AAAI

Mechanism for feature learning in neural networks and backpropagation-free machine learning models

Adityanarayanan Radhakrishnan1,2‡, Daniel Beaglehole3‡, Parthe Pandit4,5, Mikhail Belkin5,3*

1 Harvard School of Engineering and Applied Sciences, Cambridge, MA 02138, USA

2 Broad Institute of MIT and Harvard, Cambridge, MA 02142, USA

3 Computer Science and Engineering, UC San Diego, La Jolla, CA 92093, USA

4 Center for Machine Intelligence and Data Science, IIT Bombay, Mumbai 400076, India

5 Halıcıoğlu Data Science Institute, UC San Diego, La Jolla, CA 92093, USA

*** Corresponding author. Email: mbelkin@ucsd.edu**

‡ These authors contributed equally to this work.

ABSTRACT Understanding how neural networks learn features, or relevant patterns in data, for prediction is necessary for their reliable use in technological and scientific applications. In this work, we presented a unifying mathematical mechanism, known as average gradient outer product (AGOP), that characterized feature learning in neural networks. We provided empirical evidence that AGOP captured features learned by various neural network architectures, including transformer-based language models, convolutional networks, multilayer perceptrons, and recurrent neural networks. Moreover, we demonstrated that AGOP, which is backpropagation-free, enabled feature learning in machine learning models, such as kernel machines, that a priori could not identify task-specific features. Overall, we established a fundamental mechanism that captured feature learning in neural networks and enabled feature learning in general machine learning models.

INDEX TERMS Neural networks, feature learning, average gradient outer product, kernel machines, recursive feature machines, backpropagation-free learning.

I. INTRODUCTION

Neural networks have been central to major breakthroughs in biology (1), computer vision (2), and language understanding and generation (3). Yet, a major challenge for their deployment in scientific or technological applications is the limited understanding of how these models learn and use features, or relevant patterns in data, for prediction. Without understanding neural feature learning, it is difficult to establish whether neural networks produce reliable, accurate, and appropriate responses. Given the rapidly expanding reach of neural network applications, it is crucial to characterize the mechanism of neural feature learning.

The ability of neural networks to learn features from data is thought to be a central contributor to their improved effectiveness over classical machine learning models (4, 5). Despite active research effort into neural feature learning, a unified mechanism that captures features learned across neural architectures had not been identified by prior work. Most prior work on neural feature learning analyzed feature learning in fully connected networks on specific learning tasks, e.g., learning single-index functions (6, 7), multi-index functions (8, 9), or learning parity functions (4, 10). Recent work (11) analyzed feature learning in deep, nonlinear fully connected networks that fit data and minimized representation cost (given by the sum of squares of network weights). Works (5, 12–14) empirically identified specific classification and regression tasks where fully connected and convolutional neural networks outperformed non–feature learning, kernel methods. The work (12) proved that two-layer convolutional networks outperformed non–feature learning, convolutional kernel methods on tasks where relevant features for prediction are embedded in noisy background. The goal of our work was to identify a single, unifying mechanism that captured features learned across neural architectures, including effective models used in practice. Such a unifying mechanism would open new avenues for advancing interpretability of neural networks and enabling feature learning in machine learning models that cannot natively learn task-relevant features, such as kernel machines.

In this work, we presented a unifying mechanism for feature learning that captured features learned across neural network architectures and enabled feature learning with general machine learning models. We posited the neural feature ansatz (NFA), which stated that features extracted by a given neural network layer are proportional to the average gradient outer product (AGOP) with respect to the input to this layer. We presented extensive empirical evidence that our ansatz holds for state-of-the-art models such as vision transformers (15), deep convolutional networks [e.g., ResNet (2), VGG (16), AlexNet (17)], and transformer-based language models from the GPT2 family (3). We demonstrated that the AGOP of GPT2-based transformers recovered thematically related groups of tokens and that the AGOP of convolutional networks recovered features related to edge detection.

