A Unified View of BERT Representations in Fine-Tuning, Transfer, and Forgetting

Joel Ye
Computer Science
Georgia Institute of Technology
joel.ye@gatech.edu

Ayush Shrivastava
Computer Science
Georgia Institute of Technology
ayshrv@gatech.edu

Sri Vivek Vanga
Computer Science
Georgia Institute of Technology
svanga3@gatech.edu

Abstract

BERT's ability to transfer to many downstream tasks has prompted many studies into how the model's pre-training leads to its fine-tuning successes. Among these works, a number emphasize BERT's fine-tuning process. We continue a line of work that uses representational analysis, specifically using Centered Kernel Alignment (CKA) to address knowledge gaps on fine-tuning in diverse settings, e.g. unstable fine-tuning and catastrophic forgetting. We show that CKA reliably predicts model behavior, relationships between tasks and explain aspects of forgetting. We also introduce two new metrics: Mean-CKA and Area under Freezing Curve for comparing models trained in different settings, demonstrating their insights for task similarity. Code: github.com/joel99/bert-representations.

1 Introduction

BERT (Devlin et al. [2018]) models are now widely used for a variety of downstream NLP tasks. BERT's language modeling pre-training is understood to provide a kind of "universal language representation" that is easily fine-tuned for specific tasks. However, the mechanisms of the fine-tuning process is not well understood. For example, what we know about how forgetting occurs is generally intuitive – large weight updates "overwrite" prior knowledge. While BERT fine-tunes to one task easily, forgetting is still a problem in continual learning settings (Yogatama et al. [2019]): when BERT fine-tuned on task A, and then fine-tuned on task B, will lose performance on task A. To investigate this problem, we propose to use representation similarity analysis (RSA).

Representational analysis is a common tool in interpretability works. Merchant et al. [2020] used RSA to show fine-tuning BERT changes higher layer (i.e. closer to the output) representations much more than those of lower layers. Separately, Ramasesh et al. [2020] has used representational analysis to show that higher layers of large vision models are the source of forgetting in a split-CIFAR task.

These works answer preliminary questions, but there are many more phenomena we encounter when fine-tuning BERT. What makes task transfer successful? How does forgetting happen? Existing analyses emphasize the mechanics of successful transfer to a single downstream task. We propose that representational analysis can be used to fill knowledge gaps when studying diverse fine-tuning settings e.g. failed tuning, multi-task models. Studied across settings, representations will provide a more complete picture of BERT's fine-tuning mechanics.

Formally, we consider the "representations" for a given task to be the activations of a task after fine-tuning from a BERT base model. We measure task A and task B's similarity by the performance of a combination of task A's representations on task B. Finally, we measure forgetting by the performance decrease in task A when re-tuning a model tuned on task A to task B. Broadly, we propose to use representational analysis to explore a number of related questions on fine-tuning and forgetting. Specifically, we aim to use RSA to explore:

1. How does RSA similarity correlate with model behavior similarity?
2. How does representational alignment correlate with task similarity? How does representation alignment correlate with forgetting?
    - How closely do these conclusions align with those for the vision domain in Ramasesh et al. [2020]?

- Intuitively, representation alignment corresponds to high feature transfer. We seek to confirm it quantitatively for semantically diverse tasks (whereas Ramasesh et al. [2020] only studies 2 qualitatively pairs of tasks).

3. How does forgetting correlate with representation shift in forgotten domains? Why does Merchant et al. [2020] not see representation shifts in unseen domains?
4. How "general" is BERT's general knowledge, and where is it? Ramasesh et al. [2020] describes a downstream task performance dropoff when progressively freezing more layers. How do the different task dropoffs compare?
5. Prior work suggests changes in higher representations for each fine-tuned task cause forgetting. How does shuffled curriculum learning, Yogatama et al. [2019] designed to counteract catastrophic forgetting, find a new general representation over time? Is there reduced variance in representations learned?

