FOUNDATION MODEL'S EMBEDDED REPRESENTATIONS MAY DETECT DISTRIBUTION SHIFT

Max Vargas[1,*] **Adam Tsou**[1,2,*] **Andrew Engel**[1] **Tony Chiang**[1,3,4]
[1]Pacific Northwest National Laboratory [2]Stony Brook University
[3]University of Washington [4]University of Texas, El Paso
{max.vargas,andrew.engel,tony.chiang}@pnnl.gov
{adam.tsou}@stonybrook.edu
⋆ Equal Contribution

ABSTRACT

Sampling biases can cause distribution shifts between train and test datasets for supervised learning tasks, obscuring our ability to understand the generalization capacity of a model. This is especially important considering the wide adoption of pre-trained foundational neural networks — whose behavior remains poorly understood — for transfer learning (TL) tasks. We present a case study for TL on the Sentiment140 dataset and show that many pre-trained foundation models encode different representations of Sentiment140's manually curated test set M from the automatically labeled training set P, confirming that a distribution shift has occurred. We argue training on P and measuring performance on M is a biased measure of generalization. Experiments on pre-trained GPT-2 show that the features learnable from P do not improve (and in fact hamper) performance on M. Linear probes on pre-trained GPT-2's representations are robust and may even outperform overall fine-tuning, implying a fundamental importance for discerning distribution shift in train/test splits for model interpretation.

1 INTRODUCTION

Foundation models Brown et al. (2020); Touvron et al. (2023); Liu et al. (2023); Rombach et al. (2021) have quickly integrated themselves into the standard machine learning development stack, in particular for their adaptability to specialized tasks Zhai et al. (2023); Wang et al. (2020). Under the umbrella of transfer learning (TL), this specialization is often performed through fine-tuning after pre-training on a diverse corpus (Penedo et al., 2023), which is believed to result in highly generalizable feature representations Liu et al. (2023). While a variety of open-source foundation models have been released over the last decade, its unknown what their representations encode or do not encode from the original high-dimensional and feature-rich natural language input. This question is especially important under distribution shift, where the train and test datasets come from different populations (e.g., due to a biased sampling or labeling procedure).

The broad, general knowledge encoded in foundation models can provide new insight in dealing with sampling bias effects, a persistent problem in statistical science known to be present in many datasets Quionero-Candela et al. (2009). This article addresses these concerns involving distribution shift in the context of transfer learning using the Sentiment140 dataset Go et al. (2009), a sentiment classification dataset whose sampling procedures differs between train/test splits. Our observations emphasize the importance of sampling train and test data from the same population, illustrating a point of caution when fine-tuning on data which is only semantically similar to the testing task: the presence of distribution shifts makes it unclear whether fine-tuning will at all boost the performance of a foundation model. To be explicit, the main contributions of our article are:

- We show a simple examination of the final feature embedding using principal component analysis (PCA) is able to detect the train and test data are sampled from different populations (see Figure 1), giving us a data-centric method to probe distribution shifts.
- We proceed to fine-tune foundation models to study the effect of this distribution shift. Our experiments show worse generalization when fine-tuned on a distribution assumedly closer

to our test distribution compared to the original pre-training corpora. In other words: fine tuning under distribution shift obscured the capacity for the model to generalize on the test population and confounds this capacity for model robustness under the distribution shift. This reiterates the necessity to sample from the same population inference is performed on.

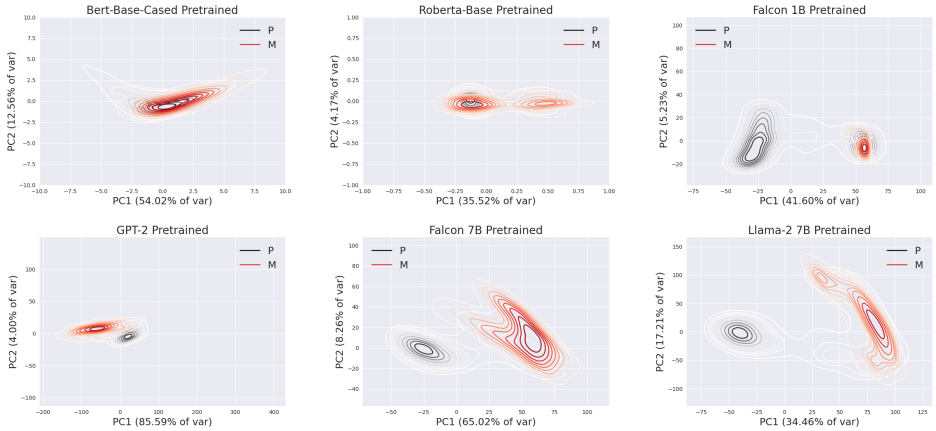


Figure 1: Kernel Density Estimates of the two largest principal components of the pre-trained final embedding representation of the automatically labeled training dataset P and the manually curated testing dataset M using various LLMs. Sub-figures are ordered in number of increasing model parameters.

2 Related Work

Transfer learning. Adapting trained models to new tasks is a popular technique, believed to improve performance by leveraging knowledge gained from a related domain or otherwise saving costs compared to training from scratch Raffel et al. (2020). Two common approaches are full fine-tuning and linear probing. In the former, all layers are allowed to update Yosinski et al. (2014), while the ladder only allows a linear, task-specific layer to update. Linear probing has been studied for its explanatory uses Belinkov (2021), Alain & Bengio (2016), Chen et al. (2020) and to provide relative baselines Kumar et al. (2022). Our experiments compare linear probes and full fine-tuning in order to measure their relative effectiveness on shifted train and test data; see Evci et al. (2022) for a different comparison of linear probes and fine-tuning.