By definition, AGOP is a backpropagation-free, data-dependent mathematical operator that can be applied to any predictive model. Thus, we used AGOP to enable feature learning in general machine learning models through an algorithm we called Recursive Feature Machine (RFM). As a model system, we used AGOP to enable feature learning with kernel machines (18), a classical machine learning model that has received renewed interest (19–21). We showed that AGOP of standard kernel machines and convolutional kernel machines (20) recovered relevant, task-specific features. For standard kernel machines operating on vectorized data, these features were transformations of input features that were useful for prediction. For convolutional kernel machines trained on standard image classification tasks, we observed that AGOP with respect to image patches recovered edge detectors, a phenomenon that had previously been observed for convolutional networks (22). Moreover, we showed that reintroducing features captured by AGOP of kernel machines improved performance of these models, thereby confirming that AGOP captured features inherent in data that were relevant for prediction.

Overall, we concluded that AGOP is a unifying mechanism for feature learning that can be used to characterize features learned by neural network architectures and enable feature learning in general machine learning models.

II. RESULTS

Neural networks are nonlinear models that implement functions of the form $f: \mathbb{R}^{d\times t} \to \mathbb{R}^{c\times t'}$ where d denotes the dimensionality of input features, c denotes the dimensionality of labels, and t, t' respectively refer to input and label sequence length for sequence models such as transformers or recurrent networks. These models are recursively defined as

$$\begin{cases} h^1(x) = x; \\ h^{l+1}(x) = \tilde{f}_l(h^l(x)) \quad \text{for} l \in \{1, ..., L\} \end{cases} \tag{1}$$

where $h^1(x) = x$, $h^l(x) \in \mathbb{R}^{d_l \times t_l}$ are the representations at the l -th layer of the network on the input x , $L \in \mathbb{Z}_+$ denotes the depth of the network, and $f^l: \mathbb{R}^{d_l \times t_l} \to \mathbb{R}^{d_{l+1} \times t_{l+1}}$ is parameterized with weight matrices $\{W_i^{(l)}\}_{i=1}^k$ as a multivariate function of the form

$$\tilde{f}_l(h^l(x)) = \tilde{f}_l(W_1^{(l)}u_{11}^{(l)}, W_1^{(l)}u_{12}^{(l)}, ..., W_k^{(l)}u_{k(m-1)}^{(l)}, W_k^{(l)}u_{km}^{(l)})] \tag{2}$$

where $u_{ij}^{(l)}$ denote subsets of entries of $h^l(x)$ (e.g., patches of images in convolutional networks or token embeddings in transformers). By adjusting values of k , m , and $\tilde{f}_l$, Eq. 2 can be used to recover standard architectures such as fully connected networks [also known as multilayer perceptrons (MLPs)], convolutional networks (CNNs), recurrent neural networks (RNNs), and transformers.

To characterize feature learning in neural networks, we used Eq. 2 to understand how weight matrices are involved in transforming hidden units. Equation 2 implies that the features extracted by the weights in layer l of a neural network are characterized by right singular vectors and singular values of $\{W_i^{(l)}\}_{i=1}^k$, because these terms rotate, reflect, and rescale components of the input into this layer, $h^l(x)$. These singular vectors and values can be recovered from eigenvectors and eigenvalues of the matrices $W_i^{(l)T}W_i^{(l)}$, which we refer to as neural feature matrices (NFMs). Suppose the neural network is trained on a dataset $\{(x^{(p)}, y^{(p)})\}_{p=1}^n$. To understand neural feature learning, we aimed to characterize the structure of NFMs at the end of training.

