Test-time Diverse Reasoning by Riemannian Activation Steering

Ly Tran Ho Khanh[*1], **Dongxuan Zhu**[*1], **Man-Chung Yue**[2], **Viet Anh Nguyen**[1]

[1]The Chinese University of Hong Kong
[2]The University of Hong Kong
hokhanhlytran@cuhk.edu.hk, dxzhu@se.cuhk.edu.hk, mcyue@hku.hk, nguyen@se.cuhk.edu.hk

Abstract

Best-of-N reasoning improves the accuracy of language models in solving complex tasks by sampling multiple candidate solutions and then selecting the best one based on some criteria. A critical bottleneck for this strategy is the output diversity limit, which occurs when the model generates similar outputs despite stochastic sampling, and hence recites the same error. To address this lack of variance in reasoning paths, we propose a novel unsupervised activation steering strategy that simultaneously optimizes the steering vectors for multiple reasoning trajectories at test time. At any synchronization anchor along the batch generation process, we find the steering vectors that maximize the total volume spanned by all possible intervened activation subsets. We demonstrate that these steering vectors can be determined by solving a Riemannian optimization problem over the product of spheres with a log-determinant objective function. We then use a Riemannian block-coordinate descent algorithm with a well-tuned learning rate to obtain a stationary point of the problem, and we apply these steering vectors until the generation process reaches the subsequent synchronization anchor. Empirical evaluations on popular mathematical benchmarks demonstrate that our test-time Riemannian activation steering strategy outperforms vanilla sampling techniques in terms of generative diversity and solution accuracy.

Code — https://github.com/lythk88/SPREAD
Extended version — https://arxiv.org/pdf/2511.08305

1 Introduction

Language models (LMs) have revolutionized tasks from code generation (Chen et al. 2021), symbolic reasoning (Wei et al. 2022), to mathematical problem solving (Lewkowycz et al. 2022). In these tasks, the quality of the final solution can improve significantly by exploring multiple plausible reasoning paths and then presenting the best solution aggregated from the information from these paths. This strategy aligns with our problem-solving intuitions, where each path of reasoning explores different techniques, generalizability, and scientific values. A simple and effective method to exploit this explore-then-aggregate idea is Best-of-N sampling (Lightman et al. 2023; Ni et al. 2023), where the model generates N candidates and uses a reward model to select the most plausible answer. The final accuracy of this Best-of-N strategy is constrained by the diversity of the generated candidates. A straightforward method to encourage exploration is to use a stochastic sampling decoder when generating the next token, leading to a zoo of emerging methods (Fan, Lewis, and Dauphin 2018; Holtzman et al. 2020; Meister et al. 2023). These methods construct a token-level search space with varying access to the internals of an LM, such as logits, next-token distributions, or probability scores.

Yet, stochastic decoding methods frequently suffer from *diversity collapse*, where the outputs of LMs may converge to nearly identical reasoning paths (Yun et al. 2025; Dang et al. 2025). This phenomenon has sparked more aggressive search strategies, such as contrastive search (Su and Collier 2023), balancing the model confidence with the degeneration penalty to avoid repetitiveness. Alternatively, Vijayakumar et al. (2016) maintains multiple diverse hypotheses during beam search by diversity-promoting objectives. Additionally, self-speculative decoding (Zhang et al. 2024) and speculative sampling (Leviathan, Kalman, and Matias 2023) inherently promote diversity through their multi-step prediction mechanisms. Most of these search strategies are computationally intensive, as they require joint consideration of reasoning trajectories distribution (Nagarajan et al. 2025).

Another challenge to promoting reasoning diversity is measuring it. Popular metrics like lexical or semantic diversity may not capture the reasoning diversity well. Lexical diversity measures the number of different meanings or perspectives conveyed among sequences, but is sensitive to text length, rephrasing, or synonyms (Bestgen 2024). Similarly, semantic diversity quantifies the different meanings or perspectives, but it is sensitive to adding or removing details from the text (Han, Kim, and Chang 2022). Computationally, both often invoke extra neural architecture for evaluation, thus adding load to the inference process.

In this paper, we look at the reasoning diversity through another proxy: the diversity of the hidden activations of the generated sequences. Hidden activations are the internal representations computed by a language model at each layer and token position as it processes input. They encode intermediate computations and abstract concepts, acting as the latent "thinking space" where reasoning, memory, and structure are implicitly formed. While there may be strong corre-

[*]These authors contributed equally.

lations between the activations and the reasoning paths, there is unfortunately no one-to-one equivalence between them. Thus, admittedly, promoting diversity of the hidden activations does not necessarily lead to diversity in the reasoning paths. However, recent progress suggests that encouraging diversity among neuron activations within the same layer increases the capacity of the model to learn a broader range of features (Laakom et al. 2023). In general, increasing the diversity of hidden activations can reduce estimation error and improve generalization. Moreover, models that facilitate diverse internal activations may be better equipped to represent and synthesize multiple reasoning strategies, as the richer internal space allows for more varied "thought paths" (Naik et al. 2023). Recent results in interpretability indicate that different features or activation clusters can sometimes correspond to different "reasoning circuits" or strategies, especially in LMs (Lindsey et al. 2025). These observations suggest that we should design fast and parameter-efficient mechanisms to promote the diversity of hidden activations during generation, hoping to induce reasoning diversity and improve the accuracy for Best-of-N sampling.

Contributions. We summarize our contributions as follows:

- We propose the **SP**herical intervention for **REA**soning **D**iversity (SPREAD), an unsupervised activation steering method that improves the diversity among reasoning trajectories. At a synchronization anchor, SPREAD extracts the hidden activations from all sequences, then computes the steering vectors that maximize the total volume spanned by all possible subsets of the intervened activations. SPREAD then adds these steering vectors to the respective activations of all subsequent tokens until the next synchronization anchor.
- We show that determining the optimal steering vectors can be reformulated as a manifold optimization problem defined over the product of spheres, where the log-determinant objective function captures the geometric diversity of the intervened activations. We propose using a Riemannian block coordinate descent algorithm, which exploits the product structure of the manifold constraints. We also study the theoretical properties of the optimization problem and prove the convergence guarantee of the algorithm for appropriate step sizes.

