Understanding the Inner-workings of Language Models Through Representation Dissimilarity

Anonymous EMNLP submission

Abstract

As language models are applied to an increasing number of real-world applications, understanding their inner-workings has become an important issue in model trust, interpretability, and transparency. In this work we show that representation dissimilarity measures, which are functions that measure the extent to which two model's internal representations differ, can be a valuable tool for gaining insight into the mechanics of language models. Among our insights are: (i) an apparent asymmetry in the internal representations of model using SoLU and GeLU activation functions, (ii) evidence that dissimilarity measures can identify generalization properties of models that are invisible via in-distribution test set performance, and (iii) new evaluations of how language model features vary as width and depth are increased. Our results suggest that dissimilarity measures are a promising set of tools for shedding light on the inner-workings of language models.

1 Introduction

The defining feature of deep neural networks is their capability of learning useful feature representations from data in an end-to-end fashion. Perhaps ironically, one of the most pressing scientific challenges in deep learning is *understanding* the features that these models learn. This challenge is not merely philosophical: learned features are known to impact model interpretability/explainability, generalization, and transferability to downstream tasks, and it can be the case that none of these effects are visible from the point of view of model performance on even carefully selected validation sets.

When studying hidden representations in deep learning, a fundamental question is whether the internal representations of a given pair of models are similar or not. Dissimilarity measures (Klabunde et al., 2023) are a class of functions that seek to address this by measuring the difference between (potentially high-dimensional) representations. In this paper, we focus on two such functions that, while popular in computer vision, have seen limited application to language models: model stitching (Lenc and Vedaldi, 2014; Bansal et al., 2021) and centered kernel alignment (CKA) (Kornblith et al., 2019). *Model stitching* extracts features from earlier layers of model f and plugs them into the later layers of model g (possibly mediated by a small, learnable, connecting layer), and evaluates downstream performance of the resulting "stitched" model. Stitching takes a task-centric view towards representations, operating under the assumption that if two models have similar representations then these representations should be reconcilable by a simple transformation to such an extent that the downstream task can still be solved. On the other hand, CKA compares the statistical structure of the representations obtained in two different models/layers from a fixed set of input datapoints, ignoring any relationship to performance on the task for which the models were trained.

In this paper we make the case that dissimilarity measures are a tool that has been underutilized in the study of language models. We support this claim through experiments that shed light on the inner-workings of language models: **(i)** We show that stitching can be used to better understand the changes to representations that result from using different nonlinear activations in a model. In particular, we find evidence that feeding Gaussian error linear unit (GeLU) model representations into a softmax linear unit (SoLU) model incurs a smaller penalty in loss compared to feeding SoLU activations into a GeLU model, suggesting that models with GeLU activations may form representations that contain strictly more useful information for the training task than the representations of models using SoLU activations. **(ii)** We show that dissimilarity measures can highlight differences in models that are invisible via test set performance. Fol-

lowing the experimental set of up of (Juneja et al., 2022), we show that both stitching and CKA detect the difference between models which generalize to an out-of-distribution test set and models that do not generalize. **(iii)** Finally, we apply CKA to the Pythia networks (Biderman et al., 2023), a sequence of generative transformers of increasing width and depth, finding a high degree of similarity between early layer features even as scale varies from 70 million to 1 billion parameters, and to a lesser degree similarity between intermediate layer features across that same range.

2 Background

In this section we review the two model dissimilarity measures appearing in this paper.

Model stitching (Bansal et al., 2021): Informally, model stitching asks how well the representation extracted by the early layers of one model can be used by the later layers of another model to solve a specific task. Let f be a neural network and for a layer l of f let $f_{\leq l}$ (resp. $f_{\geq l}$) be the composition of the first l layers of f (resp. the layers m of f with $m \geq l$). Given another network g the model obtained by stitching layer l of f to layer m of g with stitching layer φ is $g_{>m} \circ \varphi \circ f_{\leq l}$. The performance of this stitched network measures the similarity of representations of f at layer l and representations of g at layer m.

Centered kernel alignment (CKA) (Kornblith et al., 2019): Let $D = \{x_1, \ldots, x_d\}$ be a set of model inputs. For models f and g with layers l and m respectively, let $A_{f,l,g,m}$ be the covariance matrix of $f_{\leq l}(D)$ and $g_{\leq m}(D)$. Then the CKA score for models f and g at layers l and m respectively and evaluated at D is

$$\frac{||A_{f,l,g,m}||_F^2}{||A_{f,l,f,l}||_F ||A_{g,m,g,m}||_F}.$$

Here $||\cdot||_F$ is the Frobenious norm. Higher CKA scores indicate more structural similarity between representations.