Our main finding was the NFA, which connected NFMs at any intermediate layer of a trained network, $\hat{f}$, to its AGOP as follows:

$$\sum_{p=1}^{n}\sum_{j=1}^{m} \nabla_{u_{ij}^{(l)}}\hat{f}(x^{(p)}) \cdot \nabla_{u_{ij}^{(l)}}\hat{f}(x^{(p)})^T \propto (W_i^{(l)T}W_i^{(l)})^a \tag{3}$$

where $\propto$ denotes elementwise proportionality, the power a > 0 applies to the matrix, and $\nabla_{u_{ij}^{(l)}}\hat{f}(x^{(p)})$ denotes the gradient of $\hat{f}$ with respect to the input $u_{ij}^{(l)}$ of $\tilde{f}_l$. For networks with multivariate outputs, we used the Jacobian of the resulting model $\hat{f}$, which is the natural extension of the gradient. Note that a affects only the magnitude of the AGOP eigenvalues and does not affect the order of eigenvalues nor the eigenvectors. The eigenvectors

capture the task-relevant directions used for identifying features, which was the focus of our work. In supplementary text A, we proved that the ansatz with $a = \frac{1}{2}$ holds with equality when training two-layer linear networks under balanced initialization. Motivated by this theoretical insight, we chose $a = \frac{1}{2}$ for all empirical results involving neural networks. Before presenting empirical evidence for the NFA, we provide intuition for the NFA and discuss implications.

A. Intuition for the ansatz and implications

To build effective predictors on high-dimensional data, it is important to identify features that are relevant for prediction. A natural approach to identifying important features is to distinguish between directions along which the predictor varies most and least. When applied to a predictor, AGOP mathematically identifies task-relevant directions, with top eigenvectors corresponding to the most relevant directions and bottom eigenvectors corresponding to least relevant directions. Thus, our ansatz explicates an internal mechanism in neural networks that identifies task-relevant directions. Although such a mechanism is absent in standard machine learning models, including kernel machines, our result implies that AGOP can be used to enable task-relevant feature learning in these standard models operating on general data modalities.

B. Empirical evidence for the neural feature ansatz

We presented empirical evidence for the NFA across a wide variety of neural network architectures, including those commonly used in practice. We stratified our empirical evidence on the basis of architecture type: transformers (15, 23), CNNs (2, 16, 17), MLPs, and RNNs (24). We showed for each architecture that mean Pearson correlation between AGOP and NFM across all layers in trained networks was high (i.e., > 0.8 in all cases). In materials and methods and supplementary figures, we showed that this correlation was considerably higher than (i) correlation between AGOP and NFM in untrained networks and (ii) correlation between the NFM in trained models and NFM in untrained, randomly initialized models.

Given the computational simplicity of training MLPs over transformers, CNNs, and RNNs, we analyzed how activation function choice, width (defined as number of hidden units per layer), and initialization scheme affected correlation between NFM after training and AGOP in MLPs. In this setting, we found that the correlation between NFMs and AGOP was higher in situations where previous work identified benefits of feature learning, e.g., training low-width networks with default initialization (14) or training high-width networks under small initialization (5). Lastly, in supplementary text B and supplementary figures, we showed that AGOP accurately captured spurious features learned by MLPs.

C. Characterization of features identified by AGOP

Thus far, we established that NFMs after training were correlated with AGOP across various neural network architectures. We now demonstrate that AGOPs of trained networks capture features in data that are relevant for prediction. In the case of GPT2-architecture language models, AGOPs identified related groups of tokens and in the case of convolutional networks, AGOP identified edge detectors.

AGOP recovered groups of related tokens from GPT2-architecture language models

Recent work (25) demonstrated that projecting the embedding of tokens onto the right singular vectors of value matrices in GPT2 led to identification of groups of related tokens. The NFA stated that eigenvectors of AGOP were equal (up to sign) to right singular vectors of weight matrices in neural networks. Thus, the NFA implied that eigenvectors of AGOP were an alternative mechanism for capturing such related tokens. To this end, we projected token embeddings from a GPT2-architecture transformer trained on TinyStories (26) onto the eigenvectors of value-AGOPs and extracted top-weighted tokens. Our analysis indicated that eigenvectors of value-AGOPs recovered related token groups. We presented the top five eigenvectors of value-AGOPs resulting in the most thematically related token groups according to ratings by GPT4. We further demonstrated how value-AGOP eigenvectors could be used to highlight related tokens in groups of text. We presented samples of context length 256 from TinyStories analyzed by their inner product with value-AGOP eigenvectors corresponding to (i) food and cooking terms and (ii) names and animals. Overall, these results showed that AGOP captured text features learned by language models.

