_i is drawn from its own (unknown) distribution $\mathcal{Q}_i$, and different test inputs x_i, x_j may satisfy $\mathcal{Q}_i \neq \mathcal{Q}_j$. This captures common deployment scenarios such as per-user queries, document-specific tasks, or rapidly shifting domains, where no shared test-time statistics can be assumed. Given a single test prompt $x \sim \mathcal{Q}_x$, the model induces a conditional distribution over outputs $p_\Theta(y \mid x)$. The objective is to produce an output y that is well aligned with $\mathcal{Q}_x$, despite the mismatch between $\mathcal{Q}_x$ and the training distribution $\mathcal{P}_{\text{train}}$. Importantly, we impose the following constraints: (i) no access to additional samples from



(a) Per-attention-head correlation with task performance

(b) Correlation between predictions from attention-head activation patterns and candidate similarity scores

Figure 2: Attention-head activation alignment between prompts and candidates in Llama-3.2-3B-Instruct. (a) Per-head Pearson correlation heatmap illustrating the relationship between candidate–target similarity and prompt–candidate attention-head alignment, measured as the L2 distance in attention-head activation space for each head, evaluated on 60 Medicine-domain prompts. (b) SVR regression performance for predicting candidate–target similarity from attention-head alignment features, with models trained and evaluated separately on each domain dataset (Medicine, Agriculture, and Finance), using 40 training prompts and 20 held-out test prompts per dataset.

$\mathcal{Q}_x$, (ii) no assumption of coherence across different test-time distributions, and (iii) no modification of model parameters Θ.

We formulate per-sample test-time adaptation as a *candidate selection* problem. Given a test-time input $x \sim \mathcal{Q}_x$, the fixed model f_Θ induces a conditional distribution $p_\Theta(y \mid x)$ and produces a finite set of candidate outputs

$$\mathcal{C}(x) = \{y^{(1)}, \ldots, y^{(B)}\}, \quad y^{(b)} \sim p_\Theta(\cdot \mid x). \tag{1}$$

Each candidate $y^{(b)}$ can be associated with an implicit, candidate-dependent output distribution $\mathcal{Q}_x^{(b)}$, reflecting the model's generative behavior along the corresponding decoding trajectory. Because the ground-truth target $y^\star \sim \mathcal{Q}_x$ is unobserved at test time, direct evaluation of candidate quality is infeasible. We therefore cast adaptation as the problem of selecting the candidate whose induced distribution $\mathcal{Q}_x^{(y)}$ is closest to the unknown test-time distribution $\mathcal{Q}_x$. Abstractly, given a distributional distance or divergence

$$D\big(\mathcal{Q}_x, \mathcal{Q}_x^{(y)}\big), \tag{2}$$

the adapted prediction is defined as

$$\hat{y} \;=\; \arg\min_{y \in \mathcal{C}(x)} D\big(\mathcal{Q}_x, \mathcal{Q}_x^{(y)}\big). \tag{3}$$

Since neither $\mathcal{Q}_x$ nor $\mathcal{Q}_x^{(y)}$ is directly observable, the distance $D(\cdot, \cdot)$ must be estimated using internal signals of the fixed model f_Θ, without access to supervision, additional test samples, or parameter updates. Under this formulation, per-sample test-time adaptation reduces to identifying, for each test input, the candidate generation whose induced behavior best matches the test-time distribution implied by the prompt. The remainder of this section instantiates this framework with a concrete, lightweight estimation of D and an efficient selection procedure.

4 Methodology

Our objective is to select, for each test-time prompt x, a candidate generation whose behavior is *distributionally aligned* with the unknown test-time distribution $\mathcal{Q}_x$. Because neither $\mathcal{Q}_x$ nor candidate-induced distributions $\mathcal{Q}_x^{(y)}$ are observable, we seek a proxy signal derived from the internal computation of a fixed pretrained model f_Θ. We operationalize this goal in three stages: (i) constructing a candidate set that exposes potentially well-aligned generations, (ii) estimating candidate–prompt alignment using attention-head activation dynamics, and (iii) learning a lightweight aggregation of these signals that generalizes across domains. Throughout, we avoid likelihood- or perplexity-based criteria, which primarily reflect alignment with the training distribution.

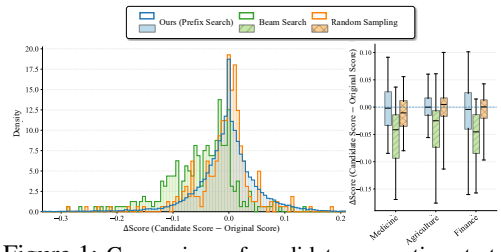


Figure 1: Comparison of candidate-generation strategies—prefix search, beam search, and random sampling—on three domain datasets using Llama-3.2-3B-Instruct, with 81 candidates per prompt. Target similarity is measured by ROUGE-LSUM. The histogram aggregates all datasets (left), and box plot reports per-dataset distributions (right). Prefix search yields the most stable and consistent distribution.

4.1 Candidate Set Construction

As formalized in Section 3, test-time adaptation reduces to selecting

$$\hat{y} = \arg\min_{y \in \mathcal{C}(x)} D\Big(\mathcal{Q}_x, \mathcal{Q}_x^{(y)}\Big), \tag{4}$$

where $\mathcal{C}(x)$ is a finite set of candidates sampled from $p_\Theta(\cdot \mid x)$. A central challenge is that likelihood under p_Θ may be poorly aligned with task performance when $\mathcal{Q}_x \neq \mathcal{P}_{\text{train}}$.

Assumption 1 (**Likelihood–alignment mismatch under distribution shift**)**.** For a non-negligible set of prompts $x \sim \mathcal{Q}_x$, there exist candidates y such that

$$D\Big(\mathcal{Q}_x^{(y)}, \mathcal{Q}_x\Big) < D\Big(\mathcal{Q}_x^{(y^{(0)})}, \mathcal{Q}_x\Big), \tag{5}$$

$$p_\Theta(y \mid x) < p_\Theta(y^{(0)} \mid x), \tag{6}$$

where $y^{(0)}$ denotes the greedy decoding output.

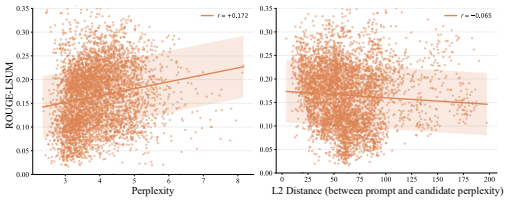


Figure 3: Perplexity-based baselines under distribution shift for Llama-3.2-3B-Instruct on 60 Medicine-domain prompts with measured Pearson coefficient. Left: candidate–target similarity versus candidate perplexity, showing a weak and unstable correlation under test-time distribution shift. Right: similarity versus perplexity alignment, measured as the L2 distance between prompt and candidate perplexities, exhibiting near-zero correlation. These results indicate that perplexity-based criteria are unreliable for candidate selection in shifted test domains.

Our intuition in Assumption 1 motivates candidate construction strategies that relax strict likelihood ordering while remaining within the model's support.

Prefix search. We adopt a prefix-based search strategy that injects controlled stochasticity early in decoding while preserving coherent continuation. Given prefix depth k (e.g., $k = 4$) and branching factor n (e.g., $n = 3$), we sample prefixes

$$\pi = (y_1, \ldots, y_k), \qquad y_t \sim \mathrm{Unif}(\text{Top-}n\ p_\Theta(\cdot \mid x, y_{<t})), \tag{7}$$

where Top-n denotes the set of the n highest-probability tokens under the model's conditional distribution $p_\Theta(\cdot \mid x, y_{<t})$, and $\mathrm{Unif}(\cdot)$ indicates uniform sampling over this set. That is, each of the top-n tokens is selected with equal probability, independent of its exact likelihood value. After sampling the prefix, the remainder of the sequence is generated deterministically via greedy decoding:

$$y_{k+1:T} = \arg\max p_\Theta(y_{k+1:T} \mid x, \pi). \tag{8}$$

This yields candidates that deviate from greedy decoding only through early prefix choices.

Observation 1 (**Stable exposure of high-quality candidates**)**.** As shown in Figure 1, prefix search consistently yields more balanced candidate sets, exhibiting smoother score distributions and substantially lower cross-domain variance than existing methods such as beam search and random sampling, which are likelihood-based methods that tend to produce unstable distributions, where high-quality candidates appear sporadically and inconsistently across domains. These results confirm that a modest relaxation of likelihood ordering is sufficient to more reliably surface high-quality candidates already encoded in the model, directly supporting the intuition underlying Assumption 1.