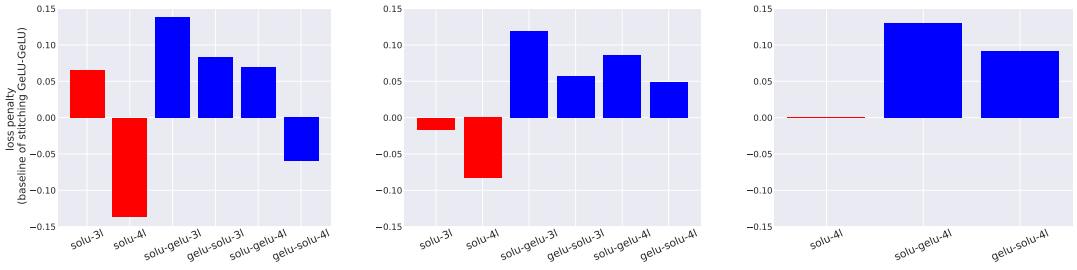
3 Model stitching reveals an asymmetry between GeLU and interpretable-by-design SoLU

The design of more interpretable models is an area of active research. Interpretable-by-design models often achieve comparable performance on downstream tasks to their non-interpretable counterparts (Rudin, 2019). However, these models (almost by definition) have differences in their hidden layer representations that can impact downstream performance.

Most contemporary transformers use the Gaussian error linear unit activation function, approximately $\mathrm{GeLU}(x) = x * \mathrm{sigmoid}(1.7x)$. The softmax linear unit $\mathrm{SoLU}(x) = x \cdot \mathrm{softmax}(x)$ is an activation function introduced in (Elhage et al., 2022) in an attempt to reduce neuron polysemanticity: the softmax has the effect of shrinking small and amplifying large neuron outputs. One of the findings of (Elhage et al., 2022) was that SoLU transformers yield comparable performance to their GeLU counterparts on a range of downstream tasks. However, those tests involved zero-shot evaluation or finetuning of the full transformer, and as such they do not shed much light on intermediate hidden feature representations. We use stitching to do this.

Using the notation from Section 2, we set our stitching layer φ to be a learnable linear layer (all other parameters of the stitched model are frozen) between the residual streams following a layer l . We stitch small 3-layer (9.4M parameter) and 4-layer (13M parameter) SoLU and GeLU models trained by (Nanda, 2022), using the PILE validation set (Gao et al., 2020) to optimize φ.

Figure 1 displays the resulting stitching penalties calculated by subtracting the PILE validation loss of the stitched models from a GeLU baseline (which is constructed by stitching between two identical GeLU models for the respective 3-layer and 4-layer cases). The red bars give the linear stitching penalty between two identical SoLU models relative to a baseline of stitching two identical GeLU models (we note that the negative penalties mean that stitched SoLU models often achieve lower loss than their GeLU counterparts). In the remaining bars, all penalties are positive and we see that when f (resp. g) is a SoLU (resp. GeLU) model — the "solu-gelu" cases — larger penalties are incurred than when f uses GeLUs and g uses SoLUs, i.e. the "gelu-solu" cases. In other words, plugging SoLU transformer features into the later layers of a GeLU transformer interferes with task performance more than plugging GeLU transformer features into the later layers of a SoLU transformer. We conjecture that this outcome results from the fact that SoLU activations effectively reduce capacity of hidden feature layers; there may be an analogy between the experiments of fig. 1 and those of (Bansal et al., 2021, Fig. 3c) stitching



(a) Stitching between layer 1 (b) Stitching between layer 2 (c) Stitching between layer 3

Figure 1: **Stitching penalties** (larger is worse) between 3-layer and 4-layer GeLU and SoLU models. The loss penalty is the stitching loss on the PILE validation set subtracted from a baseline of stitching between identical GeLU for 3 and 4 layers, respectively. "solu-gelu" stitching (stitching with a SoLU 'head' and GeLU 'tail') incurs systematic penalties compared to "gelu-solu" stitching.

vision models of different widths.

