All Roads Lead to Rome? Exploring the Invariance of Transformer Representations

Yuxin Ren[1,*] **Qipeng Guo**[2] **Zhijing Jin**[3,4] **Shauli Ravfogel**[5]
Mrinmaya Sachan[4] **Bernhard Schölkopf**[3,4] and **Ryan Cotterell**[4]
[1]Tsinghua University, [2]Fudan University,
[3]Max Planck Institute for Intelligent Systems, Tübingen, Germany,
[4]ETH Zürich [5]Bar-Ilan University

Abstract

BERT models are propelling advances in various NLP tasks. However, it is unclear how consistent the BERT models are; and if they can guarantee good performance with the same architecture but with different random seeds. A reliable model architecture should be able to yield good results in all situations and should have a consistent way of solving various tasks. Unlike previous studies on the functionality of BERT models, we probe the structure of representations learned by BERT models and discuss differences between various BERT models trained with different random seeds. We present extensive experiments on Masked Language Modeling, Text Classification, and Reading Comprehension tasks. As a result, we provide key findings to help researchers understand the BERT architecture, its pre-training process, and its fine-tuning process.[1]

1 Introduction

The Transformer model has emerged as a powerful architecture for natural language processing tasks, achieving state-of-the-art results on a wide range of benchmarks. However, one issue that has received less attention is the robustness of these models to random initializations. Specifically, it is unclear whether Transformer models learn essentially *isomorphic representation spaces* or if they learn representations that are sensitive to the random initialization of their parameters.

It has been shown that small variations in the random seed used for training can result in significant performance differences between models with the same architecture (Sellam et al., 2022; Dodge et al., 2020). Understanding the degree to which the learned representation space is sensitive to random initialization is important because it has implications for interpretability efforts, i.e, understanding *what is encoded* in those models. Such efforts can include an analysis of the learned representations of LLMs by probing the information contained within these representations (Tenney et al., 2019; Rogers et al., 2020) or by observing the activations of neurons change under intervention (Ravfogel et al., 2021; Meng et al., 2022; Lasri et al., 2022). If a model's learned representation space is highly sensitive to initialization, then it may be necessary to run probing tasks multiple times with different random initializations to obtain more reliable results. Alternatively, techniques to improve the robustness of the learned representation space can be applied to make the models more suitable for probing tasks.

The representation spaces of models trained with different random initializations may be isomorphic due to two main reasons: the properties of the training data or inherent symmetries in the models. For instance, permutation symmetry, which permits the swapping of neurons in the hidden layer without altering the loss function's value, is a typical symmetry observed in neural networks (Hecht-Nielsen, 1990). However, prior research has suggested that the permutation symmetry in neural networks may not entirely capture the underlying invariances in their training dynamics (Ainsworth et al., 2022).

This paper introduces the **bijection hypothesis**, which posits the existence of a bijection mapping between the representation spaces of different models. To investigate this hypothesis, we draw inspiration from flow-based generative models (Dinh et al., 2014, 2016; Kingma and Dhariwal, 2018) and train an invertible neural network to learn the bijection function mapping.

[*] Done during the research internship with Max Planck Institute & ETH.

[1]Our codes are at `https://github.com/twinkle0331/BERT-similarity`. For general questions, contact `jinzhi@ethz.ch`. For coding questions, contact `ryx20@mails.tsinghua.edu.cn`.

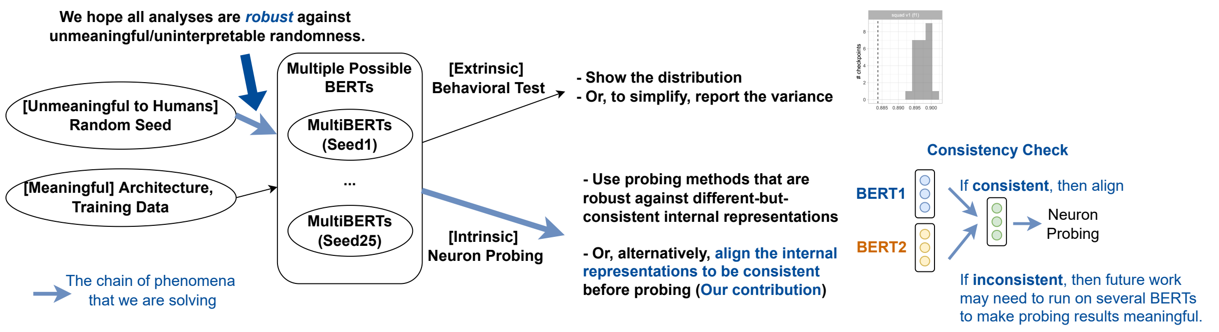

We hope all analyses are robust against unmeaningful/uninterpretable randomness.
[Unmeaningful to Humans] Random Seed
[Meaningful] Architecture, Training Data
Multiple Possible BERTs
MultiBERTs (Seed1)
...
MultiBERTs (Seed25)
[Extrinsic] Behavioral Test
[Intrinsic] Neuron Probing
- Show the distribution
- Or, to simplify, report the variance
- Use probing methods that are robust against different-but-consistent internal representations
- Or, alternatively, align the internal representations to be consistent before probing (Our contribution)
Consistency Check
BERT1
BERT2
If consistent, then align
Neuron Probing
If inconsistent, then future work may need to run on several BERTs to make probing results meaningful.
The chain of phenomena that we are solving

Figure 1: Overview of this work. Specifically, we investigate the similarity of reproduced BERT representations. Given multiple BERT models with different random seeds, we align their representation space according to our *Bijective Hypothesis*, and show clusters of BERT embeddings.

Our study aims to address several related questions. Firstly, we examine whether the representations learned by BERT models with different random seeds exhibit any degree of *invariance*. If so, we seek to determine how we can ensure that subsequent probing methods are also invariant across these varying representations. Alternatively, if the learned representations are fundamentally different, we explore whether some probing methods are still valid to some extent or if we should report the variance of probing results.

Our key contributions are as follows:

- We are, to the best of our knowledge, the first to emphasize that it is essential to measure the similarity between the hidden states across different BERT representations. [SR: it it necessary for what?]
- We pose the bijection hypothesis and train a invertible neural network to learn the bijective function mapping and recover non-linear transformations.
- We show that BERT representations are more consistent in shallower layers, and will be broken during fine-tuning, both of which have meaningful implications for future work on BERT analysis.

2 Background of LLM Analysis

2.1 Large Language Models

The recent success of nature language processing (NLP) is driven by the large language models (LLMs) (Radford et al., 2018; Devlin et al., 2019; Liu et al., 2019; Brown et al., 2020; Ouyang et al., 2022). These models are pre-trained on massive amount of text data, allowing them to learn general language representations that can be fine-tuned for specific NLP tasks. Scaling up the size of language models has been shown to confer a range of benefits, such as improved performance and sample efficiency. LLMs show strong performance in few-shot or even zero-shot tasks and a better generalization of out of distribution tasks.