AGOP recovered edge detectors from CNNs

Previous work (22) presented evidence that early layers of trained CNNs captured features akin to edge detectors. Given that the covariances of these filters were correlated with AGOP over image patches (patch-AGOP), eigenvectors of AGOP should capture similar features. We visualized the features of ImageNet images captured by AGOP of VGG19 layers. In particular, we observed that the patch-AGOP computed with respect to the input of the first convolutional layer directly acted as an edge detector. Moreover, we observed that patch-AGOP computed with respect to input of deeper layers of VGG19 amplified specific regions of images for prediction.

For example, we observed that the eyes of dogs were used as predominant features for classification, and pixels corresponding to ladybugs were predominantly used for classification. We demonstrated that eigenvectors of patch-AGOP computed for the first layer of VGG19 contained horizontal and vertical edge detectors. Furthermore, our visualization was markedly different from previous approaches using class activation mappings (28). These prior works attempted to identify features in individual images, and we identified a single operator per layer that was used to select features across all images. Overall, these results showed that AGOP captures relevant image features used by CNNs for classification.

III. AGOP ENABLED FEATURE LEARNING WITH GENERAL MACHINE LEARNING MODELS

A powerful outcome of the ansatz was that it decoupled feature extraction from choice of machine learning model. Because AGOP is an operator that can be applied independently of the model choice, it serves as a universal mechanism for enabling feature learning in machine learning models, including those that a priori could not learn task-specific features. We demonstrated the effectiveness of AGOP as a feature learning mechanism by using it to enable feature learning in kernel machines, a well-studied class of machine learning models that does not adaptively extract task-specific features.

A. AGOP of kernel machines operating on tabular data extracted relevant features for prediction

We analyzed features identified by AGOP of kernel machines operating on tabular (vectorized) data. We trained kernel machines using the Laplace kernel of the form and then computed the AGOP of the trained kernel machine. We visualized the top eigenvector of the AGOP for such kernel machines trained on prediction tasks from CelebA (29), using vectorized images. Even though kernel machines rely on a fixed kernel function regardless of the task, we observed that AGOP of trained kernel machines identified task-specific features. Moreover, in supplementary materials, we observed that these AGOP eigenvectors are similar (cosine similarity of 0.99) to the eigenvectors of first-layer NFMs in MLPs trained on the same tasks, even though these are *different* classes of models.

B. Reintroducing features identified by AGOP of kernel machines improved model performance: Recursive feature machines

The fact that AGOP identifies task-specific features provides an opportunity to improve kernel machines. To this end, letting M denote the AGOP of a trained kernel machine, we reincorporated such features into the kernel function by considering $K(M^{1/2}x, M^{1/2}z)$. Moreover, we iterated between solving kernel regression, computing AGOP of the trained kernel machine, and reintroducing AGOP features. We refer to this iterative approach as a Recursive Feature Machine (RFM). We demonstrated that reintroduction of AGOP features drastically improved performance of kernel machines [8% increase in classification accuracy on Street-View-House-Numbers (SVHN) and a double in the test R^2 on synthetic regression tasks from (8, 14)]. We provided a comprehensive analysis demonstrating the benefit of reintroducing AGOP features into kernel machines across 121 tabular datasets from (30). AGOP features improved overall average accuracy of kernel machines across the 121 tasks with up to a 31% increase in accuracy over kernel regression with the Laplace kernel. Moreover, we observed that reintroducing features led to better performance primarily on larger datasets. Indeed, the only points for which feature learning was worse was for smaller datasets, e.g., those with fewer than 1000 samples. We showed performance of RFM in relation to 181 other models on the tabular data tasks from (30) and against all other models and tasks from (31). Moreover, we found that RFM required only a fraction of the computational cost of training MLPs. Computational complexity of computing AGOP is presented in materials and methods. All evaluation metrics are described in materials and methods.

