Unlocking Tokens as Data Points for Generalization Bounds on Larger Language Models

Sanae Lotfi * 1 **Yilun Kuang** * 1 **Brandon Amos** 2
Micah Goldblum 1 **Marc Finzi** 3 **Andrew Gordon Wilson** 1

Abstract

Large language models (LLMs) with billions of parameters excel at predicting the next token in a sequence. Recent work computes non-vacuous compression-based generalization bounds for LLMs, but these bounds are vacuous for large models at the billion-parameter scale. Moreover, these bounds are obtained through restrictive compression techniques, bounding compressed models that generate low-quality text. Additionally, the tightness of these existing bounds depends on the number of IID documents in a training set rather than the much larger number of non-IID constituent tokens, leaving untapped potential for tighter bounds. In this work, we instead use properties of martingales to derive generalization bounds that benefit from the vast number of tokens in LLM training sets. Since a dataset contains far more tokens than documents, our generalization bounds not only tolerate but actually benefit from far less restrictive compression schemes. With Monarch matrices, Kronecker factorizations, and post-training quantization, we achieve non-vacuous generalization bounds for LLMs as large as LLaMA2-70B. Unlike previous approaches, our work achieves the first non-vacuous bounds for models that are deployed in practice and generate high-quality text.

1. Introduction

We do not have a good theoretical understanding for why large language models (LLMs) have such impressive empirical performance. PAC-Bayes and the related finite hypothesis generalization bounds [4; 12; 16] offer a compelling framework for understanding this good performance through the lens of compression. These bounds tell us that a model will provide good generalization if it is capable of fitting its training data while simultaneously being compressible relative to the size of its training set. The generalization bound literature includes many techniques for achieving tighter bounds on image classification problems, ranging from improved bounds themselves to new compression methods [47; 13; 17; 34; 28].

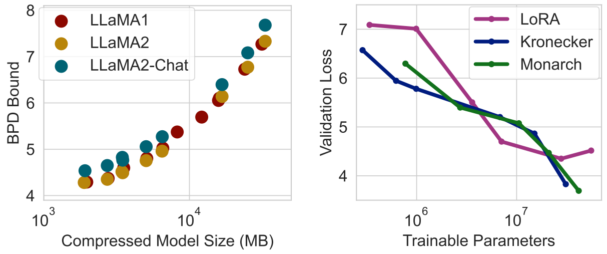

Figure 1. **Non-vacuous bounds for LLMs that scale up to 70B parameters. Left:** Bits per dimension (BPD) bounds on the Amber dataset [26] which contains 1.2 trillion tokens for different LLMs from the LLaMA family ranging in scale from 7 billion to 70 billion parameters [42]. All of these models are quantized to 2-bits, 3-bits and 4-bits per-weight using QuIP# and are publicly available [43]. The different quantization precisions are accounted for in the compressed model size. The trade-off between the empirical performance and the model complexity in our bounds favors models with a smaller compressed size in general, though we observe that across different architectures we can find larger models yielding better bounds. **Right:** Validation negative log-likelihood loss as a function of the total number of trainable parameters for different nonlinear parameterization; namely LoRA, the Kronecker factorization of dense matrices and Monarch matrices. For a fixed budget of trainable parameters, we see that the optimal compression techniques can change, hence our benchmark.

Recent work presented the first non-vacuous generalization bounds for LLMs, considering training points to be independent and identically distributed (IID) documents [29]. The authors compute generalization bounds for the expected bits-per-dimension (BPD) loss, defined for a doc-

[*] Equal contribution, order decided by coin flip [1] New York University [2] Meta AI, author was involved only in an advisory role. All experimentation and data processing were conducted at NYU [3] Carnegie Mellon University. Correspondence to: Sanae Lotfi <sl8160@nyu.edu>, Andrew Gordon Wilson <andrewgw@cims.nyu.edu>.

Work presented at TF2M workshop at ICML 2024, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

ument X composed of k tokens and a language model h as the average negative log probability $\mathrm{BPD}(h, X) = -\frac{1}{k}\sum_i^k \log_2 p_h(x_i|x_{<i})$. These bounds are only non-vacuous for compressed GPT2 variants [35] that output un-grammatical text. The term *vacuous* refers to the random guess performance on next token prediction, which is $\log_2 V$ for BPD where V is the vocabulary size.

Compression-based generalization bounds at the document level suffer from three primary limitations: (1) the number of documents in a training set is limited, and this small sample size leads to loose bounds; (2) due to the small sample size, non-vacuous generalization bounds can only be achieved using compression techniques which significantly modify the LLM pretraining routine; (3) as a result, the models which produce non-vacuous bounds generate low-quality text, so it is unclear what these bounds can tell us about more performant language models.

In this work, we address the above limitations and make the following contributions: (1) We derive a new generalization bound that considers each sample to be an individual token. Even though tokens within a document are not independent, we use properties of martingales to obtain a valid bound that benefits from the number of tokens in a language model's pretraining dataset; (2) We explore several expressive model compression techniques such as Monarch matrices, Kronecker factorizations, and post-training quantization and show that bounding the performance at the token-level favors less restrictive compression strategies; (3) Our work is the first to compute non-vacuous bounds for models compressed only through post-training quantization and without altering the pretraining procedure at all. Consequently, we obtain generalization bounds for massive pretrained LLMs like LLaMA2-70B which generate high-quality text; (4) Our experiments indicate that the chat versions of LLaMA have looser generalization guarantees, demonstrating that fine-tuning these models for dialogue negatively affects their performance on next token prediction; (5) We demonstrate that GPT2 models that are restricted to only seeing k tokens in their context obtain significantly better bounds than k-th order Markov chains for high values of k, reflecting the remarkable ability of transformer-based models in capturing longer range correlations; and (6) We show that a model's ability to recall memorized facts from its pretraining data deteriorates faster than its ability to recognize structured patterns as we decrease the size of the model through compression.

2. Background

Finite hypothesis compression bounds. Let $R(h, x) \in [a, a+\Delta]$ be a bounded risk and $h \in \mathcal{H}$ be a hypothesis drawn from a finite hypothesis space with prior $P(h)$. A classic finite hypothesis generalization bound [39] states that for any $\delta > 0$ with probability $1 - \delta$,

$$R(h) \leq \hat{R}(h) + \Delta\sqrt{\frac{\log 1/P(h) + \log 1/\delta}{2m}} \tag{1}$$

where the empirical risk is defined as $\hat{R}(h) := \frac{1}{m}\sum_{i=1}^m R(h, x_i)$ with $\{x_i\}_{i=1}^m$ being IID and $R(h) = \mathbb{E}[\hat{R}(h)]$. The complexity term depends on the prior log probability $\log 1/P(h)$. We use the Solomonoff prior $P(h) \leq 2^{-K(h)}$ [40], where $K(h)$ is the prefix Kolmogorov complexity of h defined as the length of the shortest program that produces h for a fixed programming language [21]. Consequently, our prior favors models h that have a small minimum compressed length. While the Kolmogorov complexity is incomputable, it can be bounded as $\log 1/P(h) \leq K(h)\log 2 \leq C(h)\log 2 + 2\log C(h)$, where $C(h)$ is the compressed size of the model according to a pre-specified compressor.

Compression bounds for LLMs. When constructing document-level bounds for language, the empirical risk is defined over an entire document X as $R(h, X) = -\log_2 p_h(X)/L$, where $p_h(X)$ is defined auto-regressively on the sequence of tokens $X = [x_1, x_2, \dots x_L]$ as $p_\theta(X) = \prod_{i=1}^L p_h(x_i|x_{<i})$, where $x_{<i}$ denotes $x_1, x_2, \dots, x_{i-1}$.

Prediction smoothing. Since the bound in Equation (1) only applies to a bounded risk, it is not valid for the bits-per-dimension loss that is unbounded. In this case, one can introduce a prediction smoothing probability α to the predictive model such that the generative probability distribution becomes a mixture between the next token probability according to the auto-regressive model $f(\theta)$ with parameters θ and a uniform distribution over the vocabulary of size V as follows: $p_h(x_i|x_{<i}) = (1-\alpha)p_\theta(x_i|x_{<i}) + \alpha/V$. Therefore, $R(h, X)$ can be bounded in an interval of size $\Delta = \log_2(1+(1-\alpha)V/\alpha)$. The optimal α is determined via a grid search in previous work. One of the contributions of our work is that we optimize the prediction smoothing probability at the token level $\alpha_\theta(x_{<i})$, which further improves the bounds. We describe this contribution in Appendix C.