Foundation Models Commonly built on the transformer architecture for vision and language contexts, these models first go though pre-training on a large, general use data corpus before being specialized to a downstream task Vaswani et al. (2017); Brown et al. (2020); Rombach et al. (2021). Though the pre-training corpus is typically sourced from the open web, many details are often closed the public Gemini Team et al. (2023); Touvron et al. (2023). The methods provided here implicitly take advantage of the knowledge gained from these training corpora to test for distribution shifts.

Representation Learning. Also known as feature learning, this focuses on learning useful features directly from raw data instead of hand-crafting features for a specific task Bengio et al. (2013). The problem of explaining the learned features of a deep network is famously difficult, with abundant research to probe their inner workings (e.g.Elhage et al. (2022); Wen & Li (2021); Yang & Hu (2021)) and the relations between different models' learned representations Kornblith et al. (2019); Morcos et al. (2018). Similar to the studies in Fort et al. (2021), Uppaal et al. (2023), and Jiang et al. (2023), we use PCA to perform dimensional reduction on data represented in high-dimensional space and extract key features to compare datasets and their underlying distributions.

3 Background and Methods

Data. All experiments utilize the Sentiment140 Dataset Go et al. (2009), which is a collection of tweets scraped from Twitter in the year 2009 [1]. The full dataset contains an automatically processed training dataset P and manually curated test set M which are collected using different methodologies. The 1,600,000 examples of P have labels inferred from sentimentally relevant emoticons scrubbed from the original tweets, e.g. ":)" indicates a positive tweet and ":(" negative. In contrast, M consists of 359 examples with human annotated sentiment labels. There are two distinct sampling populations: P comes from the population of tweets containing emoticons, and M from a population of tweets containing references to a varied list of subjects generated by the dataset authors. This is problematic if the emoticons do not perfectly match onto the notion of language-sentiment; Appendix B lists some datapoints from P that are likely confounders. The original authors behind Sentiment140 intended to view P and M as testing and training sets, respectively. As different experiments will train on samples coming from either P or M, we refrain from referring to P as 'the' training set or M as 'the' testing set.

Approximating Distribution Shift. In this article we deal with neural networks trained on natural language which we use to predict binary sentiment classification. For simplicity, we present such a network as a function $F : \mathcal{L} \to \{0, 1\}$, where $\mathcal{L}$ denotes the sampling space of natural language. The models considered here have a factorization $F = H \circ E$ where $E : \mathcal{L} \to \mathbb{R}^d$ is a high-dimensional embedding obtained from the final hidden layer and $H : \mathbb{R}^d \to \{0, 1\}$ is a classification head.

Given a sample of tweets $\{t_1, t_2, \ldots, t_n\}$ drawn from a distribution $p(t)$, we pass each tweet t_i through F and extract the feature activations $x_i := E(t_i) \in \mathbb{R}^d$, $i = 1, \ldots, n$ following a distribution $q(x)$. The x_i's serve as data engineered by the neural network with which we perform PCA to approximate $q(x)$. If we have another sample of tweets $\{t'_1, t'_2, \ldots, t'_{n'}\}$ drawn from $p'(t)$ then we can perform the same analysis to approximate the underlying distribution $q'(x)$ of the associated vectors $\{x'_1, x'_2, \ldots, x'_{n'}\}$. Importantly, we note that if $q(x) \neq q'(x)$, then we must have $p(t) \neq p'(t)$. That is, if the underlying distribution for the embedded samples is shifted, then so too must be the distribution for the original tweets. Further exposition is given in Appendix A.4.

Training. We use three different training methodologies in Section 4. The different methodologies contrast whether we allow additional feature learning ($\mathcal{FT}$) or use the original pre-trained model's representations ($\mathcal{LP}$).

- $\mathcal{FT}$) Starting with pre-trained weights, we attach a randomly initialized classification head and train for ten epochs, allowing *all* weights to update. We use disjoint samples of $20,000$ points from P for training and validation sets. The validation set is used to choose an early stopping epoch. Evaluation is performed on the entirety of M.
- $\mathcal{LP}$) Initialization as in $\mathcal{FT}$, allowing only the final classification layer to update. We use a balanced set of 300 points in M for training the final layer, evaluating on the remaining 59.
- $\mathcal{FT} + \mathcal{LP}$) First train using the $\mathcal{FT}$ method from above, without evaluation. Then re-initialize the final layer, training and evaluating following the instructions in $\mathcal{LP}$.

4 Results

4.1 LLMs can separate P and M with pre-trained weights.

We use various popular open-source models Devlin et al. (2019); Liu et al. (2019); Almazrouei et al. (2023); Touvron et al. (2023); Radford et al. (2019) to visualize the features of P and M using PCA on the final layer activations. We project down to the first two principal components and visualize the distribution with kernel density estimation. Results are shown in Figure 1. Five of the six language models are able to visually separate between P and M. Both RoBERTa-base and Falcon 1B do so with some overlap of the embedded approximations while other models like GPT-2, Falcon 7B, and Llama-2 7B are able to highly differentiate the shift. This suggests that a model's ability to discriminate on distribution shifts may correlate with either model complexity, the training corpus, or an interaction thereof.

[1] https://huggingface.co/datasets/sentiment140

4.2 FINE-TUNING ON DATASETS SIMILAR TO M DOES NOT GENERALIZE TO M.

Here we compare additional feature learning $\mathcal{FT}$ to re-using the pre-trained model's representations $\mathcal{LP}$, by evaluating both methods effect on generalization on M. Table 1 indicates that the pre-trained features in GPT-2 have robust capabilities to classify on M. In Appendix D.3 we undergo further analysis to give evidence that $\mathcal{LP}$ statistically outperforms $\mathcal{FT}$ (p-value 0.007). In context of Figure 1, we view the relative performance statistics between $\mathcal{LP}$ and $\mathcal{FT}$ as a consequence of the misaligned features between P and M.