C. AGOP of convolutional kernel machines extracted features on image patches

Analogously to the case of using AGOP to enable feature learning in standard kernel machines, we used patch-AGOP to enable feature learning in convolutional kernel machines. These convolutional kernels, such as the Convolutional Neural Tangent Kernel (CNTK) (20), can be viewed as functions on image patches. We thus trained convolutional kernel machines and then computed patch-AGOP to identify relevant features.

We presented examples of features learned by convolutional kernels on synthetic and real-world image datasets. We began with the illustrative synthetic task of classifying whether a noisy image contains a star pattern. Prior work (12) showed that CNNs were able to far outperform kernel machines in such synthetic settings and claimed that CNNs were able to amplify signal and reduce noise in such images in contrast to kernel machines. After training CNTK on 50 samples from this synthetic dataset, we computed the patch-AGOP, M . We observed that eigenvectors of M corresponded to patch transformations that highlight relevant pixels, indicating that the AGOP captured features relevant for this specific prediction problem. We multiplied image patches by $M^{1/2}$ and saw that such features amplified signal corresponding to the star pattern and reduced noise.

We visualized eigenvectors of AGOP for convolutional kernel machines trained on SVHN using K(x,z) corresponding to CNTK with a patch size of 3×3 . We observed that AGOP eigenvectors of the trained convolutional kernel machine corresponded to edge detectors. We demonstrated that patch-AGOPs of these models trained on SVHN automatically identified edges in images of different resolution and domains such as ImageNet. Lastly, we demonstrated that reintroducing the patch-AGOP into convolutional kernel machines led to improvements in performance across various synthetic and real-world image classification tasks. Consistent improvement across various tasks confirmed that patch-AGOP of convolutional kernels identifies relevant features for image classification.

IV. REMARKS

Our results can be extended by recursively applying AGOP to capture features on nonlinearly transformed versions of input data (see supplementary text D). We found that such deep, nonlinear feature learning through AGOP, a process we call Deep RFM, improved performance of convolutional kernel machines, closing the gap between kernel machines and neural networks. We additionally showed that AGOP could be effectively used in conjunction with alternative representations of data to substantially increase model performance. In supplementary text E and supplementary materials, we used AGOP to improve performance of kernel machine image classifiers trained on scattering transformed images (32). Lastly, as AGOP captured task-specific features on representations of data, it could be additionally used to perform transfer learning between tasks that required similar subsets of features (see supplementary text F and supplementary figures).

V. DISCUSSION

A. *Advancing transparency in deep neural networks*

Given the broad reach of deep learning today, it is crucial to understand how these models learn and use features to make predictions. Precise characterization of neural feature learning is a key step toward building deep learning systems that are demonstrably safe and aligned with human preferences (33). Our results advanced the transparency of neural networks by providing a single mechanism that could be used to characterize and visualize features extracted across various neural architectures operating on tabular, image, and text data. Thus, we envision that our results will be a key step toward building transparent and interpretable machine learning models.

B. *New avenues for building effective machine learning models*

Identification of a small set of task-relevant features from a much larger set of features is a key step for building effective, interpretable predictive models. Approaches for task-relevant feature selection have been developed and studied in classical machine learning literature, including works for feature selection in linear models (34) and nonlinear models in specific tabular data settings (e.g., multi-index models) (35–37). Prior to our work, it was unclear how to disentangle neural feature learning from predictor construction because both occur simultaneously when training deep networks using backpropagation. Thus, our work presented an important foundational advancement in framing neural feature learning from the classical machine learning perspective. We believe our findings offer an opportunity to improve current neural architectures by making them more computationally efficient, theoretically grounded, and interpretable.