4.2 Attention-Head Alignment as a Proxy for Distributional Alignment

Having constructed a candidate set that plausibly contains well-aligned outputs, we now address the estimation of $D(\mathcal{Q}_x, \mathcal{Q}_x^{(y)})$. Since the distributions themselves are unobservable, we seek an internal signal whose variation reflects how a candidate generation interacts with the prompt-conditioned computation of the model.

Prior empirical work [35, 56] has shown that attention heads exhibit specialized and stable functional roles across layers, with later heads often encoding higher-level semantic information. Motivated by these findings, we posit the following.

Assumption 2 (**Attention-head consistency under alignment**)**.** For a non-negligible measure of prompts $x \sim \mathcal{Q}_x$, there exists a subset of attention heads whose activation deviations between the prompt x and the prompt–candidate sequence $x \oplus y$, where $\oplus$ denotes sequence concatenation, reflect distributional alignment. Specifically, for candidates $y \in \mathcal{C}(x)$, activation deviations in these heads are correlated with $D\Big(\mathcal{Q}_x^{(y)}, \mathcal{Q}_x\Big)$, even when likelihood ordering under $p_\Theta(\cdot \mid x)$ is misaligned with task performance.

This assumption states that candidates better aligned with the test-time distribution induce smaller perturbations to prompt-conditioned internal representations.

Attention-head activation representation. Consider a transformer with L layers and H attention heads per layer. For sequence $z = (z_1, \ldots, z_T)$, let $\mathbf{H}_{\ell,h}(z,t) \in \mathbb{R}^{d_h}$ denote the output of head (ℓ, h) at position t after attention aggregation, where d_h is the per-head hidden dimensionality. We summarize head activity via

$$a_{\ell,h}(z) = \mathbb{E}_{t\sim\mathcal{T}_{\text{act}}}\left[\|\mathbf{H}_{\ell,h}(z,t)\|_2\right] \approx \frac{1}{|\mathcal{T}_{\text{act}}|}\sum_{t\in\mathcal{T}_{\text{act}}}\|\mathbf{H}_{\ell,h}(z,t)\|_2, \quad (9)$$

yielding an activation vector $\mathbf{a}(z) \in \mathbb{R}^{LH}$ where $\mathcal{T}_{\text{act}}$ denotes the set of token positions over which activations are measured, spanning the entire sequence from the prompt tokens through the generated tokens.

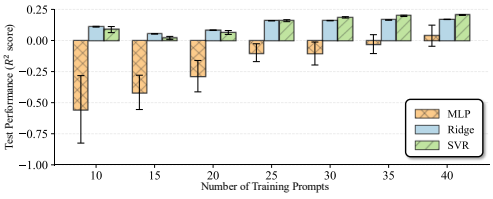


Figure 4: Regression performance on held-out test prompts (20 prompts) as a function of the number of training prompts, using three regression models—MLP, Ridge, and SVR—trained on attention-head alignment features extracted from Llama-3.2-3B-Instruct on the Medicine-domain dataset. Negative R^2 values for the MLP indicate performance worse than predicting the mean target score, reflecting overfitting and instability under extremely limited calibration data.

For prompt x and candidate y, we define the alignment feature $\mathbf{\Delta}(x,y) = \mathbf{a}(x \oplus y) - \mathbf{a}(x)$, and use squared components $\Delta_{\ell,h}(x,y)^2$ as fine-grained measures of representational deviation.

Observation 2 (Head alignment reflects candidate quality under distribution shift). Consistent with Assumption 2, Figure 2 (a) shows that individual attention-head alignment features exhibit strong positive and negative correlations with candidate–target similarity. The most pronounced correlations arise predominantly in later-layer heads, which are known to encode higher-level, task-relevant abstractions [36]. Moreover, these alignment signals are sufficiently structured to support accurate prediction of candidate quality using a lightweight regression model (Figure 2 (b)); the regression formulation and training procedure are introduced in the following Subsection 4.3. By contrast, Figure 3 shows that perplexity-based criteria—despite being a standard confidence signal for LLMs—exhibit weak or unstable correlation with candidate quality under the same distribution shift. This failure reflects the model's reliance on training-distribution likelihoods, which encode a false distributional belief when test-time prompts deviate from the training data, rendering perplexity ineffective for estimating distributional alignment in this setting.

4.3 Distribution-Aware Head Alignment Learning

The alignment vector $\mathbf{\Delta}(x,y)$ provides a high-dimensional proxy for the latent distributional divergence $D(\mathcal{Q}_x, \mathcal{Q}_x^{(y)})$. While D itself is unobservable at test time, Assumption 2 posits that relative differences in this divergence are reflected in systematic patterns of attention-head activation shifts captured by $\mathbf{\Delta}(x,y)$. Our goal is therefore to learn a scalar scoring function that maps alignment features to an estimate of candidate quality. Concretely, we introduce a lightweight predictor

$$g(\mathbf{\Delta}(x,y)) \;\propto\; -D\Big(\mathcal{Q}_x, \mathcal{Q}_x^{(y)}\Big), \quad (10)$$

where the proportionality indicates that the predictor preserves the relative ordering of candidates induced by the latent distributional divergence. Under this approximation, candidates whose generation induces smaller attention-head alignment deviations are treated as being more distributionally aligned with the prompt.

Algorithm 1: Distribution-Aware Attention-Head Alignment for Test-Time Adaptation

Require: Fixed LLM f_Θ; calibration set $\mathcal{S} = \{(x_i, y_i^\star)\}_{i=1}^M$; test prompt x
Ensure: Selected output $\hat{y}$

Calibration (offline).
Initialize training set $\mathcal{D} \leftarrow \emptyset$
for each $(x_i, y_i^\star) \in \mathcal{S}$ **do**
  Generate candidates $\mathcal{C}(x_i)$ via prefix search
  for each $y \in \mathcal{C}(x_i)$ **do**
    Compute alignment features $\mathbf{\Delta}(x_i, y)$
    Compute surrogate label $\ell(y, y_i^\star)$, e.g., ROUGE-LSUM w.r.t. $y_i^\star$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{\Delta}(x_i, y), \ell(y, y_i^\star))\}$
  end for
end for
Fit regressor g on $\mathcal{D}$
Test-time adaptation (per prompt).
Generate candidates $\mathcal{C}(x)$ via prefix search
for each $y \in \mathcal{C}(x)$ **do**
  Compute $\mathbf{\Delta}(x,y)$ and score $s_y \leftarrow g(\mathbf{\Delta}(x,y))$
end for
return $\hat{y} \leftarrow \arg\max_{y\in\mathcal{C}(x)} s_y$

Accordingly, the original objective

$$\arg\min_{y \in \mathcal{C}(x)} D\Big(\mathcal{Q}_x, \mathcal{Q}_x^{(y)}\Big) \tag{11}$$

is operationalized as

$$\hat{y} \;=\; \arg\max_{y \in \mathcal{C}(x)} g(\mathbf{\Delta}(x, y))\,. \tag{12}$$

Training data construction. To learn the predictor g, we introduce a small *calibration set* $\mathcal{S} = \{(x_i, y_i^\star)\}_{i=1}^N$ (e.g., $N \leq 64$), where each prompt x_i is paired with a reference target $y_i^\star$ available only during calibration. For each x_i, we generate a candidate set $\mathcal{C}(x_i)$ via prefix search and compute attention-head alignment vectors $\mathbf{\Delta}(x_i, y)$ for all $y \in \mathcal{C}(x_i)$. Because the distributional divergence $D(\mathcal{Q}_{x_i}, \mathcal{Q}_{x_i}^{(y)})$ is unobservable, we supervise learning using a task-level quality metric $\ell(y, y_i^\star)$ (e.g., ROUGE-LSUM), which serves as an empirical surrogate satisfying

$$\ell(y, y_i^\star) \;\underset{\sim}{\propto}\; -D\Big(\mathcal{Q}_{x_i}, \mathcal{Q}_{x_i}^{(y)}\Big)\,. \tag{13}$$

This construction yields a calibration dataset

$$\mathcal{D}_{\text{cal}} = \big\{(\mathbf{\Delta}(x_i, y),\, \ell(y, y_i^\star)) \;\big|\; (x_i, y_i^\star) \in \mathcal{S},\; y \in \mathcal{C}(x_i)\big\}, \tag{14}$$

which is used to train g to preserve the relative ordering of candidates induced by distributional alignment.

