Proceedings of Machine Learning Research **318** (2026)
The 39th Canadian Conference on Artificial Intelligence

Canadian AI 2026
25–29 May 2026, Vancouver, BC

Probabilistic TopK Sparse Autoencoder for Interpreting the Activations of Large Language Models

Raymond Li*, Chuyuan Li, Gabriel Murray, Giuseppe Carenini
University of British Columbia

Abstract

Sparse Autoencoders (SAEs) have emerged as a popular solution for extracting interpretable features from language model activations. However, existing SAE designs suffer from deterministic activations that starve gradients to "dead" components, and produce uncalibrated coefficients that provide no meaningful notion of uncertainty. To address these limitations, we introduce Probabilistic TopK SAE, a novel approach that augments the TopK SAE with probabilistic gating through a Binary Concrete distribution. This stochastic sampling improves both optimization dynamics (reducing dead features) and interpretability calibration, where coefficient magnitude more reliably reflects feature presence. Empirical experiments with GPT-2 and Qwen3 shows that our method achieves consistent Pareto improvements over the baselines in high sparsity settings (small number of activated features) while maintaining a larger set of alive dictionary features.

Keywords: NLP, LLM, Mechanistic Interpretability, SAE

1. Introduction

The aim of mechanistic interpretability in Natural Language Processing (NLP) aims to understand how language models transform inputs into outputs through their learned representations and circuits [1]. One fundamental challenge is *Superposition* [2], where language models learn to represent more features than the number of neurons, allowing models to compress a large amount of information into a limited parameter space. This compression can lead to polysemantic neurons, where individual neurons respond to multiple different features depending on the context, making their activations difficult to interpret directly.

Sparse Autoencoders (SAEs) [3–6] address superposition by learning sparse decompositions of activations using an overcomplete dictionary of features, where each disentangled feature aligns with a single concept [7]. This allows SAEs to extract a larger set of monosemantic neurons that can be isolated for interpretation through sparsity enforcement. For example, the $\mathcal{L}_1$ penalty can be applied in conjunction with the ReLU activation which pushes small coefficients toward zero [4, 5], while the TopK activation selects only the K largest pre-activation values and sets all others to zero, guaranteeing exactly K active features per input and allows linear scaling to very large models [6]. The development of such SAE-based strategies has enabled significant advancements in mechanistic interpretability, allowing the identification of interpretable features across multiple scales and layers of language models [8, 9] with successful downstream applications in circuit discovery [10] and model steering [11].

Nevertheless, a drawback of existing sparsity mechanisms is that they often **produce deterministic, uncalibrated activations**, which are brittle to early "winner-take-all" dynamics. The main culprit is that existing sparsity constraints excessively starve gradients of non-selected units (e.g., ReLU, TopK), with features that start slightly less aligned with the data being further suppressed and never recovering. Additionally, while coefficient magnitudes are used to determine feature usage, these values **do not provide any meaningful notion of uncertainty**. This is because these magnitudes are scale-dependent and can

* raymondl@cs.ubc.ca

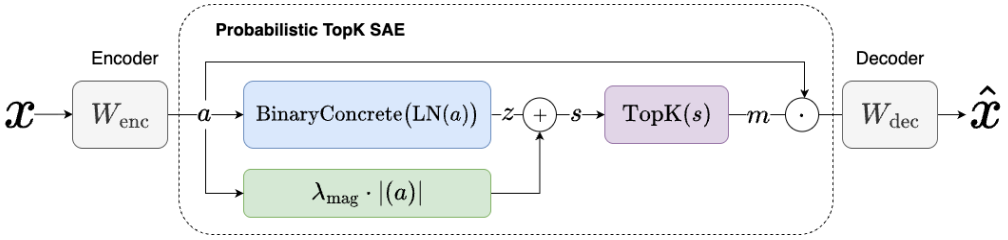

Figure 1. Overview of our proposed Probabilistic TopK SAE architecture. LN denotes the Layer Normalize layer. The samples z from the Binary Concrete gates are combined with the scaled pre-action magnitudes as scores for the TopK selector.

be arbitrarily rescaled through encoder–decoder weight trade-offs. As a result, they do not reliably reflect confidence in feature selection.