In addition to the performance gains from $\mathcal{LP}$ over $\mathcal{FT}$, we stress the substantial savings in both time and space complexity by simply training a linear classifier rather than fine-tuning all of the weights of model. Training via $\mathcal{LP}$ (using M) takes under 3.8 seconds/epoch with 2.4 GB VRAM compared to $\mathcal{FT}$ taking up to 11.9 minutes/epoch with 4.0 GB VRAM.

Table 1: Comparing Fine-Tuned GPT-2 with Targeted Linear Probes

Base Model	Method	Train Acc. (%)	Test Acc. (%)
Pre-Trained GPT-2	$\mathcal{FT}$	$91.6_{\pm 0.93}$	$84.1_{\pm 0.49}$
Random Features GPT-2	$\mathcal{LP}$	$85.7_{\pm 0.27}$	$57.8_{\pm 1.05}$
Pre-Trained GPT-2	$\mathcal{LP}$	$95.3_{\pm 0.15}$	$86.3_{\pm 0.55}$
Pre-Trained GPT-2	$\mathcal{FT} + \mathcal{LP}$	$93.9_{\pm 0.27}$	$84.7_{\pm 1.17}$

Table 1: Those equipped with the $\mathcal{FT}$ method are fine-tuned using 20k datapoints from P and tested on all 359 points of M. Those with the $\mathcal{LP}$ method have a linear probe trained using 300 points of M and tested on the remaining 59 points of M. Values are shown $\pm$ the standard error of the mean.

5 DISCUSSION

This article has presented a first approximation to visualizing the distributions of feature-embeddings, allowing us to confirm that a suspected distribution shift adversely affects a foundation model's representations. We used this tool to observe the shift between Sentiment140's automatically processed and manually curated subsets as represented by six foundation models. Five of the models' embeddings clearly separate in the first two PCs, despite a surface-level similarity in the natural text (Figure 1). In fact, the separation between these distributions grows with model size, suggesting that greater model complexity allows better separability of text drawn from different sources. It is possible this effect is causally linked to other covariates between models, including training corpus and architectural improvements. Subsequent experiments with GPT-2 showed that features learned from fine-tuning on can harm predictive capacity on M compared to a simple linear probe on M which used the pre-trained features of the pre-trained model and only required 1.5% of the data to train. This highlights both the potential consequences of training with misaligned data as well as the value of in-distribution data.

Though we only focused on Sentiment140 here, the distinct sampling procedures behind P and M reflects a common situation in data science where it is costly or impossible to sample from the true distribution of interest Zoph et al. (2016); Cai et al. (2020). Towards addressing distribution shift for TL with foundation models, our work argues for two separate yet related pre-processing steps. First, ask whether the existing pre-trained representation recognizes the test data? If yes, then a simple linear probe may be sufficient. If no, we follow up by asking: is the train/test split identically distributed? If so, then cautiously proceed with FT. In the final negative case if not, re-evaluate the data generation methodology as otherwise, full fine-tuning on shifted data can confound interpretability for model generalization and downstream analysis.

6 LIMITATIONS AND FUTURE WORK

A fundamental limitation of our approach is that it is presently an ad hoc qualitative measure to detect distribution shifts rather than a principled test statistic. As discussed above, our suggestion that the ability of foundation models to discern distribution shift is a result of the model complexity

is a simple statement of correlation. It is also possible that the ability is derivative of the improvements/differences to pre-training corpora; which re-emphasizes the need for open-source publication of pre-training data. A deeper study of the model's architecture, its overall expressivity, and the data upon which it is trained may one day lead to quantitative hypothesis testing.

ACKNOWLEDGEMENTS

The work of AD, MV, AE, and TC were partially supported by the Mathematics for Artificial Reasoning in Science (MARS) initiative via the Laboratory Directed Research and Development (LDRD) Program at PNNL.

REFERENCES

Ben Adlam and Jeffrey Pennington. Understanding double descent requires a fine-grained bias-variance decomposition. *ArXiv*, abs/2011.03321, 2020. URL https://api.semanticscholar.org/CorpusID:226278106.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *ArXiv*, abs/1610.01644, 2016. URL https://api.semanticscholar.org/CorpusID:9794990.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Maitha Alhammadi, Mazzotta Daniele, Daniel Heslow, Julien Launay, Quentin Malartic, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. The falcon series of language models: Towards open frontier models. 2023.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48:207–219, 2021. URL https://api.semanticscholar.org/CorpusID:236924832.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828, aug 2013. ISSN 0162-8828. doi: 10.1109/TPAMI.2013.50. URL https://doi.org/10.1109/TPAMI.2013.50.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and et al. Language models are few-shot learners. *Advances in neural information processing systems*, 2020.

Chenjing Cai, Shiwei Wang, Youjun Xu, Weilin Zhang, Ke Tang, Qi Ouyang, Luhua Lai, and Jianfeng Pei. Transfer learning for drug discovery. *Journal of Medicinal Chemistry*, 63(16):8683–8694, 2020. doi: 10.1021/acs.jmedchem.9b02147. URL https://doi.org/10.1021/acs.jmedchem.9b02147. PMID: 32672961.

Mark Chen, Alec Radford, Jeff Wu, Heewoo Jun, Prafulla Dhariwal, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International Conference on Machine Learning*, 2020. URL https://api.semanticscholar.org/CorpusID:219781060.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. *ArXiv*, abs/1705.02364, 2017. URL https://api.semanticscholar.org/CorpusID:28971531.

Amit Daniely, Roy Frostig, and Yoram Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *NIPS*, 2016. URL https://api.semanticscholar.org/CorpusID:217536627.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805, 2019. URL https://api.semanticscholar.org/CorpusID:52967399.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition, 2022.