ACKNOWLEDGMENT

A.R. thanks A. Philippakis for feedback on the paper, C. Uhler for support on this work, and E. Lander for discussions on transformers. We thank D. Hsu, J. Tropp, and J. Lee for valuable literature references. Funding: A.R. is currently supported by the George F. Carrier postdoctoral fellowship in applied mathematics at Harvard School of Engineering and Applied Sciences (SEAS) and was supported by the Eric and Wendy Schmidt Center at the Broad Institute of MIT and Harvard at the time of submission. P.P. is currently supported by a Thakur Family Chair at IIT Bombay and was supported by a Simons-HDSI postdoctoral fellowship at UCSD at the time of submission. We acknowledge support from the National Science Foundation (NSF) and the Simons Foundation for the Collaboration on the Theoretical Foundations of Deep Learning through awards DMS-2031883 and #814639 as well as the TILOS institute (NSF CCF-2112665). This work used the programs XSEDE (Extreme science and engineering discovery environment), which is supported by NSF grant nos. ACI-1548562, and ACCESS (Advanced cyberinfrastructure coordination ecosystem: services and support), which is supported by NSF grant nos. 2138259, 2138286, 2138307, 2137603, and 2138296. Specifically, we used the resources from SDSC Expanse GPU compute nodes, and NCSA Delta system, via allocations TG-CIS220009.
Author contributions: A.R., D.B., P.P., and M.B. designed and performed the research, analyzed the data, and wrote the paper. Competing interests: None declared. Data and materials availability: All data needed to evaluate the conclusions in the paper are present in the paper or the supplementary materials. All image datasets considered in this work, i.e., CelebA, SVHN, CIFAR10, CIFAR100, GTSRB, MNIST, STL-10, PCAM, Caltech101, DTD, QMNIST, and EMNIST, are publicly available for download via PyTorch. ImageNet32 and ImageNet are publicly available for download through the link

to the ImageNet portal provided in (27). TinyStories is publicly available for download through HuggingFace (38). Shakespeare data are provided directly from (39). Tabular data from (30) is publicly available through the link to code provided in (40). Tabular data from (31) are publicly available through the link to code provided in their paper. All code for (i) computing correlation between AGOP and NFM for various architectures (transformers, CNNs, RNNs, MLPs); (ii) visualizing AGOP features in CNNs and transformer-based language models of the GPT2-family; and (iii) training RFMs on tabular and image data are available at (41).

REFERENCES

[1] K. Tunyasuvunakool et al., Nature 596, 590–596 (2021).

[2] K. He, X. Zhang, S. Ren, J. Sun, "Proceedings of the 2016 IEEE conference on computer vision and pattern recognition" (2016), pp. 770–778.

[3] A. Radford et al., Language models are unsupervised multitask learners. OpenAI blog 1 (8), 9 (2019).

[4] Z. Shi, J. Wei, Y. Lian, "Proceedings of the 2022 International Conference on Learning Representations" (2022).

[5] G. Yang, E. J. Hu, "Tensor Programs IV: Feature learning in infinite-width neural networks" in Proceedings of the 2021 International Conference on Machine Learning (PMLR 2021), pp. 11727–11737.

[6] A. Bietti, J. Bruna, C. Sanford, M. J. Song, in Advances in Neural Information Processing Systems (Curran Associates, 2022), pp. 9768–9783.

[7] J. Ba et al., "High-dimensional asymptotics of feature learning: How one gradient step improves the representation" in Advances in Neural Information Processing Systems (Curran Associates, 2022), pp. 37932–37946.

[8] A. Damian, J. Lee, M. Soltanolkotabi, "Conference on Learning Theory" (PMLR, 2022), pp. 5413–5452.

[9] E. Abbe, E. Boix-Adserà, T. Misiakiewicz, "Conference on Learning Theory" (PMLR, 2022), pp. 4782–4887.

[10] A. Daniely, E. Malach, in Advances in Neural Information Processing Systems (Curran Associates, 2020), pp. 20356–20365.

[11] A. Jacot, in Advances in Neural Information Processing Systems (Curran Associates, 2023).

[12] S. Karp, E. Winston, Y. Li, A. Singh, in Advances in Neural Information Processing Systems (Curran Associates, 2021), pp. 24883–24897.

[13] P. M. Long, Properties of the after kernel. arXiv:2105.10585 [cs.LG] (2021).

[14] N. Vyas, Y. Bansal, P. Nakkiran, Limitations of the ntk for understanding generalization in deep learning. arXiv:2206.10012 [cs.LG] (2022).