Using the steering methods, SPREAD uses readily available hidden activations from the generation process and does not require any additional neural architectures to measure quality or reasoning diversity. Moreover, SPREAD could rely on only one hyperparameter that prescribes the relative radii of the intervention vectors, and it relieves the burden of parameter tuning at inference time.

Our paper unfolds as follows: Section 2 reviews the related works on generative diversity and activation steering, Section 3 presents the mathematical formulation of the SPREAD framework, Section 4 develops the manifold optimization algorithm for computing the optimal steering vectors, and Section 5 empirically illustrates the performance of SPREAD on mathematical reasoning tasks.

Notations. The space of p-dimensional vectors is denoted $\mathbb{R}^p$. For any $x \in \mathbb{R}^p$, $\|x\|_2$ is its Euclidean norm. For a matrix $A \in \mathbb{R}^{p\times N}$, we use $\|A\|_F$ for the Frobenius norm. We use $\nabla\ell(V)$ and $\nabla^2\ell(V)$ for the gradient of function ℓ and Hessian matrix of function ℓ with respect to V in the Euclidean sense; while $\mathrm{grad}\ \ell$ and $\mathrm{Hess}\ \ell$ are the Riemannian counterparts. We use $\nabla_i\ell(V)$, $\nabla_i^2\ell(V)$, $\mathrm{grad}_i\ \ell(V)$ and $\mathrm{Hess}_i\ \ell(V)$ for the corresponding operator with respect to the i-th block v_i while fixing all other blocks. All proofs are relegated to the appendix.

2 Literature Review

Diversity in generation. Classical approaches, including temperature sampling (Ackley, Hinton, and Sejnowski 1985), top-k sampling (Fan, Lewis, and Dauphin 2018), nucleus sampling (Holtzman et al. 2020), and typical decoding (Meister et al. 2023), promote output diversity by introducing stochasticity into the generation process. Prompt-centric techniques have also been shown to enrich the diversity of reasoning. Li et al. (2023b) focuses on prompt diversity by generating diverse prompts to explore different reasoning paths, while filtering out incorrect answers by a weighted voting scheme. Naik et al. (2023) proposes a self-reflective prompting that leverages the LLM as a guide to design a diverse set of approaches for complex reasoning tasks. Wang et al. (2025) uses multiple adaptive steering vectors for different hallucination types, though maintaining multiple vectors is computationally expensive. Chung et al. (2025) achieves diversity through parameter-efficient prefix tuning, but effectiveness is sensitive to training data quality.

Activation steering is a lightweight and interpretable method for controlling LMs. This approach injects direction vectors into the residual stream of transformer layers to steer generation toward desired attributes (e.g., truthfulness, sentiment, toxicity) without modifying model weights. *Contrastive steering methods* derive directions by comparing activations between positive and negative examples of desired behaviors. Turner et al. (2023) computes steering vectors by averaging residual stream differences between factual and hallucinatory responses, while Stolfo et al. (2025) contrasts activations with and without specific instructions. *Probe-based steering methods* instead learn to identify relevant concepts through trained classifiers, then extract steering directions from the learned representations (Li et al. 2023a; Zhang et al. 2025). Despite demonstrating effectiveness across domains like toxicity reduction (Zhang et al. 2025), and truthfulness enhancement (Turner et al. 2023), activation steering restricts its broader applicability. Most existing approaches rely on single, fixed direction vectors that constrain model outputs to narrow behavioral modes. The challenge becomes even more pronounced in mathematical reasoning, where defining clear positive and negative exemplars for contrastive learning is inherently difficult because mathematical correctness involves complex, context-dependent logical structures that resist simple binary classification. These domain-specific challenges explain why prior activation steering research has largely avoided mathematical reasoning applications.

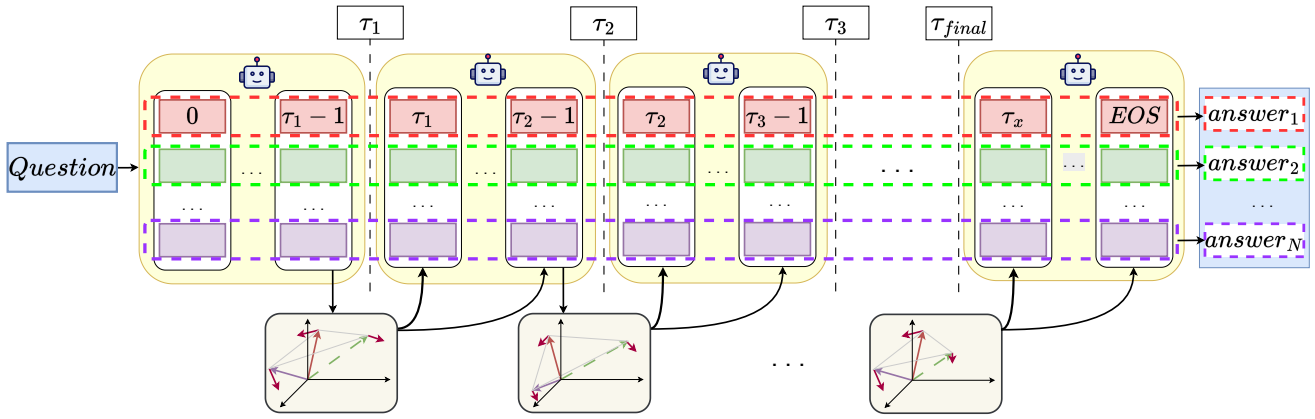

Figure 1: Overview of SPREAD for generating N diverse reasoning answers simultaneously. At each decoding step τ_t, we extract the hidden vectors corresponding to the last token in each path. These hidden vectors serve as inputs to Algorithm 1, where they are projected into a shared activation space to compute N steering vectors. This process is repeated until an end-of-sequence EOS token is generated.

3 Activation Steering for Diverse Generation Under the Optimization Lens

We aim to elicit diverse reasoning paths from an LM at test time without any fine-tuning by sampling multiple sequences per input prompt. Figure 1 shows the core idea of the SPREAD framework, which intervenes directly in the model's activation space during the autoregressive generation process. Specifically, we simultaneously generate N output sequences and extract the final hidden state vectors after τ tokens, $H = [h_1, \ldots, h_N] \in \mathbb{R}^{p \times N}$, for all sequences. We then compute an additive steering vector v_i for each hidden state h_i, yielding a new set of hidden states $H_{new} = H + V$, where $V = [v_1, \ldots, v_N]$. The core of our approach is to maximize a geometric measure of diversity for H_{new} (2), or equivalently with respect to the steering vectors V. A natural measure for the dispersion of a set of vectors is the volume of the parallelepiped they span.

