Understanding and Improving Noisy Embedding Techniques in Instruction Finetuning

Abhay Yadav
Johns Hopkins University
Baltimore, MD, USA
ayadav13@jh.edu

Abstract

Recent advancements in instructional fine-tuning have injected noise into embeddings, with `NEFTune` (Jain et al., 2024) setting benchmarks using uniform noise. Despite `NEFTune`'s **empirical findings** that uniform noise outperforms Gaussian noise, the reasons for this remain unclear. This paper aims to clarify this by offering a thorough analysis, both theoretical and empirical, indicating comparable performance among these noise types. Additionally, we introduce a new fine-tuning method for language models, utilizing symmetric noise in embeddings. This method aims to enhance the model's function by more stringently regulating its local curvature, demonstrating superior performance over the current method, `NEFTune`. When fine-tuning the LLaMA-2-7B model using Alpaca, standard techniques yield a 29.79% score on AlpacaEval. However, our approach, `SymNoise`, increases this score significantly to 69.04%, using symmetric noisy embeddings. This is a 6.7% improvement over the state-of-the-art method, `NEFTune` (64.69%). Furthermore, when tested on various models and stronger baseline instruction datasets, such as Evol-Instruct, ShareGPT, OpenPlatypus, `SymNoise` consistently outperforms `NEFTune`. The current literature, including `NEFTune`, has underscored the importance of more in-depth research into the application of noise-based strategies in the fine-tuning of language models. Our approach, `SymNoise`, is another significant step towards this direction, showing notable improvement over the existing state-of-the-art method.

1 Introduction

For Large Language Models (Vaswani et al., 2017; Devlin et al., 2019; Radford et al., 2019; Raffel et al., 2020; Zhang et al., 2022; Touvron et al., 2023b; Zhao et al., 2023) to be effective, their proficiency in executing specific instructions is crucial (Wang et al., 2022; Ouyang et al., 2022; Brown et al., 2020; Chung et al., 2022). These models typically begin with training on a vast array of unfiltered web data, after which they undergo a more focused fine-tuning stage using a smaller, selectively chosen collection of instructional data. The fine-tuning stage, centered on instructions, is fundamental in unlocking and controlling the full capabilities of LLMs. The practical value of these models is predominantly dependent on how efficiently we can leverage these concise instructional data sets for optimal performance.

In recent years, noise injection (Nukrai et al., 2022; Zang et al., 2021; Akbiyik, 2020) has been a focal point in computer vision research, yielding methods that enhance model robustness and accuracy. This strategy has recently been adapted for fine-tuning Large Language Models (LLMs), exemplified by the `NEFTune` method (Jain et al., 2024), which applies uniform random noise to improve model performance on diverse datasets. Despite `NEFTune`'s efficacy surpassing traditional fine-tuning techniques, the reasons behind its success, particularly against the commonly used Gaussian noise, are not entirely understood. Our work demystifies this by presenting a detailed theoretical and empirical analysis that reveals comparable results between noise types when appropriately scaled. Moreover, we introduce a novel noise injection approach which not only facilitates a more intuitive

Table 1: AlpacaEval Win Rate against Text-Davinci-003 when applied with LLaMA-2, trained across diverse datasets. SymNoise shows an overall improvement throughout all datasets, outperforming NEFTune on all datasets. The noise scaling factor for the Gaussian distribution is divided by $\sqrt{3}$, resulting in similar performance for both methods.

	Alpaca	Evol-Instruct	ShareGPT	OpenPlatypus	Average
LLaMA-2 7B	29.79	70.34	68.74	62.00	57.71
+NEFT	64.69	79.60	76.28	70.61	72.80
+Gaussian	64.98	78.88	75.94	70.20	72.49
+SymNoise	**69.04**	**81.38**	**78.67**	**72.23**	**75.33**

understanding but also achieves superior empirical results, outperforming NEFTune and other established fine-tuning methods by a considerable margin.

In particular, our objective is to regularize the curvature of the function learned during training. Curvature regularization has been used in domains such as computer vision (Moosavi-Dezfooli et al., 2019; Lee & Park, 2023), graph embedding (Pei et al., 2020), and deep neural networks (Huh, 2020). Specifically, we aim to ensure that the function's response changes gradually when the input is modified slightly by noise. In more technical terms, our goal is to have the gradient approach zero in the immediate vicinity of an input altered by a minimal amount. This represents a more stringent condition than merely requiring small values for the Hessian or gradient. However, considering computational efficiency, we opt to avoid the direct computation of gradients or Hessians. Instead, we employ this stringent condition, which, as our experiments on real-world benchmark datasets demonstrate, is effective in practical scenarios.

In this paper, we unveil Symmetric Noise Fine Tuning (SymNoise), a new technique that leverages symmetric Bernoulli distribution-based noise applied to the embedding vectors of training data during the finetuning stage. Each noise component is generated with an equal probability of $\frac{1}{2}$ for the values -1 and 1. This method significantly enhances instruction finetuning outcomes, often with remarkable gains, while avoiding additional computational or data resources. While maintaining simplicity, SymNoise has a profound impact on downstream conversational output quality. We show that when a large langudage model like LLaMA-2-7B (Touvron et al., 2023c) is finetuned using SymNoise, its performance on AlpacaEval (Dubois et al., 2023) rises from 29.79% to 69.04% – a substantial increase of about 39.25 percentage points.

Importantly, when compared to the existing NEFTune method (which uses random uniform noise), SymNoise demonstrates a superior performance edge, outperforming NEFTune by approximately 6.7%. Thus, SymNoise not only represents a valuable advancement over traditional finetuning methods but also establishes a new benchmark in efficiency and effectiveness for LLM finetuning.

Contributions. In our comprehensive study, we conduct a detailed theoretical and empirical examination, demonstrating that Gaussian and uniform random noise exhibit functional equivalence when adjusted with an appropriate scaling factor, leading to similar performance on real-world datasets. This insight holds significant importance, especially considering that the creators of the NEFTune method, a leading approach employing uniform noise, have openly recognized gaps in their understanding of the method's superior performance, notably in comparison to the extensively studied Gaussian noise. By establishing a connection with Gaussian noise, our study helps demystify the NEFTune method. Moreover, we introduce an innovative noise injection method that exceeds the capabilities of NEFTune and existing alternatives. Our contributions thus propel the momentum for continued exploration in this field.