As pointed out in (Bansal et al., 2021) such an analysis is not possible using CKA, which only detects distance between distributions of hidden features (displayed for our GeLU and SoLU models in fig. 4), not their usefulness for a machine learning task. Further, the differences in layer expressiveness between GeLU and SoLU models are not easily elicited by evaluation on downstream tasks, but can be seen through linear stitching.

4 Locating generalization failures

Starting with a single pretrained BERT model and fine-tuning 100 models (differing only in random initialization of their classifier heads and stochasticity of batching in fine-tuning optimization) on the MNLI dataset (Williams et al., 2018), the authors of (McCoy et al., 2020) observed a striking phenomenon: despite the fine-tuned BERTs' near identical performance on MNLI, their performance on HANS (an out of distribution variant of MNLI) (McCoy et al., 2019) was highly variable. On a specific subset of HANS called "lexical overlap" one subset of fine-tuned BERTs (the *generalizing*) models) were successful; models in its complement (the *heuristic* models) failed catastrophically. Recent work of (Juneja et al., 2022) found that partitioning the 100 fine-tuned BERTs on the basis of their HANS-LO performance is essentially equivalent to partitioning them on the basis of *mode connectivity*.

But at what layer(s) do the hidden features of the generalizing and heuristic models diverge? For example, are these two subpopulations of models distinct because their earlier layers are different or because their later layers are different? Or both? Neither analysis of OOD performance nor mode connectivity analysis can provide an answer. Fortunately, both stitching and CKA reveal that the difference between the features of the generalizing and heuristic models is concentrated in later layers of the BERT encoder.

Figure 2a displays performance of pairs of BERT models stitched with the *identity function* $\varphi = \mathrm{id}$, i.e. models of the form $g_{>m} \circ f_{\leq l}$ on the MNLI finetuning task.[1] We see that at early layers of the models identity stitching incurs almost no penalty, but that stitching at later layers (when the fraction of layers occupied by at the bottom model exceeds 75 %) stitching *between* generalizing and heuristic models incurs significant accuracy drop (high stitching penalty), whereas stitching *within* the generalizing and heuristic groups incurs almost no penalty.

A similar picture emerges in fig. 2b, where we plot CKA values between features of fine-tuned BERTs on the MNLI dataset. At early layers, CKA is insensitive to the generalizing or heuristic nature of model A and B. At later layers (in the same range where we saw identity stitching penalties appear), the CKA measure *between* generalizing and heuristic models significantly exceeds its value *within* the generalizing and heuristic groups.

5 Representation dissimilarity in scaling families

Enormous quantities of research and engineering energy have been devoted to design, training and

[1] The motivation for stitching with a constant identity function comes from evidence that stitching-type algorithms perform symmetry correction (accounting for the fact that features of model A could differ from those of model B by an architecture symmetry e.g. permuting neurons, see e.g. (Ainsworth et al., 2023)), but that networks obtained from multiple fine-tuning runs from a fixed pretrained model seem to not differ by such symmetries (see for example (Wortsman et al., 2022)).

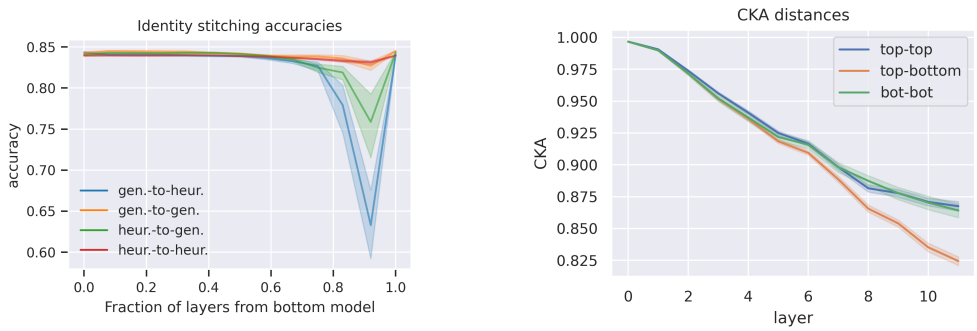
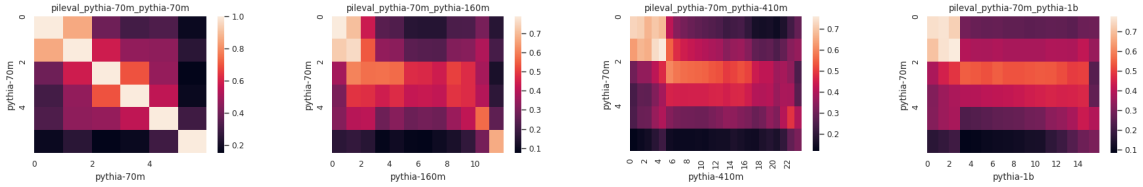