Definition 1 (Parallelepiped)**.** Given n vectors $h_1, \ldots, h_n \in \mathbb{R}^p$, the parallelepiped is their convex hull:

$$\mathcal{P}(\{h_1, \ldots, h_n\}) \triangleq \left\{ h \in \mathbb{R}^p : h = \sum_{i=1}^{n} \lambda_i h_i,\ \lambda \in [0,1]^n \right\}.$$

To encourage robust diversity, we require that not only the entire set of N steered vectors be diverse, but that *any* subset of these vectors also be diverse. This prevents degenerate solutions where, for instance, $N-1$ vectors are clustered together and only one is pushed far away. Let $\mathbb{I} \in 2^{[N]}$ be any index subset of the N sequences. We aim to maximize the sum of the squared volumes of the parallelepiped associated with all possible subsets:

$$\max_{\forall i: \|v_i\|_2^2 \le \alpha_i} \sum_{\mathbb{I} \in 2^{[N]}} \text{Volume}(\mathcal{P}(\{h_i + v_i\}_{i \in \mathbb{I}}))^2, \qquad (1)$$

where α_i is a magnitude for the intervention vector v_i on the i-th path. The squared norm constraints effectively keep the modified hidden state $h_i + v_i$ within a "trust region" of moderate radius $\sqrt{\alpha_i}$ around the original state h_i. If α_i is too big, the vector v_i could erase meaningful information stored in h_i, leading to generation collapse. The next proposition gives an explicit form of the objective function in Problem (1) as a log-determinant function.

Proposition 2 (Objective function equivalence)**.** *Problem* (1) *is equivalent to the following log-determinant optimization problem*

$$\min_{\substack{\forall i: \|v_i\|_2^2 \le \alpha_i \\ V = [v_1, \ldots, v_N]}} \ell(V) \triangleq -\log\det[I + (H+V)^\top (H+V)]. \quad (2)$$

Problem (2) has a convex feasible set, but its objective function ℓ is non-convex, as illustrated in the next example.

Example 3 (Non-convexity of ℓ)**.** *Take*

$$H = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, V_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, V_2 = \begin{bmatrix} -2 & 0 \\ 0 & -2 \end{bmatrix},$$

and $V_3 = \frac{1}{2}(V_1 + V_2)$*. Then* $\ell(V_1) = \ell(V_2) = -\log 4$ *and* $\ell(V_3) = 0$*. We find* $2\ell(V_3) - (\ell(V_1) + \ell(V_2)) = 2\log 4 > 0$*, which implies that* ℓ *is not convex.*

Problem (2) turns out to be a non-convex problem, and it is, in general, NP-hard to find its global optimum (Jin et al. 2021). However, we can show a qualitative result asserting that the optimal steering vectors of Problem (1) will make the norm constraint $\|v_i\|_2^2 \le \alpha_i$ binding, and we obtain another equivalent problem with equality constraints.

Proposition 4 (Constraint equivalence)**.** *Problem* (1) *is further equivalent to the following log-determinant optimization problem with equality constraints*

$$\min \left\{ \ell(V) \ :\ \|v_i\|_2^2 = \alpha_i, V = [v_1, \ldots, v_N] \right\}. \quad (3)$$

The equality constraints $\|v_i\|_2^2 = \alpha_i$ are no longer convex. However, the advantage of the reformulation (3) is that we can cast Problem (3) as an optimization problem over a Riemannian manifold. In the next section, we will describe the procedure for solving Problem (3) with manifold optimization methods.

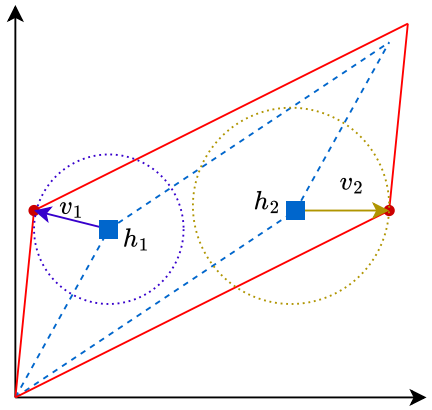

Figure 2: An illustration of the volume maximization intuition behind SPREAD. The hidden vectors h_1 and h_2 (originally blue squares) are pushed toward target positions using the corresponding steering vectors v_1, v_2, found via Riemannian Block Coordinate Descent (Algorithm 1). After intervention, the new parallelepiped (red) has a larger volume than the original parallelepiped (dashed blue).

Hyperparameters. Problem (1) and its equivalent form (3) requires N radii values $\{\alpha_i\}_{i=1}^N$ as input hyperparameters. We propose to set $\alpha_i = C\|h_i\|_2/p$ for all i, where p is the dimension of h_i, and thus the number of hyperparameters is reduced to only one relative parameter $C > 0$.

4 Manifold Optimization for Steering

This section is to devise an efficient algorithm for solving Problem (3) based on Riemannian optimization. Formally, we let $\mathcal{M}_i$ be the sphere in $\mathbb{R}^p$ of radius $\sqrt{\alpha_i}$:

$$\mathcal{M}_i = \{v_i \in \mathbb{R}^p \ : \ \|v_i\|_2^2 = \alpha_i\},$$

and define the product manifold $\mathcal{M} = \mathcal{M}_1 \times \cdots \times \mathcal{M}_N$. Problem (3) can be cast as a Riemannian optimization problem over the product manifold $\mathcal{M}$, which is naturally solved using Riemannian optimization algorithms.

4.1 Preliminaries About the Manifold $\mathcal{M}$

Riemannian optimization exploits the geometric structure of the constraint set to perform updates along curved spaces (manifolds) rather than flat Euclidean space. Key tools include (i) the Riemannian gradient, which generalizes classical gradient methods to manifold settings, and (ii) the exponential map, which generalizes the concept of moving in a straight line along a gradient in Euclidean space to moving along a shortest path on a manifold.