2 Background and Related Work

In the evolving landscape of instruction finetuning for Large Language Models (LLMs), initial efforts like FLAN and T0 marked the beginning of cross-task generalization (Sanh et al.,

Table 2: `AlpacaEval` Win Rate with and without `NEFTune`, `SymNoise` on LLaMA-2, LLaMA-1, and OPT on various datasets. `SymNoise` shows improved performance across these datasets and models.

Method/Dataset	Alpaca	Evol-Instruct	OpenPlatypus	ShareGPT
OPT-6.7B	41.4	52.2	45.7	53.4
+NEFTune	48.7	55.5	45.8	54.3
+SymNoise	50.8	57.6	46.9	55.6
LLaMA-1-7B	48.5	62.3	51.2	62.9
+NEFTune	61.7	67.5	56.9	63.6
+SymNoise	64.0	69.8	58.5	65.4
LLaMA-2-7B	48.3	62.5	57.2	63.5
+NEFTune	62.5	67.6	61.7	64.2
+SymNoise	64.9	69.6	62.1	66.1

2021; Wei et al., 2021). These models, involving encoder-decoder language architectures, underwent finetuning across a diverse spectrum of thousands NLP tasks. This progression, detailed in studies by Chung et al. (2022) and Xu et al. (2022) demonstrated the adaptability of LLMs to a variety of standard NLP tasks.

Following this trajectory, OpenAI's InstructGPT (Ouyang et al., 2022) emerged as a pioneering model adept at handling open-ended questions with remarkable efficiency. This model, an iteration of GPT-3 (Brown et al., 2020), incorporated reinforcement learning from human feedback (RLHF), leading to the development of advanced models like ChatGPT (OpenAI, 2022). ChatGPT, in particular, gained widespread attention for generating more coherent and extended texts compared to InstructGPT.

Building on these developments, Wang et al. (2022) introduced the Self-Instruct approach, utilizing InstructGPT to generate instructional pairs for further finetuning of foundational models like LLaMA into specialized variants such as Alpaca (Taori et al., 2023). Concurrently, the trend towards distilled models, as discussed by Taori et al. (2023) and Xu et al. (2023), led to the creation of diverse datasets. These datasets, including works by Xu et al. (2023) and Lee et al. (2023), focused on refining specific model capabilities like STEM question answering and logical reasoning. Another notable advancement was AlpaGasus by Chen et al. (2023), which employed a quality-filtering mechanism based on GPT-4 evaluations to enhance model performance. In a different methodology, ShareGPT, as described by Chiang et al. (2023), was developed through the crowd sourcing of real user interactions sourced from ChatGPT.

In the context of incorporating noise into model training, the pioneering work by Zhu et al. (2019) with the FreeLB method demonstrated the effectiveness of adversarial perturbations in boosting MLM model performance. This method involved introducing calculated Gaussian perturbations into the embeddings and optimizing them to maximally impact model performance. Similar strategies were later applied in various domains, such as image captioning (Nukrai et al., 2022), point cloud processing (Zang et al., 2021), graphs (Kong et al., 2022), and privacy mechanisms (Dwork et al., 2014). Curvature regularization has been used in domains such as computer vision (Moosavi-Dezfooli et al., 2019; Lee & Park, 2023), graph embedding (Pei et al., 2020), and deep neural networks (Huh, 2020). Noise based on the Bernoulli distribution, as opposed to Gaussian or Uniform noise, has been utilized, as mentioned by Spall (1998). In this approach, each outcome, either -1 or 1, is assigned an equal probability of $\frac{1}{2}$.

3 On Similarity of Uniform noise and Gaussian noise

In this section, we investigate the similarity between uniform and Gaussian noise when used for embedding perturbations. While these noise types yield different statistical properties in low dimensions, their behavior becomes increasingly similar as the number of dimensions grows. This phenomenon is especially relevant in the context of large language models (LLMs), where embeddings typically reside in high-dimensional spaces.

Lemma 1. **(Uniform Distribution L_2 Norm)** *For $P = (x_1, x_2, ..., x_d) \sim U^d([-1, 1])$, the expected L_2 norm is:*

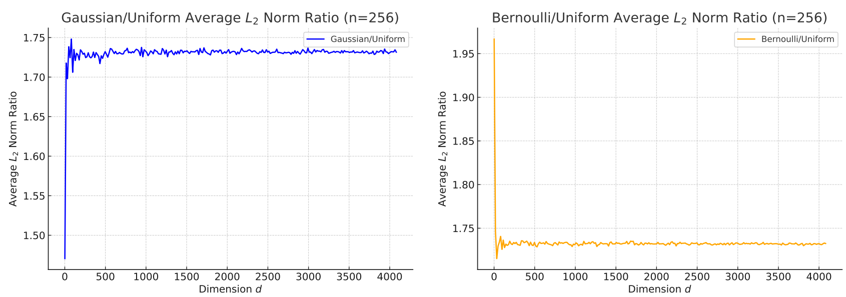
$$E[\|P\|_2] = \sqrt{\frac{d}{3}}. \tag{1}$$

The proof is deferred to Appendix B.0.1.

Lemma 2. **(Gaussian Distribution L_2 Norm)** *For $P = (x_1, x_2, ..., x_d) \sim N^d(0, 1)$, the expected L_2 norm is:*

$$E[\|P\|_2] = \sqrt{d}. \tag{2}$$

The proof is deferred to Appendix B.0.2.



(a) Gaussian/Uniform average L_2 ratio (b) Bernoulli/Uniform average L_2 ratio

Figure 1: Comparison of average L_2 norm ratios for Gaussian and Bernoulli noise relative to Uniform noise as a function of dimensionality.