We will study the fine-tuning and forgetting of BERT on a number of tasks from the GLUE Benchmark (Wang et al. [2018]): Natural language inference (MNLI), Semantic Textual Similarity (STS), Sentiment Analysis (SST), Part-of-Speech Tagging(POS). Note the related effects of shifting task domains (i.e. BioBERT, Lee et al. [2020]) will be out of scope for this work. For example, Merchant et al. [2020] notes a difference in the effects of fine-tuning on in-domain and out-domain representations. We emphasize the shifting usage of linguistic representations rather than shifting vocabulary, and consequently we will primarily analyze representation changes in the Wikipedia corpus used for pretraining.

The questions tie together prior works on BERT's fine tuning mechanics and resolve several ambiguities between them. This work thus provides explanatory power for BERT's fine-tuning and additional insight into how to successfully train BERT in multi-task settings. This explanatory power is naturally useful for creating more general NLP models, but can also tie with e.g. large vision models, providing general principles for multi-task training outside of the language domain. We broadly hypothesize a strong correlation between task transfer and representation alignment, and believe BERT's forgetting is due to misalignment in top-layer representations. We expect shuffled curriculum learning acts to find a representation closely aligned with all downstream tasks.

2 Related work

2.1 Representational Similarity Analysis

Representational similarity analysis (RSA) Kornblith et al. [2019] techniques take in a set of examples and a pair of neural networks activations (e.g. from two layers, or two models). The activations for each example are compared to output a similarity score between 0 and 1 corresponding to how similar the sets of representations are.

We plan to use Centered Kernel Alignment (CKA) introduced in Kornblith et al. [2019]. Kornblith et al. [2019] shows that CKA outperforms other RSA metrics such as PWCCA Morcos et al. [2018] in consistently identifying correspondences between representations in networks trained from different initializations.

2.2 RSA-based model behavior analysis

RSA is widely used to understand how fine-tuning affects model representations and thereby expose opportunities for better generalization and transferability in deep learning.

Ramasesh et al. [2020] use RSA on deep learning models in computer vision to show that top layers representations change much more than shallow layers during catastrophic forgetting. They also analyse catastrophic forgetting mitigation techniques and argue they primarily work by stabilizing the top layer representations.

Merchant et al. [2020] use RSA to analyse fine-tuning in BERT. They show that fine tuning primarily affects the top layers of BERT, with noteworthy variation across tasks. Interestingly, they also show that fine tuning has a weaker effect on representations of out-of-domain sentences compared to representations of in-domain sentences.

2.3 Forgetting in BERT

Yogatama et al. [2019] show the effect of training curricula on how models forget previously acquired linguistic knowledge. They perform two experiments which result in different model behavior. The first experiment is a standard continual learning setup. They start with a pretrained BERT model and finetune it on SQuAD. Then they take the SQuAD-tuned BERT and finetune it on another task – TriviaQA. When fine tuning on TriviaQA, the authors notice that

the model's performance on SQuAD drops. This experiment demonstrates that the continual learning setup results in catastrophic forgetting of SQuAD knowledge.

In the second experiment, they take a pretrained model and train it with a random curriculum. At each step, sample a batch of examples from a randomly chosen task. This results in a model which is able to perform reasonably well on many tasks. The drawback of such a curriculum is that it either requires all tasks to be presented at the beginning or retrains on all tasks after seeing a new task.

The authors mention that avoiding catastrophic forgetting in the first setting is an open question. We will analyze model behaviors in these different setups to gain insight into how we might achieve such a continual learning setup.

2.4 Fine Tuning Mechanics

In the original BERT paper, Devlin et al. [2018] mention briefly that fine-tuning on small downstream datasets can have unstable performance. Phang et al. [2018] refines this observation on several large language models and finds this instability tends to be bimodal – a model either has high performance or low performance on the downstream task, i.e. it is "successful" or "unsuccessful" at fine-tuning. Mosbach et al. [2020], Zhang et al. [2020] further investigate the mechanics of this instability and come up with simple recommendations for avoiding fine-tuning instability. Both parties conclude that smaller learning rates, increased iterations, and enabling Adam optimizer bias correction greatly increases fine-tuning stability. Unless otherwise specified, we will use Mosbach et al. [2020]'s recommended training methodology for fine-tuning.