To address these limitations, we propose **Probabilistic TopK SAE**, which augments the standard TopK autoencoders [6] with Binary Concrete gates [12, 13]. The stochasticity during training helps reduce the invariance to rescaling by introducing a probability distribution over each dictionary component. Moreover, we hypothesize that stochastic gating also mitigates the "winner-take-all" collapse by reducing the prevalence of dead units. By annealing the temperature of the Binary Concrete distribution, the model should smoothly transition from exploratory, probabilistic feature usage early in training to sharper, near-deterministic selection at convergence. We evaluate our method on the residual streams of GPT-2 [14] and Qwen3-0.6B [15], and find that it consistently achieves Pareto improvements over baseline methods, yielding better reconstruction fidelity at each sparsity level, while producing activation magnitudes that are more correlated with the presence of the underlying features.

2. Method

2.1. Probabilistic Gating

We introduce stochastic gating using the Binary Concrete distribution [12, 13], a continuous relaxation of a Bernoulli random variable. This allows each dictionary component to be probabilistically activated during training by adding Gumbel noise $u \sim \text{Uniform}(0, 1)$:

$$\text{BinaryConcrete}(\alpha) = \sigma\left(\frac{\log \alpha + \log u - \log(1-u)}{\beta}\right), \tag{2.1}$$

where α controls the activation probability and β is a temperature parameter that determines the sharpness of the distribution. This mechanism allows stochastic exploration of binary decisions while remaining amenable to gradient-based optimization.

2.2. Probabilistic TopK SAE

As shown in Figure 1, our method extends the standard TopK SAE [6] by integrating input-dependent Binary Concrete gates before the TopK operation. From input $x \in \mathbb{R}^n$, the encoder first computes the pre-activations $a \in \mathbb{R}^M$ before a layer normalization layer, preventing large-magnitude activations from dominating the Binary Concrete sampling:

$$\begin{aligned} a &= W_{\text{enc}}(x - b_{\text{dec}}), \\ z &= \text{BinaryConcrete}\big(\text{LayerNorm}(a)\big). \end{aligned} \tag{2.2}$$

To reduce noise, we compute a combined score s that incorporates both the probabilistic gate samples z and the absolute magnitudes from the deterministic pre-activations a, balanced by hyperparameter λ_{mag}. In practice, we set λ_{mag} to a extremely small value of $1e^{-4}$.

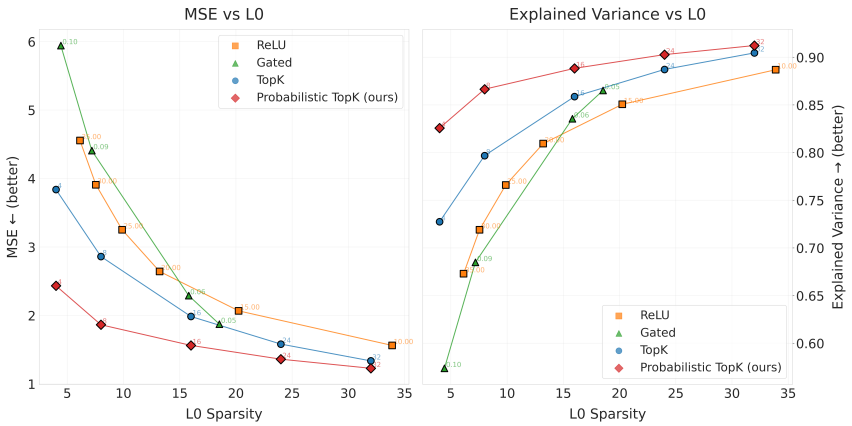

Figure 2. Reconstruction vs. sparsity Pareto curve for layer 8 of GPT-2 small. Ours has a better reconstruction-sparsity trade-off than other activation functions.

Finally, the decoder reconstructs $\hat{x} \in \mathbb{R}^n$ after the TopK operation (Equation 2.3).

$$\begin{aligned} s &= z + \lambda_{\text{mag}}|a|, \\ \hat{x} &= W_{\text{dec}}(\text{TopK(s)} \odot a) + b_{\text{dec}}. \end{aligned} \tag{2.3}$$

2.3. **Training**

The training objective is the standard mean squared error (MSE) between input and reconstructed activations $\mathcal{L}_{\text{MSE}} = \|x - \hat{x}\|_2^2$. To balance exploration and stability in feature selection, we apply temperature annealing by exponentially decaying β throughout training: $\beta_t = \beta_0 \cdot (\beta_T/\beta_0)^{t/T}$. This mitigates the "winner-take-all" collapse by enabling the model to stochastically explore more feature at the beginning of training.