Figure 2: **Left:** Identity stitching on the MNLI dataset. Between pairs of the top 10 ("generalizing") and bottom 10 ("heuristic") performing models on the lexical overlap subset of HANS (an out-of-distribution NLI dataset). **Right:** CKA values between the models of (McCoy et al., 2020) (larger means more similar). *Top-top*: pairwise CKAs between the top 10 models on HANS-LO. *Top-bottom*: pairwise CKAs between the top 10 and bottom 10 models on HANS-LO. *Bot-bot*: pairwise CKAs between the bottom 10 models on HANS-LO.



(a) (b) (c) (d)

Figure 3: CKA for Pythia family models (Biderman et al., 2023) (higher means more similar). **(a)** Features within `pythia-70m`. **(b-c)** Features between `pythia-70m` and {`pythia-160m`,`pythia-410m`,`pythia-1b`}.

evaluation of *scaling families* of language models. By a "scaling family" we mean a collection of neural network architectures with similar components, but with variable width and/or depth resulting in a sequence of models with increasing size (as measured by number of parameters). Pioneering work in computer vision used CKA to discover interesting properties of hidden features that vary with network width and depth (Nguyen et al., 2021), but to the best of our knowledge no similar studies have appeared in the natural language domain

We take a first step in this direction by computing CKA measures of features within and between the first four models of the Pythia family (Biderman et al., 2023). In the case of inter-model CKA measurements on `pythia-70m` we see that similarity between layers l and m gradually decreases as $|m - l|$ decreases. Using CKA to compare features *between* `pythia-70m` and `pythia-160m` we observe high similarity between features at equal distance from either the beginning or end of the model. A plausible reason for this early layer similarity is the common task of "de-tokenization," (Elhage et al., 2022) where early neurons in models may change unnatural tokenizations to more useful representations, for example responding strongly to compound words (e.g., `birthday | party`). To a lesser degree, we see similarity between middle layers of these two models (2-3 in `pythia-70m`, 1-9 in `pythia-160m`), visible as a rectangular pattern in fig. 3 (middle) and reminiscent of the "block structure" analyzed in (Nguyen et al., 2021). Similar remarks, with less emphasis on similarity between later layers, apply to CKA between features of `pythia-70m` and `pythia-410m`/`pythia-1b`. For additional CKA comparisons we refer to the appendix.

Limitations

In the current work we examine relatively smaller language models (< 1 billion parameters). Larger models have qualitatively different features, so it is not obvious if our experiments on the differences in layer expressiveness between GeLU and SoLU models will scale to significantly larger models. While we show that identity stitching and CKA distinguish the heuristic and generalizing BERT models, we did not attempt to use stitching/CKA to automatically cluster the models (as was done with mode connectivity in (Juneja et al., 2022)).

Ethics Statement

In this paper we present new applications of representation analysis to language model hidden features. Large language models have the potential to impact human society in ways that we are only beginning to glimpse. A deeper understanding of the features they learn could be exploited for both positive and negative effect. Positive use cases include methods for enhancing language models to be more accurate, generalizable and robust (hinted at in the experiments of section 4), methods for explaining the decisions of language models and identifying model components responsible for unwanted behavior. Unfortunately, these tools can often be repurposed to do harm, for example extracting information from model training data and inducing specific undesirable model predictions.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2016. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *International Conference On Learning Representations*.

Samuel K. Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. 2023. Git Re-Basin: Merging Models modulo Permutation Symmetries.

Yamini Bansal, Preetum Nakkiran, and Boaz Barak. 2021. Revisiting model stitching to compare neural representations. In *Advances in Neural Information Processing Systems*, volume 34, pages 225–236. Curran Associates, Inc.

Serguei Barannikov, Ilya Trofimov, Nikita Balabin, and Evgeny Burnaev. 2021. Representation topology divergence: A method for comparing neural network representations. *arXiv preprint arXiv:2201.00058*.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada. Association for Computational Linguistics.

