Representation Fine-Tuning on Vision Tasks

Zheng Wang
Stanford University
peterwz@stanford.edu

Abstract

Parameter-efficient fine-tuning (PEFT) methods, such as Low-rank Adaptation (LoRA), have significantly reduced the number of parameters required to fine-tune large language models (LLMs). Recently, new PEFT methods like Representation Fine-tuning (ReFT) have pushed this efficiency even further, reducing the fine-tuning parameter count to less than 1% of the original model while also enhancing the interpretability of the fine-tuned models. In this Stanford CS231n project, we explored the application of ReFT in fine-tuning models for vision-related tasks, including image-text understanding and visual instruction tuning. ReFT employs fewer parameters than other PEFT methods yet achieves comparable performance, particularly in image captioning. This motivates us to further investigate the application of ReFT in other domains.

1. Introduction

Pretrained large language models (LLMs) are frequently fine-tuned to adapt them to new domains or tasks (4). Through fine-tuning, a single base model can be adapted to a variety of tasks with only small amounts of domain-specific data. However, fine-tuning LLMs is expensive. Parameter-efficient fine-tuning (PEFT) methods address the high costs of full fine-tuning by updating only a small fraction of the weights. This reduces memory usage and training time, while achieving performance similar to full fine-tuning in many settings (12).

Current state-of-the-art PEFT methods, such as Low-Rank Adaptation (LoRA, (11)), modify *weights* rather than *representations*. However, much prior interpretability work in natural language processing has shown that representations encode rich semantic information, suggesting that editing representations might be a more powerful alternative to weight updates. Historical research on language representations has provided increasing evidence that human-interpretable concepts can be encoded linearly (25). It is thus possible to use linear transformations to edit language models' representations, treating that as a new PEFT method.

The recently proposed Representation Fine-tuning (ReFT (30)) method gained insight from this reasoning. ReFT trains linear low-rank interventions that manipulate a small fraction of the language model's representations to steer model behaviors for downstream tasks at inference time. By editing representations rather than weights, ReFT tunes fewer parameters than other PEFT methods like LoRA, while achieving similar fine-tuning performance. For tasks such as commonsense reasoning and instruction tuning, ReFT has been shown to use less than 1% of the model's original parameters, serving as a drop-in replacement for weight-based PEFTs like LoRA. In addition to being 15x-65x more parameter-efficient, ReFT is also more flexible and interpretable.

PEFT methods like LoRA also have broad applications in computer vision, including fine-tuning vision-language models such as VL-Bart (15), LLaVA (17), and diffusion models (23). However, vision feature representations lie in different subspaces compared to token representations in the embedding space. Therefore, a versatile PEFT method that works across both vision and language domains would have broad applications.

In this CS231n project, we applied ReFT to computer vision tasks. Due to the limited time of the course project, we focused on vision-language tasks, particularly image-text understanding. These tasks require the model to integrate information from both input texts and images, completing tasks such as (1) answering logical questions about the given image, (2) reasoning about multiple input images, and (3) adding captions to input images. Successfully completing these challenging image-text understanding tasks would demonstrate ReFT's adaptability to image features and pave the way for broader vision applications such as vision instruction-tuning and image generation.

Compared to LoRA-like methods, which use low-rank matrices to approximate additive weight updates during training, ReFT explicitly edits model activations (in our case, adding interventions to the residual stream) to steer the representation in a particular direction. In addition to improving the interpretability of fine-tuning, ReFT also fa-

cilitates easier composition of different adaptations (interventions) in the representation space. This feature is useful in the language domain and even more applicable in the vision domain. For instance, we could train one ReFT to steer an image towards a Pikachu and another ReFT to steer the image towards an Apple Vision Pro. Composing these two ReFTs in the representation space could present us with a Pikachu wearing an Apple Vision Pro. Compared to various attempts to compose multiple LoRAs together (33), composing ReFTs is a simple and interpretable vector space operation in the representation space, providing a more sharable and composable framework for image editing.