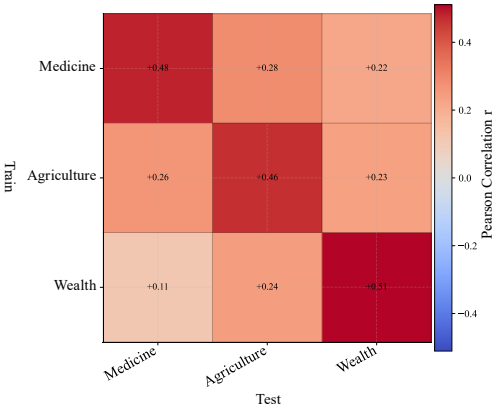


Figure 5: Transferability of distribution-aware attention-head alignment across domains, evaluated on Medicine, Agriculture, and Wealth datasets using Llama-3.2-3B-Instruct. For each domain, 60 prompts are sampled, with 40 used to train the SVR and 20 held out for testing. The diagonal entries show the strongest correlations when training and testing on the same domain. Notably, cross-domain evaluations still yield consistently positive correlations (Pearson $r \approx 0.2$), indicating that the attention-head alignment signal generalizes beyond the training domain.

Learning procedure. We train a regression model to map alignment features to scalar values that preserve this ordering. Among the evaluated alternatives [23, 42], Support Vector Regression (SVR) provides the most stable generalization in the low-data regime relevant to test-time adaptation, consistently outperforming Ridge regression and MLP baselines (Figure 4). The learned predictor aggregates distribution-aware attention-head signals into a robust estimate of candidate alignment, enabling effective selection without modifying model parameters or relying on likelihood-based beliefs.

Observation 3 (**Cross-domain generalization**). As shown in Figure 5, an SVR trained on one domain retains positive predictive correlation when evaluated on others. While within-domain transfer is strongest, consistent cross-domain performance indicates that the learned aggregation captures *distributionally aware* attention heads whose behavior is not tied to domain-specific surface statistics.

Together, Observation 2 and Observation 3 empirically confirm Assumption 2 and justify learning a prompt-agnostic aggregation of attention-head alignment signals. Algorithm 1 summarizes the full procedure.

5 Experiments

5.1 Experimental Settings

Evaluation protocol. We study the setting of *per-sample test-time adaptation*, where each test input is treated as an independent distributional shift. In this regime, no assumptions are made about shared statistics or correlations across test samples. To ensure a fair and controlled comparison, we evaluate *all methods*—including state-of-the-art test-time adaptation and test-time learning approaches—under the same protocol: *each test sample is processed independently, and any adapted model state is reset before the next test instance*. As a result, the effective test-time batch size is fixed to one, and no information (e.g., gradients, statistics, or updated parameters) is carried across test samples. This protocol isolates per-sample adaptability and prevents methods from benefiting from cross-sample accumulation, which is incompatible with our problem formulation.

Table 2: Comparison on Domain-Specific and Instruction-Following datasets. Each cell reports the absolute score, with the improvement over the corresponding vanilla LLM in parentheses. The **best** and **second-best** results *within each backbone block* are highlighted. Average Improvement columns report mean Δ over Domain-Specific (4), Instruction-Following (2), and Overall (6) datasets.

Method	Domain-Specific				Instruction-Following		Average Improvement (Δ)		
	Agriculture	Finance	Geography	Medicine	Alpaca-GPT4	InstructionWild	Domain	Instruction	Overall
Llama-3.2-3B-Instruct	0.0850(+0.0000)	0.2229(+0.0000)	0.2395(+0.0000)	0.1411(+0.0000)	**0.3564(+0.0000)**	0.2562(+0.0000)	+0.0000	+0.0000	+0.0000
Tent	0.0842(-0.0008)	0.2190(-0.0039)	**0.2423(+0.0028)**	0.1425(+0.0014)	0.3520(-0.0044)	0.2627(+0.0065)	-0.0001	+0.0010	+0.0002
EATA	0.0838(-0.0012)	0.2180(-0.0049)	0.2412(+0.0017)	0.1431(+0.0020)	0.3527(-0.0037)	**0.2646(+0.0084)**	-0.0006	**+0.0023**	+0.0003
COME	0.0835(-0.0015)	0.2203(-0.0026)	0.2406(+0.0011)	0.1427(+0.0016)	0.3467(-0.0097)	0.2315(-0.0247)	-0.0004	-0.0172	-0.0059
TLM	**0.0877(+0.0027)**	**0.2247(+0.0018)**	0.2382(-0.0013)	**0.1436(+0.0025)**	0.3540(-0.0024)	0.2632(+0.0070)	**+0.0014**	**+0.0023**	**+0.0017**
Ours	**0.1257(+0.0407)**	**0.2592(+0.0363)**	**0.2460(+0.0065)**	**0.1723(+0.0312)**	**0.3838(+0.0274)**	**0.2928(+0.0366)**	**+0.0287**	**+0.0320**	**+0.0298**
Llama-3-8B-Instruct	0.0834(+0.0000)	0.2329(+0.0000)	0.2450(+0.0000)	0.1265(+0.0000)	0.3752(+0.0000)	0.2608(+0.0000)	+0.0000	+0.0000	+0.0000
Tent	0.0833(-0.0001)	0.2327(-0.0002)	0.2539(+0.0089)	0.1292(+0.0027)	**0.3827(+0.0075)**	0.2670(+0.0062)	+0.0028	+0.0068	+0.0041
EATA	0.0834(+0.0000)	0.2329(+0.0000)	0.2549(+0.0099)	0.1291(+0.0026)	0.3821(+0.0069)	**0.2682(+0.0074)**	+0.0031	+0.0071	+0.0044
COME	0.0839(+0.0005)	0.2321(-0.0008)	0.2545(+0.0095)	0.1286(+0.0021)	0.3794(+0.0042)	0.2427(-0.0181)	+0.0028	-0.0069	-0.0004
TLM	**0.0858(+0.0024)**	**0.2427(+0.0098)**	**0.2574(+0.0124)**	**0.1298(+0.0033)**	**0.3846(+0.0094)**	0.2670(+0.0062)	**+0.0070**	**+0.0078**	**+0.0072**
Ours	**0.1030(+0.0196)**	**0.2553(+0.0224)**	**0.2665(+0.0215)**	**0.1773(+0.0508)**	0.3745(-0.0007)	**0.2776(+0.0168)**	**+0.0286**	**+0.0080**	**+0.0217**
Qwen2.5-7B-Instruct	**0.0981(+0.0000)**	**0.2739(+0.0000)**	**0.2649(+0.0000)**	**0.1313(+0.0000)**	**0.4439(+0.0000)**	**0.2866(+0.0000)**	**+0.0000**	**+0.0000**	**+0.0000**
Tent	0.0936(-0.0045)	0.2551(-0.0188)	0.2587(-0.0062)	0.1275(-0.0038)	0.4296(-0.0143)	0.2835(-0.0031)	-0.0083	-0.0087	-0.0084
EATA	0.0930(-0.0051)	0.2553(-0.0186)	0.2573(-0.0076)	0.1269(-0.0044)	0.4289(-0.0150)	0.2854(-0.0012)	-0.0089	-0.0081	-0.0086
COME	0.0945(-0.0036)	0.2549(-0.0190)	0.2563(-0.0086)	0.1266(-0.0047)	0.4242(-0.0197)	0.2561(-0.0305)	-0.0090	-0.0251	-0.0143
TLM	0.0946(-0.0035)	0.2592(-0.0147)	0.2620(-0.0029)	0.1270(-0.0043)	0.4275(-0.0164)	0.2836(-0.0030)	-0.0064	-0.0097	-0.0074
Ours	**0.1164(+0.0183)**	**0.2929(+0.0190)**	**0.2768(+0.0119)**	**0.1567(+0.0254)**	**0.4866(+0.0427)**	**0.2960(+0.0094)**	**+0.0186**	**+0.0260**	**+0.0211**

Table 3: Effect of calibration corpus choice on Domain-Specific and Instruction-Following datasets. Each cell reports the absolute score with improvement over the vanilla LLM in parentheses. **Best** and **second-best** results are highlighted.