Drawing from Lemma 1 and Lemma 2, it is apparent that the expected noise from the Gaussian distribution is $\sqrt{3}$ times that of the Uniform distribution. Consequently, to equate the noise scales for comparison, the noise scaling factor for the Gaussian distribution should be adjusted by a factor of $\sqrt{3}$.

As depicted in Figure 1a, a distinct pattern emerges in the ratio of average noise (quantified via L_2 norms) between Gaussian and Uniform distributions as dimensionality increases. Notably, the relative impact of Gaussian noise amplifies, approximating $\sqrt{3}$ times the effect induced by Uniform noise with increasing dimensions. An in-depth exploration and analysis concerning the influence of altering the sample size, while maintaining a fixed dimensionality, are detailed in Appendix A.2.

Moreover, the comparative results on real-world datasets are presented in Table 1, where all conditions are held constant except for the substitution of the Uniform distribution with a Gaussian distribution. In this context, the noise scaling factor for the Gaussian distribution is adjusted by a factor of $\sqrt{3}$, consistent with the discussion above, and one can notice that the performance of both methods thereafter is comparable. A more detailed ablation study is given in the Sec. 5.5.2.

4 Proposed Method: `SymNoise`

In the ideal scenario, our goal is to implement curvature regularization, a technique prevalent in fields such as computer vision (Moosavi-Dezfooli et al., 2019; Lee & Park, 2023), graph embedding (Pei et al., 2020), and deep neural networks (Huh, 2020). However, due to the high computational cost associated with these methods, we aim to explore an alternative approach that adheres to a more stringent condition. This approach has demonstrated superior performance in practice, surpassing current state-of-the-art methodologies. Specifically, we seek to design a function with a gradient (∇f) having value as 0 in the vicinity of the input, i.e., for $x, \epsilon \in \mathbb{R}^d$, $\nabla f = \frac{\left|\frac{f((x-\epsilon))-f(x+\epsilon)}{\epsilon}\right|}{2} \leq \delta$, when $\delta = 0$, we have $f(x+\epsilon) = f(x-\epsilon)$.

In this formulation, the noise turns out to be based on a Bernoulli distribution, diverging from the more commonly used Gaussian or Uniform noise types. Specifically, it uses values of -1 and 1 with equal probability, as in Spall (1998), to provide a balanced and predictable effect on the network's learning.

Following Jain et al. (2024), we train instruction-based models using instruction-response pairs. Unlike `NEFTune`, which adds uniform noise to token embeddings, we introduce symmetric Bernoulli noise. While we retain the same noise scaling factor, $\epsilon = \alpha/\sqrt{Ld}$ (with L as sequence length, d as embedding dimension, and α as a tunable parameter), our method differs in how noise is applied. Details of our approach, `SymNoise`, are provided in Algorithm 2, alongside `NEFTune` in Algorithm 1 for comparison.

4.1 On Similarity of Uniform noise and Bernoulli noise

Lemma 3. (Bernoulli Distribution L_2 Norm) *For $P = (x_1, x_2, ..., x_d)$, with $x_i \in \{-1, 1\}$ and $P(x_i = 1) = P(x_i = -1) = 0.5$, the expected L_2 norm is:*

$$E[\|P\|_2] = \sqrt{d}. \tag{3}$$

The proof is deferred to Appendix B.0.3.

In alignment with the discussions in Sec. 3 and corroborated by Lemma1, Lemma 2, and Fig 1b, it is evident that the noise induced by the Bernoulli distribution is amplified by a factor of $\sqrt{3}$ compared to that of the Uniform distribution. To accommodate this disparity, our proposed method `SymNoise` incorporates this $\sqrt{3}$ scaling factor, as detailed in Algorithm 2.

Algorithm 1 `NEFTune`: **N**oisy **E**mbedding Instruction **Fi**ne**tun**ing (Taken from the paper (Jain et al., 2024))

Input: $\mathcal{D} = \{x_i, y_i\}_1^N$ tokenized dataset, embedding layer $\mathrm{emb}(\cdot)$, rest of model $f_{/\mathrm{emb}}(\cdot)$, model parameters θ, $\mathrm{loss}(\cdot)$, optimizer $\mathrm{opt}(\cdot)$
Hyperparameter: base noise scale $\alpha \in \mathbb{R}^+$
Initialize θ from a pretrained model.

repeat
  $(X_i, Y_i) \sim \mathcal{D}$ {sample a minibatch of data and labels}
  $X_{\mathrm{emb}} \leftarrow \mathrm{emb}(X_i), \mathbb{R}^{B \times L \times d}$ {batch size B, seq. length L, embedding dimension d}
  $\epsilon \sim \mathrm{Uniform}(-1, 1), \mathbb{R}^{B \times L \times d}$ {sample a noise vector}
  $X'_{\mathrm{emb}} \leftarrow X_{\mathrm{emb}} + (\frac{\alpha}{\sqrt{Ld}})\epsilon$ {add scaled noise to embeds [1]}
  $\hat{Y}_i \leftarrow f_{/\mathrm{emb}}(X'_{\mathrm{emb}})$ {make prediction at noised embeddings}
  $\theta \leftarrow \mathrm{opt}(\theta, \mathrm{loss}(\hat{Y}_i, Y_i))$ {train step, e.g., grad descent}
until Stopping criteria met/max iterations.

5 Experiments

In this section, we perform numerous experiments across various models and benchmarks to demonstrate the efficacy of our proposed method `SymNoise` and compare it with existing approaches including `NEFT`.