2. Related Work

Parameter-efficient fine-tuning methods (PEFTs). PEFTs train a fraction of the model's parameters to adapt it to downstream tasks. We classify PEFTs into three categories:

1. **Adapter-based methods** train additional modules (e.g. fully-connected layers) on top of the frozen pretrained model. *Series adapters* insert components between LM attention or MLP layers (10; 22), while *parallel adapters* add modules alongside existing components (9). Since adapters add new components that cannot be easily folded into existing model weights, they impose an additional burden at inference time.

2. **LoRA** (11) and DoRA (19) use low-rank matrices to approximate additive weight updates during training, and require no additional overhead during inference, as the weight updates can be merged into the model. These are the current state-of-the-art PEFT methods.

3. **Prompt-based methods** add randomly-initialised soft tokens to the input (usually as a prefix) and train their embeddings while keeping the LM weights frozen (16). These methods are often less optimal compared to other PEFTs and come with significant inference overhead. A variant of this method, where hidden-layer activations are also tuned, was introduced as a baseline in (11), showing better performance.

Representation editing. Recent work on *activation steering* and *representation engineering* demonstrates that adding fixed or task-specific steering vectors (34; 18) or applying concept erasure (2) to the residual stream can enable a degree of control over pre-trained LM generations without the need for resource-intensive finetuning.

The success of these methods confirms that representations induced by pre-trained LMs carry a rich semantic structure.

Interventional interpretability. Recent work has increasingly used interventions on model-internal states to test hypotheses about how LMs implement various behaviors. Specifically, interventions on linear subspaces of representations have provided growing evidence that human-interpretable concepts are encoded linearly (25; 24). This includes linguistic characteristics such as gender and number (14; 1), logical and mathematical reasoning, entity attributes, and a number of other domains (21; 8).

Vision-language fine-tuning. It has become common practice to fine-tune a pretrained model to perform multiple downstream vision-language tasks. (20) first fine-tuned a vision-language model (VLM) on multiple downstream tasks simultaneously. VL-Adapter (27) fine-tuned an adapter on an LM and achieved performance comparable to full fine-tuning on vision-language tasks. LLaVA (17) fine-tuned the LLaMA LLM on vision instruction-tuning tasks, unlocking vision capabilities in large language models. Our method follows this stream and investigates whether ReFT, as a new PEFT method, could achieve performance similar to adapters and LoRA on vision-language fine-tuning.

3. Methods

3.1. ReFT

In this section we briefly introduce the ReFT method. We refer users to the ReFT paper (30) for more details. To keep the presentation simple, we assume throughout that our target model is a Transformer-based (28) LM that produces contextualized representations of sequences of tokens. Given a sequence of n input tokens $\mathbf{x} = (x_1, \ldots, x_n)$, the model first embeds these into a list of representations $\mathbf{h}^{(0)} = (\mathbf{h}_1^{(0)}, \ldots, \mathbf{h}_n^{(0)})$. Then, m layers successively compute the j-th list of hidden representations $\mathbf{h}^{(j)}$ as a function of the previous list of hidden representations $\mathbf{h}^{(j-1)}$. Each hidden representation is a vector $\mathbf{h} \in \mathbb{R}^d$. The LM uses the final hidden representations $\mathbf{h}^{(m)}$ to produce its predictions. For vision experiments, we only consider autoregressive LMs, which predict $p(x_{n+1} \mid x_1, \ldots, x_n) = (\mathbf{W}\mathbf{h}_n^{(m)})$, where $\mathbf{W}$ is a learned matrix mapping from representations to logits over the vocabulary space.

3.1.1 Motivation

The **linear representation hypothesis** claims that concepts are encoded in linear subspaces of representations in neural networks. Early connectionist work on distributed neural representations first proposed this idea (25; 24), and recent empirical work has found evidence supporting this claim in neural models trained on natural language and other input distributions (21; 8).