Utku Evci, Vincent Dumoulin, H. Larochelle, and Michael Curtis Mozer. Head2toe: Utilizing intermediate representations for better transfer learning. In *International Conference on Machine Learning*, 2022. URL https://api.semanticscholar.org/CorpusID:245837741.

Stanislav Fort, Jie Jessie Ren, and Balaji Lakshminarayanan. Exploring the limits of out-of-distribution detection. In *Neural Information Processing Systems*, 2021. URL https://api.semanticscholar.org/CorpusID:235358891.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, and Anja Hauth *et. al.* Gemini: A family of highly capable multimodal models, 2023.

Raja Giryes, Guillermo Sapiro, and Alexander M. Bronstein. Deep neural networks with random gaussian weights: A universal classification strategy? *IEEE Transactions on Signal Processing*, 64:3444–3457, 2015. URL https://api.semanticscholar.org/CorpusID:2906154.

Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. In *Stanford CS224N Project Report*, 2009. URL https://api.semanticscholar.org/CorpusID:18635269.

Yiding Jiang, Christina Baek, and J. Zico Kolter. On the joint interaction of models, data, and features, 2023.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3519–3529. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/kornblith19a.html.

Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *International Conference on Learning Representations*, 2022. URL https://api.semanticscholar.org/CorpusID:247011290.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning, 2023.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692, 2019. URL https://api.semanticscholar.org/CorpusID:198953378.

Song Mei and Andrea Montanari. The generalization error of random features regression: Precise asymptotics and the double descent curve. *Communications on Pure and Applied Mathematics*, 75, 2019. URL https://api.semanticscholar.org/CorpusID:199668852.

Ari Morcos, Maithra Raghu, and Samy Bengio. Insights on representational similarity in neural networks with canonical correlation. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/a7a3d70c6d17a73140918996d03c014f-Paper.pdf.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Neural Information Processing Systems*, 2019. URL https://api.semanticscholar.org/CorpusID:202786778.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*, 2023. URL https://arxiv.org/abs/2306.01116.

Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009. ISBN 0262170051.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, 2019. URL https://api.semanticscholar.org/CorpusID:160025533.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1), jan 2020. ISSN 1532-4435.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, A. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on Empirical Methods in Natural Language Processing*, 2013. URL https://api.semanticscholar.org/CorpusID:990233.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288, 2023. URL https://api.semanticscholar.org/CorpusID:259950998.

Rheeya Uppaal, Junjie Hu, and Yixuan Li. Is fine-tuning needed? pre-trained language models are near perfect for out-of-domain detection. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12813–12832, Toronto, Canada, jul 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.717. URL https://aclanthology.org/2023.acl-long.717.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Comput. Surv.*, 53(3), jun 2020. ISSN 0360-0300. doi: 10.1145/3386252. URL https://doi.org/10.1145/3386252.

Zixin Wen and Yuanzhi Li. Toward understanding the feature learning process of self-supervised contrastive learning. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 11112–11122. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/wen21c.html.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771, 2019. URL http://arxiv.org/abs/1910.03771.

Greg Yang and Edward J. Hu. Tensor programs iv: Feature learning in infinite-width neural networks. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 11727–11737. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/yang21c.html.

Gilad Yehudai and Ohad Shamir. On the power and limitations of random features for understanding neural networks. In *Neural Information Processing Systems*, 2019. URL https://api.semanticscholar.org/CorpusID:90262791.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *NIPS*, 2014. URL https://api.semanticscholar.org/CorpusID:362467.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11941–11952, 2023. doi: 10.1109/ICCV51070.2023.01100.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. Transfer learning for low-resource neural machine translation. In Jian Su, Kevin Duh, and Xavier Carreras (eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1568–1575, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1163. URL https://aclanthology.org/D16-1163.

A Experimental Details

All experiments were conducted on an Nvidia DGX-2 on a single A100-40 GPU using Python 3.9, via PyTorch Paszke et al. (2019). Due to memory limitations, we limited batch size to 1, while non-standard for full fine-tuning, still attained robust results.

A.1 Uncertainty Estimates

All experiments were run multiple times to ascertain both training and generalization variability. Seeds were manually set for the torch, numpy, and random modules. Experiments on M were performed 50x, with seeds 0-49, where M was shuffled with that specific seed, then split into training and evaluation sets. The experiment involving training on both P and M using the $\mathcal{FT} + \mathcal{LP}$ training method was performed 20x, with seeds 0-20. All other experiments were performed 10 times, with seeds 0-9, shuffling the data, then splitting accordingly.

A.2 Dataset and Model Sourcing

We sourced our data and models from openly available sources such as Hugging Face and Papers With Code. We obtained all training and evaluation data used in this experiment from the Stanford Sentiment140 website[2]. State-of-the-Art (SOTA) benchmarks were obtained from Papers With Code[3]. We used *GPT-2ForSequenceClassification*, and *GPT2Tokenizer* from Hugging Face for our experiments loaded from the Transformers Python library Wolf et al. (2019).

A.3 Pre-Processing Steps

We pre-processed text in Sentiment140 according to steps outlined in (Go et al., 2009, Sec 2.3). Duplicate tweets were removed. In addition, we used regular expressions to find usernames, remaining emoticons, and URLs. For example, a given username in a tweet would be converted to the token USERNAME. Likewise, a url would be converted to the token URL. Remaining emoticons were removed during our pre-processing.

A.4 Feature Extraction for Distribution Shift

LLMs and Feature Extraction. In this article we deal with neural networks trained on natural language which we use to predict binary sentiment classification. For simplicity, we present such a network as a function $F : \mathcal{L} \to \{0, 1\}$, where $\mathcal{L}$ denotes the sampling space of natural language. The models considered here actually have a factorization $F = H \circ E$ where $E : \mathcal{L} \to \mathbb{R}^d$ is an embedding function into a high-dimensional euclidean space obtained from the final hidden layer and $H : \mathbb{R}^d \to \{0, 1\}$ is a classification head obtained by applying a linear projection followed by softmax on the resulting logits.