2.2 Interpretability of LLMs

The empirical success of BERT on various NLP domains, encourages researchers to analyse and understand these architectures (Rogers et al., 2020). Kovaleva et al. (2019) visualize self-attention modules to probe how tokens in a sequence interact with each other. Clark et al. (2019) provide in-depth analysis on the self-attention mechanism, showing that some attention heads correspond to linguistic notions of syntax and correference. In addition, Zhao et al. (2020); Merchant et al. (2020) investigate the impacts of downstream task finetuning for BERT representations. Hewitt and Liang (2019); Pimentel et al. (2020) stress that controlling for important factors, e.g., probing classifier's capacity, is crucial to obtain meaningful probing results.

2.3 Random Seeds Can Make a Difference

Sellam et al. (2022) conduct experiments on pretrained models with different random seeds for pretraining, and find that the instance-level agreement on downstream tasks varies by the random seeds, and the gap on out-of-distribution tasks is more pronounced. Dodge et al. (2020) shows that varying only the random seed of fine-tuning process leads to substantial performance gap. The two factors influenced by random seed, training order and weight initalization, contribute equally to the variance of performance. Zhong et al. (2021) argue that model predictions are noisy at the instance level. On MNLI, even the same architecture with different finetuning seeds will lead to different predictions on ≈8% of the instances, due to under-specification (D'Amour et al., 2020).

3 The *Bijection Hypothesis*

3.1 Representation Spaces of BERTs

Given two BERT models $\mathcal{M}_1$ and $\mathcal{M}_2$ with identical architecture but different random initializations, the goal is to evaluate their similarity through their representation spaces.

We denote the sets of BERT parameters after the pre-training as $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$, respectively. Their representation transformation functions are thus $f_{\boldsymbol{\theta}_1}$ and $f_{\boldsymbol{\theta}_2}$, respectively. Since a BERT has 12 transformer layers (Vaswani et al., 2017), it is a hyperparameter to specify the representation space of which transformer layer we want to analyze. Let us denote the layer of interest as l. Then, the sequence transformation function works as follows. For any text sequence $\boldsymbol{t}$ with k tokens, we first get its token-wise embedding $\boldsymbol{T} \in \mathbb{R}^{k\times 768}$ at Layer l by $\boldsymbol{T} = \mathrm{Transformer}_{1,\dots,l}(\boldsymbol{t}) = \mathrm{FFN}_l(\mathrm{Att}_l(\dots \mathrm{FFN}_1(\mathrm{Att}_1(\mathrm{Emb}(\boldsymbol{t}))))),$ where FFN_i and Att_i denotes the feedforward layer and the self-attention network at Layer i, respectively, and Emb denotes the embedding layer. Then we follow the practice in Del and Fishel (2021); Wu et al. (2019); Zhelezniak et al. (2019) to do a mean-pooling operation to obtain the overall sentence embedding $\frac{1}{k}\sum_{i=1}^{k} \boldsymbol{T}_{i,*}$.

For the two transformation functions $f_{\boldsymbol{\theta}_1}$ and $f_{\boldsymbol{\theta}_2}$, we feed a large list of n arbitrary text inputs, $\boldsymbol{D} := [\boldsymbol{t}_1, \boldsymbol{t}_2, \dots, \boldsymbol{t}_n]$ to obtain the two sentence representation spaces:

$$\boldsymbol{X} := [f_{\boldsymbol{\theta}_1}(\boldsymbol{t}_1), \cdots, f_{\boldsymbol{\theta}_1}(\boldsymbol{t}_n)] \text{ for } \mathcal{M}_1 \tag{1}$$
$$\boldsymbol{Y} := [f_{\boldsymbol{\theta}_2}(\boldsymbol{t}_1), \cdots, f_{\boldsymbol{\theta}_2}(\boldsymbol{t}_n)] \text{ for } \mathcal{M}_2 \,. \tag{2}$$

Here, $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^{n\times d}$, where d is the dimension of these embeddings, or the number of features.

3.2 Similarity = Bijective Mapping

We propose the *Bijection Hypothesis*, which states that two models are perfectly similar if and only if there exists a diffeomorphism in their representation spaces. The mapping can be discovered through techniques such as linear or non-linear projections.

Namely, for the two spaces $\boldsymbol{X}$ and $\boldsymbol{Y}$, we learn an invertible function g, such that we minimize the distance metric δ between the transformed $g(\boldsymbol{X})$ and the original $\boldsymbol{Y}$. The similarity of the two spaces can be quantified by the distance metric, and if $\delta(g(\boldsymbol{X}), \boldsymbol{Y})$ is zero, then the two spaces are perfectly similar.

3.3 Reformulating Existing Similarity Indices into Linear Bijective Functions

We briefly introduce existing methods to align the two representation spaces before measuring their similarity.

Linear Regression. An intuitive way to learn a bijective function for aligning two representation spaces is linear regression, which can capture all linear transformations by minimizing the L2 norm. Namely, we learn a matrix $\boldsymbol{W}$ which satisfies the following objective:

$$\min_{\boldsymbol{W}} \delta(\boldsymbol{X}\boldsymbol{W}, \boldsymbol{Y}) = \min_{\boldsymbol{W}} ||\boldsymbol{X}\boldsymbol{W} - \boldsymbol{Y}||_2 \,.$$

Canonical Correlation Analysis (CCA). CCA projects the two matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$ into the subspaces which maximize the canonical correlation between their projections. The formulation is as follows:

$$\rho_i = \max_{\boldsymbol{w}_X^i, \boldsymbol{w}_Y^i} \mathrm{corr}(\boldsymbol{X}\boldsymbol{w}_X^i, \boldsymbol{Y}\boldsymbol{w}_Y^i) \tag{3}$$
$$\text{for} \quad \forall_{j<i} \;\; \boldsymbol{X}\boldsymbol{w}_X^i \perp \boldsymbol{X}\boldsymbol{w}_X^j \,, \tag{4}$$
$$\forall_{j<i} \;\; \boldsymbol{Y}\boldsymbol{w}_Y^i \perp \boldsymbol{Y}\boldsymbol{w}_Y^j . \tag{5}$$

Here, the objective is to maximize the correlation between all pairs of projected subspaces of $\boldsymbol{X}$ and $\boldsymbol{Y}$, and the constraint is to keep the orthogonality among all the subspaces of each original space.

Since the dimension of CCA is equal to d, by concating the projection vectors into matrices $\boldsymbol{W}_X$ and $\boldsymbol{W}_Y$, both $\boldsymbol{X}$ and $\boldsymbol{Y}$ are invertible matrices. The transformation matrix from $\boldsymbol{X}$ to $\boldsymbol{Y}$ is then equivalent to the product of $\boldsymbol{W}_X$ and the inverse of $\boldsymbol{W}_Y$:

$$\boldsymbol{W}_X \boldsymbol{W}_Y^{-1} \tag{6}$$

However, since we have to pre-determine the number of canonical correlations c, CCA suffers when the guessed c differs from the optimal one.

Noise-Robust Improvements of CCA. As CCA might suffer from outliers in the two spaces, there are two types of improvements that are more robust towards outliers or noises in the spaces.