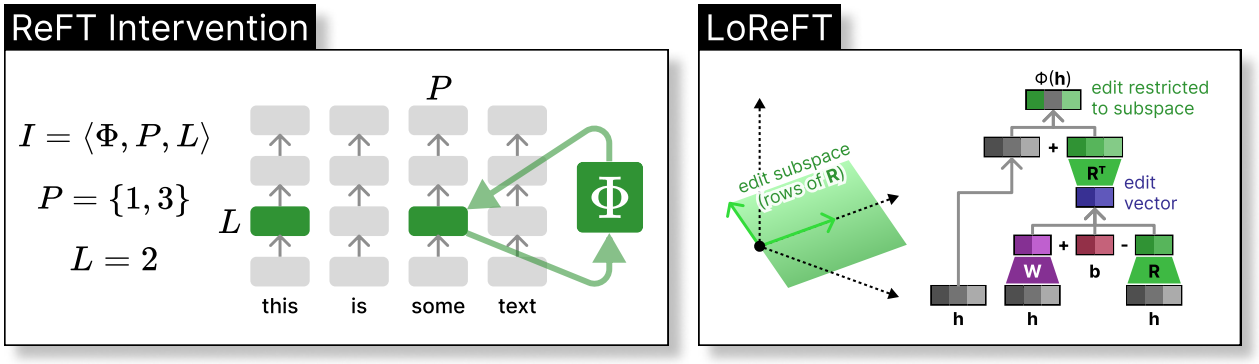

Figure 1. **Illustration of ReFT**. Image borrowed from the ReFT paper (30). **(1)** The left panel depicts an intervention I: the intervention function Φ is applied to hidden representations at positions P in layer l. **(2)** The right panel depicts the intervention function used in ReFT, which finds an edit vector that only modifies the representation in the linear subspace spanned by the rows of $\mathbf{R}$. Specifically, we show how a rank-2 ReFT operates on 3-dimensional hidden representations.

In interpretability research, the framework of causal abstraction (5) uses **interchange interventions** to causally establish the role of the components of the neural network in the implementation of particular behaviors. The logic of the interchange intervention is as follows: if one fixes a representation to what it would have been given a counterfactual input, and this intervention consistently affects model output in the way predicted by our claims about the component producing that representation, then that component plays a causal role in the behaviour being studied. Experiments investigating how such interventions affect model behavior form the evidence for claims about the causal role of a representation and the concept it encodes.

To test whether a concept is encoded in a linear subspace of a representation, as claimed by the linear representation hypothesis, one may use a **distributed interchange intervention** (DII) (6). Let $\mathbf{b}$ be the hidden representation created at row i and column k when our model processes input b, and let $\mathbf{s}$ be the corresponding representation when that same model processes input s. A distributed interchange intervention on $\mathbf{b}$ given a counterfactual source representation $\mathbf{s}$ is then defined as

$$\mathsf{DII}(\mathbf{b}, \mathbf{s}, \mathbf{R}) = \mathbf{b} + \mathbf{R}^\top(\mathbf{Rs} - \mathbf{Rb}) \tag{1}$$

where $\mathbf{R} \in \mathbb{R}^{r \times d}$ is a low-rank projection matrix with orthonormal rows, d is the representation dimensionality, and r is the dimensionality of the subspace we are intervening on. We learn the subspace $\mathbf{R}$ using distributed alignment search (DAS), which finds the subspace that maximises the probability of the expected counterfactual output after intervention (6). DAS is highly expressive, and can effectively localize concepts within model representations (32; 1). This suggests that subspace representation interventions could also be a powerful tool for model control.