Nora Belrose, Zach Furman, Logan Smith, Danny Halawi, Igor V. Ostrovsky, Lev McKinney, Stella Rose Biderman, and J. Steinhardt. 2023. Eliciting latent predictions from transformers with the tuned lens. *ARXIV.ORG*.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. 2023. Pythia: A Suite for Analyzing Large Language Models Across Training and Scaling.

Frances Ding, Jean-Stanislas Denain, and Jacob Steinhardt. 2021. Grounding representation similarity through statistical testing. *Advances in Neural Information Processing Systems*, 34:1556–1568.

Nelson Elhage, Tristan Hume, Catherine Olsson, Neel Nanda, Tom Henighan, Scott Johnston, Sheer ElShowk, Nicholas Joseph, Nova DasSarma, Ben Mann, Danny Hernandez, Amanda Askell, Kamal Ndousse, Jones, , Dawn Drain, Anna Chen, Yuntao Bai, Deep Ganguli, Liane Lovitt, Zac Hatfield-Dodds, Jackson Kernion, Tom Conerly, Shauna Kravec, Stanislav Fort, Saurav Kadavath, Josh Jacobson, Eli Tran-Johnson, Jared Kaplan, Jack Clark, Tom Brown, Sam McCandlish, Dario Amodei, and Christopher Olah. 2022. Softmax linear units. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2022/solu/index.html.

Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. 2022. The role of permutation invariance in linear mode connectivity of neural networks. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Jonathan Frankle, G. Dziugaite, Daniel M. Roy, and Michael Carbin. 2019. Linear mode connectivity and the lottery ticket hypothesis. *ICML*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv: 2101.00027*.

Charles Godfrey, Davis Brown, Tegan Emerson, and Henry Kvinge. 2022. On the symmetries of deep learning models and their internal representations. *arXiv preprint arXiv:2205.14258*.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. *Conference On Empirical Methods In Natural Language Processing*.

Jeevesh Juneja, Rachit Bansal, Kyunghyun Cho, João Sedoc, and Naomi Saphra. 2022. Linear connectivity reveals generalization strategies. *arXiv preprint arXiv: Arxiv-2205.12411*.

Max Klabunde, Tobias Schumacher, Markus Strohmaier, and Florian Lemmerich. 2023. Similarity of neural network models: A survey of functional and representational measures. *arXiv preprint arXiv:2305.06329*.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey E. Hinton. 2019. Similarity of Neural Network Representations Revisited. *ICML*.

Karel Lenc and A. Vedaldi. 2014. Understanding image representations by measuring their equivariance and equivalence. *Computer Vision And Pattern Recognition*.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Ekdeep Singh Lubana, Eric J. Bigelow, Robert P. Dick, David Krueger, and Hidenori Tanaka. 2022. Mechanistic mode connectivity. *arXiv preprint arXiv: 2211.08422*.

R. Thomas McCoy, Junghyun Min, and Tal Linzen. 2020. BERTs of a feather do not generalize together: Large variability in generalization across models with similar test set performance. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 217–227, Online. Association for Computational Linguistics.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Neel Nanda. 2022. Transformerlens.

Thao Nguyen, Maithra Raghu, and Simon Kornblith. 2021. Do Wide and Deep Networks Learn the Same Things? Uncovering How Neural Network Representations Vary with Width and Depth.

Yuxin Ren, Qipeng Guo, Zhijing Jin, Shauli Ravfogel, Mrinmaya Sachan, Bernhard Schölkopf, and Ryan Cotterell. 2023. All roads lead to rome? exploring the invariance of transformers' representations. *arXiv preprint arXiv: 2305.14555*.

Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215.

Naomi Saphra and Adam Lopez. 2019. Understanding learning dynamics of language models with SVCCA. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3257–3267, Minneapolis, Minnesota. Association for Computational Linguistics.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019. The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4396–4406, Hong Kong, China. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Alex H Williams, Erin Kunz, Simon Kornblith, and Scott Linderman. 2021. Generalized shape metrics on neural representations. *Advances in Neural Information Processing Systems*, 34:4738–4750.

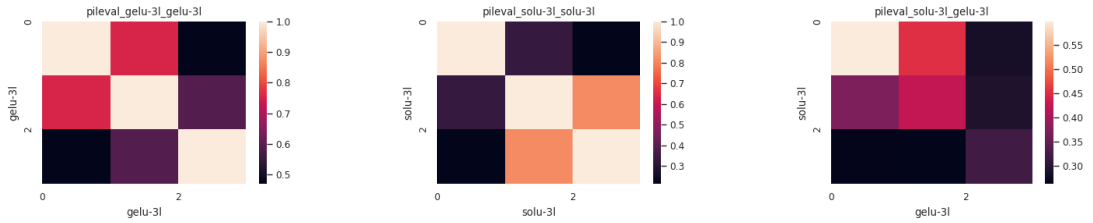
Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. 2022. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time.