3. **Experiments**

We train Probabilistic SAEs on the residual streams of GPT-2 [14] and Qwen3-0.6B [15]. Training details are provided in Appendix A. For comparison, we include three baselines: standard ReLU SAE [3, 4], TopK SAE [6], and Gated SAE [5].

3.1. **Pareto Performance**

From Figure 2 and Figure 3, our proposed Probabilistic TopK SAE achieves a Pareto improvement over the TopK SAE baselines across all K. This is more significant at high sparsity (i.e., $K = 4, 8$), where we significantly outperform the baselines. A key benefit of our design is the use of β-annealing, which encourages the model to explore more features and prevents premature commitment to suboptimal selections. However, we do see a diminished benefit at higher K (e.g., $K = 32, 64$), where exploration becomes less important.

We further analyze the number of active components during training. Maintaining a large set of active features is crucial for reconstruction performance: if many features become inactive early in training, the model's effective capacity is reduced, leaving fewer basis functions to represent the data. This can result in incomplete coverage of the input space and higher reconstruction error [16]. As shown in Figure 4, our model maintains a substantially higher number of active components throughout training compared to the TopK baseline, which helps preserve model capacity and supports improved reconstruction. A similar trend is found on layer 26 of the Qwen3 model ($K = 16, 32, 64$).

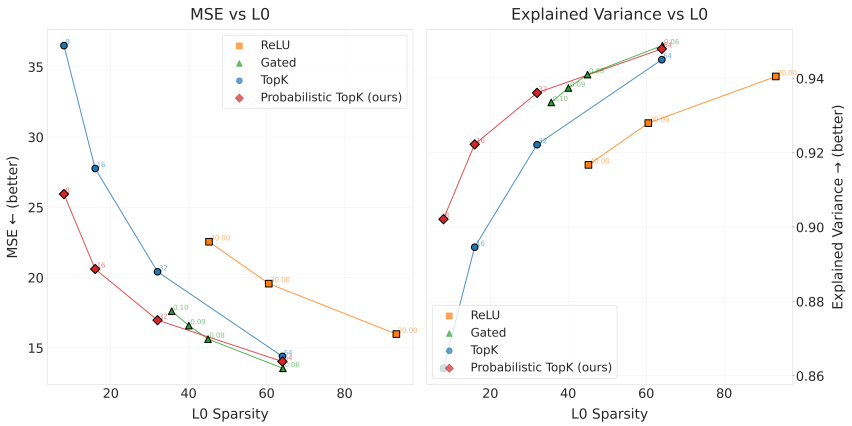

Figure 3. Reconstruction vs sparsity Pareto curve for layer 26 of Qwen3-0.6B. Probabilistic TopK (ours) has a better reconstruction-sparsity trade-off than Top-K and ReLU, and is comparable to Gated SAE at lower sparsity levels.

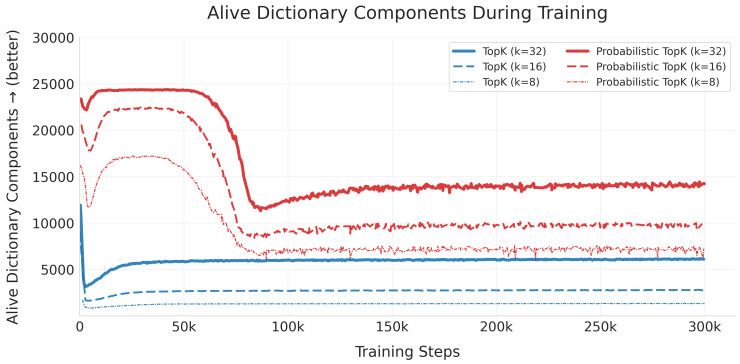

Figure 4. Number of alive dictionary components during training for TopK and Probabilistic TopK SAEs for layer 8 of GPT-2 ($K = 8, 16, 32$).

3.2. **Automatic Interpretability**

To evaluate the interpretability of dictionary components and measure the correlation between activation magnitude and interpretability, we perform automatic interpretability using the detection method provided by [17], where we stratify neuron activations into percentile buckets and measuring interpretability scores within each stratum. Specifically, we select runs with similar sparsity rate (i.e., L_0) and divide each neuron's activation distribution into 5 percentile buckets. We then generate separate explanations from examples in each activation bucket before scoring the explanation from each bucket.