3 Problem Definition

In this section, we define Transformer (Vaswani et al. [2017]) and BERT (Devlin et al. [2018]) models and show how we will use Representation Similarity Analysis for comparing layer representations across different models.

3.1 Transformer & BERT

Transformer is the composition of L transformer blocks, each with its own parameters: $f_{\theta_L} \circ \cdots \circ f_{\theta_1}(\mathbf{x}) \in \mathbb{R}^{n \times d}$. Output from each layer can be denoted by $\mathbf{z}_l$ where $l \in \{1, ..., L\}$.

BERT's model architecture is a multi-layer bidirectional Transformer encoder followed by a linear layer for solving downstream tasks. For our experiments, we will use $\mathbf{BERT_{BASE}}$ which consists of 12 transformer layers i.e. $L = 12$.

4 Methods

4.1 Tasks used

For our analysis, we perform experiments in different settings on 4 frequently used tasks.

1. **MNLI:** Multi-Genre Natural Language Inference (Williams et al. [2017]) is a dataset designed for evaluating machine learning models for sentence understanding. Its task is given 2 statements, predict the relation between them using classification on different genres. The dataset consists of 443k instances and the labels are from 10 distinct genres of written and spoken English. We report accuracy on the matched (same distribution) validation set.

2. **STS-B:** Semantic Textual Similarity Benchmark (Cer et al. [2017]) measures the meaning similarity of sentences. The task involves producing real-valued similarity scores for sentence pairs , ranging from 0 for no meaning overlap to 5 for meaning equivalence. We report Pearson's correlation between our predicted and human annotated similarity scores.

3. **SST-2:** Stanford Sentiment Treebank (Socher et al. [2013]) is a sentiment classification task consisting of roughly 200k where half of them are of positive sentiment and the other half of negative. We report validation accuracy.

4. **POS:** Part-of-Speech Tagging. We use the CoNLL-2003 dataset (Sang and De Meulder [2003]) for POS-tagging or Named Entity Recognition. We use the POS labels only, and report validation accuracy.

To ensure the diversity among tasks, note that the first 3 tasks that we have considered are Sequence-level Classification tasks and the last task is a Token-level Classification task.

4.2 Representation Similarity Analysis and Centered Kernel Alignment

We use Centered Kernel Alignment (CKA) (Kornblith et al. [2019]) for our analysis on representation similarity. A representation is nothing but a 2D matrix $R^{n \times p}$. This matrix contains activations of p neurons in the layer we are analyzing for n examples from the dataset.