The first type is singular vector CCA (SVCCA), which performs singular value decomposition (SVD) to extract directions which capture most of the variance of the original space, before running the CCA. Noisy dimensions are removed by SVD.

The other method is projection-weighted CCA (PWCCA), which computes a weighted average over the subspaces learned by CCA: [SR: I think the discussion is too verbose and complicates things. We should just say we first perform SVD to remove noisy dimensions, and then run regular CCA.] [Yuxin: Sure, both PWCCA and SVCCA try to mitgate the influence brought by noise dimesnions.]

$$\rho_{\text{PWCCA}} = \frac{\sum_{i=1}^{c} \alpha_i \rho_i}{\sum_{i=1}^{c} \alpha_i} \tag{7}$$

$$\alpha_i = \sum_j |\langle \boldsymbol{p}_i, \boldsymbol{x}_j \rangle| \tag{8}$$

$$\boldsymbol{p}_i = \boldsymbol{X} \boldsymbol{w}_X^i \ , \tag{9}$$

where $\boldsymbol{x}_j$ is the j-th feature of the $\boldsymbol{X}$ space, namely, the j-th column. The weight α_i for each correlation measure ρ_i is calculated as how similar the i-th subspace projection $\boldsymbol{p}_i$ is to all the features $\boldsymbol{x}_j$ of the original space $\boldsymbol{X}$.

However, SVCCA and PWCCA are still ***linear*** methods and unable to capture non-linear relationship between $\boldsymbol{X}$ and $\boldsymbol{Y}$. Further, Csiszárik et al. (2021) shows that the low-rank transformation learned by SVD is not as effective as learning a matching under a task loss and the effectiveness of learning a matching stems from the non-linear function in a network.

3.4 Non-Linear Bijection by INNs

All the linear bijective methods introduced previously do not necessarily suit BERTs, as BERT learns non-linear feature transformation through the non-linear activation layers GELU (Hendrycks and Gimpel, 2016). Between two BERT models, there might be a complex, non-linear correspondence between their feature spaces. Hence, we propose the use of INNs to discover bijective mappings between the two BERTs.

INNs learn a neural network-based *invertible* transformation from the latent space $\mathcal{U}$ to the observed space $\mathcal{Z}$. Using it as our bijective method, we learn an invertible transformation $g(\cdot)$ from $\boldsymbol{X}$ to $\boldsymbol{Y}$. Specifically, we optimize the parameters $\boldsymbol{\theta}_g$ of the INN function by minimizing the L2 distance δ between the transformed $g_{\boldsymbol{\theta}}(\boldsymbol{X})$ and $\boldsymbol{Y}$:

$$\min_{\boldsymbol{\theta}_g} \delta(g_{\boldsymbol{\theta}_g}(\boldsymbol{X}), \boldsymbol{Y}) \ . \tag{10}$$

Intuitively, the invertibility constraint verifies that the mapping between the two representation spaces is *isomorphic*: there is no loss of information when mapping from $\boldsymbol{X}$ to $\boldsymbol{Y}$.

For the implementation of the INN, we draw inspirations from RealNVP (Dinh et al., 2016) and propose BERT-INN, which stacks affine coupling layers to learn the bijective function g. In each affine layer, a subset of input vector is transformed through a function that is simple to invert, but which is determined by the remainder of the input vector in a non-trivial manner. [SR: I'd just say we use a standard architecture and move the details to the appendix.]

Given an input vector $\boldsymbol{x}$ of dimension K, and a subset of dimension $k < K$, the output $\boldsymbol{y}$ of an affine coupling layer is defined by the following equations following the commonly used INN formulation (XXX, 2014):

$$\boldsymbol{y}_{1:k} = \boldsymbol{x}_{1:k} \tag{11}$$

$$\boldsymbol{y}_{k+1:K} = \boldsymbol{x}_{k+1:K} \odot \exp(s(\boldsymbol{x}_{1:k})) + t(\boldsymbol{x}_{1:k}) \ , \tag{12}$$

where s and t stand for scale and translation, and are functions from $\mathbb{R}^k \to \mathbb{R}^{K-k}$.

A forward transformation in one affine coupling layer preserves a subset of input vectors, hence an alternating pattern is adopted such that the input vectors that are preserved in one coupling layer are transformed in the next.

4 Alignment Experiments

4.1 Experimental Setup

Reproduced BERT Models We conduct the experiments using the 25 reproduced BERT models released by Sellam et al. (2022). The BERT models vary only by random seeds, but adopt the same training procedure as in the original BERT model (Devlin et al., 2019).

Datasets To get the BERT representations, we run the 25 trained BERT models in the inference mode, and plug in a variety of datasets and tasks. Specifically, we choose three diverse tasks from the commonly used GLUE benchmark (Wang et al., 2019): the natural language inference dataset MNLI (Williams et al., 2018), the sentiment classification dataset SST-2 (Socher et al., 2013), and the paraphrase detection dataset MRPC (Dolan and Brockett, 2005). These datasets also cover a variety of domains from news, social media, and movie reviews, which are the common domains for many NLP datasets. The dataset statistics are in Table 1.

Dataset	Task	Train	Test	Domain
MNLI	Inference	393K	20K	Mixed
SST-2	Sentiment	67K	1.8K	Movie reviews
MRPC	Paraphrase	3.7K	1.7K	News

Table 1: Dataset statistics from the GLUE benchmark (Wang et al., 2019).

Baselines We compare four common existing bijective methods introduced in Section 3.4, namely linear regression, CCA, SVCCA, and PWCCA. Note that PWCCA sometimes generates results same as or similar to CCA, because it learns the mapping using the same principles as the CCA, and then computes a weighted average over the coefficients, which could generate similar results to CCA under some conditions.

Evaluation Setup For each method g, we evaluate their alignment performance by measuring the distance between two spaces: the ground-truth target space $\boldsymbol{Y}$, versus the learned space $g(\boldsymbol{X})$ transformed from the original $\boldsymbol{X}$ with the alignment method $g(\cdot)$. We adopt the a widely used distance function, Euclidean distance, also known as the L2 distance, to measure the distance between the two spaces, namely $\delta(g(\boldsymbol{X}), \boldsymbol{Y}) := ||\boldsymbol{Y} - g(\boldsymbol{X})||_2$. Since our goal is to align the two spaces, the smaller the L2 distance, the better, with the best value being zero. For SVCCA, since it reduces the dimension of the original spaces, we compute the L2 norms between the dimension-reduced matrices.

Since we have 25 BERT checkpoints, to save the computational cost to not iterate through all C(25, 2) = 300 combinations, we randomly select 8 model pairs to report their L2 distances under all five methods. [SR: can we increase this number to, like, 50? :p] [Yuxin: The pre-training is expensive, so the MultiBERTs only release 25 BERTs under different random seeds for training, hence, 25 BERTs are all we can use.] [SR: But I think the sentence says we use 8 random pairs out of 300? I meant increasing the number of pairs.] [Yuxin: Since for each model pair, we train the models on the representations from 12 layers. The training cost for all methods over 50 model pairs is much more expensive, 8 model pairs is a trade-off. So for the majority of our experiments, we only conduct on 8 model pais. For the boxplot figure, we compute the score on all model pairs(300 pairs).] [SR: I'm not sure I understand. can we add a short paragraph explaining which models we use and when?] [Yuxin: We randomly choose 8 model pairs from the 300 possible model pairs, so the models are randomly choosen.]