Given a pre-trained model $F = H \circ E$ and a sample of tweets $\{t_1, t_2, \ldots, t_n\}$, we pass each tweet t_i through F and extract the feature activations $x_i := E(t_i) \in \mathbb{R}^d$, $i = 1, \ldots, n$. The x_i's serve as data engineered by the LLM which we then use for PCA; using the matrix $\Phi := [x_1, x_2, \ldots, x_n]$, singular value decomposition gives a factorization $\Phi = U\Sigma V^\intercal$. The k-th principal component is then given by the k-th column of V.

Distribution shift. Let $X = \{(x_i, y_i)\}_{i \in I}$ (resp. $X' = \{(x'_i, y'_i)\}_{i \in I'}$) be a labeled training (resp. testing) dataset drawn from a distribution $p(x, y)$ (resp. $p'(x, y)$). We say that there is a distribution shift if $p(x, y) \neq p'(x, y)$. That is, X and X' are not drawn from the same distribution. Abusing language, in this case we also say there is a distribution shift between X and X'. Fixing a choice of LLM and expressing it as $F = H \circ E$, we can study distribution shifts by working in the ambient embedding space, $\mathbb{R}^d$, by applying E. If the transformed distributions in this new space are distinct, then there must be a shift between the original distributions $p(x, y)$ and $p'(x, y)$. Examples of distribution shift appear throughout data science including covariate shift, sample-selection bias, and more Quionero-Candela et al. (2009).

[2] https://huggingface.co/datasets/sentiment140

[3] https://paperswithcode.com/sota/text-classification-on-sentiment140

A.5 Additional Experiments on Distributional Shift via Linear Classifiers

In order to understand whether P and M are linearly separable, we create a dataset $(Bias)$ to identify the model's ability to classify the selection bias of Sentiment140. We choose 200 points from P and M respectively to form a training set B_{train} (balanced by sentiment labels in both categories), and we test generalization on 160 points of P and 159 points of M, also balanced by sentiment labels, B_{test}.

We perform linear probing for random feature, pre-trained, and fine-tuned initializations of GPT-2 training on the $(Bias)$ dataset. We train a logistic regression classifier with training data B_{train} for ten epochs. We use B_{train} to determine an epoch to evaluate test accuracy on B_{test}.

A.6 Additional Experiments on P and M

We list our full set of experiments in table 4, some of which were not highlighted in the main text.

To understand the relative effect of training on P, we also test linear probes of random feature and pre-trained GPT-2 initializations. We take the same train/val/test split for sampling on P as in 3, training on P_{train}, validating on P_{val} and testing on $P_{\text{test}} = M$.

We additionally investigate whether fine-tuning on M would overfit compared to a linear probe. We fine-tune on M, taking the same train/test split as described for sampling on M in 3, training on M_{train} and testing on M_{test}. Furthermore, we perform linear probing with samples of 100-250 points of M, increment by 50 per iteration and test on a held-out set of 59 points. Throughout, we use the same hyperparameters as in our original experiments in 3.

A.7 Hyperparameters

Hyperparameters used for all our experiments are shown in Table: 2. *BCEWithLogitsLoss* fuses a binary cross entropy loss and a sigmoid scaling operation.

Table 2: Hyperparameters used in linear probing and fine-tuning

Hyperparameter	Linear Probing	Fine-Tuning
Epochs	10	10
Optimizer	Adam	Adam
Learning Rate	10^{-2}	10^{-5}
β_1	.9	.9
β_2	.999	.999
ϵ	10^{-8}	10^{-8}
Batch Size	1	1
Scheduler-Type	LinearLR	LinearLR
Start Factor	.33	.33
End Factor	1.0	1.0
Scheduler Iterations	5	5
Dropout	0.1	0.1
Gradient Norm Clip	1.0	1.0
Seeds	$0-49$ for M, $0-9$ otherwise	$0-49$ for M, $0-9$ otherwise
Loss Function	BCEWithLogitsLoss	BCEWithLogitsLoss
Random Feature Initialization	Sample from $\mathcal{N}(0, 0.02)$, residual layers scaled by $\frac{1}{\sqrt{n}}$	–

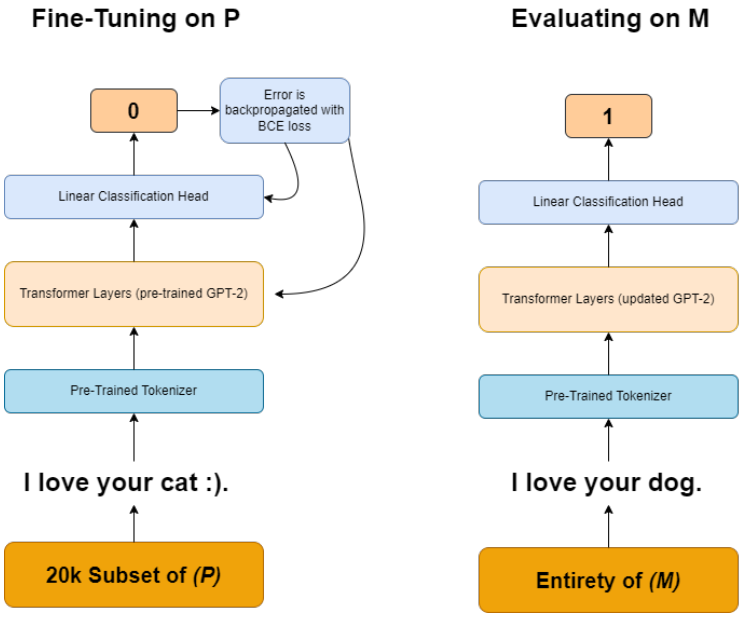


Figure 2: Full-fine-tuning of pre-trained GPT-2, trained on a sample of P, evaluated on M