In Figure 5, we present the calibration curves showing interpretability scores as a function of activation percentile, where we find that our Probabilistic TopK achieves highest correlation ($r = 0.540$ for $K = 8$, and $r = 0.559$ for $K = 16$) as well as the highest interpretability score for neurons in the top 20%. The strong calibration of our proposed methods suggests that the probabilistic gating strengthens the relationship between a neuron's activation strength and its confidence to possess the underlying feature. This property allows the better sampling of activations for explanations as well as implications to improved downstream performance such as circuit discovery and model steering.

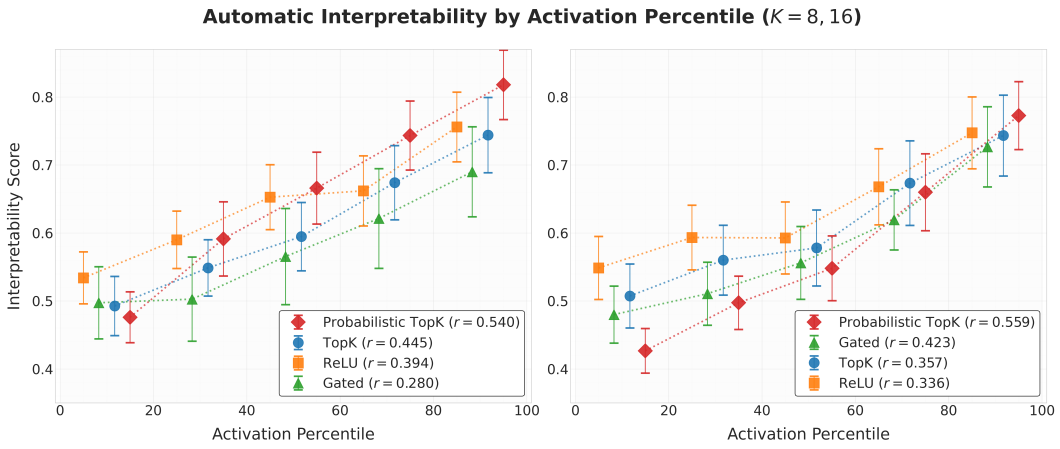

Figure 5. Mean interpretability scores against activation percentile buckets for four SAE variants trained on GPT-2 small (layer 8). Error bars indicate 95% confidence intervals (150 random neurons). Runs with similar L_0 are selected at each sparsity level.

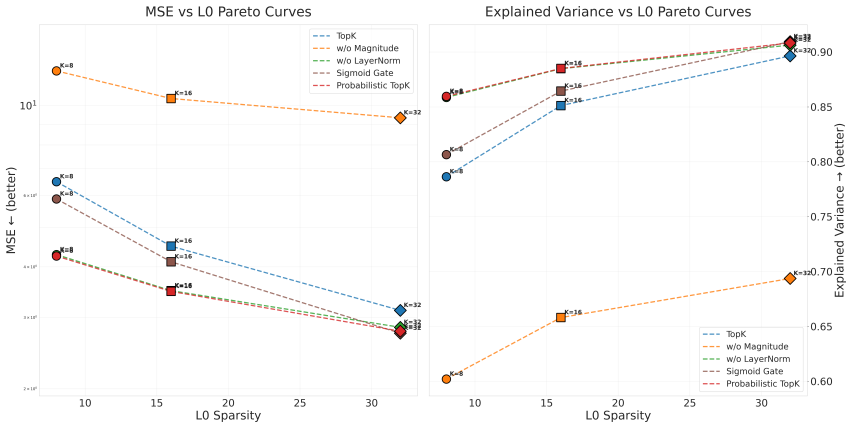

Figure 6. Ablation studies on Layer 8 of GPT-2 ($K = 8, 16, 32$) for the 3 variants include our full model and the TopK SAE baseline without any additional augmentations.