4.2 Theoretical Properties

	Transformation Type	Non-Lin.	Noise-Robust
Lin. Reg.	$\|\|Q_Y^{\mathrm{T}} X\|\|_{\mathrm{F}}^2 / \|\|X\|\|_{\mathrm{F}}^2$	✗	✗
CCA	$\|\|Q_Y^{\mathrm{T}} Q_X\|\|_{\mathrm{F}}^2 / d$	✗	✗
SVCCA	$\|\|(U_Y T_Y)^{\mathrm{T}} U_X T_X\|\|_{\mathrm{F}}^2 / \min(\|\|T_X\|\|_{\mathrm{F}}^2, \|\|T_Y\|\|_{\mathrm{F}}^2)$	✗	✓
PWCCA	$\sum_{i=1}^{d} \alpha_i \rho_i / \|\|\alpha\|\|_1$	✗	✓
INN (Ours)	Any invertible function	✓	✓

Table 2: Theoretical properties of all the alignment methods. We focus on properties such as whether the method can handle non-linear mapping between the two spaces ("Non-Lin."), and whether the method is robust against noise in the transformation ("Allow Noise").

We compare the theoretical properties of all the alignment methods in Table 2. [SR: I'd consider removing this table. I'd also consider removing the following paragraph. Both CCA and regression are linear, so the comparison doesn't make that much sense.] First, we base our analysis on the transformation types that each method can handle: linear regression allows for a linear transformation; CCA enhances the alignment of network representations by transforming the original data into a set of canonical variables, which maximizes the correlation between the subspaces; both PWCCA and SVCCA extend CCA by filtering out noise from meaningful signals. In contrast, our proposed INN can serve as a general function approximator to model any invertible transformation between the two spaces.

As a result of nature of the transformation types, we can see in the two right-most columns of Table 2 that all previous methods can only handle linear transformations, whereas our INN can model non-linear, invertible mappings. Moreover, linear regression and CCA are sensitive towards noise in the embedding space; SVCCA and PWCCA improves upon CCA by filtering out the noise; and our model is also robust against noises in that the training objective of INN is to minimize the overall loss, and is robust against outliers. [SR: I don't understand this argument. INN uses a variant of the L_2 loss which is by definition sensitive to outliers.]

4.3 Empirical Performance on Synthetic Data

We also show empirical evidence of the effectiveness of our method over the previous baselines. The first empirical test is on synthetic data, where we compose the two spaces, and make sure the ground-truth relation between the two spaces is perfect alignment. [SR: Consider presenting this differently. We want to understand whether INNs are capable of *identifying* the correspondence between the two representation spaces. We cannot do that with raw BERT representations, since we (1) don't know whether such correspondence even exists, and (2) don't know its form, if it exists. Therefore we generate synthetic data where the correspondence is known, and we measure the ability of INNs to identify it.]

	L2 Distance $\delta(g(\boldsymbol{X}), \boldsymbol{Y})$ (↓)
Linear Regression	$1.0882_{\pm 0.3867}$
CCA	$0.3060_{\pm 0.1948}$
SVCCA	$0.1396_{\pm 0.1246}$
PWCCA	$0.3060_{\pm 0.1948}$
INN (Ours)	$\mathbf{0.0059}_{\pm 0.0037}$

Table 3: Empirical performance of all the alignment methods to recover a non-linear, invertible transformation. For L2 distance (↓), the smaller, the better, and the best possible L2 distance is zero, which means the learned mapping recovers the ground-truth transformation between the two spaces.

Specifically, we make the first space $\boldsymbol{X}$ a real BERT representation space generated by Sellam et al. (2022). For the second subspace $\boldsymbol{Y}$, we generate it by applying a randomly initialized INN f to the first space $\boldsymbol{X}$, namely $\boldsymbol{Y} = f(\boldsymbol{X})$ as the ground truth.

Then, without any knowledge of the transformation f, we apply all methods to obtain their corresponding mapping function g given the two spaces $\boldsymbol{X}$ and $\boldsymbol{Y}$. For each method, we evaluate their alignment performance by measuring the distance between the learned $g(\boldsymbol{Y})$ and ground-truth $\boldsymbol{Y}$ using L2 distance, namely $\delta(g(\boldsymbol{Y}), \boldsymbol{X}) := ||g(\boldsymbol{Y}) - \boldsymbol{X}||_2$. Note that since the ground truth mapping f gives an L2 distance of zero, when interpreting the results, the lower L2 distance, the better an alignment method is.

We show the performance of our method compared with all the baselines in Table 3. All baselines (linear regression, CCA, SVCCA, and PWCCA) struggle to learn a good mapping between the two spaces, with an L2 distance always larger than 0.13, which is quite large. In contrast, our INN method gives a very small L2 distance, 0.0059, which almost recovers the ground-truth transformation. The results of this experiment echoes with the theoretical properties analyzed previously in Section 4.2.

4.4 Representation Similarity on Real Data

We now move from synthetic data to real NLP datasets. As introduce in Section 4.1, we use three diverse datasets: MNLI (Williams et al., 2018), SST-2 (Socher et al., 2013), and MRPC (Dolan and Brockett, 2005).

We report the alignment results in Table 4. For each method, we use the splits in the original data, learn the mapping using the training set, (if applicable,) tune the hyperparameters on the validation set, and finally report the performance on the test set. For some additional references, we also report the training set performance (to check how well the method fits the data), in addition to the test set performance (as the final alignment performance).

Among all the bijective methods in Table 4, our INN method leads to the smallest L2 distance on all datasets by a clear margin.

Additionally, since each BERT has 12 layers, we visualize the L2 norm across all layers in **??**. The visualization provide a more fine-grained view into the layer-wise alignment quality behind the overall numbers reported in Table 4. And consistent to Table 4, INN can effectively align the neurons across layers with a much lower L2 distance.