5.1 Datasets

In this section, we delve into finetuning datasets that have either gained widespread popularity or have recently achieved state-of-the-art results. Due to the memory limitations

Algorithm 2 `SymNoise`: **Sym**metric **Nois**y **E**mbedding Instruction Finetuning (Proposed Method)

Input: $\mathcal{D} = \{x_i, y_i\}_1^N$ tokenized dataset, embedding layer emb$(\cdot)$, rest of model $f_{/\text{emb}}(\cdot)$, model parameters θ, loss$(\cdot)$, optimizer opt$(\cdot)$
Hyperparameter: base noise scale $\alpha \in \mathbb{R}^+$
Initialize θ from a pretrained model.

repeat
$(X_i, Y_i) \sim \mathcal{D}$ {sample a minibatch of data and labels}
$X_{\text{emb}} \leftarrow \text{emb}(X_i), \mathbb{R}^{B\times L\times d}$ {batch size B, seq. length L, embedding dimension d}
$\epsilon \sim \text{Bernoulli}\{-1, 1\}, \mathbb{R}^{B\times L\times d}$ {sample a noise vector}
$X'_{\text{emb}} \leftarrow X_{\text{emb}} + (\frac{\alpha}{\sqrt{Ld}})\frac{\epsilon}{\sqrt{3}}$ {add scaled noise to embeds [2]}
$X''_{\text{emb}} \leftarrow X_{\text{emb}} - (\frac{\alpha}{\sqrt{Ld}})\frac{\epsilon}{\sqrt{3}}$ {subtract same symmetric noise from embeds}
$\hat{Y}_i \leftarrow f_{/\text{emb}}(concat(X'_{\text{emb}}, X''_{\text{emb}}))$ {make prediction at noised embeddings}
$\theta \leftarrow \text{opt}(\theta, \text{loss}(\hat{Y}_i, Y_i))$ {train step}
until Stopping criteria met/max iterations.

of our hardware setup, our focus is exclusively on single-turn datasets following similar protocol as used in Jain et al. (2024). The chosen datasets are: Alpaca (Taori et al., 2023), ShareGPT (Chiang et al., 2023), Evol-Instruc (Xu et al., 2023), and Open-Platypus (Lee et al., 2023). More details about these datasets are in Appendix A.1

In the fine-tuning phase, each model, with the exception of ShareGPT, utilizes the prompt from the Alpaca system. Conversely, ShareGPT is fine-tuned using the prompt from the Vicuna system. Our approach to hyperparameter tuning, including the selection of values, aligns with the methodologies suggested by Jain et al. (2024). We adhered strictly to the same set of hyperparameters as those employed in `NEFTune` (Jain et al., 2024).

5.2 Models

Following Jain et al. (2024) setup for experimentation, our experiments predominantly utilize Large Language Models (LLMs) with a parameter size of 7 billion. Specifically, our focus is on models such as LLaMA-1 (Touvron et al., 2023a), LLaMA-2 (Touvron et al., 2023c), and OPT-6.7B (Zhang et al., 2022). These transformer-based models primarily differ in the amount of training data they've been exposed to, with OPT-6.7B, LLaMA-1, and LLaMA-2 being trained on 180 billion, 1 trillion, and 2 trillion tokens, respectively. This variance in training data volume is expected to manifest in their performance across benchmarks like MMLU, where LLaMA-2 typically outperforms the others.

5.3 Evaluation Protocols

Our experimental framework, adapted from the original `NEFTune` (Jain et al., 2024) setup, primarily utilizes single-turn data for training. We assess the models' conversational skills using `AlpacaEval` and examine their performance on tasks from the OpenLLM Leaderboard. This was done to verify that the introduction of our `symnoise` augmentation does not negatively impact the models' performance on standard multiple-choice tasks. Notably, the results demonstrate that our augmented models consistently outperform the original `neftune` models, albeit by a modest margin.

- **AlpacaEval**: Introduced by Dubois et al. (2023), `AlpacaEval` is crucial for appraising generative quality. It functions as an automated model-based evaluator, comparing Text-Davinci-003's generations with our model's over 805 prompts, focusing on the *Win Rate*. The Win Rate indicates how often our model is preferred over Text-Davinci-003, as judged by model evaluator (GPT-4, ChatGPT etc). The dataset's 805 test prompts, sourced from various platforms, ensure a comprehensive testing scope. AlpacaEval's high human agreement rate (Dubois et al., 2023), validated by 20K annotations, highlights its usefulness and accuracy. We employ both GPT-4 and ChatGPT as evaluators, using ChatGPT initially due to GPT-4's API limitations and costs.

- **Hugging Face OpenLLM Leaderboard**: For leaderboard assessments, datasets like ARC (Clark et al., 2018), HellaSwag (Zellers et al., 2019), and MMLU (Hendrycks et al., 2020) are utilized. These verbalized multiclass classification datasets test the LLM's capability in factual questioning and reasoning. Our evaluations confirm that the SymNoise method does not diminish the models' proficiency in these domains.

5.4 Results

The methodology we employed for tuning hyperparameters and choosing their values adheres closely to the protocols proposed by Jain et al. (2024). Specifically, we meticulously adopted the identical hyperparameter set as delineated in NEFTune by Jain et al. (2024).

5.4.1 Improvement in generated text quality

Our results demonstrate an enhanced metric performance compared to NEFTune in terms of generated text quality. As evident from Table 1, there is a notable improvement across all datasets at the 7B size, with an average increase of 17.6% (compared to NEFTune's improvement of 15.1%). This indicates that the implementation of SymNoise significantly enhances conversational capabilities and answer quality. These findings are supported by evaluations using AlpacaEval, where SymNoise notably outperforms NEFTune. Furthermore, as shown in Table 2, enhancements are also observed in older models like LLaMA-1 and OPT-6.7B, with SymNoise consistently surpassing NEFTune in these models as well. An interesting observation is the comparatively smaller gain by NEFTune in ShareGPT, as per ChatGPT's analysis, a trend not mirrored in GPT-4's evaluation. However, SymNoise consistently excels over NEFTune for ShareGPT in evaluations by both GPT-4 and ChatGPT. In Table 1, the Win Rate shows a significant increase from 29.79% to 69.04% for Alpaca, thereby outperforming the state-of-the-art method NEFTune by 6.7%.