LoReFT. The formulation of DII in eq. 1 immediately suggests a way to control model generations via interventions. The guiding intuition is that we can learn how to perform interventions that steer the model towards predicting our task labels. The resulting method, Representation Fine-tuning (ReFT), is defined by the following variant of 1:

$$\Phi(\mathbf{h}) = \mathbf{h} + \mathbf{R}^\top(\mathbf{Wh} + \mathbf{b} - \mathbf{Rh}) \tag{2}$$

This equation is identical to equation 1, except we use a *learned projected source* $\mathbf{Rs} = \mathbf{Wh} + \mathbf{b}$. ReFT thus edits the representation in the r-dimensional subspace spanned by the rows of $\mathbf{R}$ to take on the values obtained from our linear projection $\mathbf{Wh} + \mathbf{b}$. We depict this operation in Figure 1. The learned parameters are $\phi = \{\mathbf{R}, \mathbf{W}, \mathbf{b}\}$; the parameters of the LM are frozen. As with DII, $\mathbf{R} \in \mathbb{R}^{r \times d}$ is a low-rank matrix with orthonormal rows where d is the hidden-state dimensionality and $r \leq d$ is the rank of the subspace. We further define a linear projection $\mathbf{W} \in \mathbb{R}^{r \times d}$ and bias vector $\mathbf{b} \in \mathbb{R}^r$.

Relationship between ReFT and LoRA. ReFT does not limit the particular form of intervention Φ that could be applied to the representation $\mathbf{h}$. To analyze the similarities and differences between LoRA and ReFT, we can consider an ablated form of LoReFT, which removes the orthogonality constraint and the difference operation:

$$\Phi(\mathbf{h}) = \mathbf{h} + \mathbf{W}_2^\top(\mathbf{W}_1\mathbf{h} + \mathbf{b}) \tag{3}$$

Both $\mathbf{W}_1, \mathbf{W}_2 \in \mathbb{R}^{r \times d}$ are low-rank projection matrices. The above intervention 3 resembles LoRA and can be considered as applying the low-rank matrix transformation to the representation directly, instead of to the model weights. The ReFT paper (30) suggests that removing orthogonal constraints in Φ decreases performance only slightly.

In both 2 and 3, we apply the intervention to the representation of the model $\mathbf{h}$ at the transformer's residual stream, where it combines the output of the MLP layer and the Attention layer. The closest form of LoRA to ReFT would be to apply LoRA to the attention layer output projection matrix and the MLP projection matrix. Mechanically, ReFT uses a single low rank transformation 3 to encode multiple LoRA matrices, so it has less expressive power than LoRA. Also, ReFT only applies to specific tokens in the prompt, instead of LoRA, which applies to all the tokens in the prompt and during decoding. However, since ReFT is grounded in the linear representation hypothesis of language, it is a much more parameter-efficient way to unlock the task-specific knowledge and capabilities in the pre-trained model.

Training objective. In the vision-language experiments performed in this project, we only consider generation tasks using encoder-decoder LMs (such as Bart (15)). The pre-trained language model induces a distribution over token sequences $p(\cdot)$. We denote the model that results from the ReFT intervention Φ on $p(\cdot)$ as $p_\Phi(\cdot)$ with trainable parameters ϕ. To simplify notation, we refer to the hidden representations produced by the LM on input $\mathbf{x}$ as $\mathbf{h}(\mathbf{x})$, and those by the intervened LM as $\mathbf{h}_\Phi(\mathbf{x})$.

For vision-language generation tasks, our training objective is language modeling. In our case, the input sequence $\mathbf{x} = (x_1, \ldots, x_n)$ contains n tokens as the prompt, where the first few tokens $\mathbf{x_t} = (x_1, \ldots, x_p)$ are the embedded text tokens, and $\mathbf{x_v} = (x_{p+1}, \ldots, x_n)$ are the embedded image features. The embedded image features have been pre-processed by a pre-trained ResNet-101 network from the dataset's images. Each image token corresponds to a bounding box of image features. Image embeddings and text embeddings are directly concatenated:

$$\mathbf{x} = \textbf{Concat}(\text{prompt}, \mathbf{x_t}, \mathbf{x_v}) \tag{4}$$