At a point on a manifold, a tangent space is a vector space that "touches" the manifold at that point and contains all possible directions in which one can move along the manifold from that point. For an individual sphere, the tangent space at $v_i \in \mathcal{M}_i$ is

$$T_{v_i}\mathcal{M}_i = \{z \in \mathbb{R}^p : z^\top v_i = 0\}.$$

For the product manifold, the tangent space at $V = [v_1, \ldots, v_N]$ is

$$T_V\mathcal{M} = \{Z = (z_1, \ldots, z_N) \ : \ z_i \in T_{v_i}\mathcal{M}_i \ \forall i\}.$$

The Riemannian gradient of ℓ defined on the manifold $\mathcal{M}$ is the steepest ascent direction that lies within the tangent space of the manifold at a given point. It is computed by first taking the Euclidean gradient of ℓ in the ambient space $\mathbb{R}^{p\times N}$, and then projecting this gradient onto the tangent space of the manifold. Let $G \in \mathbb{R}^{p\times N}$ be the computed Euclidean gradient of ℓ at V, the projection of G onto $T_V\mathcal{M}$ is decomposable into N projections of the columns g_i onto the tangent space of the individual sphere $\mathcal{M}_i$

$$\mathrm{Proj}_{T_{v_i}\mathcal{M}_i}(g_i) = (I - \frac{1}{\alpha_i} v_i v_i^\top) g_i \qquad \forall i.$$

The next lemma formalizes the computation of the Riemannian gradient for ℓ.

Lemma 5. *For any $V = [v_1, \ldots, v_N] \in \mathcal{M}$, the Riemannian gradient of ℓ in the i-th block is given by*

$$\mathrm{grad}_i \ell(V) = g_i - \frac{1}{\alpha_i} g_i^\top v_i v_i \in \mathbb{R}^p, \tag{4}$$

where $g_i = -2((H+V)M^{-1})_i \in \mathbb{R}^p$ and $M = I + (H+V)^\top(H+V)$. Furthermore, the Riemannian gradient of ℓ at point V corresponding to the product manifold $\mathcal{M}$ is given by $\mathrm{grad}\ell(V) = [\mathrm{grad}_1\ell(V), \ldots, \mathrm{grad}_N\ell(V)]$.

The *negative* Riemannian gradient at V gives the direction of steepest descent within the tangent space. The exponential map then moves the incumbent solution along this direction, not in a straight line, but along a curved path that fits the spherical surface. This allows us to take meaningful steps while staying on the manifold throughout the iterations. Given the descent direction $d_i = -\mathrm{grad}_i\ell(V)$ and a step size η_i for the i-th block, the exponential map gives (Boumal 2023, Section 10.2)

$$\mathrm{Exp}_{v_i}(\eta_i d_i) = \tag{5}$$
$$\cos(\frac{\eta_i\|d_i\|_2}{\sqrt{\alpha_i}})v_i + \sin(\frac{\eta_i\|d_i\|_2}{\sqrt{\alpha_i}})\frac{d_i}{\|d_i\|_2}\sqrt{\alpha_i} \in \mathcal{M}_i.$$

Given $\mathrm{Exp}_{v_i}(\eta_i d_i)$ in (5), the exponential map on the product manifold $\mathcal{M}$ is given by

$$\mathrm{Exp}_V([\eta_1 d_1, \ldots, \eta_N d_N]) = \tag{6}$$
$$[\mathrm{Exp}_{v_1}(\eta_1 d_1), \ldots, \mathrm{Exp}_{v_N}(\eta_N d_N)] \in \mathcal{M}.$$

4.2 Block-Coordinate Riemannian Descent

Since our manifold $\mathcal{M}$ decomposes as a product of scaled spheres, its natural block structure allows one global update to be replaced by N cheaper spherical updates. Consequently, Riemannian Block-coordinate Descent (Gutman and Ho-Nguyen 2023) provides an intuitive and efficient method for Problem (3).

In the outer iteration k, Algorithm 1 updates the blocks v_i sequentially for $i = 1, \ldots, N$. Fixing all other blocks at their most recent values, it first computes the Euclidean gradient g_i. In Line 5, g_i is then projected onto the tangent space to

Algorithm 1: Riemannian Block Coordinate Descent on Product of Spheres with Exponential Maps

Require: Hidden activation vectors $\{h_i\}_{i=1}^N$, magnitudes $\{\alpha_i\}_{i=1}^N$, learning rate $\eta_i > 0$, max iterations K

1: Initialize $v_i^{(0)}$ using (7)
2: **for** $k = 1$ to K **do**
3: **for** $i = 1$ to N **do**
4: Compute Euclidean gradient in the variable v_i by $g_i \leftarrow \nabla_i \ell(v_1^{(k)}, \ldots, v_i^{(k-1)}, \ldots, v_N^{(k-1)})$
5: Compute the descent direction $d_i \leftarrow -g_i + \frac{1}{\alpha_i}((v_i^{(k-1)})^\top g_i)\, v_i^{(k-1)}$
6: Compute $v_i^{(k)} \leftarrow \cos(\frac{\eta_i \|d_i\|_2}{\sqrt{\alpha_i}}) v_i^{(k-1)} + \sin(\frac{\eta_i \|d_i\|_2}{\sqrt{\alpha_i}}) \frac{d_i}{\|d_i\|_2} \sqrt{\alpha_i}$
7: **end for**
8: **end for**
9: **return** $V^{(K)} = [v_1^{(K)}, \ldots, v_N^{(K)}]$

yield the Riemannian descent direction d_i, and then Line 6 moves $v_i^{(k-1)}$ along the geodesic on $\mathcal{M}_i$ via the exponential map (5), thus preserving its feasibility on the sphere.

Initialization. To get a feasible initial point, for each i, we can initialize $v_i^{(0)}$ by

$$v_i^{(0)} = \sqrt{\alpha_i} \frac{h_i + \varepsilon_i - \bar{h}}{\|h_i + \varepsilon_i - \bar{h}\|_2} \qquad \forall i. \tag{7}$$

where $\varepsilon_i \sim \mathcal{N}(0, \sigma^2 I_d)$ are independent Gaussian noise with small variances and $\bar{h} = \frac{1}{N}\sum_{i=1}^N h_i$ is the centroid of the hidden vectors $\{h_i\}_{i=1}^N$.