A Related Work

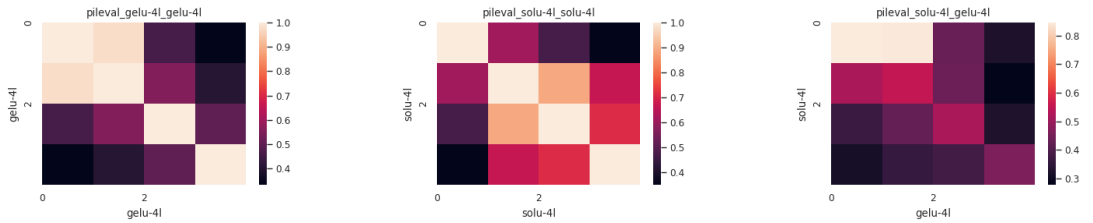
The problem of understanding how to compare the internal representations of deep learning models in a way that captures differences meaningful to the machine learning task has inspired a rich line of research (Klabunde et al., 2023). Popular approaches to quantifying dissimilarity range from shape based measures (Ding et al., 2021; Williams et al., 2021; Godfrey et al., 2022) to topological approaches (Barannikov et al., 2021). The vast majority of research into these approaches has concentrated on computer vision applications rather than language models. Exceptions have looked at the evolution of model hidden layers with respect to a training task (Voita et al., 2019) or downstream tasks (Saphra and Lopez, 2019), statistical testing of sensitivity and specificity of dissimilarity metrics applied to BERT models (Ding et al., 2021), and more flexible representation dissimilarity measures aligning hidden features with invertible neural networks (Ren et al., 2023). While representation dissimilarity measures have not been extensively explored in language models, there is a rich line of work examining the structure of and relationship between hidden layers of NLP transformer models via probing for specific target tasks (Adi et al., 2016; Liu et al., 2019; Belinkov et al., 2017; Hewitt and Liang, 2019). Our Section 4 borrows an experiment set-up from (Juneja et al., 2022), which found that partitioning BERT models using geometry of the fine-tuning loss surface (namely, linear mode connectivity (Frankle et al., 2019; Entezari et al., 2022)) can differentiate between models that learn *generalizing* features and *heuristic* features for a natural language inference task. Similar mode connectivity experiments for computer vision models were performed in (Lubana et al., 2022).

B Additional results



(a) (b) (c)

Figure 4: CKA for three layer GeLU and SoLU models (higher means more similar). **(a, b)** Inter-model CKA for GeLU (resp. SoLU) models with 3 transformer blocks. **(c)** Between-model CKA for 3-block GeLU,SoLU models (rows correspond to SoLU layers, columns to GeLU layers dbl check -CG.



(a) (b) (c)

Figure 5: CKA for four layer GeLU and SoLU models (higher means more similar). **(a, b)** Inter-model CKA for GeLU (resp. SoLU) models with 3 transformer blocks. **(c)** Between-model CKA for 3-block GeLU,SoLU models (rows correspond to SoLU layers, columns to GeLU layers dbl check -CG.

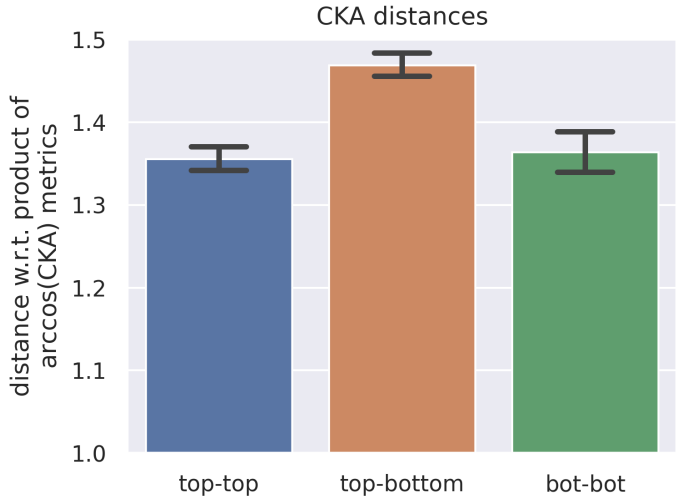
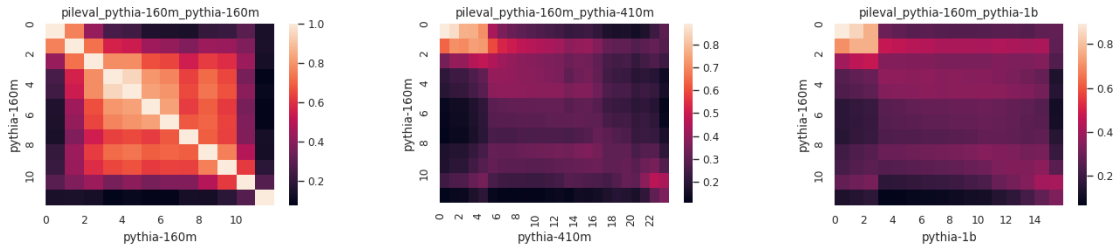