4.5 Functional Similarity on Real Data

[SR: I added this intro.] The results presented thus far showcase the effective mapping capability of INNs between the representation spaces of two models. However, it remains unclear how applying this mapping would impact the actual *behavior* of the models. If an accurate bijection between the representation spaces of two models is established, it should be possible to *intervene* during the

	MNLI		SST2		MRPC	
	Train	Test	Train	Test	Train	Test
CCA	$0.1484_{\pm 0.0763}$	$0.1534_{\pm 0.0802}$	$0.3064_{\pm 0.1791}$	$0.2535_{\pm 0.1784}$	$0.1455_{\pm 0.0663}$	$0.1502_{\pm 0.0668}$
SVCCA	$0.1216_{\pm 0.1011}$	$0.1180_{\pm 0.0989}$	$0.2399_{\pm 0.2322}$	$0.1997_{\pm 0.2375}$	$0.1033_{\pm 0.1136}$	$0.0852_{\pm 0.0730}$
PWCCA	$0.0546_{\pm 0.0441}$	$0.0529_{\pm 0.0418}$	$0.1095_{\pm 0.1214}$	$0.0877_{\pm 0.1080}$	$0.0424_{\pm 0.0344}$	$0.0391_{\pm 0.0288}$
Lin. Reg.	$0.1003_{\pm 0.0799}$	$0.1024_{\pm 0.0822}$	$0.0884_{\pm 0.0533}$	$0.0896_{\pm 0.0623}$	$0.0618_{\pm 0.0392}$	$0.0641_{\pm 0.0424}$
INN (Ours)	$\mathbf{0.0520}_{\pm 0.0388}$	$\mathbf{0.0461}_{\pm 0.0336}$	$\mathbf{0.0563}_{\pm 0.0315}$	$\mathbf{0.0523}_{\pm 0.0401}$	$\mathbf{0.0299}_{\pm 0.0165}$	$\mathbf{0.0310}_{\pm 0.0180}$
Sanity Check:						
Non-Bij. NN	$0.0360_{\pm 0.0256}$	$0.0371_{\pm 0.0272}$	$0.0479_{\pm 0.0279}$	$0.0449_{\pm 0.0352}$	$0.0247_{\pm 0.0145}$	$0.0263_{\pm 0.0165}$

Table 4: Alignment performance on MNLI, SST2, and MRPC datasets using CCA, SVCCA, PWCCA, linear regression (Lin. Reg.), and our INN method. We report the L2 norm δ ($\downarrow$) after aligning the sentence space on the training set (just for a reference) and test set (as the main result). The training and test samples were randomly selected from the respective datasets, with a sample size of 1600. For sanity check, we also report the performance of a non-bijective neural network (Non-Bij. NN), which should lead to the smallest L2 norm because it has less constraint than INN, although at the expense of violating the Bijective Hypothesis. [SR: Reminder: need to show **relative** reconstruction errors across the paper.] [Yuxin: Since the rerunning all experiment takes too much time, here we only compute the mean MSE loss among the 12 layers. The inputs keep the same for all methods, so the comparsion is fair. For the boxplot figure in Figure 2, we show the relative reconstruction errors.] [SR: Not sure I understand. Why does it matter that the input is the same? two models can differ in mean feature magnitude across all layers.]

forward pass of one model by substituting its representation with the corresponding representation from the other model, without causing significant loss. In this section, we conduct a behavioral analysis to investigate this hypothesis. We design the following "layer injection" experiment: For each pair of BERTs $\mathcal{M}_1$ and $\mathcal{M}_2$ finetuned on a downstream classification task (e.g., MNLI, SST2, or MRPC), we denote the embedding spaces of the last layer before the classification as $\boldsymbol{X}$ for model $\mathcal{M}_1$, and $\boldsymbol{Y}$ for model $\mathcal{M}_2$. We learn a transformation function g to map $\boldsymbol{X}$ to $\boldsymbol{Y}$. , we generate the transformed embedding $g(\boldsymbol{X})$ and inject it into the second model $\mathcal{M}_2$. With this direct replacement, we keep all the other weights in $\mathcal{M}_2$ unchanged, and run it in the inference mode to get the performance on all three datasets. Our experiment design carries a similar spirit to the existing work in computer vision on convolutional neural networks (Csiszárik et al., 2021).

As we can see in Table 5, our model generates the best performance across all datasets, which is quite impressive compared with the close-to-random performance of many other methods. Moreover, our performance is even better than that of the non-bijective neural networks, which might indicate that the non-bijective transformation is not the proper way to go. It might distort some features in the embedding space, making it incompatible with the following layers of the injected

	MNLI	SST2	MRPC
Random Embe.	$33.12_{\pm 14.63}$	$53.97_{\pm 26.83}$	$42.49_{\pm 23.89}$
Random Label	$33.55_{\pm 0.43}$	$50.44_{\pm 1.20}$	$52.9_{\pm 2.51}$
Lin. Reg.	$37.15_{\pm 24.26}$	$54.14_{\pm 31.27}$	$45.93_{\pm 28.48}$
CCA	$30.37_{\pm 9.30}$	$45.46_{\pm 12.36}$	$51.72_{\pm 18.60}$
PWCCA	$39.70_{\pm 4.57}$	$50.24_{\pm 0.90}$	$50.00_{\pm 18.38}$
Non-Bij. NN	$29.61_{\pm 15.37}$	$64.25_{\pm 19.78}$	$53.58_{\pm 25.76}$
INN (Ours)	$\mathbf{75.75}_{\pm 8.97}$	$\mathbf{74.01}_{\pm 15.61}$	$\mathbf{86.34}_{\pm 1.07}$

Table 5: Behavioral similarity check for bijective function mapping methods across three datasets (MNLI, SST2, and MRPC). We report accuracy performance for the three datasets. The performance are evaluated on the validation set. The first method is pretrained BERT embedding connected with a randomly initialized classifier.

BERT. Previous work also shows a discrepancy between representational similarity and behavior similarity.

5 Findings: Variance and Invariance across BERTs

In this section, we utilize the INN alignment method on 25 replicated BERTs to address a the following question:

> *Are these BERTs (which only differ by random seeds) fundamentally similar, or different?*

This question holds significance for the NLP community because both answers matter for future work directions: If these BERTs are similar, then all the work built on one single BERT will generalize to other BERTs. And if these BERTs differ, then follow-up work on BERT should also test their findings on the other BERTs varied by random seeds, to make sure the findings are robust.

To answer this overall question, we ask three subquestions from different perspectives: (1) Across the 12 representation layers, which ones are similar, and which ones are different? (2) How does the way how hidden states interact with each other (i.e., attention weights) vary? (3) How does finetuning enlarge or reduce the differences?

To get more representative conclusions, we run BERT models on the largest dataset, MNLI. [SR: not sure this argument is convincing...][Yuxin: There maybe slight difference in different tasks, but it is hard to show empirical evidence to explain the differences. So the main experiments is conducted on MNLI.The experiments on other datasets is shown in the appendix.]

5.1 Deep vs. Shallow Layers

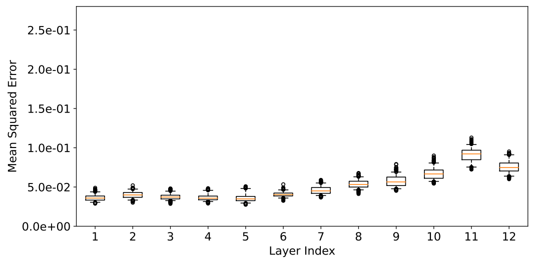


Figure 2: Similarity of the *hidden states* learned in the 12 layers of BERTs. For each layer, we align the 25 BERTs and plot the distribution of their L2 distances after alignment. The shallow layers (Layer 1 – 6) are similar, with a small L2 distance of around 0.02, whereas the deep layers (Layer 7 – 12) start to be more different, all the way to a large L2 distance of over 0.06.