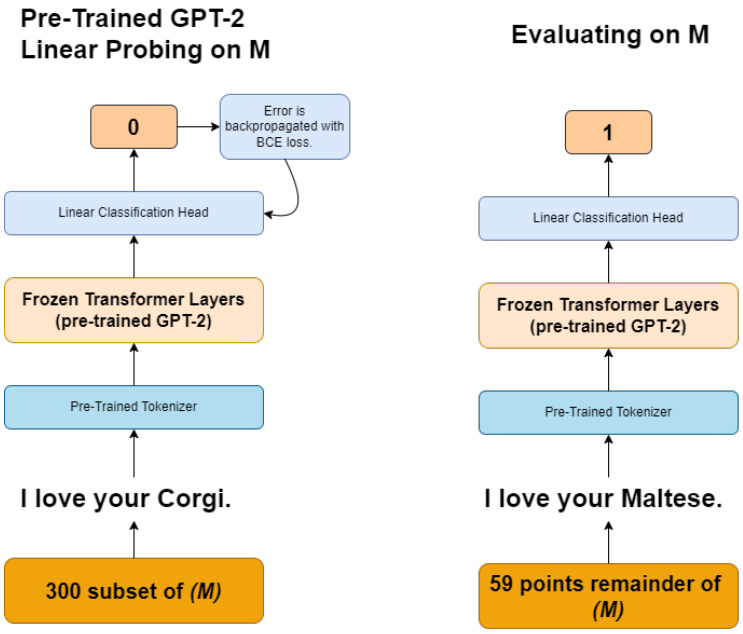


Figure 3: Linear probe of pre-trained GPT-2, trained on a sample of M, evaluated on remainder of M.

A.8 Visualizing the Methods FT and LP

Figures 2, 3, and 4 illustrate the experiments described in section 3. Specifically, figure 2 presents the typical transfer learning paradigm of learning on one dataset (P in this case) and evaluating on a separate 'expectedly similar' dataset (M). Figures 3 and 4 diagram attaching a linear head to examine feature representations for pre-trained and random feature initializations of GPT-2.

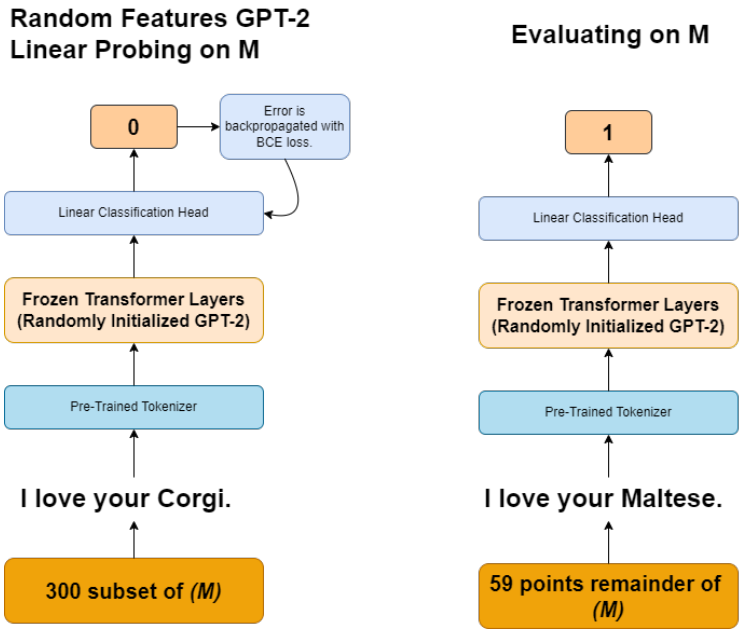


Figure 4: Linear Probe of Random Feature GPT-2, trained on a sample of M, evaluated on the remainder.

B Examining P for Examples of Mislabeled Data

Table 3: Possibly Mislabeled Data in P

Label	ID	Tweet
:(	1467863072	@twitterhandle1
:(	1467871226	@twitterhandle2 Congrats!! i totally forgot to submit photos
:(	1467979094	@twitterhandle3 OHH! OMG. LMAO. I'm crying right now, LOL! KUTNERRRR was the best!
:)	1836399555	@twitterhandle4
:)	1879932456	@twitterhandle5 ????????????????
:)	1833894264	Just another SARS is coming...

Table: 3 We present selected examples of potentially mislabeled in dataset P. The curators of Sentiment140 scrubbed the tweets in P of emoticons. A ':(' corresponds to a 0 label, ':)' corresponds to 1 label.

In retrospect, using emoticons as a proxy to label sentiment is an imperfect method; one that the creators of Sentiment140 acknowledged as a key difficulty of creating their dataset Go et al. (2009). Table 3 showcases typical examples of such statements. Rows one and four consist of tweets only consisting of a social media handle after the relevant emoticon is removed. After pre-processing, the handle is changed to the token USERNAME, making these tweets a possible source of bias in the training data as inputs consisting of the exact same sequence of tokens would have opposite labels.

Row two includes an example of mixed sentiment. One aspect of the tweet is positive ("Congrats!!"), yet one is negative "i totally forgot to submit photos", nevertheless a human annotator may label this tweet as positive considering it starts by mentioning another user's handle and congratulating them. Row three is arguably an example of a false-negative, and we believe that this

sentence expresses positive sentiment. To us, row six shows a false-positive, referencing the spread of a disease. In its original context, with a smiling emoticon, the tweet in row six would have been an example of sarcasm. Row five is also a potential false-positive as a long sequence of question marks is traditionally associated with disbelief or anger. Given that M was hand-labeled, we can assume that false-negatives and false-positives as shown in Table 3 would not be as prevalent in the data.