Figure 6: CKA-derived distances between the models of (McCoy et al., 2020) (higher means *less* similar, see appendix C for details on this distance metric). *Top-top*: pairwise CKAs between the top 10 models on HANS-LO. *Top-bottom*: pairwise CKAs between the top 10 and bottom 10 models on HANS-LO. *Bot-bot*: pairwise CKAs between the bottom 10 models on HANS-LO.



(a) (b) (c)

Figure 7: CKA for Pythia family models (Biderman et al., 2023) (higher means more similar). **(a)** Features within `pythia-160m`. **(b-c)** Features between `pythia-160m` and {`pythia-410m`, `pythia-1b`}.

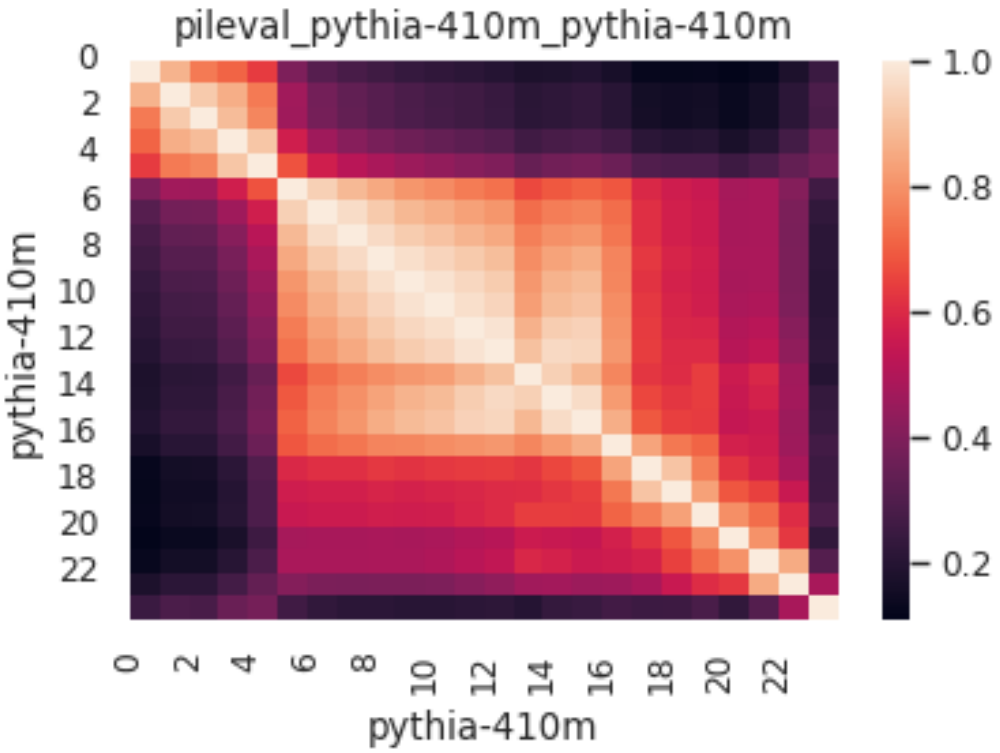
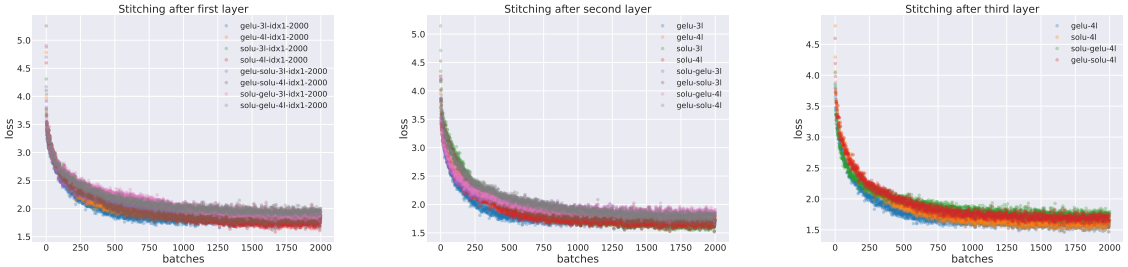