We plot the similarity of the representation spaces across the 12 layers of BERT in Figure 2. As we can see, the shallow layers (Layer 1 – 6) are relatively similar, all having an L2 distance close to 0.02, which is very small. However, the deeper layers (Layer 7 – 12) start to see an increase in inconsistency, with a larger L2 norm which rises from 0.03 to over 0.06, tripling the distance of shallow layers.

One potential explanation for the observed results is that the shallower layers of the model tend

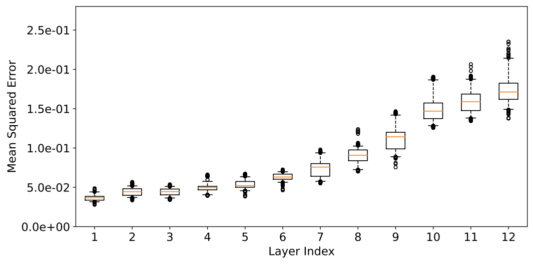


Figure 3: Similarity of the 12 layers of a BERT. We can see that the transformation in different layers are more similar for layers next to each other, and differ a lot when the layers are far away from each other.

to learn more low-level features that are shared across various texts, whereas the deeper layers are capable of capturing more complex semantic features that can have multiple plausible interpretations. To support this claim, we present the results of our INN method, which measure the similarity across all 12 layers, in Figure 3. Here are three notable findings from our analysis:

(1) Two areas of high similarity, represented by whiteness, are present in the plot, suggesting that the representations of the shallow layers are similar among themselves, as are the representations of the deep layers. (2) The 12th layer behaves similiarly to the 7th layer, which might mean that it needs to contain information from both low-level features and high-level features in order to predict the word correctly. [SR: I'd remove point 2.] (3) The difference in the representation space changes gradually across the layers, without any abrupt changes.

5.2 Representation vs. Interaction

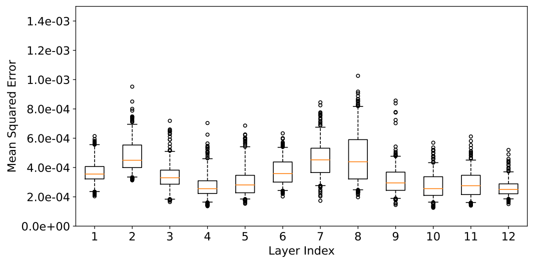
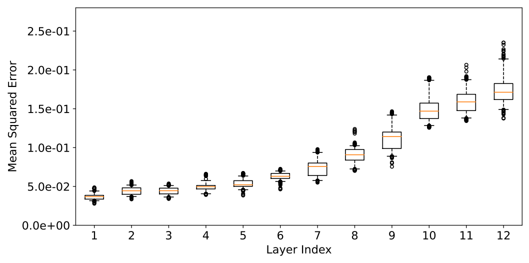


Figure 4: Similarity of the *attention weights* in the 12 layers of BERTs. For each layer, we align the 25 BERTs and plot the distribution of their L2 distances after alignment.

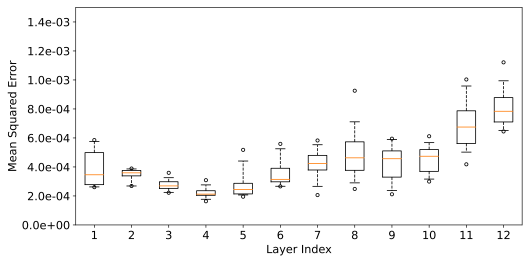
The transformer architecture of BERT learns not only the hidden representations, but also the way how the hidden states should *interact*, which are expressed by the weight matrices in the attention layers.

We inspect the similarity of the attention weights in Figure 4. Surprisingly, the attention weights are far more invariant than the representations, with L2 distances lower by two orders of magnitude, ranging from 0.0002 to 0.0004. This indicates that the way the hidden states interact tend to be very similar across all BERTs.

5.3 Before vs. After Finetuning



(a) Similarity of the *hidden states after* fintuning. We can see that the shallow layers stay invariant, whereas the deep layers vary even more than those before finetuning in Figure 2.



(b) Similarity of the *attention weights after* fintuning. We can see that the invariance is mostly kept across all layers.

Figure 5: Similarity across the 12 layers of BERTs after finetuning. The deep representations tend to vary more, whereas the shallow representations and attention mechanisms tend to stay similar.

Finally, we inspect how finetuning affects the similarity among BERTs. In Figure 5(a), we can see that, although the shallow layers (1 – 6) remain similar after finetuning, the deep layers (7 – 12) vary drastically, reaching a high L2 distance of up to 0.12, which doubles its original maximum L2 distance of 0.06 in Figure 2. On the contrary, the way how the hidden states interact stays more invariant, as we can see that the attention weights stay similar across the BERTs, with L2 distances still mostly around 0.0002 to 0.0004 in Figure 5(b).

Ethical Considerations

This work aims to measure the similarity of the learned representation space of the commonly used NLP model, BERT. This is a step towards safer model deployment, and more transparency of the black-box LLMs. As far as we are concerned, there seem to be no obvious ethical concerns for this work. All datasets used in this work are commonly used benchmark datasets in the community, and there is no user data involved.

This project originates from a discussion between Zhijing and Qipeng in 2020. Inspired by Bernhard's causal representation learning (Schölkopf et al., 2021) and promising results in CV, they did some preliminary exploration into running ICA on BERT embeddings as a first step of Causal Representation Learning for NLP. When Yuxin joins MPI and ETH as an intern with Zhijing, he discovers that ICA shows some results that could vary a lot by the random seed, which inspires this work on inspecting the what random seeds bring to the BERT models. [...]

Acknowledgment

This material is based in part upon works supported by the German Federal Ministry of Education and Research (BMBF): Tübingen AI Center, FKZ: 01IS18039B; by the Machine Learning Cluster of Excellence, EXC number 2064/1 – Project number 390727645; by the John Templeton Foundation (grant #61156); by a Responsible AI grant by the Haslerstiftung; and an ETH Grant (ETH-19 21-1). Zhijing Jin is supported by PhD fellowships from the Future of Life Institute and Open Philanthropy.

References

Samuel K Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. 2022. Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint arXiv:2209.04836*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286.

Adrián Csiszárik, Péter Kőrösi-Szabó, Ákos Matszangosz, Gergely Papp, and Dániel Varga. 2021. Similarity and matching of neural network representations. *Advances in Neural Information Processing Systems*, 34.

Maksym Del and Mark Fishel. 2021. Establishing interlingua in multilingual language models. *arXiv preprint arXiv:2109.01207*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Association for Computational Linguistics (ACL)*, pages 4171–4186.

Laurent Dinh, David Krueger, and Yoshua Bengio. 2014. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. 2016. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Finetuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv*.

Bill Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Third International Workshop on Paraphrasing (IWP2005)*.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, et al. 2020. Underspecification presents challenges for credibility in modern machine learning. *Journal of Machine Learning Research*.

Robert Hecht-Nielsen. 1990. On the algebraic structure of feedforward network weight spaces. In *Advanced Neural Computers*, pages 129–135. Elsevier.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. 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 2733–2743, Hong Kong, China. Association for Computational Linguistics.