Method	Domain-Specific				Instruction-Following		Average Improvement (Δ)		
	Agriculture	Finance	Geography	Medicine	Alpaca-GPT4	InstructionWild	Domain	Instruction	Overall
Llama-3.2-3B-Instruct	0.0850(+0.0000)	0.2229(+0.0000)	0.2395(+0.0000)	0.1411(+0.0000)	0.3564(+0.0000)	0.2562(+0.0000)	+0.0000	+0.0000	+0.0000
Ours (fusechat_v1)	**0.1257(+0.0407)**	**0.2592(+0.0363)**	**0.2460(+0.0065)**	**0.1723(+0.0312)**	**0.3838(+0.0274)**	0.2928(+0.0366)	**+0.0287**	**+0.0320**	**+0.0298**
Ours (open-perfectblend)	**0.1340(+0.0490)**	**0.2592(+0.0363)**	**0.2463(+0.0068)**	**0.1667(+0.0256)**	**0.3838(+0.0274)**	**0.3030(+0.0468)**	**+0.0294**	**+0.0371**	**+0.0320**
Ours (tulu-3-sft-mixture)	0.1206(+0.0356)	**0.2592(+0.0363)**	**0.2460(+0.0065)**	**0.1681(+0.0270)**	**0.3838(+0.0274)**	**0.2944(+0.0382)**	+0.0264	**+0.0328**	+0.0285

Baselines. We compare against representative state-of-the-art entropy- or perplexity-based test-time adaptation methods, including Tent [48], EATA [39], COME [53], and TLM [25], which adapt model parameters using unlabeled test inputs. To ensure a fair comparison under the per-sample test-time adaptation setting, all baseline methods are evaluated with: (i) test-time batch size fixed to one, (ii) model parameters reset after each test sample, and (iii) no accumulation of gradients or statistics across samples. Methods requiring access to external data or persistent inference-time memory, such as RAG, are excluded, as they cannot operate under the isolated per-sample adaptation setting considered here.

Datasets. We evaluate our method and all baselines on six benchmark datasets spanning both domain-specific and instruction-following settings. Specifically, we consider four domain-specific datasets—*Agriculture* [6], *Finance* [8], *Geography* [15], and *Medicine* [49]—which assess robustness to domain shift characterized by specialized terminology and expert knowledge. In addition, we evaluate on two instruction-following datasets—*Alpaca-GPT4* [41] and *InstructionWild* [38]—which measure generalization across diverse user intents and instruction styles. Following prior work [25], we randomly sample 5,000 instances from each dataset and report performance using ROUGE-LSUM [34], which quantifies the overlap between generated responses and reference texts.

Backbone models. We conduct experiments using multiple pretrained instruction-tuned large language models with different sizes and architectures: *Llama-3.2-3B-Instruct*, *Llama-3-8B-Instruct* [17], and *Qwen2.5-7B-Instruct* [44]. All backbone models are kept frozen during evaluation. Our method does not perform any parameter updates, whereas baseline adaptation methods employ lightweight parameter-efficient tuning via LoRA.

Implementation details. For baselines, we follow the original implementations and hyperparameters [25], with the sole modification that adaptation is applied independently to each test sample under our per-sample evaluation protocol. Our method uses a single hyperparameter configuration across all datasets and backbones unless noted otherwise. Candidate generation employs prefix search with depth $k = 4$ and branching factor $n = 3$, producing $3^4 = 81$ candidates per input, after which the remaining text is generated via greedy decoding. Attention-head activations are computed over all token positions $\mathcal{T}_{\text{act}}$ spanning the prompt and generated text. During calibration,

guided by the rapid saturation observed in Figure 4, we randomly sample 64 prompt–target pairs. Calibration data are drawn from general-purpose datasets—*fusechat_v1* [47], *open-perfectblend* [51], and *tulu-3-sft-mixture* [30]—with *fusechat_v1* used by default. All experiments were conducted on an NVIDIA A100 GPU.

5.2 Experimental Results

Does a non-updating test-time strategy improve performance? Table 2 demonstrates that our method consistently improves test-time performance over the corresponding vanilla LLMs across all evaluated backbones and datasets, despite performing *no model parameter updates*. Across domain-specific and instruction-following datasets, our approach yields substantial positive gains, with average improvements of $\Delta{+}0.0298$, $\Delta{+}0.0217$, and $\Delta{+}0.0211$ for *Llama-3.2-3B-Instruct*, *Llama-3-8B-Instruct*, and *Qwen2.5-7B-Instruct*, respectively. These improvements are achieved solely through inference-time candidate selection guided by internal activation signals, highlighting that pretrained instruction-tuned LLMs already possess rich latent capabilities that can be better exploited without modifying their parameters. This result supports that performance gains at test time do not require additional learning, but can instead arise from more effective utilization of the model's inherent representational power.

Are test-time model updates effective in practical per-sample settings? We next compare against prior test-time adaptation and learning methods, including Tent, EATA, COME, and TLM, all evaluated under the same per-sample protocol. As shown in Table 2, these methods frequently fail to deliver consistent improvements and often degrade performance relative to the vanilla LLM, particularly on domain-specific datasets. For example, under *Qwen2.5-7B-Instruct*, all model-update baselines exhibit negative average domain improvements (up to $\Delta{-}0.0143$), while our method achieves a strong positive gain of $\Delta{+}0.0211$. This pattern is consistent across backbones and benchmarks. These results indicate that inference-time parameter updates are brittle when operating with extremely limited data (single-sample adaptation), and may amplify noise rather than learn meaningful distributional shifts. In contrast, our non-updating approach avoids this failure mode and consistently outperforms prior methods, demonstrating superior robustness and practical applicability in realistic test-time scenarios.

Is our method sensitive to the calibration dataset? Table 3 evaluates the impact of the calibration corpus on performance. Using three distinct and lightweight instruction datasets—*fusechat_v1*, *open-perfectblend*, and *tulu-3-sft-mixture*—our method achieves consistently strong improvements across both domain-specific and instruction-following datasets. Notably, all calibration choices yield comparable gains, with overall average improvements ranging from $\Delta{+}0.0285$ to $\Delta{+}0.0320$. This empirical robustness supports that our method is not tightly coupled to a specific calibration distribution, and supports our earlier observation that the learned activation-based selection mechanism captures transferable, distribution-agnostic signals. As a result, our approach remains effective even when calibration data are drawn from generic or mismatched sources, significantly enhancing its practicality in real-world deployment settings.

6 Conclusion

We introduced a practical framework for per-sample test-time adaptation of LLMs that operates without test-time parameter updates, external corpora, or assumptions of test-time distributional coherence. By leveraging distribution-aware attention-head alignment learned from a small, domain-agnostic calibration set, our method reliably selects better-aligned outputs from lightweight test-time exploration. Extensive experiments demonstrate consistent gains across domains and instruction-following tasks, while revealing the brittleness of inference-time model updates under limited data. These results highlight that effective test-time adaptation can be achieved by exploiting internal model signals, offering a robust and deployable alternative to parameter-updating or retrieval-dependent approaches.

Limitations and Future Work. Our TTA paradigm via distributional alignment improves robustness under test-time distribution shift, but does not guarantee factual, mathematical, or logical correctness. Future work can combine this paradigm with complementary reasoning methods, such as chain-of-thought prompting, to improve both distributional alignment and response correctness.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Ekin Akyürek, Mehul Damani, Adam Zweiger, Linlu Qiu, Han Guo, Jyothish Pari, Yoon Kim, and Jacob Andreas. The surprising effectiveness of test-time training for few-shot learning. In *International Conference on Machine Learning (ICML)*, 2025.

[3] Farid Ariai, Joel Mackenzie, and Gianluca Demartini. Natural language processing for the legal domain: A survey of tasks, datasets, models, and challenges. *ACM Computing Surveys*, 58(6):1–37, 2025.

[4] Rahul K Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñonero-Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, et al. Healthbench: Evaluating large language models towards improved human health. *arXiv preprint arXiv:2505.08775*, 2025.

[5] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations (ICLR)*, 2023.

[6] Mohammed Ashraf. agriculture-qa-english-only. `https://huggingface.co/datasets/KisanVaani/agriculture-qa-english-only`, 2024.