Compressing LLMs with SubLoRA. To achieve the extreme compression level necessary to obtain non-vacuous document-level bounds, Lotfi et al. [29] propose SubLoRA, a non-linear subspace parameterization of an LLM's weights θ. Using SubLoRA, these weights can be written as $\theta = \theta_0 + \mathrm{LoRA}(Pw)$. Here $\theta_0 \in \mathbb{R}^D$ are the model weights at random initialization and $\mathrm{LoRA}(Pw)$ combines low rank adaptation (LoRA) [18] with subspace training [28] via the projector $P \in \mathbb{R}^{D\times d}$. The LoRA decomposition parameterizes a dense matrix $W \in \mathbb{R}^{a\times b}$ as the product of two low-rank matrices $A \in \mathbb{R}^{a\times r}, B \in \mathbb{R}^{r\times b}$ with a small rank r. As for the linear subspace parameterization Pw, the projection matrix P is defined as a Kronecker product $P = Q_1 \otimes Q_2$ produced by orthogonalizing

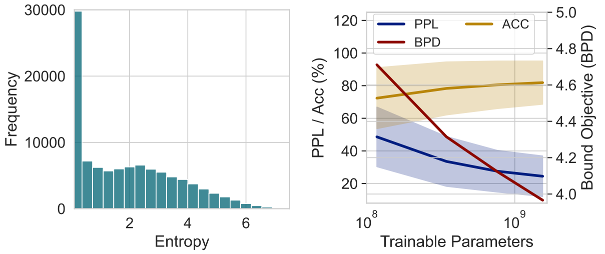

Figure 2. **Our bounds analyze a quantity that is meaningful and predictive of generalization. Left:** Using LLaMA2-7B, we compute the entropy of $p(x_i|x_{<i})$, where the context $x_{<i}$ is fixed and sampled from the Amber training dataset. The distribution over next tokens given a fixed context from the training data is indeed diffuse and characterized by high entropy values. **Right:** On the left y-axis, we plot the average zero-shot accuracy (ACC) and perplexity (PPL) achieved by GPT2 models ranging in scale from 117M to 1.5B averaged over downstream datasets, as reported in Radford et al. [35]. On the right y-axis, we plot an approximation of the conditional BPD expectation that we bound in Equation (2) where we resample x_i from a LLaMA2-7B given fixed training contexts $x_{<i}$ from the Amber dataset. The approximation of the BPD objective that we bound achieves 97.9% and 99.1% correlation with the accuracy and perplexity, respectively.

$Q_1, Q_2 \sim \mathcal{N}(0, 1/\sqrt{D})^{\sqrt{D}\times\sqrt{d}}$ via a QR decomposition.

3. Token-Level Generalization Bounds

We construct a novel bound that naturally accommodates the non-IID structure of the tokens as they occur in documents:

Theorem 3.1. *With probability at least $1-\delta$ over the randomness in a sampled sequence $\{x_1, x_2, \ldots, x_m\}$, if the negative log likelihood of a model $h \in \mathcal{H}$ can be bounded $-\log_2 p_h(\cdot|x_{<i}) \in [a, a+\Delta_i]$, then the negative log likelihood of the data for model h satisfies*

$$\frac{1}{m}\sum_{i=1}^{m}\mathbb{E}[-\log_2 p_h(X_i|x_{<i})|x_{<i}] \leq -\frac{1}{m}\log_2 p_h(x_{\leq m}) + \hat{\Delta}\sqrt{\frac{\log 1/P(h) + \log 1/\delta}{2m}}, \tag{2}$$

where $\hat{\Delta} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\Delta_i^2}$, the expectation is taken over $X_i \sim p(X_i|x_{<i})$ from the data generating process, and $P(h)$ is any normalized prior over a discrete hypothesis space $\mathcal{H}$ that does not depend on $\{x_i\}_{i=1}^{m}$.

On the right-hand side of the bound is the conventional empirical risk: $-\frac{1}{m}\log_2 p_h(x_{\leq m}) = -\frac{1}{m}\sum_i \log_2 p_h(x_i|x_{<i})$ on the measured sequence and a complexity term $\log 1/P(h)$. We describe how we sample sequence $x_{\leq m}$ and compute the empirical risk in Sections B.2 and B.3. We provide the full proof of Theorem 3.1 in Appendix B.1.

The meaningfulness and interpretation of our bounds. It is important to note the difference between the quantity that we bound $\frac{1}{m}\sum_{i=1}^{m}\mathbb{E}[-\log_2 p_h(X_i|x_{<i})|x_{<i}]$, which is conditioned on contexts seen at training, and the expected risk $\mathbb{E}[-\log_2 p_h(X_i|x_{<i})]$ under resampling from the data generating process where new contexts can be sampled from this process. However, the resampled next tokens $x_i|x_{<i}$ are not necessarily from the training set, and to the extent that the distribution over next tokens is entropic, we are measuring a different quantity than the empirical training performance of the hypothesis h. Moreover, we know that the distribution over next tokens is often indeed diffuse; for instance, many words have common synonyms. The distribution over next tokens is especially diffuse when we start a new sentence, for example. We demonstrate how diffuse the distribution $p(x_i|x_{<i})$ is for fixed contexts $x_{<i}$ from the publicly available Amber training dataset [26] (see Appendix F.6) by sampling $x_i|x_{<i}$ using LLaMA2-7B to approximate the generative process. Figure 2(Left) shows that, indeed, the distribution $p(x_i|x_{<i})$ is characterized by a high entropy for a large number of tokens. Given how diffuse the distribution is and the large number of possible sentences, it is broadly infeasible to make predictions on new resampled tokens from the empirical distribution alone. Moreover, Figure 2(Right) shows that our bounds are predictive of downstream performance for GPT2 models.

In short, our bounds go significantly beyond the observation that the empirical distribution converges to the true distribution, and are predictive of generalization on downstream tasks. See Appendix B.4 for an extended discussion.

4. Compressing LLMs Effectively

In shifting from document-level to token-level bounds, the number of data points m increases considerably, and thus we can afford to pay significantly more bits in the complexity of the compressed model. In this new regime, SubLoRA becomes very restrictive. Therefore, we benchmark several expressive compression strategies that can be applied in the pretraining of LLMs or post-training. We provide a more detailed description of these techniques in Appendix D.

Enhanced LoRA. We revisit LoRA [18] and instead of applying it only to self-attention layer weights, we extend it to all linear layers in the model. We also include the biases and layer normalization weights in the projection.

Kronecker Product. For this compression technique, all dense layers in the model are parameterized using a Kronecker product of two low-rank matrices.

Monarch Matrices. We can also parameterize dense matrices W using Monarch matrices [8], where W can be written as the product of two block diagonal matrices and a reshape or permutation operation.

Compression Approach	Bits Per Dimension	Top-1 Error	Top-10 Error	Top-100 Error
SubLoRA [29]	10.49	90.44	71.33	49.77
Enhanced SubLoRA (Ours)	10.44	89.38	69.54	49.84
Enhanced LoRA (Ours)	7.85	78.15	52.48	31.64
Monarch Only (Ours)	**7.65**	**75.87**	**47.47**	**28.34**
Kronecker Only (Ours)	8.03	80.80	52.77	30.14
Kronecker + Subspace (Ours)	10.02	88.75	67.91	47.14
Random Guess	15.62	99.99	99.98	99.80

Table 1. **Non-vacuous generalization bounds using different compression techniques to pretrain variants of GPT2 small.** We find that with the larger complexity budget afforded by the token-level bounds, subspace compression is no longer necessary or even beneficial for the bounds. Of the structures we consider, the Monarch parametrization performs best.

QuIP 2-Bit quantization. In addition to the above compression techniques used to pretrain LLMs in efficient nonlinear subspaces, we consider QuIP post-training quantization [5]. This approach effectively compresses the weights of the LLM into fewer bits while maintaining a good performance.