4.3 Convergence Analysis

We now study the convergence guarantee of Algorithm 1. The next theorem asserts the convergence of Algorithm 1 for solving Problem (3) with well-chosen step sizes. To this end, let $\bar{\alpha} \triangleq \sum_{i=1}^N \alpha_i$ be the total radii, and for each sequence i, define the quantity

$$L_i \triangleq 2 + 4(\|H\|_F + \sqrt{\bar{\alpha}})^2 + \frac{2}{\sqrt{\alpha_i}}(\|H\|_F + \sqrt{\bar{\alpha}}) \quad \forall i. \tag{8}$$

Theorem 6 (Convergence of Algorithm 1)**.** *Let $V^{(k)} = [v_1^{(k)}, \ldots, v_N^{(k)}]$ be the sequence generated from Algorithm 1 with learning rate $\eta_i = 1/L_i$ for $i = 1, \ldots, N$. Define*

$$C \triangleq \frac{L_{\min}^2}{4L_{\max}(L_{\min}^2 + L^2 N(N-1))},$$

where $L_{\min} = \min_i L_i$ and $L_{\max} = \max_i L_i$. Then, the following hold:

- *We have $\lim_{k\to\infty} \|\mathrm{grad}\, \ell(V^{(k)})\|_F = 0$ and*

$$\min_{s \le k} \|\mathrm{grad}\, \ell(V^{(s)})\|_F \le \sqrt{\frac{\ell(V^{(0)}) - \ell^\star}{Ck}},$$

  where $\ell^\star$ is the optimal value of Problem (3)*.*
- *Any limit point $V^\infty \in \mathcal{M}$ of $\{V^{(k)}\}_{k\ge 1}$ is a stationary point, i.e.,* $\mathrm{grad}\, \ell(V^\infty) = 0$.

Theorem 6 establishes both asymptotic and non-asymptotic convergence results for Algorithm 1. It guarantees that any limit point of the generated sequence is a stationary point of the objective function. The sublinear rate $O(1/\sqrt{k})$ for the minimum gradient norm bounds the number of iterations needed to achieve a desired accuracy level.

To analyze the convergence of the algorithm, we study the smoothness of the objective function. Firstly, we recite the L-smoothness definition in the Riemannian sense (Gutman and Ho-Nguyen 2023, equation (3)).

Definition 7 (L-smoothness)**.** The function $\ell : \mathcal{M} \to \mathbb{R}$ is L-smooth if for all $V \in \mathcal{M}$, $U \in T_V\mathcal{M}$ and $Z = \mathrm{Exp}_V(U)$ it holds that

$$\|\Gamma_{V\to Z}^U\, \mathrm{grad}\ell(V) - \mathrm{grad}\ell(Z)\|_F \le L\|U\|_F,$$

where $\Gamma_{V\to Z}^U : T_V\mathcal{M} \to T_Z\mathcal{M}$ is the parallel transport operator along the curve $\gamma(t) = \mathrm{Exp}_V(tU)$.

The following lemma shows that the objective function $\ell(V)$ of Problem (3) satisfies Definition 7 with an explicit Lipschitz constant L.

Proposition 8 (Smoothness)**.** *Let $\alpha_{\min} \triangleq \min_i \alpha_i > 0$. The objective function $\ell(V)$ is L-smooth, with constant*

$$L = 2 + 4(\|H\|_F + \sqrt{\bar{\alpha}})^2 + \frac{2}{\sqrt{\alpha_{\min}}}(\|H\|_F + \sqrt{\bar{\alpha}}). \tag{9}$$

We further show that the function ℓ also satisfies the block smoothness, which is defined by restricting Definition 7 to each individual sphere $\mathcal{M}_i$.

Proposition 9 (Block smoothness)**.** *Let $V = [v_1, \ldots, v_N] \in \mathcal{M}$, $U = [u_1, \ldots, u_N] \in T_V\mathcal{M}$ and $Z = \mathrm{Exp}_V(U)$. For any v_i and $u_i \in T_{v_i}\mathcal{M}_i$, it holds that*

$$\|\Gamma_{v_i\to z_i}^{u_i}\, \mathrm{grad}_i \ell(V) - \mathrm{grad}_i \ell(Z)\|_F \le L_i \|u_i\|_2 \tag{10}$$

with L_i defined in (8)*. Here $\Gamma_{v_i\to z_i}^{u_i} : T_{v_i}\mathcal{M}_i \to T_{z_i}\mathcal{M}_i$ is the parallel transport operator along the curve $\gamma(t) = \mathrm{Exp}_{v_i}(tu_i)$.*

Proposition 9 is crucial both for establishing the convergence of Algorithm 1 and for guiding the selection of its step-sizes. Equation (10) together with Boumal (2023, Corollary 10.54) implies that for each i it holds

$$\ell(v_1, \ldots, \mathrm{Exp}_{v_i}(u_i), \ldots, v_i) \le \ell(V) + \langle \mathrm{grad}_i\, \ell(V), u_i\rangle + \frac{L_i}{2}\|u_i\|_2^2. \tag{11}$$

Thus, in particular, if one sets $\eta_i = 1/L_i$ then each coordinate-descent update yields a provable decrease in the objective (Gutman and Ho-Nguyen 2023, Lemma 1). This obviates the need for extensive tuning of learning rates η_i.

Model	Temp.	AIME24			MATH500			OlympiadBench		
		SPREAD		Samp.	SPREAD		Samp.	SPREAD		Samp.
		$C=1$	$C=10$		$C=1$	$C=10$		$C=1$	$C=10$	
Qwen2.5-1.5B	1.0	3.3	**6.7**	0.0	43.2	**43.4**	42.8	**21.5**	19.7	19.0
	0.8	**6.7**	**6.7**	3.3	51.8	**54.2**	51.4	25.3	**27.0**	25.9
	0.6	**10.0**	3.3	3.3	**55.0**	54.0	53.4	28.4	28.3	**29.3**
	0.4	6.7	6.7	6.7	56.2	**57.6**	55.8	**30.8**	30.1	30.7
	0.2	**10.0**	6.7	6.7	**55.6**	53.8	52.2	**28.0**	27.1	26.4
Qwen2.5-Math -1.5B-Instruct	1.0	**26.7**	16.7	20.0	83.8	83.6	**84.6**	47.6	**49.5**	48.3
	0.8	**26.7**	23.3	16.7	**85.2**	85.0	83.6	49.9	49.2	**51.0**
	0.6	**26.7**	**26.7**	20.0	**85.4**	84.4	84.6	50.4	**51.0**	50.8
	0.4	23.3	23.3	23.3	**84.6**	84.2	84.0	**51.7**	48.8	51.0
	0.2	20.0	**26.7**	16.7	82.4	84.4	84.4	**52.3**	51.1	49.9

Table 1: Pass@N $\uparrow$ (in %) performance comparison across model variants and sampling temperature on three mathematical reasoning benchmarks. Bold indicates the best method in each dataset.