The goal is to predict the output sequence $\mathbf{y} = (y_1, \ldots, y_m)$ with m tokens. We minimize the cross-entropy loss with teacher forcing over all output positions.

$$\min_{\phi} \left\{ -\sum_{i=1}^{m} \log p_\Phi \left(y_i \mid \mathbf{x}\mathbf{y}_{<i} \right) \right\} \tag{5}$$

4. Dataset

We use the following fine-tuning datasets during the training, validation, and testing of vision-language models:

- **VQA v2** (7) - Visual Question Ansering. This dataset contains various questions about input images, covering aspects like color, shape, texture, relationships between image elements, and semantic understanding. Figure 2 provides an example of the VQA dataset.

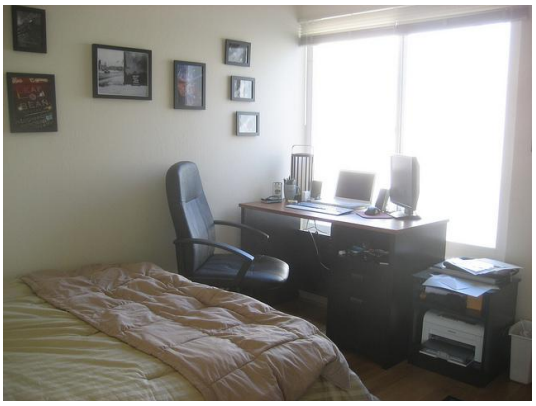

Figure 2. **Sample VQA Image**. VQA questions about this image include: (1) Is the bed white? (2) How many frames are on the wall? (3) What kind of room is this?

- **GQA** (13). An enhanced version of VQA, focusing more on visual reasoning and compositional question answering. Instead of simpler questions like "What color are the gym shoes?" in VQA, GQA includes more complex questions, such as "Is there any milk in the bowl to the left of the apple?"

- **NLVR v2** (26). This dataset involves questions comparing pairs of images. For instance, one might ask whether the left image contains twice as many dogs as the right image. These questions require a deeper understanding of the logical relationships between the pair of input images.

- **COCO Caption** (3). This dataset contains a large collection of images for which the model must generate captions. The generated captions are evaluated against a pool of human-generated captions using metrics such as BLEU, CIDEr, and ROUGE. Following the approach in the DoRA paper, we report the CIDEr (29) evaluation score for ReFT.

All these datasets use images collected for the COCO captioning task. The COCO images are preprocessed into image features using a ResNet-101 backbone. Our experiments utilize the preprocessed image features downloaded directly from the DoRA (19) project site.

5. Experiments

5.1. Experiment Setup

We adopted the codebase from VL-Adapter (27), DoRA (19), ReFT (30), and Pyvene (31) and integrated these libraries together to fine-tune the Facebook Bart-base (15) on the aforementioned datasets.

In our experiments, we apply ReFT exclusively to the language model encoder, editing the representation at the

Methods	% Params	VQA	GQA	NLVR	COCO-Caption	Avg
FT	100	66.9	56.7	73.7	112.0	77.3
LoRA	5.93	65.2	53.6	71.9	115.3	76.5
DoRA	5.96	65.8	54.7	73.1	115.9	77.4
ReFT-64 (Ours)	2.10	61.4	50.0	65.5	114.7	72.9

Table 1. **Image-Text Understanding results of ReFT**. These results are reported on the test sets of the relative datasets under the hyperparameter selected according to Table 2.

transformer block residual stream across all encoder layers. This approach may be less powerful than editing the decoder layers as well, as done in LoRA and DoRA. We chose to apply ReFT only to the encoder partly because ReFT paper (30) indicated that intervening on the prompt yielded the best performance, and the encoder representations correspond to the prompt.