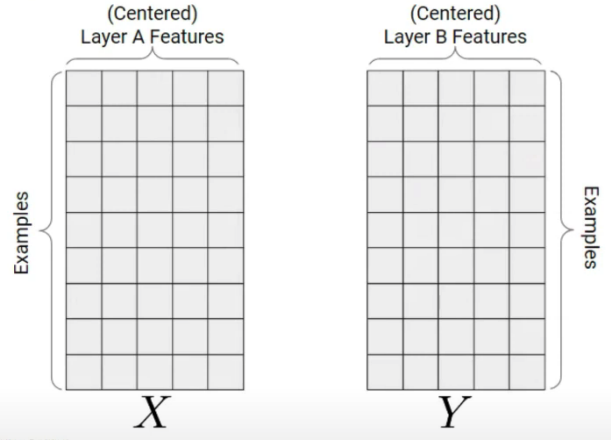


Figure 1: Comparison between 2 representations using the CKA metric

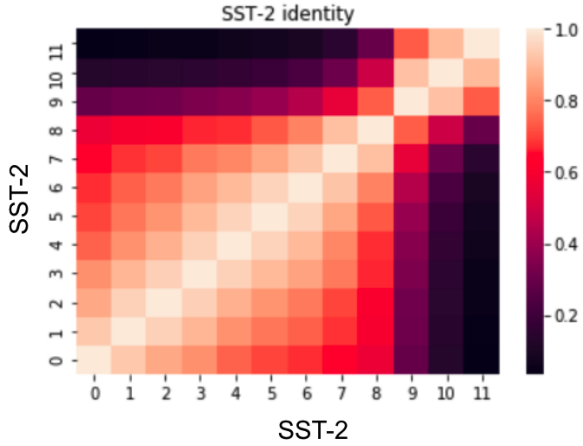


Figure 2: Visualization of CKA matrix for SST-2 task

Let $X \in \mathbb{R}^{n \times p_1}$ and $Y \in \mathbb{R}^{n \times p_2}$ be two representations from two layers in same/different models and on the same n examples. We assume that these matrices have been preprocessed to center the columns. Then, CKA $= ||Y^T X||_F^2 / (||X^T X||_F ||Y^T Y||_F)$. CKA outputs a similarity score between 0 and 1 corresponding to how similar the sets of representations from the two layers are.

The above described methodology gives us a way of comparing representational similarity between two layers. We taken two different variants of $\textbf{BERT}_{\textbf{BASE}}$ and compute the scalar CKA score for each layer pair between the two variants. As there are 12 layers in $\textbf{BERT}_{\textbf{BASE}}$, this will gives us a 12x12 matrix which we call "CKA Profile". This CKA profile gives a wholistic view of how the two variants of $\textbf{BERT}_{\textbf{BASE}}$ differ across all layer pairs. Please note that we take 5000 most frequently occurring tokens from the task dataset to generate CKA profile.

As scalar CKA score lies between 0 and 1, the 12x12 matrix has entries ranging from 0 to 1. This is visualised in a heatmap. The X axis corresponds to layers from 1 to 12 for one variant of $\textbf{BERT}_{\textbf{BASE}}$ and the Y axis corresponds to layers from 1 to 12 for the other variant of $\textbf{BERT}_{\textbf{BASE}}$ model.

4.3 New Metrics for measuring Task Transfer

Understanding how performance varies when we finetune models trained on task A to solve task B helps us perform principled finetuning and provide insights on how to train models in multi-task setting. Can we create metrics that quantity the task transfer performance using a scalar? Specifically, with task space $\mathcal{T}$, we want to find $\mathcal{M} : \mathcal{T} \longmapsto [0, 1]$. To answer this question, we introduce two metrics:

1. **Mean-CKA:** Mean-CKA summarizes the layer by layer matrix produced by CKA to evaluate overall model similarities. We compute it by first calculating the similarity between a model tuned on task A and a model tuned on task B, for tokens drawn from dataset B, and subtracting out the expected self-similarity between two random seeds on task B. The subtraction accounts for the intuition that we would not expect task A representations to align with task B if task B does not align with itself. Taking the mean of resulting matrix (and adding 1) gives us a scalar in $[0, 1]$.

2. **Area under Freezing Curve (AuFC):** For vision tasks (Zamir et al. [2018]), task transfer is typically measured by the performance on task B of a shallow network fed representations from task A. In NLP, however, fine-tuning is the dominant form of model adaptation. However, when we naively finetune models already trained on

task A for the second task B, models will simply forget task A, as shown by the low variance in performances in Figure 4. Note that to make performance interpretable across tasks, we report a performance normalized by the upper bound given by tuning from $\textbf{BERT}_{\textbf{BASE}}$ to task B. We see that all tasks can be further finetuned for all possible combinations to achieve their optimal performance, i.e., the prior task A is forgotten. Consequently, we develop a more nuanced metric by instead measuring how task B performance degrades as we freeze the model's layers. If task A features are perfectly aligned with task B, then no layer representations need to change for high performance on task B. More typically, task B performance degrades as shown in Figure 5. We can convert this performance curve into a metric by measuring the area under the curve (AuC). An AuC of 1.0 represents perfect representation alignment, whereas a value of 0.0 implies that task B cannot use any task A representations.

Figure 3: Mean-CKA computed for task transfer from MNLI to POS.