5. Non-vacuous Bounds For LLMs With Billions of Parameters

We compute generalization bounds for: (i) models that are trained through non-linear subspace compression in the form of LoRA, Kronecker product or Monarch matrices on the OpenWebText dataset, then quantized using the same setup as Lotfi et al. [29], or (ii) pretrained models to which we either apply aggressive quantization, which is the case for GPT2, or use QuIP 2-bit, 3-bit and 4-bit quantized models, which is the case for LLaMA. In the pretrained LLMs setting, we evaluate our bounds for both the OpenWebText (9B tokens) and Amber (1.2T tokens) datasets. Additional details on the experimental setting can be found in Appendix F.

Token-level Bounds via Nonlinear Parametrizations. We report the generalization bounds we obtain when we pretrain variants of GPT2 small with different nonlinear parametrizations in Table 1. We note several important results: (1) We significantly improve upon the LoRA compression with and without subspace as LoRA only led to vacuous bounds in previous works; (2) Among all subspace compression strategies that we explore, Monarch without subspace leads to the tightest bounds. It is also important to note that the Monarch approximation outperforms the 2 other approximations as we increase the number of trainable parameters as shown in Figure 1(Right); (3) applying linear subspace compression on top of the non-linear representations does not help, as further reducing the number of trainable parameters through linear subspace projection leads to a worse trade-off between the empirical performance of the compressed model and its compressed size. We provide an extended discussion of these results in Appendix E.1.

Non-vacuous bounds for pretrained LLMs. We also explore intensive post-training quantization as an effective compression technique for pretrained, publicly available LLaMA1, LLaMA2 and LLaMA2-Chat models. We report the bounds obtained for 2-bit LLaMA2 on the Amber dataset (1.2 trillion tokens) in Table 2. The full set of results is reported in Table 5 and a full discussion of bounds for pretrained LLMs can be found in Appendix E.2 and includes pretrained GPT2 models as well. We obtain non-vacuous bounds for all these models, despite their large scale ranging from 7B to 70B parameters. We also observe that the LLaMA-Chat variants obtain worse bounds, demonstrating the negative effect of fine-tuning these models for dialogue use cases on next token prediction performance.

Model	BPD	Top-1 Error	Top-100 Error
LLaMA2-7B	**4.28**	**47.5**	**12.56**
LLaMA2-13B	4.51	47.85	14.44
LLaMA2-70B	6.39	58.26	25.04
Random Guess	14.97	99.99	99.68

Table 2. Pretrained LLaMA2 models achieve non-vacuous token-level bounds on the Amber dataset via post-training quantization.

In Appendix E, we additionally show that models for which we obtain non-vacuous bounds generate high-quality text. We also contextualize our bounds against Markov chains and show that our compressed GPT2 models go beyond storing simple n-gram statistics. Finally, we demonstrate that highly compact models are capable of reasoning and encoding patterns since they are compressible.

6. Conclusion

We introduce novel token-level generalization bounds for LLMs which are able to accommodate the non-IID nature of the tokens within the training corpus. Combined with different compression techniques, we achieve non-vacuous bounds for LLMs with up to 70 billion parameters. These models are capable of producing high quality text, unlike those in prior work. While we still have a gap to close between the typical validation BPD and the constraint of our bounds, our bounds are meaningful and make statements that go beyond what is achieved by simple n-gram statistics.

References

[1] Akinwande, V., Jiang, Y., Sam, D., and Kolter, J. Z. Understanding prompt engineering may not require rethinking generalization. *arXiv preprint arXiv:2310.03957*, 2023.

[2] Azuma, K. Weighted sums of certain dependent random variables. *Tohoku Mathematical Journal, Second Series*, 19(3):357–367, 1967.

[3] Benesty, J., Chen, J., Huang, Y., and Cohen, I. Pearson correlation coefficient. In *Noise reduction in speech processing*, pp. 37–40. Springer, 2009.

[4] Catoni, O. Pac-bayesian supervised classification: the thermodynamics of statistical learning. *arXiv preprint arXiv:0712.0248*, 2007.

[5] Chee, J., Cai, Y., Kuleshov, V., and Sa, C. D. Quip: 2-bit quantization of large language models with guarantees, 2024.

[6] Chugg, B., Wang, H., and Ramdas, A. A unified recipe for deriving (time-uniform) pac-bayes bounds. *Journal of Machine Learning Research*, 24(372):1–61, 2023.

[7] Computer, T. Redpajama: an open dataset for training large language models, 2023. URL `https://github.com/togethercomputer/RedPajama-Data`.

[8] Dao, T., Chen, B., Sohoni, N. S., Desai, A., Poli, M., Grogan, J., Liu, A., Rao, A., Rudra, A., and Ré, C. Monarch: Expressive structured matrices for efficient and accurate training. In *International Conference on Machine Learning*, pp. 4690–4721. PMLR, 2022.

[9] Dettmers, T., Lewis, M., Shleifer, S., and Zettlemoyer, L. 8-bit optimizers via block-wise quantization, 2022.

[10] Dettmers, T., Shmitchell, S., Roberts, A., Lee, K., Brown, T. B., Song, D., and Raffel, C. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*, 2023.

[11] Deutsch, P. Rfc1952: Gzip file format specification version 4.3, 1996.

[12] Dziugaite, G. K. and Roy, D. M. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017.

[13] Dziugaite, G. K., Hsu, K., Gharbieh, W., Arpino, G., and Roy, D. On the role of data in pac-bayes bounds. In *International Conference on Artificial Intelligence and Statistics*, pp. 604–612. PMLR, 2021.

[14] Edalati, A., Tahaei, M., Kobyzev, I., Nia, V. P., Clark, J. J., and Rezagholizadeh, M. Krona: Parameter efficient tuning with kronecker adapter. *arXiv preprint arXiv:2212.10650*, 2022.

[15] Frantal, Z., Gruslys, A., and Kiela, D. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

[16] Goldblum, M., Finzi, M., Rowan, K., and Wilson, A. G. The no free lunch theorem, kolmogorov complexity, and the role of inductive biases in machine learning. *arXiv preprint arXiv:2304.05366*, 2023.

[17] Hayou, S., He, B., and Dziugaite, G. K. Probabilistic fine-tuning of pruning masks and pac-bayes self-bounded learning. *arXiv preprint arXiv:2110.11804*, 2021.

[18] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[19] Jin, T., Clement, N., Dong, X., Nagarajan, V., Carbin, M., Ragan-Kelley, J., and Dziugaite, G. K. The cost of down-scaling language models: Fact recall deteriorates before in-context learning. *arXiv preprint arXiv:2310.04680*, 2023.

[20] Kim, J., Lee, J. H., Kim, S., Park, J., Yoo, K. M., Kwon, S. J., and Lee, D. Memory-efficient fine-tuning of compressed large language models via sub-4-bit integer quantization. *arXiv preprint arXiv:2305.14152*, 2023.

[21] Kolmogorov, A. N. On tables of random numbers. *Sankhyā: The Indian Journal of Statistics, Series A*, pp. 369–376, 1963.

[22] Kuznetsov, V. and Mohri, M. Generalization bounds for non-stationary mixing processes. *Machine Learning*, 106(1):93–117, 2017.

[23] Li, R., Allal, L. B., Zi, Y., Muennighoff, N., Kocetkov, D., Mou, C., Marone, M., Akiki, C., Li, J., Chim, J., Liu, Q., Zheltonozhskii, E., Zhuo, T. Y., Wang, T., Dehaene, O., Davaadorj, M., Lamy-Poirier, J., Monteiro, J., Shliazhko, O., Gontier, N., Meade, N., Zebaze, A., Yee, M.-H., Umapathi, L. K., Zhu, J., Lipkin, B., Oblokulov, M., Wang, Z., Murthy, R., Stillerman, J., Patel, S. S., Abulkhanov, D., Zocca, M., Dey, M., Zhang, Z., Fahmy, N., Bhattacharyya, U., Yu, W., Singh, S., Luccioni, S., Villegas, P., Kunakov, M., Zhdanov, F., Romero, M., Lee, T., Timor, N., Ding, J., Schlesinger, C., Schoelkopf, H., Ebert, J., Dao, T., Mishra, M., Gu, A., Robinson, J., Anderson, C. J.,

Dolan-Gavitt, B., Contractor, D., Reddy, S., Fried, D., Bahdanau, D., Jernite, Y., Ferrandis, C. M., Hughes, S., Wolf, T., Guha, A., von Werra, L., and de Vries, H. Starcoder: may the source be with you!, 2023.