ReFT edits the prefix and suffix of the input tokens. Since we concatenate text embeddings and visual embeddings as inputs in the vision-language experiments, we create LoReFT interventions that separately edit the prefix and suffix of both the text tokens and the image tokens. Image interventions are also fine-tuned differently with text interventions. For example, image interventions have a much higher rank than text interventions. We cap the text token length at 20 and image token length (number of feature boxes) at 36. These are kept the same as DoRA (19).

For the trainable modules, we follow the same setup as described in the DoRA paper (19). In addition to the trainable representation interventions, which include all LoReFT interventions such as the $\mathbf{R}$ matrix and the $\mathbf{Wh}+\mathbf{b}$ linear transformation, we also train the input visual embedding, the batch norm and layer norm statistics, and the model's biases.

We use the validation set of the VQA task at epoch 20 for hyperparameter selection, then apply these hyperparameters to multi-task training across VQA, GQA, NLVR, and COCO Caption. Details of the hyperparameter selection can be found in Table 2.

For VQAv2, we report the VQA Score, which is a weighted average indicating whether the model's predictions match the pool of human-provided answers. GQA and NLVR evaluations are similar to VQA. For COCO Captioning, we report the CIDEr-D score, which measures the Term Frequency Inverse Document Frequency (TF-IDF) weighting for each n-gram. The CIDEr-D metric best matches human evaluation on captioning compared to other metrics reported in the MSCOCO Captioning paper (3). For more details on the CIDEr metric, refer to (29). Number of beams are set to 1 for VQA, GQA, and NLVR, but set to 5 for COCO Captioning. We keep the generation temperature at 1 during both training and evaluation.

Fine-tuning on VQAv2 dataset alone takes about 20 hours on a single Nvidia A100 GPU with 40GB RAM. Fine-tuning on the multi-task dataset takes about 3.5 days with the same setup.

5.2. Results

5.2.1 Overall Results

Table 1 shows our preliminary quantitative results. In general, on vision language tasks, ReFT's performance still lags behind LoRA or DoRA's. However, for specific fine-tuning tasks such as COCO Captioning, ReFT achieved a higher CIDEr score than the full fine-tuning of VL-Bart. This indicates that in free-form generation tasks like COCO Captioning, ReFT can achieve relatively good performance. Additionally, ReFT models use fewer parameters than LoRA or DoRA. We will explain in the "Rank selection" section why the parameter count cannot be further reduced.

ReFT lags behind LoRA by about 4% on VQA, 3% on GQA, and 6% on NLVR. These tasks increasingly require less free-form generation and more logical reasoning. Figure 3 presents a failure case of ReFT where DoRA succeeds. In this example, ReFT can identify one of the traffic signs as a stop sign but fails to identify the meaning of the "yellow sign" and ignores the pedestrian crossing sign below the stop sign. However, ReFT may be good at recognizing detailed image features, as shown in Figure 4.

One hypothesis is that ReFT can unlock capabilities already present in the pretrained model, such as image captioning. However, ReFT lacks the capabilities for more fine-grained steering of the pretrained model towards complex reasoning. Figure 3, for example, requires the model to distinguish the three signs in the image by color. If the image embeddings are out of distribution for the original model, it may be difficult for ReFT to complete such tasks.

5.2.2 Rank selection

In Figure 5 we analyze the effect of ReFT rank on VQA's validation performance.

First, unlike models in the LoRA family, increasing the parameter count of ReFT (such as increasing the rank of ReFT's steering matrix) above 64 did not lead to a significant increase in performance. This suggests that a smaller number of parameters might lead to overfitting with ReFT.