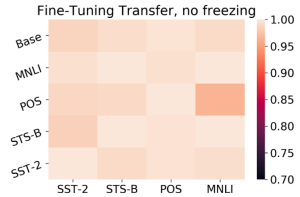


Figure 4: Normalised task performance for all combinations of finetuning tasks

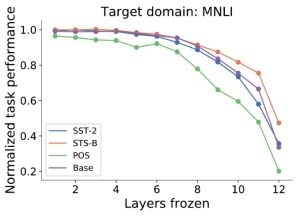


Figure 5: Area under Frozen Curve for target task MNLI

5 Analysis and Results

We have built our codebase on top of HuggingFace transformer and dataset repositories (Wolf et al. [2019]), which provides the infrastructure to setup a pretrained model. On top of this, we developed finetuning scripts to finetune BERT models on different sets of tasks in either single-task or multi-task settings. In our experiments, we first show that CKA metric is a reliable metric for visualizing representations and making inferences from their visualizations. Then we show results on the newly introduced metrics, Mean-CKA and AuFC. Then we perform analysis on catastrophic forgetting for BERT.

5.1 CKA profile for different settings

Before using CKA to make claims about the manner in which representations in BERT change, we first wanted to check if it is a reliable tool in measuring behavioural similarity. Please note that we are considering similarity in accuracies as behavioral similarity. So we performed three tests with the CKA profile for a model trained on task A.

1. **CKA profile for identity models:** We took a model trained on task A and computed the CKA matrix for representations of that model with itself. In Figure 6 (top row), we show that this matrix visualized for all tasks. We see that the diagonal elements for all 4 tasks are 1.0 which makes sense because we are comparing the representations of a model with itself. Figure 6 (top row, first column) shows the model finetuned for SST-2 which got 91.8% acuracy on the task metric.

2. **CKA profile for models with different random seeds:** Then, we compared the representations of a model trained with 2 different random seeds. Figure 6 (bottom row) show the visualizations for these models. We note that the maximum values of the CKA-matrix still appear at the diagonal but are slightly lesser bright (slightly less than 1.0) which makes sense as these representation should be very similar as there are extracted from same model with only difference in their random seeds. Figure 6 (bottom row, first column) shows the 2 models finetuned for SST-2 which got 91.8% and 92.7% acuracy on the task metric.

3. **CKA profile for successful vs failed finetuning:** Then, we wanted to compare representations for dissimilar models trained on the same task. So, we took a model which trained for the SST-2 task which chanegd its learning rate such that it resulted in failed training. The successfull model got 92.7% and the failed model got 49% accuracy on the task metric. We visualize their CKA-matrix in Figure 7. From the visualization, it is very clear that the representations for these models are very different and their representations are only similar at the very first transformer layer.

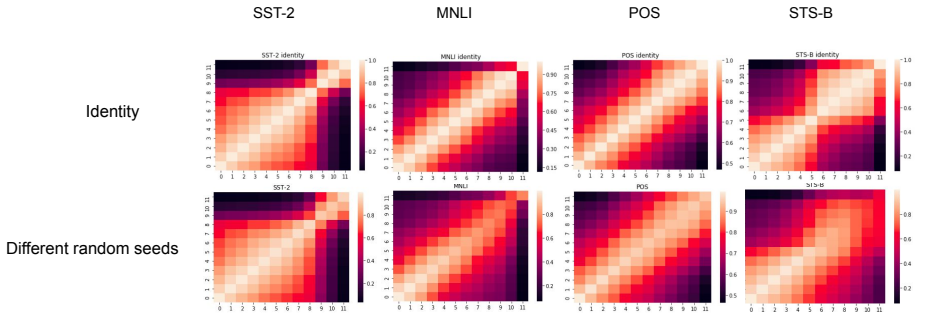


Figure 6: (Top row): CKA-matrix for identity representations for all tasks. (Bottom row): CKA-matrix for model representations trained with different random seeds for all tasks.

Figure 7: CKA-matrix for successfull vs failed training for SST-2 task

5.2 Task Transfer performance

To measure if the models trained on task A can be transferred to task B, we compute a matrix of task transfer performance using all combinations of tasks. There are 2 metrics that we can use to perform this analysis: Mean-CKA and AuFC.