Durk P Kingma and Prafulla Dhariwal. 2018. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of bert. 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 4365–4374.

Karim Lasri, Tiago Pimentel, Alessandro Lenci, Thierry Poibeau, and Ryan Cotterell. 2022. Probing for the usage of grammatical number. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8818–8831.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *arXiv preprint arXiv:2202.05262*.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What happens to bert embeddings during fine-tuning? *arXiv preprint arXiv:2004.14448*.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *CoRR*, abs/2203.02155.

Tiago Pimentel, Naomi Saphra, Adina Williams, and Ryan Cotterell. 2020. Pareto probing: Trading off accuracy for complexity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3138–3153, Online. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Technical report, OpenAI.

Shauli Ravfogel, Grusha Prasad, Tal Linzen, and Yoav Goldberg. 2021. Counterfactual interventions reveal the causal effect of relative clause representations on agreement prediction. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 194–209.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics (TACL)*, 8:842–866.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. 2021. Towards causal representation learning. *CoRR*, abs/2102.11107.

Thibault Sellam, Steve Yadlowsky, Jason Wei, Naomi Saphra, Alexander D'Amour, Tal Linzen, Jasmijn Bastings, Iulia Turc, Jacob Eisenstein, Dipanjan Das, et al. 2022. The multiberts: Bert reproductions for robustness analysis. In *ICLR*. OpenReview.net.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Alex Wang, Amapreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations (ICLR)*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Association for Computational Linguistics (ACL)*, pages 1112–1122.

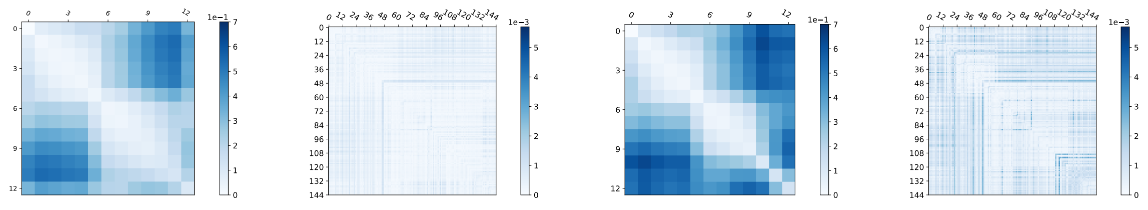
Shijie Wu, Alexis Conneau, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Emerging cross-lingual structure in pretrained language models. *arXiv preprint arXiv:1911.01464*.

Mengjie Zhao, Philipp Dufter, Yadollah Yaghoobzadeh, and Hinrich Schütze. 2020. Quantifying the contextualization of word representations with semantic class probing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1219–1234, Online. Association for Computational Linguistics.

Vitalii Zhelezniak, April Shen, Daniel Busbridge, Aleksandar Savkov, and Nils Hammerla. 2019.

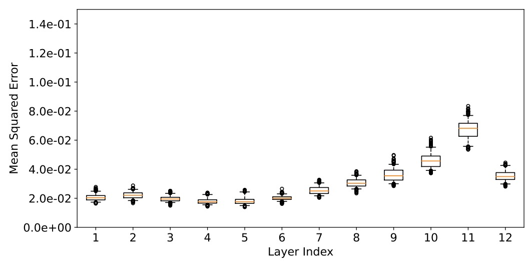
Correlations between word vector sets. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 77–87.

Ruiqi Zhong, Dhruba Ghosh, Dan Klein, and Jacob Steinhardt. 2021. Are larger pretrained language models uniformly better? comparing performance at the instance level. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3813–3827, Online. Association for Computational Linguistics.

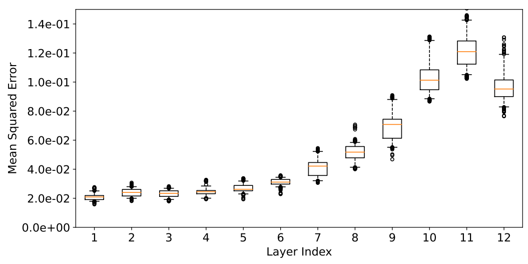


(a) (b) (c) (d)

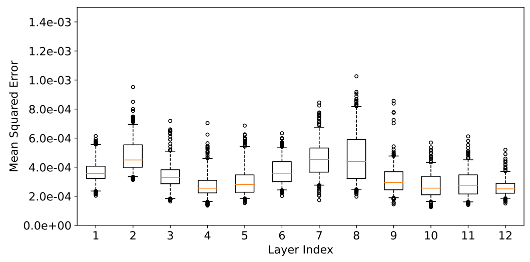
Figure 6: (a) Mutual feature similarity of fine-tuned models on MRPC. (b) Mutual attention similarity of pre-trained models on MRPC. (c) Mutual feature similarity of fine-tuned models on SST-2. (d) Mutual attention similarity of fine-tuned models on SST-2.



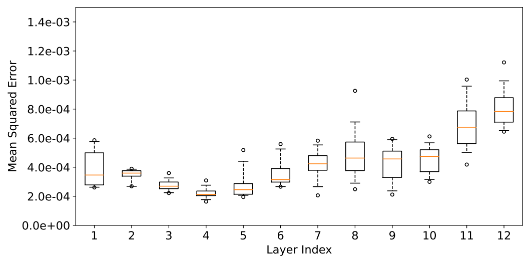
(a) Similarity of *hidden states before* fintuning.



(b) Similarity of *hidden states after* fintuning.



(c) Similarity of *attention weights before* fintuning.



(d) Similarity of *attention weights after* fintuning.

Figure 7: The fine-tuning task is MNLI.

A Self-similarity for Other Tasks

B Mutual-similarity for Other Tasks

B.1 Copy of 2x2 plots

B.2 Invariance in Shallower Layers & Variance in Deeper Layers

Conjecture BERT stacks SAN and FFN layers repeatedly to form a complete model without changing the inner structure, so the depth of a layer becomes an important factor to describe intrinsic properties. Low-level layers close to the input while the high-level layers close to the output.

A clear fact is that random seeds do not affect the data distribution, so a rational conjecture is that the layers which is close to the input should be consistent since all of them close to the input distribution. The same conjecture may not hold for the layers close to the output since super power of non-linearity can match any distribution to the output distribution. It is promised if the neural network is large enough according to Inverse transform sampling () theory.

Results By observing embedding layers and low-level layers, shown in Figure-7, we verified that their representations are almost invariant to random seeds, and this trend still holds in mid-level layers (up to the 8th layer). However, the consistency become weak in high-level layers, we can see a significant drop in 9th layer. This phenomena suggests that models trained with different random seeds tend to use different logic to make the final prediction. From a traditional machine learning view, we can treat low- and mid-level layers as a feature extractor, and the last few layers as a classifier. Thus, our probing suggests that pre-training brings a consistent and reliable feature extractor but the classifier is divided and sensitive to random seeds.