5 Numerical Experiment

We evaluate SPREAD across three established mathematical reasoning benchmarks: AIME24 (30 problems), MATH500 (500 problems), and OlympiadBench (675 problems). We evaluate using the following metrics:

- *Pass@N*: The proportion of problems for which at least one of the N generated solutions produces the correct final answer.
- *Solution Diversity*: The float score indicating the overall diversity among N solutions.
- *Unique Solution Count*: The number of distinct solution approaches among N solutions.

For the latter two metrics, we employ a language-model-as-a-judge paradigm using GPT-4.1-mini. We prompt the model with the question and N generated solutions, requesting a diversity score (float in $[0, 1]$) and a count of unique approaches (integer). The specific prompts and parameters for this evaluation are detailed in the Appendix. Our experiments utilize two model variants: Qwen2.5-1.5B (base) and Qwen2.5-Math-1.5B-Instruct (math-specialized). Steering vectors are applied at layer 28, which corresponds to the final layer in both architectures. The hidden activations have dimension $p = 1536$. The magnitude α_i is chosen as $\alpha_i = C\|h_i\|_2/p$, where $C \in \{1, 10\}$ is a scaling constant. Steering vectors are computed using Algorithm 1 at token positions $\tau \in \{100, 600, 1100, 1600\}$ with $K = 20$, and the learning rates are calibrated using the input activations following Theorem 6. All experiments are conducted on NVIDIA RTX A5000 24GB hardware using temperature sampling for solution generation with temperature $\in \{0.2, 0.4, 0.6, 0.8, 1.0\}$ and a maximum generation length of 2048 tokens. To ensure reproducibility, we fix the random seed to 42 for all experimental runs. Additional numerical results are presented in the appendix.

5.1 Experiment Results

Table 1 summarizes the performance of SPREAD compared to temperature sampling across various benchmarks and temperature settings. SPREAD with either $C = 1$ or $C = 10$ consistently performs at least as well as vanilla temperature sampling, and often achieves improvements of several percentage points. Overall, SPREAD decoding (especially with $C = 1$) offers the strongest results under most conditions. Table 2 demonstrates that SPREAD consistently outperforms temperature sampling in generating unique solutions across multiple benchmarks. These results highlight SPREAD's robustness and effectiveness in promoting correct and diverse reasoning trajectories across various model variants and datasets.

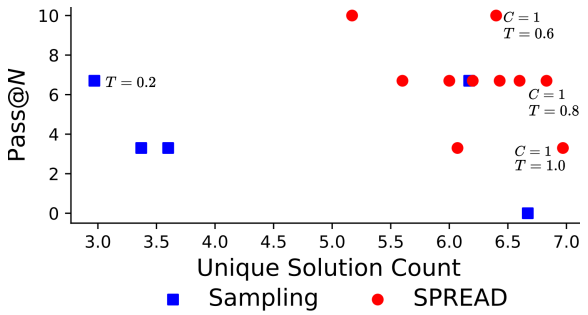

Figure 3: Comparison of SPREAD and sampling methods on AIME24 dataset using Qwen2.5-1.5B model, showing Pass@N and Unique Solution Count.

To strengthen the comparison, we aggregate the metrics and analyze the accuracy-diversity frontiers in Figure 3 and 4 for the AIME24 dataset. We can observe that the performance of SPREAD (red circles) dominates that of vanilla sampling methods with different temperatures (blue squares). The Pareto plots for the MATH and OlympiadBench datasets are presented in the appendix.

5.2 Computational Efficiency

We analyze the execution time of Algorithm 1 for different hidden vector dimensions p and different numbers of sequences N. We generate synthetic activations with p varying from 1536 up to $2^{14} = 16384$, which corresponds to the hidden size of large-scale models such as LLaMA-3.1-405B. Figure 5 reports the average solution time of Algorithm 1 with $K = 20$ over 30 independent runs. The ex-

Metric	Model	Temp.	AIME24			MATH500			OlympiadBench		
			SPREAD		Samp.	SPREAD		Samp.	SPREAD		Samp.
			$C=1$	$C=10$		$C=1$	$C=10$		$C=1$	$C=10$	
Unique Solution Count ↑	Qwen2.5 -1.5B	1.0	**6.97**	6.60	6.67	3.14	**3.21**	3.14	6.12	6.14	6.14
		0.8	**6.83**	6.43	3.6	3.03	**3.05**	2.97	5.99	5.99	**6.04**
		0.6	**6.40**	6.07	3.37	**2.91**	2.86	2.90	5.57	5.65	**5.58**
		0.4	6.00	**6.20**	6.17	2.73	2.74	**2.77**	**5.30**	5.30	5.26
		0.2	5.17	**5.60**	2.97	2.47	2.56	**2.59**	4.65	4.69	4.69
	Qwen2.5 -Math -1.5B -Instruct	1.0	6.63	**7.03**	3.67	1.92	**1.94**	1.93	**4.59**	4.53	4.57
		0.8	6.63	**6.73**	3.47	1.87	**1.90**	1.89	4.28	4.36	**4.39**
		0.6	6.27	**6.47**	3.5	1.81	**1.84**	1.82	4.10	**4.16**	4.11
		0.4	**6.23**	5.87	6.17	1.77	1.73	**1.78**	**3.99**	3.92	3.89
		0.2	**5.87**	**5.87**	5.53	1.70	1.74	1.74	3.67	3.65	**3.68**
Diversity Score ↑	Qwen2.5 -1.5B	1.0	**0.37**	0.33	0.40	**0.40**	0.39	0.39	0.38	**0.40**	0.39
		0.8	**0.51**	0.45	0.37	0.42	**0.44**	0.41	0.46	0.46	0.46
		0.6	**0.52**	0.46	0.39	**0.41**	0.40	0.40	0.46	0.46	0.46
		0.4	0.41	**0.47**	0.42	0.36	0.36	0.36	0.42	**0.43**	0.42
		0.2	0.31	**0.34**	0.33	**0.29**	**0.29**	0.28	0.33	**0.35**	**0.35**
	Qwen2.5 -Math -1.5B- Instruct	1.0	0.64	**0.65**	0.63	0.25	0.25	0.25	**0.39**	**0.39**	0.38
		0.8	0.60	**0.62**	0.57	0.23	0.23	0.23	0.36	**0.38**	0.37
		0.6	**0.57**	0.55	0.55	0.21	**0.22**	0.20	0.34	0.34	0.34
		0.4	0.51	0.51	0.51	**0.19**	0.18	0.18	**0.31**	0.30	0.31
		0.2	0.46	**0.52**	0.41	0.15	**0.16**	0.15	0.26	0.26	0.26