3.3. **Ablation Study**

To validate our design choices, we conduct comprehensive ablation studies: (1) **Top-K**: standard Top-K SAE without probabilistic gating (Equation 2.1); (2) **w/o Magnitude**: setting $\lambda_{\text{mag}} = 0$ and using only sampled z for scoring (Equation 2.3); (3) **w/o LayerNorm**: removing layer normalization layer before binary concrete operation (Equation 2.2); (4) **Sigmoid Gate**: replacing Binary Concrete with sigmoid gates (Equation 2.2). All variants are conducted across different sparsity levels ($K = 8, 16, 32$) on GPT-2. Additional analysis of temperature effects is provided in Appendix B.

From Figure 6, we see that scoring without magnitude contributions (w/o Magnitude) performs significantly worse than all other variants. Using only z fails to capture fine-grained differences in feature alignment, leading to noisy reconstructions, whereas even a small $\lambda_{\text{mag}} = 1e^{-4}$ effectively biases selection toward better-aligned features. We also find that replacing the Binary Concrete with a standard Sigmoid Gate degrades reconstruction

performance for smaller K (4, 8), where the TopK activation defaults to a small set of dictionary components, leading to suboptimal reconstruction without the exploration. Lastly, removing the layer normalization layer does not show a significant performance drop.

4. Conclusion

We introduced the Probabilistic TopK SAE, an extension of TopK SAE with stochastic gate through the Binary Concrete distribution. This encourages exploration during training, where the resulting models achieve a stronger Pareto frontier on the sparsity–reconstruction trade-off than deterministic TopK baselines, reflecting both higher dictionary utilization and better stability. For future work, we will perform additional downstream experiments (e.g., circuit discovery, model steering) and improve the existing architecture at higher K values.

Acknowledgements

We thank Anji Ma for helping with the experiments and providing valuable discussion.

References

[1] L. Bereska and S. Gavves. "Mechanistic Interpretability for AI Safety - A Review". In: *TMLR 2024* (2024).

[2] N. Elhage et al. "Toy Models of Superposition". In: *Transformer Circuits Thread* (2022).

[3] T. Bricken et al. "Towards Monosemanticity: Decomposing Language Models With Dictionary Learning". In: *Transformer Circuits Thread* (2023).

[4] H. Cunningham et al. "Sparse Autoencoders Find Highly Interpretable Features in Language Models". In: *ICLR*. 2024.

[5] S. Rajamanoharan et al. "Improving Sparse Decomposition of Language Model Activations with Gated Sparse Autoencoders". In: *NeurIPS*. Vol. 37. 2024, pp. 775–818.

[6] L. Gao et al. "Scaling and evaluating sparse autoencoders". In: *ICLR*. 2025.

[7] D. Shu et al. "A Survey on Sparse Autoencoders: Interpreting the Internal Mechanisms of Large Language Models". In: *EMNLP Findings*. Nov. 2025, pp. 1690–1712.

[8] A. Templeton et al. "Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet". In: *Transformer Circuits Thread* (2024).

[9] T. Lieberum et al. "Gemma Scope: Open Sparse Autoencoders Everywhere All At Once on Gemma 2". In: *BlackboxNLP Workshop*. Miami, Florida, US, Nov. 2024, pp. 278–300.

[10] S. Marks et al. "Sparse Feature Circuits: Discovering and Editing Interpretable Causal Graphs in Language Models". In: *ICLR*. 2025.

[11] R. Bayat et al. "Steering Large Language Model Activations in Sparse Spaces". In: *COLM*. 2025.

[12] C. J. Maddison et al. "The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables". In: *ICLR*. 2017.

[13] C. Louizos et al. "Learning Sparse Neural Networks through L_0 Regularization". In: *ICLR 2018*.

[14] A. Radford et al. "Language models are unsupervised multitask learners". In: *OpenAI blog* (2019).

[15] A. Yang et al. "Qwen3 technical report". In: *arXiv preprint arXiv:2505.09388* (2025).

[16] J. Bloom. *Open Source Sparse Autoencoders for all Residual Stream Layers of GPT2 Small*. 2024.

[17] G. S. Paulo et al. "Automatically Interpreting Millions of Features in Large Language Models". In: *ICML*. 2025.

[18] N. Nanda and J. Bloom. *TransformerLens*. 2022.

[19] A. Gokaslan and V. Cohen. *OpenWebText Corpus*. 2019.

[20] G. Penedo et al. "The FineWeb Datasets: Decanting the Web for the Finest Text Data at Scale". In: *NeurIPS: Datasets and Benchmarks Track*. 2024.