Hypothesis We present two possible explanations of why low- and mid-level layers are consistent but high-level does not: the insufficient data issue and the division of labor of layers. The first guess is that training data is insufficient to provide enough information for optimizing all layers since the deeper layer has a larger feature space and thus needs more data to converge to a unique point. The second guess is that there is a division of labor of layers in the network, and their different duties cause some of them to be invariant, but others are not. We can see the consistency split layers into two groups, the invariant layers (1-8th layers) and the variant layers (9-12th layers). A simple understanding is that the former is a feature extractor, and the latter is a classifier. In that case, we can explain the invariant layers as extracting features from input data since a generic feature space that can resolve various tasks is almost bijective to the input data, and the variant layers can be explained as classifying labels based on the extracted features. Since the classification process only concerns a subspace of the feature space related to labels, many different many-to-one mappings can be established, so the classification modules are variant.

Verification We verified the second guess by two approaches, probing the self-correlation of layers and estimating the complexity of feature spaces.

Although the visualization indicates the division of labor in a network, it does not suggest the labors are feature extraction and classification. To verify it, we estimate the complexity of feature spaces since a key difference between feature extraction and classification is that the latter feature space is much smaller than the former. This is due to the nature that a subset of features is often enough to determine the label. Therefore, we estimate the complexity of the two groups to see whether there is a significant complexity drop in the feature space. To measure the complexity of feature space, we sample a bunch of data and compute the nuclear norm of their features as an approximation of their complexities.

The low-level layers of large language models, such as BERT, include the sub-word embedding, which is a fundamental representation of the words in the vocabulary. This representation is learned during the pre-training stage and is designed to be consistent across different tasks and fine-tuning processes. This consistency is achieved through the use of shared weights, which ensures that the sub-word embedding is the same across different tasks and scenarios.

However, as we move to higher-level layers, the representations learned by the model become more task-specific and fine-tuning dependent. These higher-level layers are designed to capture more complex and nuanced relationships between the words and their context, and as a result, they become adapted to the specific task at hand. This adaptation leads to a divergence in the representations learned by these higher-level layers, even when fine-tuning on similar tasks. This divergence is a natural result of the fine-tuning process, and it is necessary for the model to be able to perform well on the specific task.

B.3 What Happens after Finetuning?

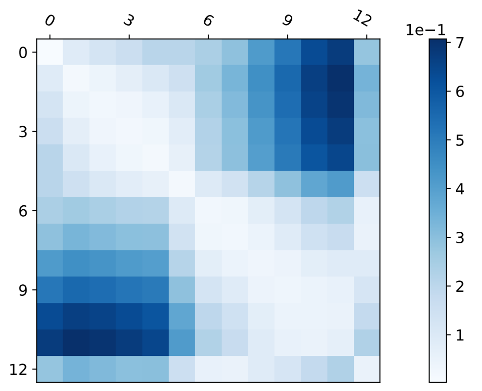
Higher Variance in Embeddings Previous works show that while fine-tuning significantly improves the performance on downstream tasks, the changes often arise in the top layers and vary a lot across different tasks (Merchant et al., 2020). Figure shows the mutual-correlation cross random seeds after fine-tuning, and we observe that the consistency are significantly weaker than pre-training, even low-level layers diverge a lot.

The representation for text classification contains task-specific and task-agnostic information. Task-agnostic information is irrelevant to downstream tasks. If a certain textual pattern occur coupled with labels, just task-agnostic information detecting the correlative patterns would reach the right prediction. For example, figure shows two examples where relying on task-agnostic shortcut makes appropriate prediction.

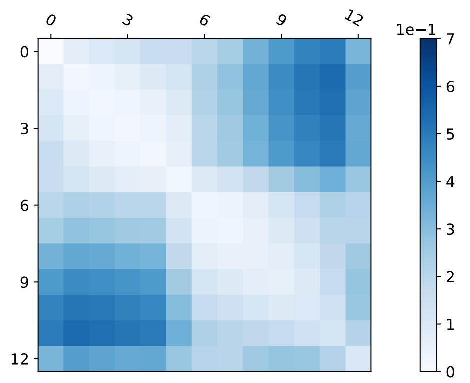
If the training set contains multiple shortcuts, model trained on different random seeds may detect different correlative pattern. For pre-training, the corpora come from various domains, and the ground truth are one-hot labels over the whole sub-word vocabulary while fine-tuning are often restricted to a specific domain or scenario. The short-cuts correlated with sub-word are relatively diverse, so the proportion of a certain short-cut to the training samples for pre-training are much more lower than fine-tuning. Then, the models for fine-tuning may wandering for distinguishing spurious artifacts from the task-specific inductive bias, then the consistency acquired in pre-training is broken.

Despite the fine-tuning process harming the consistency between models, which indicates it loses generalizable knowledge and tends to overfit the fine-tuned data, there should be stationary and effective patterns remaining in the network since the pre-train plus fine-tune formula achieve significant improvements against the models trained from scratch. By analyzing extensive experiment results, we find that how tokens interact with each other via self-attention modules holds after the fine-tuning.

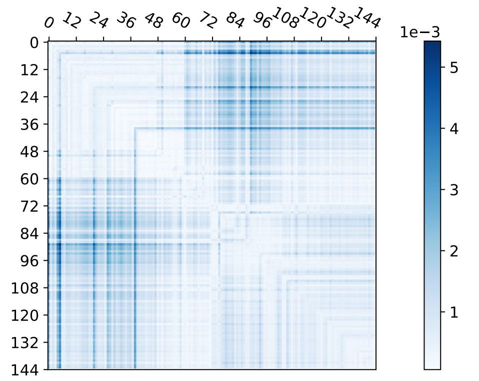
In addition, fine-tuning has little influence on attention maps, showing that the knowledge preserved in the self-attention module could be largely transferred to downstream tasks. Thus, we believe the interactions among tokens that are controlled by self-attention modules are stationary across the fine-tuning and optimization processes. The attention interaction patterns are invariant to both random seeds and fine-tuning processes, which shows it is a core primitive in terms of generalization abilities.



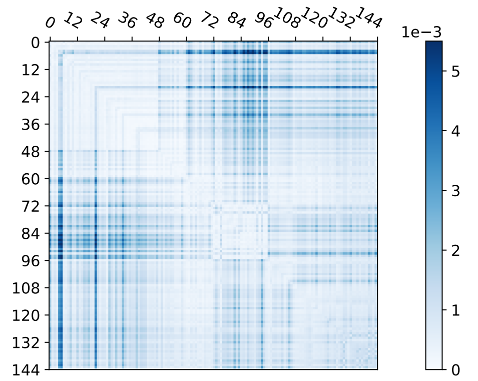
(a) Feature similarity (after pretraining)



(b) Hidden states similarity (after finetuning)



(c) Attention weight similarity (after pretraining)



(d) Attention weight similarity (after finetuning)

Figure 8: Hidden states similarity on MNLI. *what story will we have for this figure? I sort of understand what (a) and (b) mean. What message is (c) and (d) bringing to us?*– Zhijing

As shown in Figure 8, the consistency of attention maps increased gradually according to the depth of layers.