5.4.2 Improvement in OpenLLM Leaderboard tasks

In addressing the potential that SymNoise could enhance conversational abilities at the expense of traditional skills, we conducted evaluations using tasks from the OpenLLM Leaderboard. Employing the LM-Eval Harness framework (Gao et al., 2021), we assessed our model's performance on benchmarks such as MMLU (Hendrycks et al., 2020), ARC (Clark et al., 2018), and HellaSwag (Zellers et al., 2019). These tests shed light on the model's knowledge base, reasoning capabilities, and adherence to factual information. As illustrated in Figure 3, the results indicate that SymNoise not only stabilizes scores but also actively preserves and, in some cases, enhances the model's capabilities. Notably, SymNoise consistently outperforms NEFTune in terms of performance, highlighting its effectiveness in striking a balance between conversational proficiency and traditional computational skills.

5.5 Analysis

As shown in NEFTune (Jain et al., 2024) and related work, adding noise to embeddings during training helps mitigate overfitting to dataset-specific quirks like formatting or phrasing. This shifts the model from memorizing instructions to leveraging the broader capabilities of the pretrained base model. A direct effect is that models produce longer, more coherent responses—preferred by both human and automated evaluators (Dubois et al., 2023). While increased verbosity contributes to performance gains, our analysis shows that SymNoise improves both response quality and quantity beyond what NEFTune achieves.

Conceptually, SymNoise assigns probability mass to multiple noisy variants of instructions, encouraging the model to learn a broader, more uniform distribution rather than overfitting to the training data or a single perturbed version. This promotes better generalization and reduces overfitting.

Table 3: For OpenLLM Leaderboard tasks, the influence of NEFTune and SymNoise is investigated on LLaMA-2, encompassing Alpaca, Evol-Instruct, and OpenPlatypus datasets, alongside LLaMA-1 trained on the Evol-Instruct dataset. Comparative observations reveal a uniformity in performance metrics across the diverse datasets and models, indicating negligible impact of NEFTune but slightly better performance of SymNoise on the overall effectiveness. We follow the similar procedure as mentioned in Jain et al. (2024), and report their results for completeness. In order to minimize computational expenses, we refrained from conducting thorough hyper-parameter optimization, which may have further improved the results.

Task	Llama-2 7B (Alpaca)	+NEFT	+SymNoise
ARC	56.4	56.1	56.0
HellaSwag	80.0	80.1	80.2
MMLU	47.9	47.7	47.9
	Llama-2 7B (OpenPlatypus)	+NEFT	+SymNoise
ARC	54.2	55.4	55.7
HellaSwag	80.4	80.6	80.8
MMLU	43.9	45.3	45.5
	Llama-1 7B (Evol-Instruct.)	+NEFT	+SymNoise
ARC	53.7	54.1	56.0
HellaSwag	77.9	78.0	78.9
MMLU	38.3	38.4	39.1

5.5.1 *Longer responses vs repetition*

In this section, our objective is to determine whether the lengthier responses produced using SymNoise are a result of increased repetition or if they contribute to more diverse and detailed content.

Echoing the insights from Jain et al. (2024) and supporting evidence from leaderboard performances, a notable correlation emerges between extended response lengths and improved performance in the AlpacaEval task. This raises the question of whether the augmentation of response length by SymNoise could lead to diminished text diversity and quality. Our analysis scrutinized the frequency of N-gram repetitions in responses generated by LLaMA-2, trained on various datasets, both with and without SymNoise application.

Following the methodology of Jain et al. (2024), our analysis was restricted to the initial segments of each response to maintain consistency. Specifically, we examined the first 50 words for Alpaca-trained models, 100 words for Evol-Instruct, and 150 words for OpenPlatypus, ensuring that at least half of the responses exceeded these thresholds. Responses shorter than these limits were excluded from the analysis.

As delineated in Table 5, the findings reveal that SymNoise typically yields lengthier responses. However, importantly, the frequency of 2-gram repetitions and overall token log-diversity remain largely consistent, paralleling the results observed with NEFTune. This suggests that the increased length of responses under SymNoise is not simply due to repetitive content, but rather indicates the inclusion of additional, relevant information, thereby enriching the depth and value of the generated responses.

5.5.2 *Ablation study with different strength of noise*

In this section, we explored the efficacy of employing different noise distributions, specifically uniform (NEFTune) versus Gaussian noise, versus within the SymNoise algorithm. From the Table 4, one can notice that Gaussian noise tends to produce longer outputs. However, this increased length does not correlate with a corresponding enhancement in performance. While it is generally observed that longer generations are associated with improved scoring, none of the generation-time strategies employed matched the effectiveness of models trained with SymNoise. Interestingly, our innovative approach, SymNoise, exhibits superior performance, surpassing benchmark results. It demonstrates an approximate improvement of 6.7% over the models utilizing `NEFTune`. Furthermore, we conducted a comparative analysis with Bernoulli noise to underscore the effectiveness of the symmetric opposing noise component in `SymNoise`.

Moreover, we maintained consistent experimental conditions while substituting the Uniform distribution with a Gaussian distribution. In alignment with our theoretical framework, we adjusted the noise scaling factor for the Gaussian distribution by dividing it by $\sqrt{3}$. This adjustment led to comparable performance between the two methods across various `NEFTune` noise levels, reinforcing the validity of our noise scaling approach.

Table 4: `AlpacaEval` Win Rate and Average Character Count assessed by ChatGPT across various noise settings.

Setting	Alpaca	Evol-Instruct	OpenPlatypus
LLaMA-2-7b	48.26 (375)	62.55 (864)	57.20 (1101)
+`NEFT` Noise 5	62.55 (1062)	67.58 (1404)	60.99 (1428)
+`NEFT` Noise 10	61.18 (1010)	65.59 (1697)	60.62 (1834)
+`NEFT` Noise 15	61.86 (820)	66.58 (1651)	61.74 (1694)
+Gaussian Noise $5/\sqrt{3}$	62.6 (1073)	68.01 (1431)	60.31 (1437)
+Gaussian Noise $10/\sqrt{3}$	61.01 (1211)	65.29 (1783)	60.32 (1878)
+Gaussian Noise $15/\sqrt{3}$	61.93 (835)	65.99 (1767)	61.38 (1806)
+Gaussian Noise 5	60.93 (1371)	65.09 (2066)	59.13 (2061)
+Bernoulli Noise 5	61.32 (1272)	65.10 (1840)	60.22 (1968)
+`SymNoise` Noise 5	**64.92** (1186)	**69.62** (1700)	**62.14** (1689)

Table 5: Transposed view of average lengths and 2-gram repetition rates in `AlpacaEval` responses for different training methods.