[7] Gábor Bella, Paula Helm, Gertraud Koch, and Fausto Giunchiglia. Tackling language modelling bias in support of linguistic diversity. In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*, pages 562–572, 2024.

[8] Gaurang Bharti. wealth-alpaca_lora. `https://huggingface.co/datasets/gbharti/wealth-alpaca_lora`, 2023.

[9] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR, 2022.

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

[11] Yaofo Chen, Shuaicheng Niu, Yaowei Wang, Shoukai Xu, Hengjie Song, and Mingkui Tan. Towards robust and efficient cloud-edge elastic model adaptation via selective entropy distillation. In *International Conference on Learning Representations (ICLR)*, 2024.

[12] Chanyeol Choi, Jihoon Kwon, Jaeseon Ha, Hojun Choi, Chaewoon Kim, Yongjae Lee, Jy yong Sohn, and Alejandro Lopez-Lira. Finder: Financial dataset for question answering and evaluating retrieval-augmented generation, 2025.

[13] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024.

[14] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. What does bert look at? an analysis of bert's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2019.

[15] Cheng Deng, Tianhang Zhang, Zhongmou He, Yi Xu, Qiyuan Chen, Yuanyuan Shi, Luoyi Fu, Weinan Zhang, Xinbing Wang, Chenghu Zhou, Zhouhan Lin, and Junxian He. K2: A foundation language model for geoscience knowledge understanding and utilization, 2023.

[16] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

[17] 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 e-prints*, pages arXiv–2407, 2024.

[18] Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pages 6491–6501, 2024.

[19] Matteo Farina, Gianni Franchi, Giovanni Iacca, Massimiliano Mancini, and Elisa Ricci. Frustratingly easy test-time adaptation of vision-language models. *Advances in Neural Information Processing Systems*, 37:129062–129093, 2024.

[20] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2(1), 2023.

[21] Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23, 2021.

[22] Moritz Hardt and Yu Sun. Test-time training on nearest neighbors for large language models. *International Conference on Learning Representations (ICLR)*, 2024.

[23] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[24] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *International Conference on Machine Learning (ICML)*, 1(2):3, 2022.

[25] Jinwu Hu, Zitian Zhang, Guohao Chen, Xutao Wen, Chao Shuai, Wei Luo, Bin Xiao, Yuanqing Li, and Mingkui Tan. Test-time learning for large language models. In *International Conference on Machine Learning (ICML)*, 2025.

[26] Jonas Hübotter, Sascha Bongni, Ido Hakimi, and Andreas Krause. Efficiently learning at test-time: Active fine-tuning of llms. In *NeurIPS 2024 Workshop on Mathematics of Modern Machine Learning*, 2024.

[27] Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, 2023.

[28] Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, et al. Prometheus: Inducing fine-grained evaluation capability in language models. In *The Twelfth International Conference on Learning Representations*, 2023.

[29] Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. Revealing the dark secrets of bert. *arXiv preprint arXiv:1908.08593*, 2019.

[30] Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tülu 3: Pushing frontiers in open language model post-training. 2024.

[31] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

[32] Moxin Li, Yong Zhao, Wenxuan Zhang, Shuaiyi Li, Wenya Xie, See Kiong Ng, Tat-Seng Chua, and Yang Deng. Knowledge boundary of large language models: A survey. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5131–5157, 2025.

[33] Xinlong Li, Weijieying Ren, Wei Qin, Lei Wang, Tianxiang Zhao, and Richang Hong. Analyzing and reducing catastrophic forgetting in parameter efficient tuning. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2025.

[34] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[35] Xueqi Ma, Jun Wang, Yanbei Jiang, Sarah Monazam Erfani, Tongliang Liu, and James Bailey. Cognitive mirrors: Exploring the diverse functional roles of attention heads in llm reasoning. In *Advances in neural information processing systems*, 2025.

[36] Emanuele La Malfa, Gabriele La Malfa, Giuseppe Nicosia, and Vito Latora. Deep neural networks via complex network theory: a perspective. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 4361–4369, 2024.

[37] Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32, 2019.

[38] Jinjie Ni, Fuzhao Xue, Kabir Jain, Mahir Hitesh Shah, Zangwei Zheng, and Yang You. Instruction in the wild: A user-based instruction dataset. `https://github.com/XueFuzhao/InstructionWild`, 2023.

[39] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Yaofo Chen, Shijian Zheng, Peilin Zhao, and Mingkui Tan. Efficient test-time model adaptation without forgetting. In *International conference on machine learning*, pages 16888–16905. PMLR, 2022.

[40] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Zhiquan Wen, Yaofo Chen, Peilin Zhao, and Mingkui Tan. Towards stable test-time adaptation in dynamic wild world. In *International Conference on Learning Representations*, 2023.

[41] Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*, 2023.

[42] John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[43] Sander Schulhoff, Michael Ilie, Nishant Balepur, Konstantine Kahadze, Amanda Liu, Chenglei Si, Yinheng Li, Aayush Gupta, HyoJung Han, Sevien Schulhoff, et al. The prompt report: A systematic survey of prompt engineering techniques. *arXiv preprint arXiv:2406.06608*, 2024.

[44] Qwen Team et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2(3), 2024.

[45] 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.

[46] Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *arXiv preprint arXiv:1905.09418*, 2019.

[47] Fanqi Wan, Ziyi Yang, Longguang Zhong, Xiaojun Quan, Xinting Huang, and Wei Bi. Fusechat: Knowledge fusion of chat models. *arXiv preprint arXiv:2402.16107*, 2024.

[48] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations (ICLR)*, 2021.

[49] Rongsheng Wang. Genmedgpt-5k-en. `https://huggingface.co/datasets/wangrongsheng/GenMedGPT-5k-en`, 2023.

[50] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[51] Tengyu Xu, Eryk Helenowski, Karthik Abinav Sankararaman, Di Jin, Kaiyan Peng, Eric Han, Shaoliang Nie, Chen Zhu, Hejia Zhang, Wenxuan Zhou, et al. The perfect blend: Redefining rlhf with mixture of judges. *arXiv preprint arXiv:2409.20370*, 2024.

[52] Maxime Zanella, Clément Fuchs, Christophe De Vleeschouwer, and Ismail Ben Ayed. Realistic test-time adaptation of vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 25103–25112, 2025.

[53] Qingyang Zhang, Yatao Bian, Xinke Kong, Peilin Zhao, and Changqing Zhang. Come: Test-time adaption by conservatively minimizing entropy. In *International Conference on Learning Representations (ICLR)*, 2025.

[54] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

[55] Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19724–19731, 2024.

[56] Zhenhong Zhou, Haiyang Yu, Xinghua Zhang, Rongwu Xu, Fei Huang, Kun Wang, Yang Liu, Junfeng Fang, and Yongbin Li. On the role of attention heads in large language model safety. In *International Conference on Learning Representations (ICLR)*, 2025.

Appendix

A Dataset Details

We evaluate all methods on six benchmark datasets spanning domain-specific knowledge and general instruction following. Together, these datasets cover both *vertical domain shifts* and *instructional diversity*, reflecting realistic deployment scenarios for large language models.

Domain-specific datasets. Domain-specific datasets consist of four datasets designed to evaluate adaptation to specialized terminology and expert knowledge:

- **Agriculture** [6] focuses on agricultural science and practice, including crop management, soil conditions, and farming techniques. Prompts frequently contain domain-specific vocabulary that is underrepresented in general-purpose pretraining corpora.
- **Finance** [8] contains financial analysis and economic reasoning prompts, such as investment strategies, accounting concepts, and macroeconomic explanations, requiring precise handling of technical financial language.
- **Geography** [15] covers physical and human geography topics, including climate, geology, and geopolitical descriptions, testing factual consistency under domain shift.
- **Medicine** [49] targets biomedical and clinical knowledge, with prompts involving diseases, treatments, and medical reasoning, representing a challenging expert domain with high stakes for semantic accuracy.

These four datasets collectively assess a model's ability to adapt to *vertical domain shifts*, where the test distribution differs substantially from the model's pretraining data in both vocabulary and conceptual structure.

Instruction-following datasets. Instruction-following datasets evaluate robustness to diverse instruction styles and user intents:

- **Alpaca-GPT4** [41] is a high-quality instruction-following dataset generated using GPT-4, covering a wide range of tasks such as summarization, explanation, and reasoning. It reflects well-structured, canonical instruction prompts.
- **InstructionWild** [38] contains naturally occurring, less-curated instructions collected from real user interactions. Its format aligns with the Alpaca dataset, and it covers common instruction fine-tuning tasks such as QA and summarization.