Table 2: Unique Solution Count ↑ and Diversity Score ↑ comparison across model variants and sampling temperature on three mathematical reasoning benchmarks. Bold indicates the best method in each dataset.

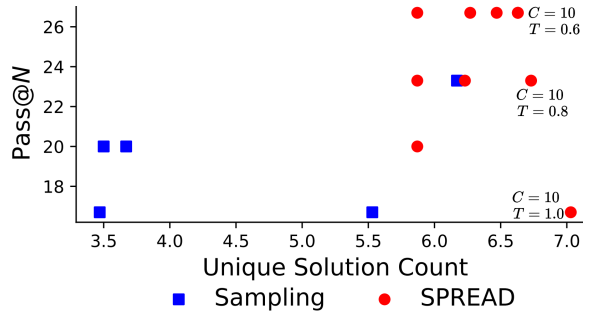

Figure 4: Comparison of SPREAD and sampling methods on AIME24 dataset using Qwen2.5-Math-1.5B-Instruct model, showing Pass@N and Unique Solution Count.

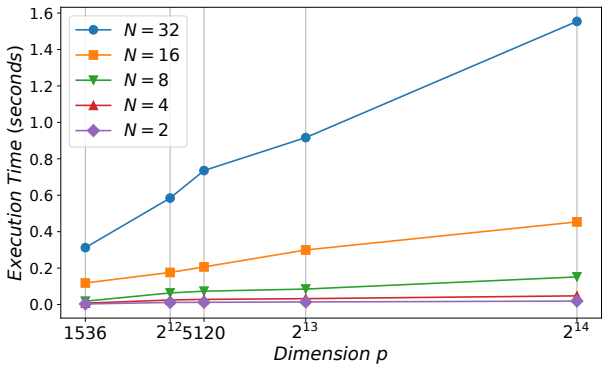

Figure 5: Average running time of Algorithm 1 with varying problem dimension p and number of steering vectors N. Lower execution time is better.

ecution time for $N = 32$ remains under 1.8 seconds even for the largest dimension 16384, showing that our SPREAD method remains highly efficient. These results demonstrate the scalability and practical efficiency of our algorithm for inference-time language model intervention.

6 Conclusions

We introduced SPREAD, a novel unsupervised activation steering framework at test time designed to improve the reasoning diversity of language models. By addressing the well-known challenge of low output variance in best-of-N sampling, SPREAD intervenes meaningful diversity into generation trajectories. Our key innovation lies in reformulating the activation steering problem as a Riemannian optimization over the product of spheres, maximizing the log-determinant volume spanned by intervened representations. This principled formulation enables us to efficiently compute diverse steering directions using Riemannian block-coordinate descent. Empirical evaluations on mathematical reasoning benchmarks such as AIME24, MATH, and OlympiadBench validate the effectiveness of our approach, as SPREAD significantly improves both the diversity and accuracy of generated solutions, outperforming temperature-sampling techniques. These results demonstrate the potential of geometric test-time intervention methods to enhance reasoning in language models.

References

Ackley, D. H.; Hinton, G. E.; and Sejnowski, T. J. 1985. A learning algorithm for Boltzmann machines. *Cognitive science*, 9(1): 147–169.

Bestgen, Y. 2024. Measuring lexical diversity in texts: The twofold length problem. *Language Learning*, 74(3): 638–671.

Boumal, N. 2023. *An Introduction to Optimization on Smooth Manifolds*. Cambridge University Press.

Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; de Oliveira Pinto, H. P.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; Ray, A.; Puri, R.; Krueger, G.; Petrov, M.; Khlaaf, H.; Sastry, G.; Mishkin, P.; Chan, B.; Gray, S.; Ryder, N.; Pavlov, M.; Power, A.; Kaiser, L.; Bavarian, M.; Winter, C.; Tillet, P.; Such, F. P.; Cummings, D.; Plappert, M.; Chantzis, F.; Barnes, E.; Herbert-Voss, A.; Guss, W. H.; Nichol, A.; Paino, A.; Tezak, N.; Tang, J.; Babuschkin, I.; Balaji, S.; Jain, S.; Saunders, W.; Hesse, C.; Carr, A. N.; Leike, J.; Achiam, J.; Misra, V.; Morikawa, E.; Radford, A.; Knight, M.; Brundage, M.; Murati, M.; Mayer, K.; Welinder, P.; McGrew, B.; Amodei, D.; McCandlish, S.; Sutskever, I.; and Zaremba, W. 2021. Evaluating large language models trained on code. arXiv:2107.03374.

Chung, H.-L.; Hsiao, T.-Y.; Huang, H.-Y.; Cho, C.; Lin, J.-R.; Ziwei, Z.; and Chen, Y.-N. 2025. Revisiting test-time scaling: A Survey and a diversity-aware method for efficient reasoning. arXiv:2506.04611.

Dang, X.; Baek, C.; Wen, K.; Kolter, J. Z.; and Raghunathan, A. 2025. Weight ensembling improves reasoning in language models. In *Second Conference on Language Modeling*.