5.2.1 Using Mean-CKA

In Figure 8, we compute the task transfer matrix using Mean-CKA. Using this, we are able to visualize which task can be easily transferred to another task. We see that STS-B transfers well to MNLI and vice-versa, This has been highlighted using green rectangles. Figure 9 shows the same plot, and demonstrates how the cells that are averaged to produce the statistic we use to measure the general transferability of a task. This statistic suggests SST-2 and POS do not transfer well.

5.2.2 Using AuFC

Now, we want to see whether a representation-based metric like Mean-CKA is consistent with a performance-based metric like AuFC. Thus, we repeat the experiment from Sec 5.2.1, this time using AuFC as the task transfer metric. Figure 10 shows the results, and indeed produces similar findings. STS-B and MNLI are similar and transfer well among each other, and transfer to other tasks. Again, SST-2 and POS tasks do not transfer well among each other.

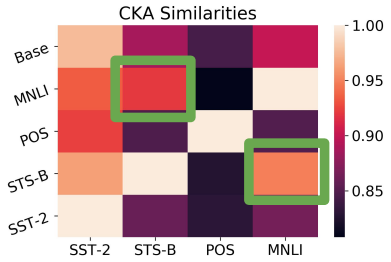
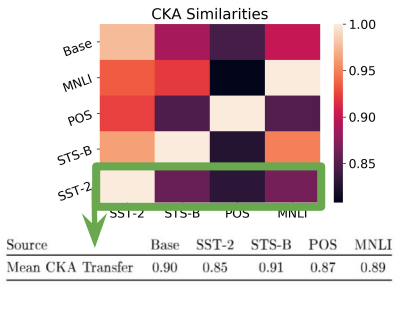
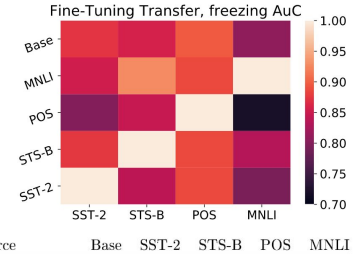


Figure 8: Task Transfer performance for all tasks using mean-CKA higlighting similarity between MNLI and STS-B in green



Source	Base	SST-2	STS-B	POS	MNLI
Mean CKA Transfer	0.90	0.85	0.91	0.87	0.89

Figure 9: Task Transfer performance for all tasks using mean-CKA higlighting the fact that POS task does not transfer well.



Source	Base	SST-2	STS-B	POS	MNLI
AuFC Transfer	0.86	0.84	0.87	0.79	0.89

Figure 10: Task Transfer performance for all tasks using AuFC

5.3 Analysing Catastrophic Forgetting

In prior sections we have built out a connection between representations and behavior. We now use model representations to understand catastrophic forgetting, a particularly undesirable model behavior. Specifically, we investigate specific facets of forgetting again using the framework of first tuning on task A and then tuning on task B.

1. **Why is forgetting asymmetric?** Specifically, as shown in the performance plots of Figure 11, why does POS not get forgotten when the model is tuned on SST-2, and not vice versa? The representational view clearly shows that the forgetting going from SST-2 to POS is due to a large shift in upper layers. Once tuned on POS, the model's upper layers resemble intermediate SST-2 layers (as highlighted by the rectangle). Conversely, the upper layers do not shift when learning SST-2. Interestingly, this means that POS representations are sufficient to perform well on SST-2, but that is not what would be found by only tuning on SST-2.

2. **How does shuffling mitigate forgetting?** Instead of first learning task A, and then learning task B, we can provide the model shuffled batches from either task A or task B. We then have a multi-task learning setup, so neither task will be forgotten. The representation view shown in Figure 12 show this model immediately resembles fully tuned POS representations. As SST-2 is learned, its necessary representations do emerge in the upper layers. This presents a transitivity puzzle – CKA is only invariant to isotropic scaling and orthogonal transforms, both of which are transitive. Then, how are upper layer representations similar to both SST-2 and POS single task representations, which are themselves dissimilar? Since the CKA profiles do not reflect perfect alignment, we speculate that shuffling is simply finding an averaged representation somewhat close to both single-task representations.