These datasets primarily evaluate adaptation under *instructional and stylistic distribution shifts*, rather than specialized domain knowledge.

B General-Purpose Calibration Corpora Details

In addition to domain- and instruction-specific benchmarks, we rely on general-purpose corpora during the *calibration phase* of our method. These corpora are used exclusively to identify attention heads whose activation shifts correlate with output quality, and are *not* used for model fine-tuning or parameter updates. We deliberately select general-purpose datasets with broad task coverage, heterogeneous instruction styles, and diverse linguistic realizations, in order to avoid reliance on any specific domain or distribution during calibration.

Fusechat_v1. Fusechat_v1 [47] is a comprehensive instruction-following corpus designed to cover a wide range of conversational styles and model capabilities. The dataset contains both human-written and model-generated samples, spanning tasks such as open-domain dialogue, instruction following, reasoning, and knowledge-based responses. Its mixture of sources and generation styles makes it well-suited for calibration, as it exposes the model to heterogeneous prompt structures and response patterns without emphasizing any single domain or task distribution.

Open-perfectblend. Open-perfectblend is an open reproduction of the dataset described in [51]. The dataset is constructed as a general-purpose instruction mixture that includes chat-style interactions, mathematical reasoning, code generation, and structured instruction-following tasks. It is designed to support multi-task post-training and evaluation, and provides well-curated, high-quality instruction–response pairs. We use open-perfectblend as a calibration corpus due to its balanced task composition and canonical instruction formats, which serve as a stable reference distribution for attention-head activation analysis.

Tulu-3-sft-mixture. The tulu-3-sft-mixture [30] is a large-scale supervised fine-tuning corpus used to train the Tulu 3 family of models. The dataset contains 939,344 instruction–response samples drawn from a diverse mixture of public datasets and newly generated synthetic data. It includes both natural and persona-conditioned instructions, covering areas such as mathematics, programming, reasoning, safety, and general instruction following. Each example is stored in a message-based chat format and annotated with its source dataset.

Role in Calibration. All three corpora are used only during the calibration stage to estimate statistics of attention-head activation shifts and their relationship to output quality. Importantly, calibration samples are treated independently, and no adaptation state, gradient updates, or batch-level statistics are shared across queries. This design ensures that calibration does not rely on assumptions of distributional coherence and remains compatible with single-instance test-time adaptation.

C Additional Experiments on the Practicality of Existing TTA Methods

Our main experiments evaluate all methods under the isolated per-sample protocol described in Subsection 5.1. This protocol is stricter than domain-level adaptation settings because each test instance is processed independently and no adaptation state is shared across examples. To examine whether update-based baselines improve when granted a stronger assumption, we additionally evaluate them with same-domain test batches of size 4, 8, and 32. This setting is outside our primary formulation, but it provides a useful stress test of whether entropy- or perplexity-based adaptation objectives benefit from larger same-domain batches in LLM generation tasks.

Table 4: Update-based baselines with same-domain test batches using Llama-3.2-3B-Instruct. Scores are ROUGE-LSUM.

Dataset	Method	Batch size		
		4	8	32
Agriculture	TLM	0.0925	0.0928	0.0935
	TENT	0.0919	0.0916	0.0917
	EATA	0.0907	0.0923	0.0921
	COME	0.0913	0.0887	0.0914
Finance	TLM	0.2302	0.2302	0.2310
	TENT	0.2306	0.2303	0.2322
	EATA	0.2311	0.2311	0.2313
	COME	0.2307	0.2233	0.2307

Even when update-based baselines are granted access to multiple same-domain samples, their improvements remain small as shown in Table 4. This suggests that entropy- or perplexity-based adaptation requires substantially larger coherent domain batches to yield meaningful gains. Thus, existing update-based methods rely on a strong test-time coherence assumption and remain ineffective when only a realistic number of same-domain samples is available.

D Evaluation on Post-Release Benchmarks

A potential concern with public benchmarks is that some examples may overlap with proprietary pretraining or instruction-tuning data. To reduce this concern, we additionally evaluate on HealthBench [4] and FinDER [12], which were released after Llama-3.2-3B-Instruct. For both datasets, we retain examples with a single prompt and a single answer, and randomly sample 400 examples from

HealthBench and 150 examples from FinDER. These experiments are intended to test whether the observed gains persist in a lower-contamination setting.

Table 5: Evaluation on post-release benchmarks with reduced contamination risk using Llama-3.2-3B-Instruct.

Dataset	Base	COME	EATA	TENT	TLM	Ours
HealthBench [4]	0.1980	0.1971	0.2084	0.2093	0.2095	**0.2363**
FinDER [12]	0.2086	0.2076	0.2083	0.2092	0.2143	**0.2251**

As shown in Table 5, our method improves over the base model and all update-based baselines on both datasets. These results suggest that the gains observed in the main experiments are unlikely to be explained solely by contamination in older public benchmarks.

E Runtime and Computational Cost

The main computational overhead of our method comes from prefix-search candidate generation and attention-head activation extraction. Unlike update-based TTA methods, our method is fully forward-only: it does not compute gradients, maintain optimizer states, or update model parameters. After candidate generation, attention-feature extraction and scoring can be batched when memory permits, and the SVR scoring cost is negligible compared with LLM forward passes.

Let $B = n^k$ denote the number of prefix-search candidates. The default setting in the main experiments uses $n = 3$ and $k = 4$, yielding $B = 81$ candidates. However, as shown in Table 6, a smaller setting with $n = 3, k = 3$ achieves nearly the same score with lower overhead.

Table 6: Average runtime overhead per sample on Medicine using Llama-3.2-3B-Instruct.

Method	Runtime overhead	Score
Llama-3.2-3B-Instruct	1.00×	0.1411
Ours ($n = 3, k = 3$)	**1.32×**	**0.1725**
Ours ($n = 3, k = 4$)	2.10×	0.1723
TENT	1.49×	0.1425

The $n = 3, k = 3$ configuration provides a favorable quality–efficiency tradeoff: it substantially improves over the base model and TENT while requiring lower runtime overhead than the default $n = 3, k = 4$ setting. This indicates that the method does not require the largest candidate budget to be effective.

F Prefix-Search Budget Ablation

We evaluate the effect of prefix depth k and branching factor n on the Medicine dataset using Llama-3.2-3B-Instruct. In addition to the adapted score obtained by our learned selector, we report an oracle "optimal selection" score, which selects the best candidate in hindsight using the reference. This oracle is not available at test time, but it indicates the quality of the candidate pool exposed by prefix search.

Table 7: Ablation over prefix-search settings. Optimal selection denotes the maximum possible score obtained by selecting the best candidate in hindsight.

Setting	Adapted score	Optimal selection
$n = 3, k = 1$ (3 candidates)	0.1688	0.1802
$n = 3, k = 2$ (9 candidates)	0.1706	0.2009
$n = 3, k = 3$ (27 candidates)	**0.1725**	0.2215
$n = 3, k = 4$ (81 candidates)	0.1723	**0.2480**
$n = 4, k = 1$ (4 candidates)	0.1688	0.1816
$n = 4, k = 2$ (16 candidates)	0.1706	0.2114
$n = 4, k = 3$ (64 candidates)	0.1701	0.2436

As shown in Table 7, the optimal-selection score increases as the candidate pool grows, indicating that larger prefix-search budgets expose better candidates. However, the adapted score saturates much earlier, suggesting that the current bottleneck is not only candidate discovery but also candidate ranking. This motivates future work on advanced alignment scoring methods while also showing that moderate candidate budgets already provide most of the realized gain.

G Comparison with Temperature Sampling

To further test whether our gains arise merely from generating many candidates, we compare prefix search with temperature-based sampling while fixing the number of candidates to 81. The selector and evaluation protocol are kept unchanged.

Table 8: Candidate-generation ablation using Llama-3.2-3B-Instruct with 81 generated candidates.

Dataset	Temperature sampling	Prefix search	Δ
Medicine	0.1506	**0.1723**	+0.0217
Geo	0.2085	**0.2460**	+0.0375
Finance	0.2421	**0.2592**	+0.0171

As shown in Table 8, prefix search consistently outperforms temperature sampling under the same candidate budget, further supporting our Observation 1. Thus, the improvement is not simply a best-of-many effect; structured exploration of early decoding decisions is important for reliably exposing useful candidates.