[24] Li, Y., Ildiz, M. E., Papailiopoulos, D., and Oymak, S. Transformers as algorithms: Generalization and stability in in-context learning. In *International Conference on Machine Learning*, pp. 19565–19594. PMLR, 2023.

[25] Liu, Y., Xu, Q., Xu, W., and Zhu, J. Llm-qat: Data-free quantization aware training for large language models. *arXiv preprint arXiv:2305.17888*, 2023.

[26] Liu, Z., Qiao, A., Neiswanger, W., Wang, H., Tan, B., Tao, T., Li, J., Wang, Y., Sun, S., Pangarkar, O., Fan, R., Gu, Y., Miller, V., Zhuang, Y., He, G., Li, H., Koto, F., Tang, L., Ranjan, N., Shen, Z., Ren, X., Iriondo, R., Mu, C., Hu, Z., Schulze, M., Nakov, P., Baldwin, T., and Xing, E. P. Llm360: Towards fully transparent open-source llms, 2023.

[27] Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[28] Lotfi, S., Finzi, M., Kapoor, S., Potapczynski, A., Goldblum, M., and Wilson, A. G. Pac-bayes compression bounds so tight that they can explain generalization. *Advances in Neural Information Processing Systems*, 35:31459–31473, 2022.

[29] Lotfi, S., Finzi, M., Kuang, Y., Rudner, T. G., Goldblum, M., and Wilson, A. G. Non-vacuous generalization bounds for large language models. *arXiv preprint arXiv:2312.17173*, 2023.

[30] Mohri, M. and Rostamizadeh, A. Stability bounds for non-iid processes. *Advances in Neural Information Processing Systems*, 20, 2007.

[31] Norris, J. R. *Markov chains*. Number 2. Cambridge university press, 1998.

[32] Park, G., Kim, J., Kim, J., Choi, E., Kim, S., Kim, S., Lee, M., Shin, H., and Lee, J. Lut-gemm: Quantized matrix multiplication based on luts for efficient inference in large-scale generative language model. *arXiv preprint arXiv:2206.09557*, 2022.

[33] Penedo, G., Malartic, Q., Hesslow, D., Cojocaru, R., Cappelli, A., Alobeidli, H., Pannier, B., Almazrouei, E., and Launay, J. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only, 2023.

[34] Pérez-Ortiz, M., Rivasplata, O., Shawe-Taylor, J., and Szepesvári, C. Tighter risk certificates for neural networks. *Journal of Machine Learning Research*, 22 (227):1–40, 2021.

[35] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[36] Rakhlin, A. and Sridharan, K. On equivalence of martingale tail bounds and deterministic regret inequalities. In *Conference on Learning Theory*, pp. 1704–1722. PMLR, 2017.

[37] Ralaivola, L., Szafranski, M., and Stempfel, G. Chromatic pac-bayes bounds for non-iid data: Applications to ranking and stationary β-mixing processes. *The Journal of Machine Learning Research*, 11:1927–1956, 2010.

[38] RelaxML, C. Quip#: Quip with lattice codebooks. `https://github.com/Cornell-RelaxML/quip-sharp`, 2024.

[39] Shalev-Shwartz, S. and Ben-David, S. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[40] Solomonoff, R. J. A formal theory of inductive inference. part i. *Information and control*, 7(1):1–22, 1964.

[41] Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[42] Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. Llama 2: Open foundation and fine-tuned chat models, 2023.

[43] Tseng, A., Chee, J., Sun, Q., Kuleshov, V., and De Sa, C. Quip#: Even better llm quantization with hadamard incoherence and lattice codebooks. *arXiv preprint arXiv:2402.04396*, 2024.

[44] Xu, Q., Xu, W., and Zhu, J. Tensorgpt: Efficient compression of the embedding layer in llms based on the tensor-train decomposition. *arXiv preprint arXiv:2307.00526*, 2023.

[45] Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021.

[46] Zhang, R.-R. and Amini, M.-R. Generalization bounds for learning under graph-dependence: A survey. *arXiv preprint arXiv:2203.13534*, 2022.

[47] Zhou, W., Veitch, V., Austern, M., Adams, R. P., and Orbanz, P. Non-vacuous generalization bounds at the imagenet scale: a pac-bayesian compression approach. In *International Conference on Learning Representations*, 2019.

A. Related Work

Generalization bounds for neural networks. Deep neural networks are challenging to be understood using generalization theory due to their many parameters [45]. However over the past years, there has been success in constructing meaningful bounds such as for image classification models [12], vision-language models [1], and tabular data [16], often through the methodology of compression [47; 28]. Lotfi et al. [29] extend compression-based generalization bounds to the LLM setting, and obtain non-vacuous bounds at the document level. Li et al. [24] explore generalization in few-shot learning, establishing bounds based on in-context examples while maintaining a fixed pretrained model. In contrast, we investigate pretraining generalization bounds to understand why models do not overfit at training time, despite the increased dataset complexity.

Non-IID Generalization bounds. Ralaivola et al. [37] analyze the dependence graph of the random variables, deriving a bound based on the graph coloring number, fitting into a broader line of work making use of properties of the dependence graph [46]. Unfortunately for text data, the dependencies are unknown or assumed to follow the triangular autoregressive dependency structure for all pairs in the sequence. A related line of work has been to explicitly estimate coefficients which quantify the extent that random variables relate to each other, see e.g. Mohri & Rostamizadeh [30]; Kuznetsov & Mohri [22]. However, it is unclear how best to apply these methods to neural networks. Martingale tail bounds are sometimes used in online learning and reinforcement learning, e.g., for establishing regret bounds [36]. Chugg et al. [6] present a large collection of generalization bounds both in the IID and martingale settings. These results extend and generalize many existing bounds. We view our contribution as orthogonal to the efforts since we focus on constructing the components necessary to generate practical bounds for LLMs rather than abstractly innovating on concentration inequalities.

Large language models and compression. Parameter-efficient finetuning methods, such as LoRA [18], parametrize weight matrices as products of two trainable low-rank matrices on top of frozen pretrained weights. QLoRA uses 4-bit NormalFloat (NF4) and double quantization, enabling single-GPU finetuning for a 65B parameter LLM without performance degradation [9; 10]. Post-training quantization approaches, such as GPTQ [15], rely on second-order information and quantize each row of weight matrices independently. QuIP uses adaptive rounding and incoherence processing of second-order Hessian matrices, enabling 2-bit quantization of LLMs [5]. Other compression techniques for LLMs include replacing most of the 16-bit operations with 8-bit matrix multiply [9], using data-free distillations [25], designing custom kernels and sub-4-bit integer quantization [20; 32], and compressing embeddings as low-rank matrix-product state [44].

B. Token-Level Martingale Bound

B.1. Proof of the Main Theorem

Theorem B.1. *With probability at least $1-\delta$ over the randomness in a sampled sequence $x_1, x_2, \dots, x_m$, if the negative log likelihood of a model $h \in \mathcal{H}$ can be bounded $-\log_2 p_h(\cdot|x_{<i}) \in [a, a+\Delta_i]$ for some Δ_i (possibly a function of h), then the negative log likelihood of the data of a given hypothesis h satisfies*

$$\frac{1}{m}\sum_{i=1}^{m}\mathbb{E}[-\log_2 p_h(X_i|x_{<i})|x_{<i}] \leq -\frac{1}{m}\log_2 p_h(x_{\leq m}) + \hat{\Delta}\sqrt{\frac{\log 1/P(h) + \log 1/\delta}{2m}}, \tag{3}$$

where $\hat{\Delta} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\Delta_i^2}$, the expectation is taken over $X_i \sim p(X_i|x_{<i})$ from the data generating process, and $P(h)$ is any normalized prior over a discrete hypothesis space $\mathcal{H}$ that does not depend on $\{x_i\}_{i=1}^{m}$.