Figure 8: CKA for features within `pythia-410m` (higher means more similar).



(a) (b) (c)

Figure 9: Learning linear stitching layers for the stitching GeLU and SoLU models in Figure 1.

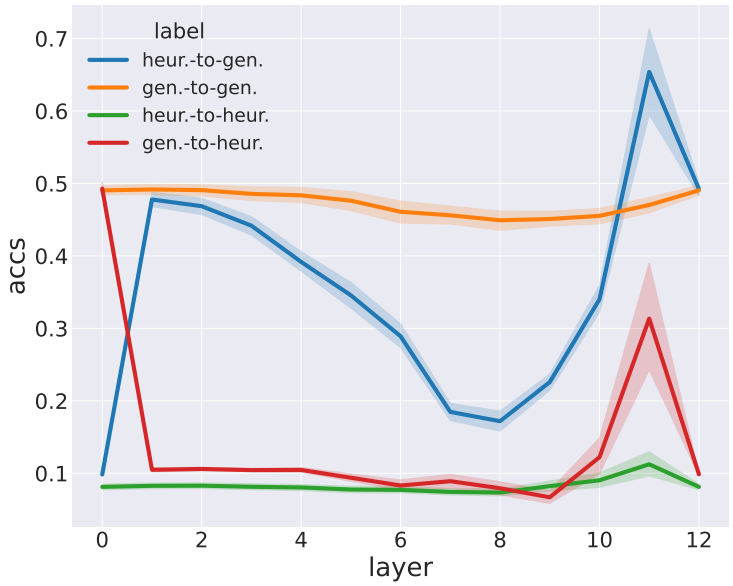


Figure 10: Identity stitching on the HANS-LO subset. The "heur.-to-heur." and "gen.-to-gen." models still exhibit distinct behavior.

C Experimental details

For the SoLU-GeLU and Pythia experiments we use the TransformerLens library (Nanda, 2022). We use a heavily adopted version of the tuned-lens library to performing stitching (Belrose et al., 2023). When computing CKA, we extract features after each transformer block (*after* addition with the residual, technically the `hook_resid_post` hook point of TransformerLens). We then flatten the tensors of shape (batch, sequence, feature) to a matrix of features,[2] and then compute CKA values in batches using an implementation of the unbiased estimator described in (Nguyen et al., 2021).

For the fine-tuned BERTs experiments we use a fork of the codebase made available by (Juneja et al., 2022) (as well as their models made available on HuggingFace (Wolf et al., 2020)). When computing CKA values, we extract features after each transformer block (in this case before addition with the residual, as it was not immediately clear how to obtain features after the residual addtion). The rest of our CKA calculation is identical to that described above.

To collapse layer-by-layer CKA to a single scalar value representing distance between the hidden features of two models with the same depth (as in fig. 6), letting $H_1, \ldots, H_L$ and $H'_1, \ldots, H'_L$ be the matrices of hidden features in the respective models and compute

$$\sqrt{\sum_{i=1}^{L} \arccos(\mathrm{CKA}(H_i, H'_i))^2}. \tag{1}$$

The explanation of this expression is as follows: as pointed out in (Williams et al., 2021), while CKA is not a metric ($H = H'$ implies $\mathrm{CKA}(H, H') = 1$, but for a metric it would have to be 0), the $\arccos$ if CKA is a metric in the mathematical sense. Equation (1) is then a metric on a product of hidden feature spaces obtained from the $\arccos$-CKA metric on each of the factors.

[2] In other words, weare comparing activations at the token level. An alternative which we have not yet evaluated would be flattening the sequence and feature indices to a single dimension and comparing activations at the sample level.