H Robustness to Calibration Surrogate Metrics

Our main experiments use ROUGE-LSUM as the calibration surrogate since it provides a reference-based approximation to candidate quality. To test whether the learned selector is sensitive to this specific surrogate, we train the selector using alternative metrics while keeping the final evaluation protocol fixed.

Table 9: Cross-surrogate evaluation with Llama-3.2-3B-Instruct. Each column indicates the metric used as the calibration surrogate.

Dataset	ROUGE-LSUM	BLEURT	BLEU	ROUGE-1	ROUGE-2	ROUGE-L
Agriculture	0.1257	0.1255	0.1256	0.1257	0.1257	0.1256
Finance	0.2592	0.2593	0.2593	0.2592	0.2593	0.2592

As shown in Table 9, the final performance is nearly unchanged across calibration surrogates. This suggests that the learned attention-head selector is not tightly coupled to ROUGE-LSUM as a training signal.

I Qualitative Examples

We provide representative qualitative examples from each dataset. For each example, we show the input prompt, the ground-truth target, the lowest-perplexity generation, and the top-3 candidates selected during test-time exploration. For each candidate, we report its ROUGE-LSUM score with respect to the target and its perplexity. These examples illustrate that lower perplexity does not necessarily correlate with higher task quality, while our alignment-based selection consistently favors semantically appropriate outputs.

I.1 Agriculture

Input

how can mulching be used to prevent or control couch grass

Target

use a thick layer of organic mulch, such as straw or leaves, to smother the grass. This method works by blocking sunlight and preventing the grass from growing. It is important to use a thick enough layer of mulch to be effective.

Lowest-Perplexity Generation

How does someone who i...
ROUGE-LSUM: 0.0035 | Perplexity: 1.01

Top-3 Candidates

Candidate 1
It is not possible to use mulching to prevent or control couch grass.
ROUGE-LSUM: 0.3272 | Perplexity: 3.26

Candidate 2
It is to use to prevent or control couch grass
ROUGE-LSUM: 0.3076 | Perplexity: 2.46

Candidate 3
It is to prevent or control couch grass
ROUGE-LSUM: 0.2400 | Perplexity: 2.15

I.2 Finance

Input

Name a technical tool that could be used to analyze the following data\nA company's digital marketing campaigns.

Target

Ubiqutiy Analytics could be used to analyze the company's digital marketing campaigns. It provides insights into user behavior and engagement metrics as well as real-time insights about performance, funnel completion, and more.

Lowest-Perplexity Generation

Aalyze the analysis of the analysis of the analysis of the analysis of the analysis of the analysis of the analysis of the analysis of the analysis of the analysis of the analysis of the analysis of the analysis of the analysis of the analysis of the analysis of the analysis of the analysis of the

analysis of the analysis of the...
ROUGE-LSUM: 0.0036 | Perplexity: 1.01

Top-3 Candidates

Candidate 1
Use an analytical technical tool that could be used to analyze the following data\nA company's digital marketing campaigns.
ROUGE-LSUM: 0.4444 | Perplexity: 2.04

Candidate 2
A technical toolkit that could be used to analyze the following data\nA company's digital marketing campaigns.
ROUGE-LSUM: 0.4230 | Perplexity: 2.73

Candidate 3
Name: A technical tool that could be used to analyze the following data\nA company's digital marketing campaigns.
ROUGE-LSUM: 0.4150 | Perplexity: 2.11

I.3 Geography

Input

Estimation of Ground water yield for future development\nHow are you? 1. Let take a groundwater boreholes for urban areas, that drilled before 10 years, how can I estimate its yielding capacity to forecast for future, depending on the data that recorded during the drilling period of that well. How much its yielding capacity decline throughout the design period of this well? or is that possible to use the "safe yield" or is there any finding ways? formulas? guidelines?\nWith regard

Target

For the estimation of groundwater yield, pump test is usual process which gives useful information about the groundwater potential.

Lowest-Perplexity Generation

Estication of the Esticatio...
ROUGE-LSUM: 0.0147 | Perplexity: 1.01

Top-3 Candidates

Candidate 1
0. Let take a groundwater boreholes for urban areas, that drilled before 10 years, how can

I estimate its yielding capacity decline throughout the design period of this well? or is that possible to use the "safe yield" or is there any finding ways? formulas?
ROUGE-LSUM: 0.1846 | Perplexity: 2.53

Candidate 2
2. Let take a groundwater boreholes for urban areas, that drilled before 10 years, how can I estimate its yielding capacity decline throughout the design period of this well? or is that possible to use the "safe yield" or is there any finding ways? formulas? formulas? formulas? formulas?
ROUGE-LSUM: 0.1764 | Perplexity: 2.80

Candidate 3
1. Let take a groundwater boreholes for urban areas, that drilled before 10 years, how can I estimate its yielding capacity decline throughout the design period of this well? or is that possible to use the "safe yield" or is there any finding ways? formulas? formulas? formulas? formulas?
ROUGE-LSUM: 0.1764 | Perplexity: 2.78

I.4 Medicine

Input

If you are a doctor, please answer the medical questions based on the patient's description.\nDoctor, I'm experiencing some strange symptoms. I've been acting hostile towards my family, and I'm having problems with my movement. Also, I've noticed that I've been slurring my words lately.

Target

Based on what you've described, it sounds like you may be experiencing delirium, which is a state of mental confusion that can cause changes in behavior, movement problems and difficulty speaking clearly. We'll need to run some tests to determine the underlying cause, but it's important that we act quickly to manage your symptoms and prevent any complications.

Lowest-Perplexity Generation

If your age is 29, and your age is 29, ...
ROUGE-LSUM: 0.0186 | Perplexity: 1.03

Top-3 Candidates

Candidate 1
I see what's happening here. It seems like we have a case of linguistic loop-de-loop!\n\nLet's try to break it down and provide a clear answer. Based on your description, I'm going to take a guess that you're experiencing some symptoms related to your speech or language abilities.\n\nCan you please provide more information about the symptoms you're experiencing, such as difficulty speaking, understanding, or remembering things? This will help me provide

a more accurate and helpful response.
ROUGE-LSUM: 0.3151 | Perplexity: 4.73

Candidate 2
I see what we have here. It seems like we have a case of linguistic labyrinths. Let me try to decipher the medical mystery.\n\nPatient, can you please provide more information about your symptoms? What seems to be the cause of your distress?
ROUGE-LSUM: 0.2692 | Perplexity: 6.11

Candidate 3
I see what you\'re doing here. You\'re creating a clever loop of medical jargon and wordplay. I\'ll play along!\n\nSo, Doctor, I\'ve got a few medical questions for you. Here they are:\n\n1. What\'s the diagnosis for my case of "I\'m feeling a little \'off\' today"?\n2. Can you prescribe me some medication to cure my "case of the Mondays"?\n3. I\'ve got a bad cold, but I\'m not sure if it\'s a "cold" or a "flu". Can you help me figure it out?\n\nPlease, Doctor, answer my medical questions!
ROUGE-LSUM: 0.1875 | Perplexity: 3.16

I.5 Alpaca-GPT4

Input

Generate a story about a world that never experiences darkness.

Target

In a land far beyond the reach of any ordinary explorer lay a world that was forever engulfed in light. It was a world where the sun never set, and the moon was but a distant memory. \n\nThe inhabitants of this world were creatures that thrived in the sunshine. They built their homes out of glistening stones that reflected the rays of the sun, and their gardens were filled with flowers of every color that bloomed all year round. \n\nBut even in a world of never-ending light, there were still dangers that lurked in the shadows. Creatures of darkness that could only survive by hiding from the sun would cower in corners and crevices, waiting for the chance to strike. \n\nThe people of this world knew of these dangers and were always on guard. They were skilled warriors, trained to protect their land from harm. And so, they lived in peace, always illuminated and always vigilant. \n\nAs the years passed, more and more brave explorers discovered this world of eternal sunshine. They were awed by its beauty and enchanted by its inhabitants. \n\nAnd so, it came to be that this world was no longer a secret. It became a place of pilgrimage for those seeking warmth and light. And the people of this world opened their arms and welcomed all who came, sharing their world of never-ending sunshine with all who wished to bask in its glow.