Metric	LLaMA-2 7B	+`NEFT`	+`SymNoise`
Character Length			
Alpaca	375	1061	1186
Evol-Instruct	864	1403	1700
OpenPlatypus	1100	1694	1689
Whitespace Length			
Alpaca	60	169	176
Evol-Instruct	138	225	245
OpenPlatypus	170	264	270
2-Gram Repetition %			
Alpaca	1.49	1.72	1.80
Evol-Instruct	3.87	3.79	3.80
OpenPlatypus	2.73	3.21	3.30

6 CONCLUSION

In this work, we rigorously establish that Gaussian and uniform random noise are functionally analogous, contingent on appropriate scaling, and demonstrate similar effectiveness on real-world datasets. This revelation is pivotal, particularly in light of the `NEFTune` creators' admission of the method's unexplained superiority, especially over the well-examined Gaussian noise. This insight not only sheds light on the previously opaque superiority of the `NEFTune` method but also bridges the gap with the well-understood Gaussian noise.

Furthermore, we have introduced `SymNoise`, a novel noise injection technique that outperforms `NEFTune` and other existing methods by a large margin. The advancements showcased by `SymNoise` in training large language models (LLMs) emphasize the importance of innovative algorithmic strategies and regularization techniques. Echoing the sentiments of (Jain et al., 2024), the field of LLMs, unlike its counterpart in computer vision, has often favored standardized training methods focusing on model scaling and dataset expansion. However, `SymNoise` underscores the potential of fine-tuning techniques in enhancing model performance, particularly in situations where overfitting to limited instruction datasets is a concern.

References

Sharegpt. https://sharegpt.com/, 2023.

Murtaza Eren Akbiyik. Data augmentation in training {cnn}s: Injecting noise to images, 2020. URL https://openreview.net/forum?id=SkeKtyHYPS.

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.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, et al. Alpagasus: Training a better alpaca with fewer data. *arXiv preprint arXiv:2307.08701*, 2023.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90 https://lmsys.org/blog/2023-03-30-vicuna/, Mar 2023.

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. *arXiv preprint arXiv:2210.11416*, 2022.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, 2019. URL https://aclanthology.org/N19-1423/.

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Alpacafarm: A simulation framework for methods that learn from human feedback. *arXiv preprint arXiv:2305.14387*, 2023.

Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, et al. A framework for few-shot language model evaluation. *Version v0. 0.1. Sept*, 2021.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Dongsung Huh. Curvature-corrected learning dynamics in deep neural networks. In *International Conference on Machine Learning*, pp. 4552–4560. PMLR, 2020.

Neel Jain, Ping-yeh Chiang, Yuxin Wen, John Kirchenbauer, Hong-Min Chu, Gowthami Somepalli, Brian R Bartoldson, Bhavya Kailkhura, Avi Schwarzschild, Aniruddha Saha, et al. Neftune: Noisy embeddings improve instruction finetuning. In *ICLR*, 2024.

Kezhi Kong, Guohao Li, Mucong Ding, Zuxuan Wu, Chen Zhu, Bernard Ghanem, Gavin Taylor, and Tom Goldstein. Robust optimization as data augmentation for large-scale graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 60–69, 2022.

Ariel N Lee, Cole J Hunter, and Nataniel Ruiz. Platypus: Quick, cheap, and powerful refinement of llms. *arXiv preprint arXiv:2308.07317*, 2023.

Yonghyeon Lee and Frank C Park. On explicit curvature regularization in deep generative models. In *Topological, Algebraic and Geometric Learning Workshops 2023*, pp. 505–518. PMLR, 2023.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9078–9086, 2019.

David Nukrai, Ron Mokady, and Amir Globerson. Text-only training for image captioning using noise-injected clip. *arXiv preprint arXiv:2211.00575*, 2022.

OpenAI. Introducing chatgpt. https://openai.com/blog/chatgpt, 2022.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

Hongbin Pei, Bingzhe Wei, Kevin Chang, Chunxu Zhang, and Bo Yang. Curvature regularization to prevent distortion in graph embedding. *Advances in Neural Information Processing Systems*, 33:20779–20790, 2020.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8), 2019. https://cdn.openai.com/better-language-models/language$_models_are_unsupervised_multitask_learners.pdf$.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*, 2021.

James C Spall. Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Transactions on aerospace and electronic systems*, 34(3):817–823, 1998.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Stanford alpaca: An instruction-following llama model, 2023.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023b.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023c.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*, 2022.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*, 2023.

Hanwei Xu, Yujun Chen, Yulun Du, Nan Shao, Yanggang Wang, Haiyu Li, and Zhilin Yang. Zeroprompt: scaling prompt-based pretraining to 1,000 tasks improves zero-shot generalization. *arXiv preprint arXiv:2201.06910*, 2022.

Xiao Zang, Yi Xie, Siyu Liao, Jie Chen, and Bo Yuan. Noise injection-based regularization for point cloud processing. *arXiv preprint arXiv:2103.15027*, 2021.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Wayne Xin Zhao, Yujian Shao, Jingyuan Li, Yuan Wang, Xinyu Li, Zihan Yu, Yujia Ji, Jing Chen, Fei Wang, and Ji-Rong Li. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. Freelb: Enhanced adversarial training for natural language understanding. *arXiv preprint arXiv:1909.11764*, 2019.