Proof sketch. *The proof of Theorem 3.1 is an application of Azuma's inequality [2] and can be broken down into the following steps:*

- *Construct a martingale difference sequence from the difference between the NLL on token x_i, and its expectation given the tokens $x_{<i}$. From the boundedness of NLL one can show that the differences are bounded.*
- *Apply Azuma's inequality for each hypothesis, choosing failure probability proportional to the chosen prior $P(h)$.*
- *Perform a union bound of the failure probabilities over all hypotheses. If all of the hypotheses satisfy the bound simultaneously, then so does the data dependent hypothesis h^*.*

Proof. Given the autoregressive predictions $R(h, x_i, x_{<i}) := -\log_2 p_h(x_i|x_{<i})$ where $x_{<i} := \{x_1, x_2, \ldots, x_{i-1}\}$. Let $\{x_i\}$ denote the actual values of the sequence that were found empirically, and $\{X_i\}$ be the random variables for these quantities.

The collection of random variables (indexed by i) $Z_i = \mathbb{E}[R(h, X_i, x_{<i})|x_{<i}] - R(h, X_i, x_{<i})$ form a Martingale difference sequence with respect to $x_{<i}$. Note here that the expectation is over the distribution $X_i \sim p(X_i|x_{<i})$. From the construction, $\mathbb{E}[Z_i|x_{<i}] = 0$ and the sequence is bounded: $A_i = \mathbb{E}[R(h, X_i, x_{<i})|x_{<i}] - a \leq Z_i \leq \Delta_i + \mathbb{E}[R(h, X_i, x_{<i})|x_{<i}] - a = B_i$, with $B_i - A_i = \Delta_i$.

Δ_i may depend on $x_{\geq i}$ but only through it's dependence on the hypothesis $h(\{x\}_{i=1}^m)$. For a fixed h we may conclude that $\sum_{i=1}^m Z_i$ is bounded difference Martingale sequence (with respect to $\{x_{<i}\}_{i=1}^m$), and we can apply Azuma's inequality [2] to derive that for any $t > 0$:

$$P\big(\sum_{i=1}^m Z_i > mt\big) \leq \exp\big(-2m^2t^2/\sum_{i=1}^m \Delta_i^2\big)$$
$$P\big(\frac{1}{m}\sum_{i=1}^m Z_i > t\big) \leq \exp\big(-2mt^2/\hat{\Delta}^2\big).$$

Judiciously choosing

$$t(h) = \hat{\Delta}\sqrt{\frac{\log 1/P(h) + \log 1/\delta}{2m}},$$

we have that $P\big(\frac{1}{m}\sum_{i=1}^m Z_i > t(h)\big) = P(h)\delta$.

Applying a union over the events $\bigcup_{h\in\mathcal{H}}\big[\frac{1}{m}\sum_{i=1}^m Z_i(h) > t(h)\big]$, we have

$$P\big(\frac{1}{m}\sum_{i=1}^m Z_i > t(h)\big) \leq \sum_h P(h)\delta = \delta,$$

therefore $P\big(\frac{1}{m}\sum_{i=1}^m Z_i \leq t(h)\big) > 1 - \delta$. Unpacking the definition of Z_i, we have that with probability at least $1 - \delta$

$$\frac{1}{m}\sum_{i=1}^m \mathbb{E}[R(h, X_i, x_{<i})|x_{<i}] \leq \frac{1}{m}\sum_{i=1}^m R(h, x_i, x_{<i}) + \hat{\Delta}\sqrt{\frac{\log 1/P(h) + \log 1/\delta}{2m}}.$$

Expressed in terms of the log likelihood, we can write this as:

$$\frac{1}{m}\sum_{i=1}^m \mathbb{E}[-\log_2 p_h(X_i|x_{<i})|x_{<i}] \leq -\frac{1}{m}\log_2 p_h(x_{\leq m}) + \hat{\Delta}\sqrt{\frac{\log 1/P(h) + \log 1/\delta}{2m}}$$

□

B.2. Sampling and Empirical Risk Evaluation

In this section, we define more precisely the sequence $x_{\leq m}$ for which we compute the empirical risk in Equation (2). We construct a sample $x_{\leq m}$ from the stochastic process p_{data} by first sampling independent and identically distributed documents, e.g., the documents that form the OpenWebText dataset. Then, we concatenate these documents deterministically using end of text (EOT) tokens. Consequently, the ground truth stochastic process has the following property:

$$p_{\text{data}}(x_i|x_{<i}) = p_{\text{data}}(x_i|x_k, ..., x_{i-1}), \tag{4}$$

where x_k is the previous EOT token. This equality holds exactly due to how the stochastic process is implemented.

On the other hand, it would not be guaranteed that a generative model $p_h(x)$ satisfies the property in Equation (4) apriori if the model were allowed to attend to tokens $x_{<k}$, even when the data generating process has this property. However, we explicitly prohibit our generative model h from attending to tokens $x_{<k}$ through the attention mask, as we have the flexibility to do so in defining our hypothesis class and model family. Therefore, our model p_h that we bound also satisfies this property $p_h(x_i|x_{<i}) = p_h(x_i|x_k, ..., x_{i-1})$ exactly, and not approximately.

In conclusion, the empirical risk for our generative model h and a sequence $x_{\leq m}$ sampled from the stochastic process defined above can be written as follows:

$$-\frac{1}{m}\log_2 p_h(x_{\leq m}) = -\frac{1}{m}\sum_i \log_2 p_h(x_i|x_{<i}) = -\frac{1}{m}\sum_i \log_2 p_h(x_i|x_k, \dots x_{i-1}),$$

where x_k is the nearest EOT token occurring before x_i. Given the large size of the OpenWebtext and Amber datasets, containing 9 billions and 1.2 trillion tokens respectively, we use subsampling for the evaluation of the empirical risk. More details can be found in Appendix B.3.

B.3. Empirical Risk Subsampling

We evaluate our bounds for the OpenWebtext and Amber datasets which contain 9 billion and 1.2 trillion tokens, respectively. Computing the exact empirical risk for these datasets would be prohibitively expensive. Therefore, we use subsampling for the evaluation of the empirical risk to accelerate bound computation. In Equation (2), we use the following inequality which holds with probability at least $1 - \delta_2$:

$$-\frac{1}{m}\log_2 p_h(x_{\leq m}) \leq -\frac{1}{n}\sum_{j=1}^{n} \log_2 p_h(x_{\sigma(j)}|x_{<\sigma(j)}) + \hat{\Delta}\sqrt{\frac{\log 1/\delta_2}{2n}} \tag{5}$$

for a subsample of size n where σ is a random permutation. We choose δ_1 in Equation (2) with respect to a new overall failure probability δ to be $\delta_1 = \delta n/(n + m)$ and choose $\delta_2 = \delta m/(n + m)$ so that the overall failure probability is still δ. The proof is simple and similar to that provided in Lotfi et al. [29].

B.4. Interpretation of Our Token-level Bounds

Token-level vs. document-level bounds. In contrast to document-level bounds, our token-level bounds increase the number of samples, driving down the size of the complexity term, and do not require the IID assumption. Whereas the number of samples previously would be the number of documents, it now is simply the number of tokens in the dataset, a far higher number. As a consequence of decreasing the complexity term, the empirical risk will be a more significant contributor to our bounds compared to document-level bounds. Therefore, we achieve non-vacuous bounds for much larger and more performant models that generate high-quality text. This development brings our theoretical bounds a large step closer to aligning with empirical generalization.

Our bounds are predictive of downstream performance. To provide further empirical evidence of the meaningfulness of our bounds, we compute an approximation of the quantity that we bound in Equation (2) by sampling next tokens x_i using LLaMA2-7B given fixed contexts $x_{<i}$ from the Amber dataset. We plot this quantity on the right y-axis of Figure 2(Right), and show on the left y-axis the performance of GPT2 models of varying sizes on downstream datasets as reported in Radford et al. [35]; see Appendix F.4 for more details. Not only does the approximation of the BPD objective show the same trend as the downstream performance for different GPT2 variants, but it also achieves 97.9% and 99.1% correlation [3] with downstream task accuracy and perplexity metrics, respectively.