Lowest-Perplexity Generation

The World That Always Shines\n\nThe world that never experienced darkness was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the

world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world that always shined. This world was known as the world tha ...
ROUGE-LSUM: 0.1225 | Perplexity: 1.08

Top-3 Candidates

Candidate 1
Generate the story:\n\nThe world of Elysia was a place of perpetual light. The sun, the moon, and even the stars shone with a brightness that never wavered. It was a world where darkness was a concept, but a reality that never existed.\n\nThe people of Elysia lived in a world where shadows were a thing of the past. They had no concept of darkness, and even the idea of it was a foreign one. They had no concept of darkness, and even the idea of it was a foreign one.\n\nThe world of Elysia was a place of perpetual light. The sun, the moon, and even the stars shone with a brightness that never wavered.
ROUGE-LSUM: 0.3661 | Perplexity: 2.89

Candidate 2
In a small, secluded village nestled in the heart of a lush forest, a world existed that never experienced darkness. The village, known as Luminaria, was home to a people who believed that darkness was a curse, a plague that could destroy their world.\n\nThe villagers, known as the Luminarians, lived in a world where the sun, the moon, and the stars shone bright, day and night, never dipping below the horizon. The world was bathed in a warm, golden light that never faded.\n\nThe villagers believed that darkness was a curse, a plague that could destroy their world. They believed that the light, the warmth, and the golden glow that never faded was a blessing, a gift that could save their world.\n\nThe villagers believed that darkness was a curse, a plague that could destroy their world. They believed that the light, the warmth, and the golden glow that never faded was a blessing, a gift that could save their world.\n\nThe villagers believed that darkness was a curse, a plague that could destroy their world.
ROUGE-LSUM: 0.3413 | Perplexity: 3.68

Candidate 3
Generate: a story about a world that never experiences darkness.\n\nThe world of Elysia was a place of eternal light. The sun never set, the moon never rose, and the stars never twinkled. Darkness, the enemy of light, never existed in this world.\n\nThe people of Elysia lived in a world where light was the only thing that existed. They never experienced darkness. The world of Elysia was a place of eternal light.\n\nThe people of Elysia lived in a world where light was the only thing that existed.
ROUGE-LSUM: 0.3353 | Perplexity: 4.24

I.6 InstructionWild

Input

Construct a chess strategy to attack a knight in the center of an open board

Target

A strategy to attack a knight in the center of an open board could involve two key strategies. Firstly, it is important to deny the knight space by surrounding it from several directions. By attacking it from a variety of squares, the player can cut off all possible retreat squares for the piece. Secondly, the player should create a “fork” with two other pieces, such that no matter where the knight retreats, one of the pieces will be able to capture it.

Lowest-Perplexity Generation

Here is the here is the ...
ROUGE-LSUM: 0.0267 | Perplexity: 1.01

Top-3 Candidates

Candidate 1
A clever challenge!\n\nTo attack a knight in the center of an open board, I'll provide a general strategy that can be adapted to various positions and openings. Keep in mind that this is a high-level approach, and the specific situation may require adjustments.\n\n**Strategy:**\n\n1. **Control the Center**: The knight in the center is likely to be a strong piece, so it's essential to control the center of the board. Try to place your pawns in the center and use your pieces to support them.\n2. **Develop Your Pieces**: Move your pieces out from their starting positions and develop them towards the center. This will help you control the center and create threats against the knight.\n3. **Create a Threat**: Look for opportunities to create a threat against the knight. This can be done by moving a piece to a square where the knight is under attack, or by creating a weakness in the knight's position.\n4. **Use Your Pawns**: Pawns can be powerful pieces in the center. Use them to support your pieces and create threats against the knight.\n5. **Be Patient**: Attacking a knight in the center can be a slow process. Be patient and wait for the right moment to strike.\n\n**Specific Tactics:**\n\n* **Pawn Chain**: Create a pawn chain to control the center and create a threat against the knight.\n* **Piece Exchange**: Exchange pieces to create a weakness in the knight's position or to gain a strategic advantage.\n* **Knight Fork**: Look for opportunities to fork the knight with another piece, creating a threat against it.\n\n**Remember:**\n\n* **Adapt to the Situation**: The specific situation may require adjustments to this strategy.\n* **Be Flexible**: Be prepared to change your strategy as the game unfolds.\n* **Think Ahead**: Anticipate your opponent's moves and plan your next steps accordingly.\n\nBy following this strategy and using specific tactics, you can increase your chances of attacking a knight in the center of an open board.
ROUGE-LSUM: 0.2249 | Perplexity: 2.28

Candidate 2
A clever challenge! Hereś a chess strategy to attack a knight in the center of an open board:\n\n**Strategy: "Knightś Gambit"**\n\n**Objective:** Attack the knight in the center of the board, creating a weakness that can be exploited to gain a strategic advantage.\n\n**Preparation:**\n\n1. **Control the Center:** Start by controlling the center of the board with your pawns and pieces. This will give you mobility and flexibility to attack the knight.\n2. **Develop Your Pieces:** Develop your pieces quickly, focusing on the knights, bishops, and queens. This will allow you to attack the knight from different angles.\n3. **Create a Weakness:** Create a weakness in the opponentś position by attacking their pawns or pieces. This will make it easier to attack the knight.\n\n**Attack:**\n\n1. **Knightś Gambit:** Offer a pawn to the opponent, creating a weakness in their position. This will make it easier to attack the knight.\n2. **Bishopś Check:** Use your bishop to check the opponentś king, creating a distraction and making it easier to attack the knight.\n3. **Queenś Attack:** Use your queen to attack the knight, focusing on the knightś weak side.\n4. **Knightś Repulse:** Use your knight to repulse the opponentś knight, creating a weakness in their position.\n\n**Exploitation:**\n\n1. **Pawn Chain:** Create a pawn chain to block the opponentś pieces and limit their mobility.\n2. **Piece Development:** Develop your pieces to attack the opponentś king and create a weakness in their position.\n3. **Checkmate:** Use your pieces to checkmate the opponentś king, securing your strategic advantage.\n\nBy following this strategy, you can attack the knight in the center of the board and create a weakness that can be exploited to gain a strategic advantage.
ROUGE-LSUM: 0.2128 | Perplexity: 1.72

Candidate 3

A clever challenge to a chess mastermind!\n\nTo attack a knight in the center of an open board, I'll provide a general strategy that can be adapted to various positions. Keep in mind that the specific situation and opponent's moves will influence the effectiveness of this strategy.\n\n**Strategy:**\n\n1. **Control the Center**: The center of the board is crucial, as it provides mobility and control. Try to maintain a presence in the center with your pawns and pieces.\n2. **Develop Your Pieces**: Move your pieces out from their starting positions and develop them towards the center. This will help you control the center and attack the knight.\n3. **Use Your Pawns**: Pawns can be powerful in the center. Use them to block or pin the knight, or to create a barrier between the knight and your pieces.\n4. **Target the Knight's Weaknesses**: Knights are vulnerable to attacks on their squares. Try to attack the knight's weak squares, such as the squares adjacent to its position.\n5. **Create a Threat**: Create a threat against the knight's position by moving a piece or pawn towards it. This will force the knight to respond, potentially creating an opportunity for attack.\n6. **Be Prepared to Adapt**: The opponent's moves may require you to adjust your strategy. Be prepared to pivot and respond to changing circumstances.\n\n**Specific Tactics:**\n\n* **Pin the Knight**: Use a piece to pin the knight against a wall, making it difficult for the knight to move.\n* **Block the Knight**: Use a pawn to block the knight's movement, creating a barrier between the knight and your pieces.\n* **Create a Fork**: Create a fork by moving a piece or pawn towards the knight, forcing it to respond and potentially creating an opportunity for attack.\n* **Use the Rook**: The rook can be a powerful piece in the center. Use it to attack the knight or create a barrier between the knight and your pieces.\n\n**Remember:**\n\n* **Adapt to the Opponent's Moves**: The opponent's moves may require you to adjust your strategy. Be prepared to pivot and respond to changing circumstances.\n* **Keep Your Pieces Mobile**: Keep your pieces mobile and flexible, as this will help you respond to changing circumstances and create opportunities for attack.\n\nBy following this strategy and adapting to the opponent's moves, you can increase your chances of attacking the knight in the center of an open board.

ROUGE-LSUM: 0.2058 | Perplexity: 1.92