[15] A. Kolesnikov et al., "Proceedings of the 2021 International Conference on Learning Representations" (2021).

[16] K. Simonyan, A. Zisserman, "Proceedings of the 2015 International Conference on Learning Representations" (2015).

[17] A. Krizhevsky, I. Sutskever, G. E. Hinton, in Advances in Neural Information Processing Systems (Curran Associates, 2012), pp. 1097–1105.

[18] B. Schölkopf, A. J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (MIT Press, 2002).

[19] A. Jacot, F. Gabriel, C. Hongler, in Advances in Neural Information Processing Systems (Curran Associates, 2018), pp. 8571–8580.

[20] S. Arora et al., "On exact computation with an infinitely wide neural net" in Advances in Neural Information Processing Systems (Curran Associates, 2019), pp. 8141–8150.

[21] R. Novak et al., "Proceedings of the 2020 International Conference on Learning Representations" (2020).

[22] M. D. Zeiler, R. Fergus, "Proceedings of the 2014 European Conference on Computer Vision" (Springer, 2014), pp. 818–833.

[23] A. Vaswani et al., "Attention is all you need" in Advances in Neural Information Processing Systems (Curran Associates, 2017), pp. 5998–6008.

[24] K. Cho et al., "Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing" (Association for Computational Linguistics, 2014), pp. 1724–1734.

[25] G. Dar, M. Geva, A. Gupta, J. Berant, Analyzing transformers in embedding space. arXiv:2209.02535 [cs.CL] (2022).

[26] R. Eldan, Y. Li, Tinystories: How small can language models be and still speak coherent english? arXiv:2305.07759 [cs.CL] (2023).

[27] O. Russakovsky et al., Int. J. Comput. Vis. 115, 211–252 (2015).

[28] R. R. Selvaraju et al., "Proceedings of the 2017 International Conference on Computer Vision" (IEEE, 2017), pp. 618–626.

[29] Z. Liu, P. Luo, X. Wang, X. Tang, "Proceedings of the 2015 International Conference on Computer Vision" (IEEE, 2015), pp. 3730–3738.

[30] M. Fernández-Delgado, E. Cernadas, S. Barro, D. Amorim, J. Mach. Learn. Res. 15, 3133–3181 (2014).

[31] L. Grinsztajn, E. Oyallon, G. Varoquaux, in Advances in Neural Information Processing Systems Datasets and Benchmarks (2022), pp. 1–48.

[32] J. Bruna, S. Mallat, IEEE Trans. Pattern Anal. Mach. Intell. 35, 1872–1886 (2013).

[33] L. Ouyang et al., "Training language models to follow instructions with human feedback" in Advances in Neural Information Processing Systems (Curran Associates, 2022), pp. 27730–27744.

[34] R. Tibshirani, J. R. Stat. Soc. B 58, 267–288 (1996).

[35] S. Trivedi, J. Wang, S. Kpotufe, G. Shakhnarovich, in Uncertainty in Artificial Intelligence (Morgan Kaufmann, 2014), pp. 819–828.

[36] W. Härdle, T. M. Stoker, J. Am. Stat. Assoc. 84, 986–995 (1989).

[37] S. Mukherjee, D.-X. Zhou, J. Shawe-Taylor, J. Mach. Learn. Res. 7, 519–549 (2006).

[38] T. Wolf et al., "Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations" (Association for Computational Linguistics, 2020), pp. 38–45.

[39] A. Karpathy, nanogpt (2022); https://github.com/karpathy/nanoGPT.

[40] S. Arora et al., "Proceedings of the 2020 International Conference on Learning Representations" (2020).

[41] A. Radhakrishnan, D. Beaglehole, P. Pandit, M. Belkin, version 1, agop_feature_learning (2024); https://doi.org/10.5281/zenodo.10676950.

[42] Y. Netzer et al., "Reading digits in natural images with unsupervised feature learning" in Advances in Neural Information Processing Systems Workshop on Deep Learning and Unsupervised Feature Learning (2011).