A Appendix

A.1 Datasets

- **Alpaca Taori et al. (2023)**: Developed using the Self-Instruct method by Wang et al. (2022) and the Text-Davinci-003 Ouyang et al. (2022) model (Ouyang et al., 2022), Alpaca leverages a small set of seed tasks to generate new instruction tuning tasks and filter out ineffective ones. This dataset has been instrumental in advancing instruction-based learning.
- **ShareGPT Chiang et al. (2023)**: Comprising 70k voluntarily shared ChatGPT conversations Sha (2023), ShareGPT provides a rich source of real-world interaction data. While originally multi-turn, we adapt it to a single-turn format using the Vicunav1.1 dataset version for consistency with our experimental setup.
- **Evol-Instruc Xu et al. (2023)**: This dataset, comprising 70k single-turn instructions, is considered more complex than Alpaca. Originating from the Alpaca dataset, Evol-Instruct employs ChatGPT to refine and evolve the initial instructions, thus broadening the scope and complexity of the tasks.
- **Open-Platypus Lee et al. (2023)**: Formed by combining 11 open-source datasets, Open-Platypus is tailored to enhance LLM performance in STEM and logical reasoning domains. It includes approximately 25k questions, with around 10% generated by LLMs and the rest by human experts. This dataset emphasizes the importance of variety and complexity in question formats.

A.2 Analysis of Distributional Characteristics in High-Dimensional Spaces

In this section, we analyze the behavior of Gaussian, Bernoulli, and Uniform distributions in high-dimensional spaces. We explore how the average L_2 norm ratios of these distributions change with respect to varying dimensions and sample sizes, providing insights into their geometric properties and implications for high-dimensional data analysis.

A.2.1 Average L_2 Norm Ratio with Varying Dimensionality

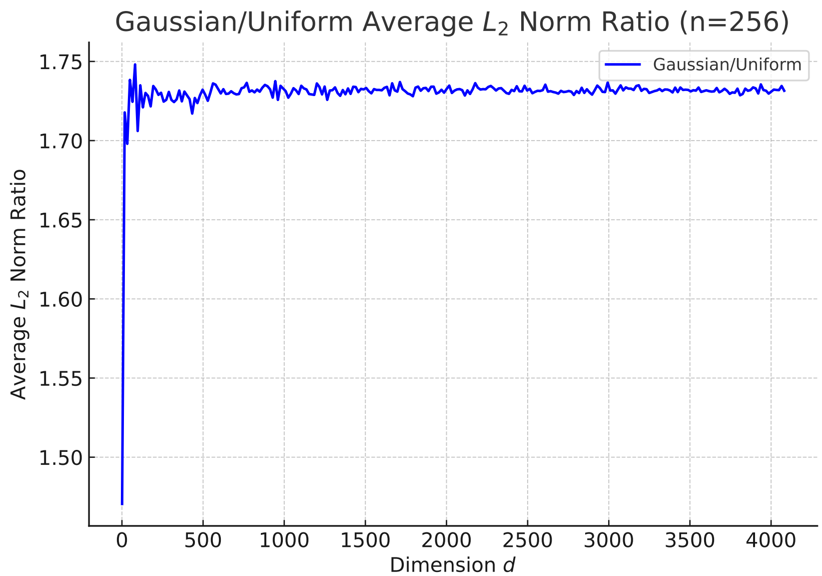


Figure 2: Gaussian/Uniform Average L_2 Norm Ratio as a Function of Dimensionality. The plot illustrates the ratio of the average L_2 norm of points drawn from a Gaussian distribution to that of a Uniform distribution, with the number of points fixed at 256 and the dimensionality varying from 1 to 4096.

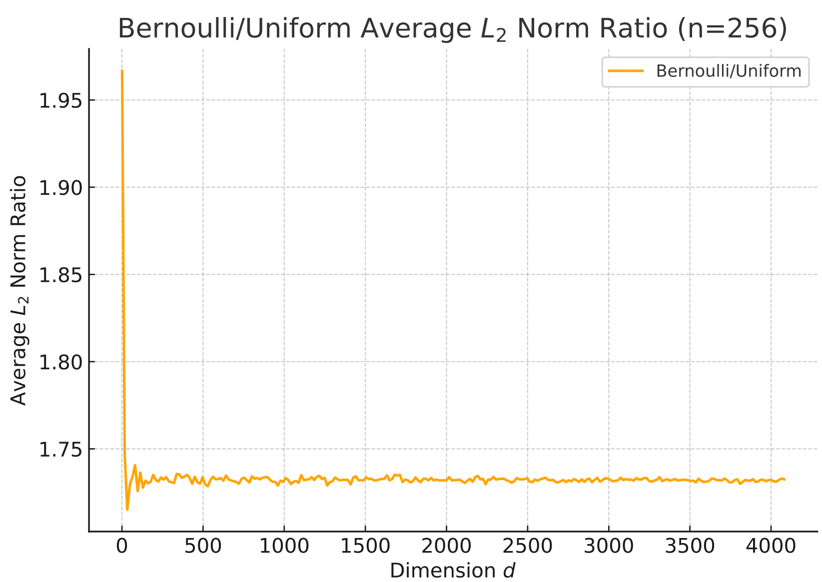


Figure 3: Bernoulli/Uniform Average L_2 Norm Ratio as a Function of Dimensionality. The plot depicts the ratio of the average L_2 norm of points drawn from a Bernoulli distribution to that of a Uniform distribution, with the number of points fixed at 256 and the dimensionality varying from 1 to 4096.