C HIGHLIGHTING A RANDOM FEATURE BASELINE

In their paper, Conneau et. al. Conneau et al. (2017) demonstrate that linear probing of BiLSTM encoders initialized with random weights can achieve a peak of 80.7% accuracy in the SST-2 sentiment analysis dataset Socher et al. (2013). On our dataset, we average 57.8% test accuracy and reach 76% peak as shown in Table 1 and Figure 5. We extend the work of Conneau et. al. Conneau et al. (2017) with LSTMs to LLM models such as GPT-2 in our case study.

Recently, compelling work has been done investigating the generalization performance of random feature classification Mei & Montanari (2019), Adlam & Pennington (2020), Yehudai & Shamir (2019). These are closely related to work on random initialization of neural networks as studied in Giryes et al. (2015). Daniely, Frostig and Singer Daniely et al. (2016) show that with a random initialization, linear probing of the last hidden layer (equivalently last-layer training) can learn linear functions as well as more compliciated classes of functions obtained by non-linear kernel composition. These learning guarantees may be sufficient to achieve the generalization shown in our sentiment analysis task. In light of their work, studying random feature regression presents an exciting opportunity for AI research.

D ADDITIONAL RESULTS

Table 4: Accuracy and Comparison to Other Works

Model	Training Accuracy	Validation Accuracy	Test Accuracy
RF GPT-2 + LP (P)	$60.5 \pm 0.20\%$	$59.6 \pm 0.14\%$	$53.6 \pm 0.62\%$
Pre-Trained GPT-2 + LP (P)	$77.3 \pm 0.1\%$	$76.4 \pm 0.09\%$	$78.7 \pm 0.75\%$
Pre-Trained GPT-2 + FT (P)	91.6 ± 0.93 %	83.5 ± 0.09 %	$84.1 \pm 0.49\%$
RF GPT-2 + LP (M)	$85.7 \pm 0.27\%$	–	$57.8 \pm 1.05\%$
Pre-Trained GPT-2 + LP (M)	$95.3 \pm 0.15\%$	–	86.3 ± 0.55 %
Pre-Trained GPT-2 + FT (M)	99.9 ± 0.02 %	–	83.7 ± 0.81 %
RF GPT-2 + LP (Bias)	$95.9 \pm 0.2\%$	–	$86.9 \pm 0.8\%$
Pre-Trained GPT2 + LP (Bias)	$94.5 \pm 0.38\%$	–	$91.4 \pm 0.36\%$
FT GPT-2 + LP (Bias)	93.4 ± 0.28 %	–	92.3 ± 0.64 %
FT RoBERTa	–	–	89.3 %
FT ALBERT	–	–	85.3 %
FT XLNET	–	–	84.0 %

Table 4: Group one and two in the table are trained using data from P and M, respectively. They are both evaluated using testing data from M. The third grouping is trained and tested on the Bias dataset described in A.5. The final group are state-of-the-art (SoTA) fine-tuned model instances described on the online repository Papers With Code.

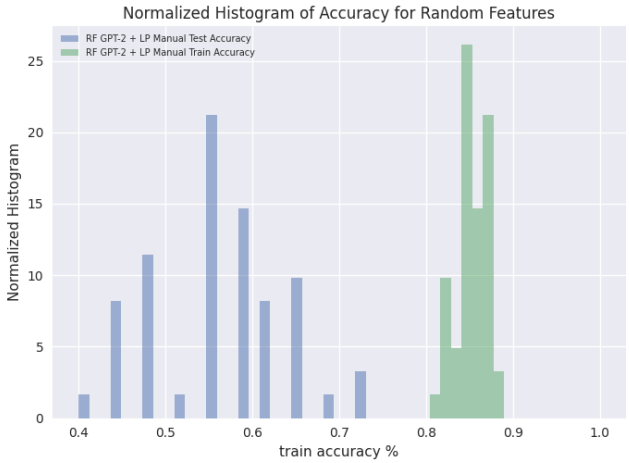


Figure 5: Training and Test Accuracy of a linear probe on the random features from GPT-2 architecture, trained on a held-out sample of 300 points from M, evaluated on the remainder.

D.1 Linear Probes separate P and M with high accuracy

Table 5: Linear Probes of the $(Bias)$ dataset

Base Model	Training Accuracy	Test Accuracy
Random Features GPT-2	$95.9 \pm 0.2\%$	$86.9 \pm 0.8\%$
Pre-Trained GPT-2	$94.5 \pm 0.38\%$	$91.4 \pm 0.36\%$
Fine-Tuned GPT-2	93.4 ± 0.28 %	92.3 ± 0.64 %

We present linear probes from three different initializations trained to classify whether a tweet belonged to either P or M. Our models were trained on B_{train}, and evaluated on B_{test} from the $(Bias)$ dataset described in A.5.

Expanding on our exploratory analysis in 4.1, we trained a series of linear classifiers to determine if tweets belonged to either P or M according to the experimental setup described in A.5. Our results show that even a linear probe off of random features was able to perform this task with 86.9% accuracy as shown in 5. Moreover, we see that probes of the pre-trained and fine-tuned models can separate the distribution shift with even greater accuracy.

D.2 Training on $\boldsymbol{P}$ under-performs on $\boldsymbol{P}$ compared to $\boldsymbol{M}$

Using GPT-2's pretrained feature representations to examine P and M gives a surprising result. The validation accuracies on P_{val} in rows 2 and 3 of Table 4 are lower than the testing accuracies on M. This result should give us pause by itself, since we typically expect the generalization metric to be lower than the training metric. This is especially the case with using P and M as training and testing distributions since our bias experiments indicate that P and M are linearly separable, hence are drawn from different distributions. We conjecture that the data of M is more closely aligned to the pre-training corpus of GPT-2, than the data of P. Consequentially, one might view sentiment classification on P as a more out-of-distribution task than sentiment classification on M.