C. Token-Level Prediction Smoothing

Rather than using a single label smoothing α for all data points, we propose to use the network itself to determine which tokens warrant more confidence and which ones require more smoothing to limit their worst-case behavior. We perform token-level prediction smoothing by adding a linear head to the LLM that outputs the probability α for each token, such that $p_h(x_i|x_{<i}) = \big(1 - \alpha_\theta(x_{<i})\big)p_\theta(x_i|x_{<i}) + \alpha_\theta(x_{<i})/V$. The training objective corresponds to the upper bound in Equation (2) rather than the empirical risk alone, where the α parameter factors into the bound via the interval size $\Delta_i = \log_2\big(1 + (1 - \alpha_\theta(x_{<i}))V/\alpha_\theta(x_{<i})\big)$. Therefore, the values of $\alpha_\theta(x_{<i})$ are adjusted to achieve the best trade-off

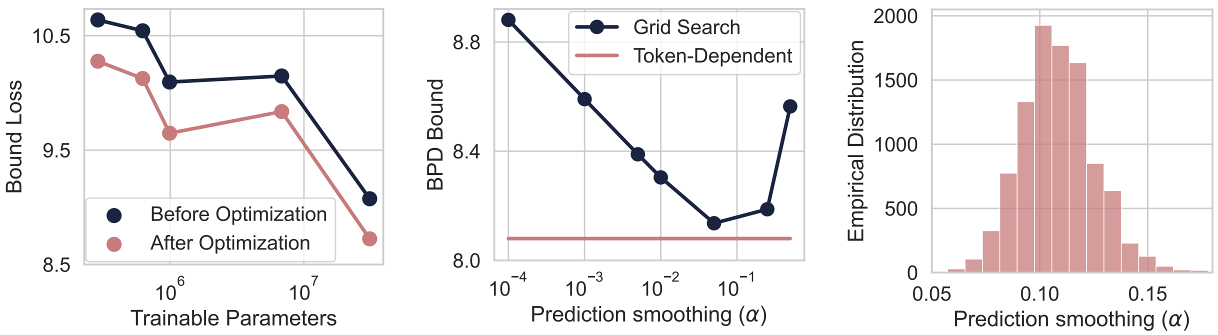

Figure 3. **Token-level prediction smoothing improves our bounds. Left:** After training, we optimize a conservative upper bound on the generalization bound that we would get from Equation (2) with respect to the α head parameters. Doing so yields a noticeable reduction in the value of the bound. **Middle:** BPD generalization bound as a function of a single global parameter chosen from a discrete number of values vs. the generalization bound for the token-dependent α after optimization. **Right:** Histogram of the values taken by $\alpha(x_{<i})$ over different inputs.

between the empirical risk and the compressed model size. We perform this optimization post-training using a subset of the training dataset.

We demonstrate in Figure 3(Left) that using this token-dependent α significantly improves the value of the bounds. In Figure 3 (Middle), we compare to the setting where the optimal α is obtained through a grid search, and in Figure 3(Right) we examine the distribution of α produced by the model.

D. Exploring Different Compression Techniques

D.1. Efficient Nonlinear Parametrizations

In addition to LoRA, we explore two expressive nonlinear parametrizations $f(\theta)$ that make efficient use of the parameter space: Kronecker structures [14] and Monarch matrices [8]. We can use these nonlinear parametrizations directly, or in conjunction with subspace compression, parametrizing the full parameters as $\theta = \theta_0 + f(Pw)$ for a projection matrix $P \in \mathbb{R}^{D\times d}$. After training, the parameters are quantized as in and coded using arithmetic coding. We describe these structures below.

LoRA. With LoRA [18], the weight matrices of linear layers are parametrized via low rank updates. Each weight matrix $W \in \mathbb{R}^{a\times b}$ is parametrized $W = W_0 + AB$ for $A \in \mathbb{R}^{a\times r}, B \in \mathbb{R}^{r\times b}$ with a small rank r, where W_0 is given by the initialization and A, B form the trainable parameters in each layer. Rather than considering only self-attention layer weights [18; 29], we extend SubLoRA to all linear layers in the model and compress the biases and layernorm weights in the subspace projection.

Kronecker Product. We can represent W as a Kronecker product $W = A \otimes B$, where $\otimes$ is the Kronecker product, $A \in \mathbb{R}^{a_1\times b_1}, B \in \mathbb{R}^{a_2\times b_2}$ and $a_1a_2 = a$, $b_1b_2 = b$, which reduces the parameters over the dense layer. This approach has been used in recent work for parameter-efficient finetuning [14] and as an alternative structure for pretraining.

Monarch Matrices. We also consider Monarch matrices [8], which employ two block diagonal matrices A, and B typically with A and B formed by $\sqrt{a}$ blocks of size $\sqrt{a} \times \sqrt{b}$ and a reshape or permutation operation R: $W = ARB$. The matrix multiplication is implemented by reshaping the input axis a into $(\sqrt{a}, \sqrt{a})$, applying matrix A as a batched matrix multiply on one axis, and then applying B to the other axis by permuting the axes. Monarch matrices have shown considerable promise as an expressive and hardware-efficient replacement for linear layers.

D.2. QuIP 2-Bit Quantization of LLM

In addition to pretraining LLMs in efficient nonlinear subspaces, we explore recent post-training quantization methods to reduce the model complexity. Quantization with Incoherence Process (QuIP) compresses LLM weights to a smaller number of bits while preserving model performance [5].

Model	BPD	Top-1 Error (%)	Top-100 Error (%)
GPT2 (124M)	**7.61**	**74.82**	**26.98**
GPT2 (355M)	8.50	79.19	32.72
GPT2 (774M)	10.47	89.50	44.23
Random Guess	15.62	99.99	99.80

Table 3. Pretrained GPT2 models achieve non-vacuous bounds for next token prediction on OpenWebText through post-training quantization only and without altering the pretraining.

Adaptive Rounding. For a weight matrix $W \in \mathbb{R}^{a \times b}$, QuIP minimizes the proxy quadratic objective $\ell(\hat{W}) = \mathbb{E}[\|(\hat{W} - W)x\|^2] = \mathrm{tr}((\hat{W} - W)H(\hat{W} - W)^\top)$, where $\hat{W} \in \mathbb{R}^{a \times b}$ are the quantized weights, $x \in \mathbb{R}^b$ is a vector drawn randomly from a calibration set, and H is the second moment matrix of these vectors used as a proxy hessian [5].

Incoherence Processing. Based on the observation that incoherences between the weights W and the proxy Hessian H benefit quantization, QuIP further applies incoherence post-processing using Kronecker products of random orthogonal matrices $U \in \mathbb{R}^{a \times a}, V \in \mathbb{R}^{b \times b}$ such that $\tilde{H} \leftarrow VHV^\top, \tilde{W} \leftarrow UWV^\top$. Here $U = U_1 \otimes \cdots \otimes U_k$ and $V = V_1 \otimes \cdots \otimes V_k$.

To compute the compressed size $C(h)$ of QuIP-quantized models, we use `gzip` [11] to compress the quantized model checkpoint and obtain the term $C(h)$ as the bits required for the storage afterawrds.

E. Additional Results

E.1. Token-level Bounds via Nonlinear Parametrizations

As discussed in Appendix D.1, we experiment with LoRA in addition to the Kronecker and Monarch subspace parametrizations in order to train compressed versions of GPT2 small (124M parameters). Compared to previous work, we enhance both LoRA and SubLoRA by not only applying the low-rank decomposition to the attention layers and the linear head, but to all the fully-connected layers in the LLM. Additionally, we train all the bias and layer normalization parameters instead of keeping them fixed at their values at initialization. We also use rotary position embeddings [41] to directly encode the positional information into the LLM. Combined with our proposed token-level optimization of the label smoothing probability α, we significantly improve upon the LoRA subspace compression, as shown in Table 1. It is worth noting the LoRA alone led to vacuous BPD document-level bounds obtained by Lotfi et al. [29] while our version is non-vacuous.