[21] D. P. Kingma and J. Ba. "Adam: A Method for Stochastic Optimization". In: *ICLR*. 2015.

Appendix A. Experiment Settings and Hyperparameters

We train Probabilistic SAEs on the residual streams of GPT-2 [14] and Qwen3-0.6B [15] using the TransformerLens library [18], following prior work [6]. The models are trained on OpenWebText [19] and FineWeb [20] for GPT-2 and Qwen, respectively, with a context length of 1024. Following Gao et al. [6], we apply SAEs on a layer near the end of the network, which should contain more meaningful features without being specialized for next-token predictions. Specifically, we use layer 8 for GPT-2, and layer 26 for Qwen3.

Inputs are mean-centered across the hidden dimension and normalized to unit norm before being passed to the autoencoder. We use the Adam optimizer [21] with a learning rate (lr=$1e^{-3}$, $\beta_1 = 0.9$ and $\beta_2 = 0.99$), gradient clipping at 10.0, and a cosine learning-rate schedule with exponential decay after warm-up, while annealing the temperature β from 5.0 to $1e^{-4}$. Decoder weights are orthogonally initialized and encoder weights are set as their transpose, with the dictionary size fixed to 16x the hidden dimension. Evaluation uses a context window of 64 over 50M tokens. GPT-2 experiments run on a single A6000 (48GB), and Qwen on a single H100 (80GB), each completing within 12 hours.

Appendix B. Analysis on Temperature Values

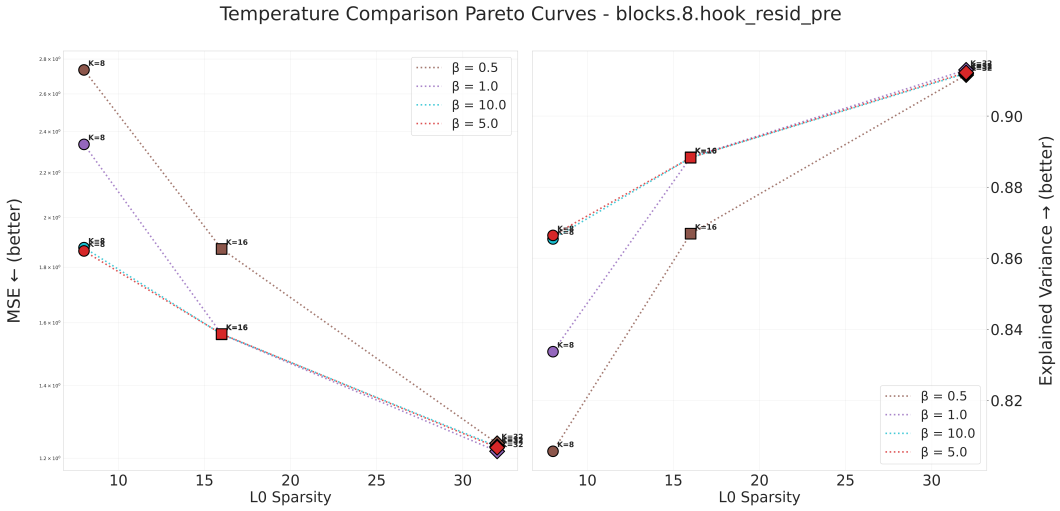

Figure 7. Pareto curve for different temperatures values ($\beta = 0.5, 1, 5, 10$) on Layer 8 of GPT-2 ($K = 8, 16, 32$).

We also experiment with different temperature values to assess their impact on the reconstruction fidelity by running ($\beta \in \{0.5, 1.0, 5.0, 10.0\}$) in GPT-2. From the results illustrated in Figure 7, we find that reconstruction quality is highly sensitive to the temperature parameter, particularly at low sparsity levels. At $K = 8$, increasing β from 0.5 to 5.0 reduces MSE by approximately 27% (from 5.83 to 4.25 on layer 10). However, the benefits plateau beyond $\beta = 5.0$, with minimal improvements observed at $\beta = 10.0$. The temperature effect diminishes as sparsity increases. At $K = 32$, the performance gap between $\beta = 0.5$ and $\beta = 5.0$ narrows significantly, suggesting that higher sparsity naturally provides more gradient paths, reducing the importance of exploration through stochastic gating. In high sparsity settings, temperature acts as a regularizer that prevents premature feature specialization.