3. **Is forgetting an overfitting phenomena?** Merchant et al. [2020] shows that forgetting occurs more in the new domain (task B) than old domain (task A), as measured by CKA changes in each layer. We wonder whether this "overfitting" of representational changes ask whether this changes as we continue training, or whether forgetting will eventually change out of domain representations as well. Figure 13 shows a model forgetting SST-2 representations while learning MNLI. Representations from both SST-2 and MNLI datasets are compared using weights from the start of tuning and from two intermediate points in training (0.3 and 0.7). At 0.3 progress through the trial, new-domain MNLI representations have always changed drastically, and they stay that way as forgetting continues. Old-domain SST-2 representations, however, are initially less changed, as expected. By time 0.7, upper layers have also largely forgotten SST-2 representations, but intermediate layers are still quite stable. These results suggest forgetting is mostly due to upper layer changes in new-domain representations.

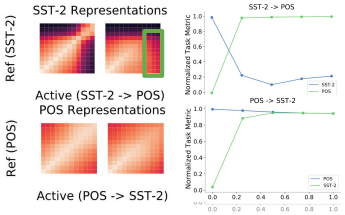


Figure 11: SST-2 and POS forgetting is asymmetric since POS is a subtask of SST-2.

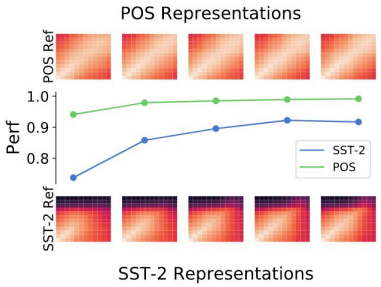


Figure 12: Shuffled training finds an average representation similar to both POS and SST-2 single task representations.

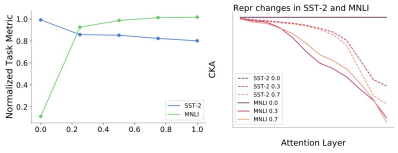


Figure 13: Even as forgetting continues, early layer representations of tokens in the old domain will change minimally.

6 Conclusion

In this project, we have explored different model adaptation strategies for the BERT model using 4 NLP tasks. We have shown that CKA is correlated to behavior similarity and is a useful tool for predicting task similarity consistently. We introduced Mean-CKA and AuFC which are used to quantitatively measure the task transfer performance. we have also analysed different factes of catastrophic forgetting in BERT model for NLP tasks.

7 Contributions

The members of this team contributed equally in both the implementation and presentation of this project.

References

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*, 2017.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. *arXiv preprint arXiv:1905.00414*, 2019.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.

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

Ari Morcos, Maithra Raghu, and Samy Bengio. Insights on representational similarity in neural networks with canonical correlation. In *Advances in Neural Information Processing Systems*, pages 5727–5736, 2018.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. *arXiv preprint arXiv:2006.04884*, 2020.

Jason Phang, Thibault Févry, and Samuel R. Bowman. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks, 2018.

Vinay V Ramasesh, Ethan Dyer, and Maithra Raghu. Anatomy of catastrophic forgetting: Hidden representations and task semantics. *arXiv preprint arXiv:2007.07400*, 2020.

Erik F Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*, 2003.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

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

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.

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. Huggingface's transformers: State-of-the-art natural language processing, 2019.

Dani Yogatama, Cyprien de Masson d'Autume, Jerome Connor, Tomas Kocisky, Mike Chrzanowski, Lingpeng Kong, Angeliki Lazaridou, Wang Ling, Lei Yu, Chris Dyer, et al. Learning and evaluating general linguistic intelligence. *arXiv preprint arXiv:1901.11373*, 2019.

Amir R. Zamir, Alexander Sax, William B. Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. Revisiting few-sample bert fine-tuning. *arXiv preprint arXiv:2006.05987*, 2020.