When these results are interpreted in context with the linear probe baseline of pre-trained GPT-2 on M, we deduce that training on P may bias sentiment classification on M (also supported from the RF+LP on P, row 1 of Table 4). Additionally, FT models on M show a clear overfit onto the training data (near 100%) with a substantial drop-off for test accuracy (approximately 83%).

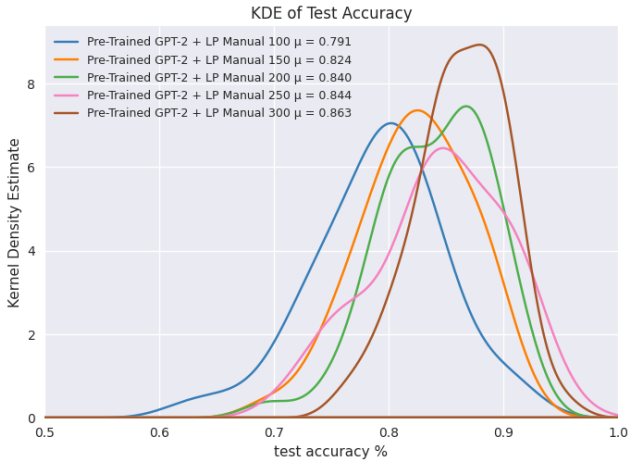


Figure 6: KDE plots of test accuracy for linear probes trained on 100/150/200/250/300 points of M and tested on 59 points in the remainder of M

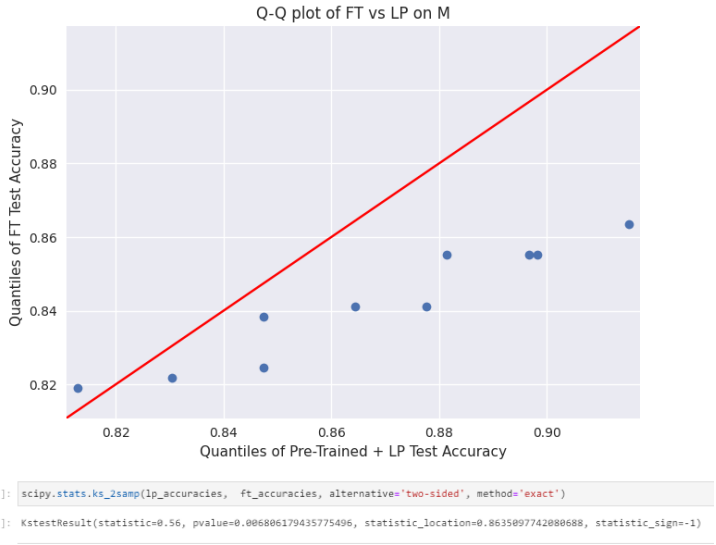


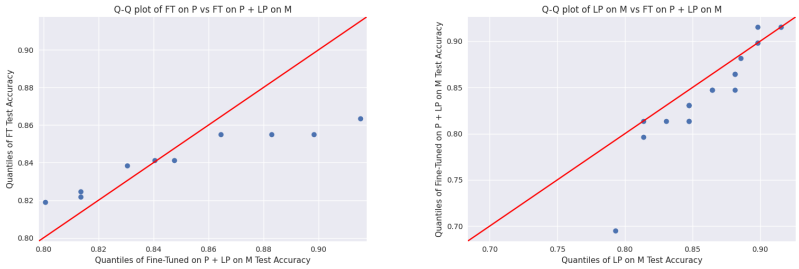
Figure 7: A QQ plot and a 2-sided KS test of test accuracies for linear probes of pre-trained GPT-2 and fine-tuned GPT-2 on M are presented.

D.3 Training with more data on M leads to increased average performance

Figure 6 presents KDEs and means of the distributions of test accuracy for linear probing on varying size samples of M. As the sample size increases, we see a clear rightward trend. Moreover, going from 100 points to 300 points gives a relative gain of roughly 8%. If this trend were to continue past 300 points, we could potentially see linear probe models with higher average test accuracy than existing state-of-the-art models.

D.4 FT on P and LP on M are not the same distribution

The QQ plot for the quantiles of the test accuracies for LP on M (on the x-axis) against FT on P (on the y-axis) implies differing distributions with a larger dispersion for test accuracies of linear probes




```
[22]: scipy.stats.ks_2samp(ft_lp_accuracies,  ft_accuracies, alternative='two-sided', method='exact')

[22]: KstestResult(statistic=0.4, pvalue=0.2213910693670814, statistic_location=0.8635098, statistic_sign=-1)

[24]: scipy.stats.ks_2samp(ft_lp_accuracies,  pretrained_manual_accuracies, alternative='two-sided', method='exact')

[24]: KstestResult(statistic=0.23, pvalue=0.3908401298500383, statistic_location=0.8305085, statistic_sign=1)
```

Figure 8: A QQ plot and a 2-sided KS test of test accuracies for linear probes of pre-trained GPT-2 and fine-tuned GPT-2 on M are presented.

fit. Moreover, applying a 2-sided KS test returns a p-value of 0.007, showing that the distributions of fine-tuned models and linear probe models are likely not the same.

D.5 Features learned from P do not improve the capacity to learn on M

While there might be a slight skew on the QQ plot comparing $\mathcal{FT}(P)$ against $\mathcal{FT}(P) + \mathcal{LP}(M)$, there is no significant evidence of a statistical difference as seen by p-value on the KS-test.