Among all subspace compression strategies that we explore, Monarch without subspace leads to the tightest token-level. This result can be attributed to two key factors. Firstly, the substantial scale of our dataset, comprising 9 billion tokens, significantly changes the trade-off between the empirical risk and the compressed model size compared to previous work, since the compressed size factor in the bound is divided by the size of the dataset. Consequently, we have greater flexibility in selecting larger models that achieve an improved empirical risk. The second factor is that the Monarch approximation indeed outperforms the 2 other approximations as we increase the number of trainable parameters; this effect can be seen in Figure 1(Right). Combining the two factors, we conclude that the Monarch approximation achieves the best bounds by striking the right trade-off between empirical risk and compressed model size. This argument also explains why subspace compression does not help, as further reducing the number of trainable parameters through linear subspace projection leads to a worse trade-off between the empirical performance of the compressed model and its compressed size.

E.2. Non-vacuous Bounds for Pretrained LLMs: GPT2, LLaMA1 and LLaMA2

Intensive quantization is another way we can achieve model compression, and therefore tighter generalization bounds. We explore the setting where we only apply post-training quantization to pretrained LLMs and compute the corresponding token-level generalization bounds.

Pretrained GPT2 models. We apply the post-training quantization [28] to the publicly available GPT2 models [35] of sizes 124M (GPT2 small), 354M (GPT2 medium), and 773M (GPT2 large) parameters that were pretrained on the WebText dataset and report the numbers in Table 3. We find that GPT2 small not only yields non-vacuous bounds, but these bounds are quite comparable to those obtained using aggressive compression techniques in Table 1. GPT2 medium and large also achieve non-vacuous bounds despite having almost a billion parameters.

Training Context Length	0	1	2	4	1024
GPT2-S-Quantized	13.9	11.1	9.0	7.9	**7.6**
Markov Chain	11.3	10.5	15.3	22.4	-

Table 4. Our LLM bounds provide a much stronger statement than what would be explained by low order Markov models.

Pretrained LLaMA models. In this set of experiments, we use pretrained and pre-quantized publicly available LLaMA1, LLaMA2 and LLaMA2-Chat models and plug in their empirical risk and compressed size directly into our token-level bounds. We report the bounds obtained for 2-bit LLaMA2 in Table 2. The full set of results is reported in Table 5. The bounds are computed for the next token prediction task on the Amber dataset, which contains 1.2T tokens. We obtain non-vacuous bounds for these models despite their large scale, ranging from 7 billion to 70 billions parameters. Our experiments show that the LLaMA2-Chat models achieve worse generalization bounds as reported in Table 5 and Figure 1(Left), demonstrating that fine-tuning Chat models for dialogue use cases hurts their generalization performance on next token prediction. Although we do not know what data was used to pretrain the LLaMA models, our bounds remain valid since they do not require for the models to be trained on the same data that the empirical risk is evaluated on.

High-quality text generation. A significant limitation of document-level bounds is that the SubLoRA model achieving the best document-level bound generates un-grammatical, low-quality text as demonstrated by Lotfi et al. [29] and shown in Table 6. In contrast, our top-performing model in terms of token-level BPD bounds on the OpenWebText dataset, which is the quantized GPT2 small model, generates high-quality text, ensuring a unique combination of practical usefulness and tight guarantees on the population risk.

E.3. Contextualizing GPT2 Bounds Against Markov Chains

The best token-level bound that we achieve for BPD on the OpenWebText dataset is 7.6. But what does this value exactly mean? One might consider the possibility that our bounds are describing only the simplest components of fitting the data that exist in the model, such as the predictions of a 0th or 1st order Markov chain [31].

In Table 4 we show that this is not the case, by explicitly training a sparse k-th order Markov chain on OpenWebText and computing our token-level bounds for the result. Sweeping over different numbers of n-grams to use for the Markov chains, our bounds for these models cap out at 10.5 BPD and rapidly degrade with higher order as more statistics need to be stored. We also train and compress versions of GPT2 that are restricted to only seeing k tokens as context, mirroring the restrictions of the Markov chains. We find that for the simple 0 and 1st order Markov chains, our compression via the transformer is slightly worse. However, the LLM performs much better for higher orders.

E.4. Memorization vs. Reasoning

LLMs are capable of memorizing facts from their pretraining data, but they also can learn highly structured patterns. As we compress a model more and more, it must lose its ability to recall memorized facts, but it may still remember patterns, since they are compressible. In this section, we examine the difference between memorization and reasoning by measuring the ability of LLMs to compress structured and unstructured sequence data. To generate structured sequences, we first use short binary expression trees to generate numerical sequences of integers [16]. These sequences are highly compressible as they are generated using short and deterministic programs. To generate unstructured sequences, we collect the set of all unique integers from the structured sequences and form random sequences composed of IID samples from the set of unique integers (see Appendix F.5 for details). We train standard GPT2 models from scratch on structured and random sequences separately. In Figure 4, we show the integer prediction training accuracy with varying degrees of post-training quantization. We observe that as models are quantized more aggressively, i.e. the number of quantization levels decreases, they forget unstructured sequences far faster than structured sequences. These results parallel the findings of Jin et al. [19] who show that smaller models can retain in-context learning capabilities but lose their ability to recall facts.

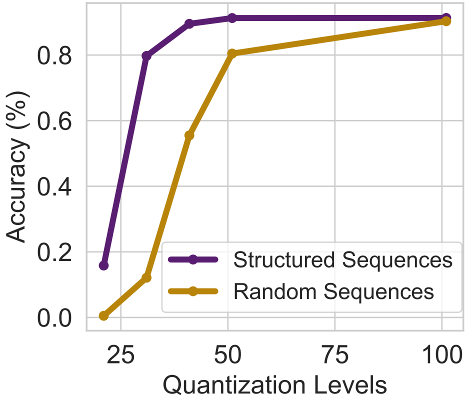

Figure 4. **As language models are compressed, they retain their understanding of patterns, but they forget highly random and unstructured data rapidly.** Experiments performed on GPT-2 models with datasets created as detailed in Appendix E.4. Compression performed via post-training quantization.

F. Experimental Details

F.1. Pretraining with Nonlinear Parametrizations

To achieve necessary model compressions for computing non-vacuous bounds, we consider GPT2 Small with 124M parameters for pretraining on OpenWebText[1] based on the nanoGPT implementation[2] [35]. We parametrize the linear layers of `CausalSelfAttention`, `MLP`, and the `LinearHead` for the GPT2 models with our nonlinear compression (LoRA, Kronecker, Monarch), where we turn bias to be on except for `LinearHead`. For LoRA and Kronecker, we turn on weight-tying between the token embedding and the final `LinearHead` parametrized by nonlinear compressions. For LoRA and Kronecker, we train the layer norm parameters in addition to all of the nonlinear compressions applied to the linear layers. For Monarch, we only train the linear layers parametrized by Monarch matrices. For all pretraining runs, we use a batch size of 8, a sequence length of 1024, and a standard AdamW optimizer [27] with a learning rate of 0.0002. We perform learning rate warm-up for 500 iterations, and we apply rotary embedding [41] to all three nonlinear parametrizations.

F.1.1. Hyperparameter Sweeps for LoRA

LoRA. We sweep over rank $r \in \{1, 4, 16, 32, 64, 128, 256\}$.

We choose a learning rate of 0.0002 with a LoRA dropout value of 0.1 and LoRA alpha value of 32.

SubLoRA. We report the rank r and corresponding subspace dimension that we sweep over for SubLoRA in Table 7.

F.1.2. Hyperparameter Sweeps for Kronecker

For Kronecker $W = A \otimes B$, we factorize the A, B matrices with $A \in \mathbb{R}^{a_1 \times b_1}, B \in \mathbb{R}^{a_2 \times b_2}$ for $a_1 a_2 = a, b_1 b_2 = b$. We sweep over all possible combinations of $\{a_1, a_2\}$ and $\{b_1, b_2\}$ by performing prime factorizations with multiplicity on the numbers a, b and enumerating all possible combinations. All of our jobs use a learning rate of 0.0002.