Fan, A.; Lewis, M.; and Dauphin, Y. 2018. Hierarchical neural story generation. In Gurevych, I.; and Miyao, Y., eds., *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 889–898. Melbourne, Australia: Association for Computational Linguistics.

Gutman, D. H.; and Ho-Nguyen, N. 2023. Coordinate descent without coordinates: Tangent subspace descent on Riemannian manifolds. *Mathematics of Operations Research*, 48(1): 127–159.

Han, S.; Kim, B.; and Chang, B. 2022. Measuring and improving semantic diversity of dialogue generation. In Goldberg, Y.; Kozareva, Z.; and Zhang, Y., eds., *Findings of the Association for Computational Linguistics: EMNLP 2022*, 934–950. Association for Computational Linguistics.

Holtzman, A.; Buys, J.; Du, L.; Forbes, M.; and Choi, Y. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Jin, C.; Netrapalli, P.; Ge, R.; Kakade, S. M.; and Jordan, M. I. 2021. On nonconvex optimization for machine learning: Gradients, stochasticity, and saddle points. *Journal of the ACM (JACM)*, 68(2): 1–29.

Laakom, F.; Raitoharju, J.; Iosifidis, A.; and Gabbouj, M. 2023. WLD-Reg: A data-dependent within-layer diversity regularizer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(7): 8421–8429.

Leviathan, Y.; Kalman, M.; and Matias, Y. 2023. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*.

Lewkowycz, A.; Andreassen, A.; Dohan, D.; Dyer, E.; Michalewski, H.; Ramasesh, V.; Slone, A.; Anil, C.; Schlag, I.; Gutman-Solo, T.; Wu, Y.; Neyshabur, B.; Gur-Ari, G.; and Misra, V. 2022. Solving quantitative reasoning problems with language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*.

Li, K.; Patel, O.; Viégas, F.; Pfister, H.; and Wattenberg, M. 2023a. Inference-time intervention: Eliciting truthful answers from a language model. In Oh, A.; Naumann, T.; Globerson, A.; Saenko, K.; Hardt, M.; and Levine, S., eds., *Advances in Neural Information Processing Systems*, volume 36, 41451–41530.

Li, Y.; Lin, Z.; Zhang, S.; Fu, Q.; Chen, B.; Lou, J.-G.; and Chen, W. 2023b. Making language models better reasoners with step-aware verifier. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 5315–5333.

Lightman, H.; Kosaraju, V.; Burda, Y.; Edwards, H.; Baker, B.; Lee, T.; Leike, J.; Schulman, J.; Sutskever, I.; and Cobbe, K. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Lindsey, J.; Gurnee, W.; Ameisen, E.; Chen, B.; Pearce, A.; Turner, N. L.; Citro, C.; Abrahams, D.; Carter, S.; Hosmer, B.; Marcus, J.; Sklar, M.; Templeton, A.; Bricken, T.; McDougall, C.; Cunningham, H.; Henighan, T.; Jermyn, A.; Jones, A.; Persic, A.; Qi, Z.; Thompson, T. B.; Zimmerman, S.; Rivoire, K.; Conerly, T.; Olah, C.; and Batson, J. 2025. On the Biology of a Large Language Model. https://transformer-circuits.pub/2025/attribution-graphs/biology.html. Accessed: 2025-08-02.

Meister, C.; Pimentel, T.; Wiher, G.; and Cotterell, R. 2023. Locally typical sampling. *Transactions of the Association for Computational Linguistics*, 11: 102–121.

Nagarajan, V.; Wu, C. H.; Ding, C.; and Raghunathan, A. 2025. Roll the dice & look before you leap: Going beyond the creative limits of next-token prediction. In *Forty-second International Conference on Machine Learning*.

Naik, R.; Chandrasekaran, V.; Yüksekgönül, M.; Palangi, H.; and Nushi, B. 2023. Diversity of thought improves reasoning abilities of large language models. *CoRR*, abs/2310.07088.

Ni, A.; Iyer, S.; Radev, D.; Stoyanov, V.; Yih, W.-t.; Wang, S.; and Lin, X. V. 2023. Lever: Learning to verify language-to-code generation with execution. In *International Conference on Machine Learning*, 26106–26128. PMLR.

Stolfo, A.; Balachandran, V.; Yousefi, S.; Horvitz, E.; and Nushi, B. 2025. Improving instruction-following in language models through activation steering. In *The Thirteenth International Conference on Learning Representations*.

Su, Y.; and Collier, N. 2023. Contrastive search is what you need for neural text Generation. *Transactions on Machine Learning Research*.

Turner, A. M.; Thiergart, L.; Leech, G.; Udell, D.; Vazquez, J. J.; Mini, U.; and MacDiarmid, M. 2023. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*.

Vijayakumar, A. K.; Cogswell, M.; Selvaraju, R. R.; Sun, Q.; Lee, S.; Crandall, D. J.; and Batra, D. 2016. Diverse beam search: decoding diverse solutions from neural sequence models. *CoRR*.

Wang, T.; Jiao, X.; Zhu, Y.; Chen, Z.; He, Y.; Chu, X.; Gao, J.; Wang, Y.; and Ma, L. 2025. Adaptive activation steering: A tuning-free LLM truthfulness improvement method for diverse hallucinations categories. In *Proceedings of the ACM on Web Conference 2025*, 2562–2578. Association for Computing Machinery.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; brian ichter; Xia, F.; Chi, E. H.; Le, Q. V.; and Zhou, D. 2022. Chain of thought prompting elicits reasoning in large language models. In Oh, A. H.; Agarwal, A.; Belgrave, D.; and Cho, K., eds., *Advances in Neural Information Processing Systems*.

Yun, L.; An, C.; Wang, Z.; Peng, L.; and Shang, J. 2025. The price of format: Diversity collapse in LLMs. arXiv:2505.18949.

Zhang, H.; Wang, X.; Li, C.; Ao, X.; and He, Q. 2025. Controlling large language models through concept activation vectors. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(24): 25851–25859.

Zhang, J.; Wang, J.; Li, H.; Shou, L.; Chen, K.; Chen, G.; and Mehrotra, S. 2024. Draft& Verify: Lossless large language model acceleration via self-speculative decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 11263–11282. Association for Computational Linguistics.