A.2.2 Average L_2 Norm Ratio with Varying Number of Points

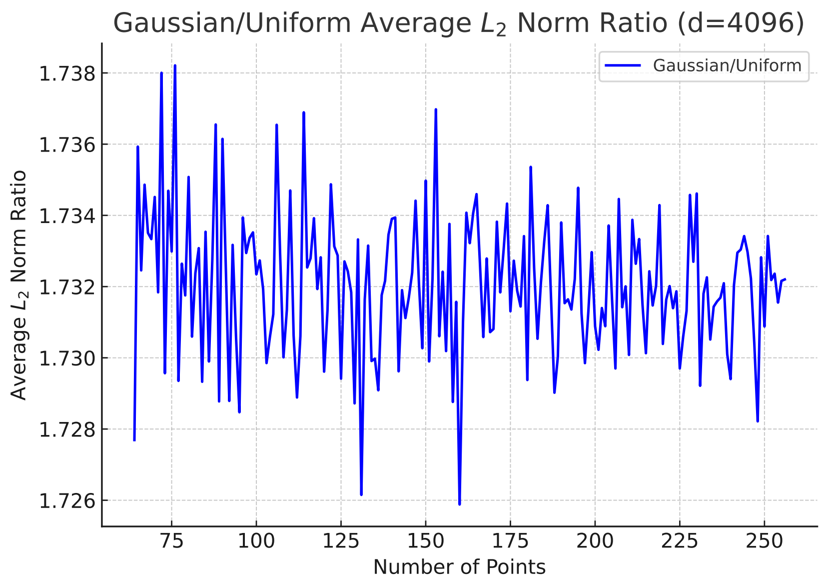


Figure 4: Gaussian/Uniform Average L_2 Norm Ratio for Varying Number of Points. The plot illustrates the ratio of the average L_2 norm of points drawn from a Gaussian distribution to that of a Uniform distribution, with the dimensionality fixed at 4096 and the number of points varying from 64 to 256.

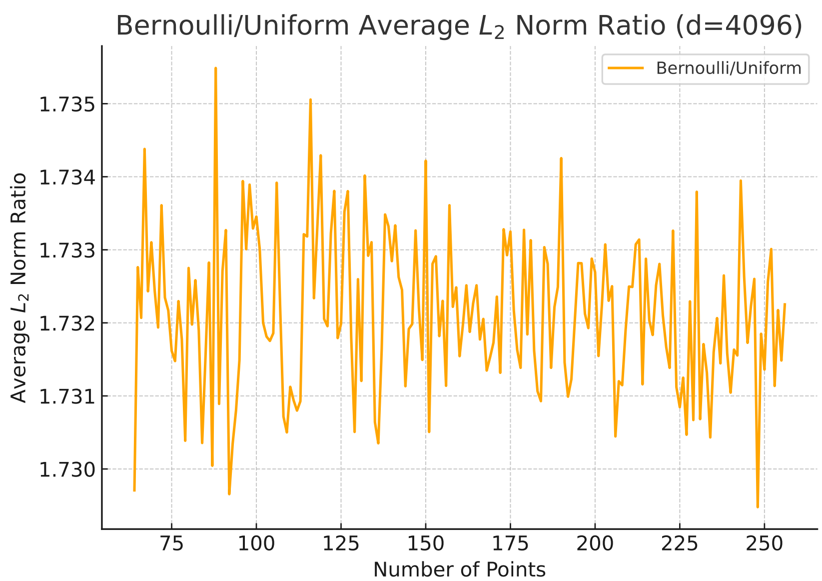


Figure 5: Bernoulli/Uniform Average L_2 Norm Ratio for Varying Number of Points. The plot depicts the ratio of the average L_2 norm of points drawn from a Bernoulli distribution to that of a Uniform distribution, with the dimensionality fixed at 4096 and the number of points varying from 64 to 256.

B Deferred proofs

In this section, we show the proofs omitted from Sec. 3 and Sec. 4.1.

B.0.1 *Proof of Lemma 1*

We state again Lemma 1 from Sec. 3 and present the proof.

Lemma 1. (Uniform Distribution L_2 Norm) *For $P = (x_1, x_2, ..., x_d) \sim U^d([-1, 1])$, the expected L_2 norm is:*

$$E[\|P\|_2] = \sqrt{\frac{d}{3}}.$$

Proof: *Each x_i is uniformly distributed over $[-1, 1]$. The second moment about the origin for a uniform distribution $U(a, b)$ is given by $\frac{b^3 - a^3}{3(b-a)}$. For $U(-1, 1)$, this yields $E[x_i^2] = \frac{1}{3}$. The components x_i are independent, hence the sum of their squares, which represents the L_2 norm squared, is the sum of their expected values: $E[\|P\|_2^2] = \sum_{i=1}^{d} E[x_i^2] = d \cdot \frac{1}{3}$. Taking the square root gives the expected L_2 norm: $E[\|P\|_2] = \sqrt{E[\|P\|_2^2]} = \sqrt{\frac{d}{3}}$.*

B.0.2 *Proof of Lemma 2*

We state again Lemma 2 from Sec. 3 and present the proof.

Lemma 2. (Gaussian Distribution L_2 Norm) *For $P = (x_1, x_2, ..., x_d) \sim N^d(0, 1)$, the expected L_2 norm is:*

$$E[\|P\|_2] = \sqrt{d}.$$

Proof: *Each x_i is distributed according to $N(0, 1)$. The square of a standard normal variable, x_i^2, follows a chi-squared distribution with 1 degree of freedom, for which the mean (expected value) is 1. Given the independence of the components x_i, the expected value of the sum of their squares, representing the L_2 norm squared, is: $E[\|P\|_2^2] = \sum_{i=1}^{d} E[x_i^2] = d \cdot 1 = d$. The expected L_2 norm is the square root of this sum: $E[\|P\|_2] = \sqrt{E[\|P\|_2^2]} = \sqrt{d}$.*

B.0.3 *Proof of Lemma 3*

We state again Lemma 3 from Sec. 4.1 and present the proof.

Lemma 3. (Bernoulli Distribution L_2 Norm) *For $P = (x_1, x_2, ..., x_d)$, with $x_i \in \{-1, 1\}$ and $P(x_i = 1) = P(x_i = -1) = 0.5$, the expected L_2 norm is:*

$$E[\|P\|_2] = \sqrt{d}.$$

Proof: *Each x_i takes values -1 or 1 with equal probability, leading to $x_i^2 = 1$ irrespective of x_i's actual value. Hence, $E[x_i^2] = 1$. Given the independence of the components x_i, the expected value of the sum of their squares, which represents the L_2 norm squared, is simply the sum of the expected values: $E[\|P\|_2^2] = \sum_{i=1}^{d} E[x_i^2] = d \cdot 1 = d$. Therefore, the expected L_2 norm is the square root of this value: $E[\|P\|_2] = \sqrt{E[\|P\|_2^2]} = \sqrt{d}$.*