F.1.3. Hyperparameter Sweeps for Monarch

For Monarch, we relax the restrictions for the number of blocks to be strictly $\sqrt{a}$ and instead by a number divisible by a to sweep over different numbers of blocks. We also include runs for Monarch where we're using absolute position encodings and runs where we're only applying the Monarch matrices factorizations to the attention layers and the linear classification heads.

[1] http://Skylion007.github.io/OpenWebTextCorpus

[2] https://github.com/karpathy/nanoGPT

Model	Bits per Dimension	Top-1 Error (%)	Top-10 Error (%)	Top-100 Error (%)
2 bits				
LLaMA1-7B	4.291	48.08	22.82	12.83
LLaMA1-13B	4.598	48.87	24.23	14.59
LLaMA1-30B	5.373	52.91	28.06	19.14
LLaMA1-65B	6.100	56.63	32.29	24.14
LLaMA2-7B	**4.282**	47.55	**22.48**	**12.56**
LLaMA2-Chat-7B	4.536	49.10	24.18	13.50
LLaMA2-13B	4.515	47.85	23.54	14.44
LLaMA2-Chat-13B	4.764	49.82	24.95	15.10
LLaMA2-70B	6.140	56.24	32.61	24.32
LLaMA2-Chat-70B	6.396	58.26	34.16	25.04
3 bits				
LLaMA1-7B	4.371	47.42	22.87	13.63
LLaMA1-13B	4.801	48.97	25.23	16.14
LLaMA1-30B	5.694	53.54	29.91	21.63
LLaMA1-65B	6.728	59.56	36.14	28.08
LLaMA2-7B	4.351	**47.15**	22.75	13.62
LLaMA2-Chat-7B	4.648	48.84	24.23	14.24
LLaMA2-13B	4.754	48.45	24.67	15.95
LLaMA2-Chat-13B	5.056	50.90	26.26	16.66
LLaMA2-70B	6.772	59.35	36.27	28.56
LLaMA2-Chat-70B	7.081	61.66	38.00	29.30
4 bits				
LLaMA1-7B	4.502	47.52	23.53	14.52
LLaMA1-13B	5.023	49.96	26.46	17.47
LLaMA1-30B	6.054	55.55	32.09	23.93
LLaMA1-65B	7.269	62.56	39.38	31.54
LLaMA2-7B	4.493	47.64	23.64	14.53
LLaMA2-Chat-7B	4.825	49.49	25.15	15.12
LLaMA2-13B	4.959	49.46	25.67	17.21
LLaMA2-Chat-13B	5.271	51.61	27.23	18.12
LLaMA2-70B	7.331	62.53	39.89	32.11
LLaMA2-Chat-70B	7.679	65.32	41.59	32.87
Random Guess	14.965	99.99	99.96	99.68

Table 5. Token-level generalization bounds for open-source pretrained LLM checkpoint on the Amber dataset. All of these models were quantized post-training using QuIP# to different numbers of bits as shown above. All the bounds are non-vacuous compared to random guess performance.

F.2. Quantization

Quantization Following Lotfi et al. [28], we apply quantization over the weights from the subspace training or the trainable LoRA, Kronecker, Monarch parameters along with layer norm weights depending on the compression setup.

Experiments on QuIP-quantized Models. We compute token-level bounds on pretrained LLaMA1 and LLaMA2 models [42] quantized with QuIP with publicly-available checkpoint [38]. Although we do not know what data was used to pre-train these models, we can evaluate the generalization bound on the Amber dataset and consider other tokens used in training as a data-dependent prior.

F.3. Bounds Evaluation

In the sequence of text, we use end of text tokens (EOT) which separate the documents. In this way, we can consider concatenating many documents together to form one long sequence. As a result of the EOT tokens and the structure of the text, the distribution $p(x_i|x_{<i})$ can be simplified into $p(x_i|x_k, x_{k+1}, \dots x_{i-1})$ where k is the index of the most recent EOT token because the documents are sampled independently. In the evaluation of the LLM we likewise have no dependence on tokens outside the given document in question.

	Generated Text
GPT2 (124M) Quantized (BPD Bound: 7.61)	The study, published in Proceedings of the National Academy of Sciences, examined the relationships between brain activity, gene expression and inflammation in diseases including Alzheimer's disease, dementia, Parkinson's disease, glioblastoma and Alzheimer's disease. "Our study demonstrates that omega-3 fatty acids play a role in the link between inflammation and brain function," said lead author Dr Richard Collins, PhD, of Duke University's Duke Center for Bioethomics and Bioengineering. After controlling for.
GPT2 (124M) SubLoRA [29]	th he the startedt at its,, the a more be power and- by. S and, of of -'s on. The UK I The, are the on the the under, but the then the day,. The. The. It for the! a,. M an they first the the speak have times. cover that (illegal In the day where I The who when and $ In We ∷[{∵ As she I WeP spirituality. The all And one which a more says thought the other (ed 15: And P It as/ T - 2 But We The The theah It who the full of that to was 'The they (It As We A and each (. The It - We The M I"

Table 6. Examples of generated text from the GPT2 small quantized model that achieves the best token-level bounds compared to the SubLoRA-pretrained GPT2 small model in Lotfi et al. [29]. In contrast to the text generated by the best performing model in terms of BPD bounds by Lotfi et al. [29], our quantized GPT2 small generates significantly higher-quality text while simultaneously achieving the best BPD and Top-$1/10/100$ error bounds.

Rank r	Subspace Dimension d
1	25000
4	50000
8	50000
16	50000
32	10000, 750000
64	25000, 2000000
128	7000000, 15000000

Table 7. Hyperparameter sweep for SubLoRA. For all runs, we used a learning rate of 0.0002, a LoRA dropout value of 0.1, and a LoRA alpha value of 32.

To compute token-level bounds, we evaluate all of our generalization bounds with failure probability $\delta = 0.05$, subsample size of $n = 10\text{k}$ on the OpenWebText training dataset of size $m = 9\text{B}$ tokens.

F.4. Correlation with Downstream Performance

We retrieve the downstream task performance of difference GPT2 variants ranging in scale from 117M to 1.5B averaged over the downstream datasets as shown in Table 8. To obtain an approximation of the conditional BPD expectation that we bound in Equation (2), we resample x_i from a LLaMA2-7B given fixed training contexts $x_{<i}$ from the Amber dataset. We use a sample size equal to $10,000$ samples.

F.5. Memorization Experiment

Following Goldblum et al. [16], we picked a complexity value of 4 and sequence length of 30 and generated 984 sequences as the training dataset. IID sampled elements from a uniform distribution over the set of unique integers from the structured

Model Size	LAMBADA (PPL)	LAMBADA (ACC)	CBT-CN (ACC)	CBT-NE (ACC)	WikiText2 (PPL)	PTB (PPL)	1BW (PPL)
117M	35.13	45.99	87.65	83.4	29.41	65.85	75.20
345M	15.60	55.48	92.35	87.1	22.76	47.33	55.72
762M	10.87	60.12	93.45	88.0	19.93	40.31	44.575
1542M	8.63	63.24	93.30	89.05	18.34	35.76	42.16

Table 8. Zero-shot downstream task performance for GPT2 models with different model sizes as reported in Radford et al. [35].

sequences are used to build the baseline random sequences dataset. Our vocabulary size is 12 as we only have integers, beginning of text token, and an additional delimiter token. The delimiter token are placed between every distinct integers during our tokenization process. We selected a GPT-2 Small model with 124M parameters and trained on the structured and random sequences separately with a learning rate of 0.0001 for 1000 epochs. Our accuracy evaluations are performed on the single integer levels. Our quantization procedure is the same as described in Appendix F.2. We show the results for this experiment in Figure 4.

F.6. Amber Dataset

We used a subset of the pretraining dataset for Amber 7B LLM [26] for our bound evaluations. This dataset contains RedPajama V1 [7] (arxiv, C4, GitHub, StackExchange, Wikipedia), StarCoder [23] (The Stack), RefinedWeb [33] (CommonCrawl) with around 1.2 trillion tokens. We tokenize the entire dataset using a LLaMA tokenizer and then sample tokens from a uniform distribution over the tokenized dataset.