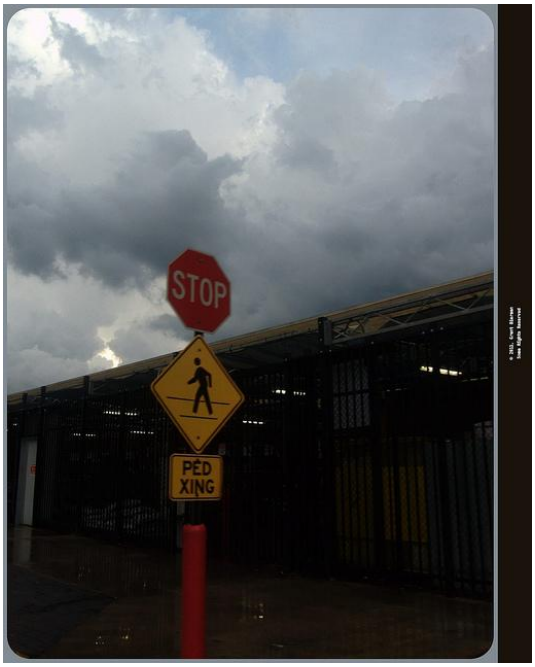

Figure 3. **VQA Sample where ReFT fails and DoRA succeeds**. The prompt is: "What does the yellow sign say?" ReFT responds with "stop". DoRA responds correctly with "pedestrian crossing".

Figure 4. **VQA Sample where ReFT succeeds and DoRA fails**. The prompt is: "Is the TV a tube or flat screen?" ReFT responds with "flat screen". DoRA responds incorrectly with "tube".

Increasing the number of ReFT fine-tuning parameters beyond a certain limit results in diminishing returns.

Second, for Figure 5, instead of an inverted-U shape observed in the text domain (which is also the behavior of DoRA), ReFT on image tokens shows a positive U-shape. This chart highlights the significant differences in the behavior of image tokens and text tokens under ReFT. It may be possible that image token embeddings are originally out of the distribution of the text model, so merely steering the representation with a low-rank matrix is insufficient to align the image tokens with the language model's representation distribution. However, a rank that is too large may lead to overfitting. [1] This partly explains why the model parameter savings of ReFT on image tasks are not as substantial as those on text tasks, as the optimal rank for ReFT on images is higher. Note that even when ReFT's rank is 1, the trainable parameters still count as 1.1% of the original model's parameters due to the need to tune the visual embeddings, bias terms, and the layer norm/batch norm statistics. These practices are kept the same as those in DoRA to ensure a fair comparison.

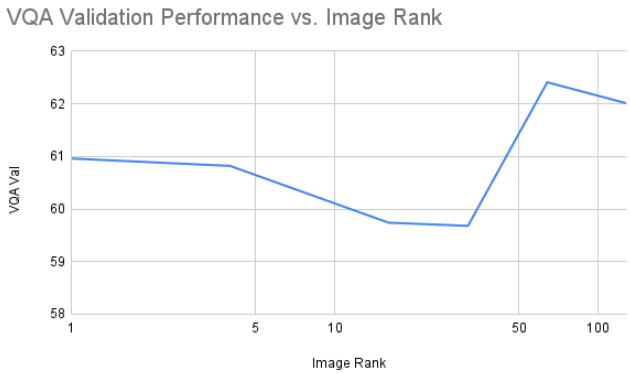

Figure 5. **VQA Validation Performance against Image Rank**. In this experiment, the text intervention's ranks are fixed at 4. Interventions are applied on the first 6 and last 6 text tokens, and the first 6 and last 6 image tokens.

6. Conclusion and Future Work

As a newly discovered PEFT method, Representation Fine-tuning (ReFT) showed promising results when fine-tuning large language models. In this project, we explored the potential for applying ReFT to vision-language tasks, specifically focusing on image-text understanding. We found that ReFT performs well on image tasks, including the COCO Captioning dataset, where it generates free-form responses. However, ReFT does not perform as well as LoRA on image tasks that require higher levels of reasoning about image elements. Additionally, ReFT requires higher ranks and thus more parameters for image tasks than for text tasks.

We believe that this project can serve as a starting point for further exploration of ReFT on other downstream tasks. For example, DoRA was applied only to the Query and Key matrices of the attention layer, so ReFT may be applicable to representations beyond on the residual streams only. ReFT could also be applied to the decoding layers of the encoder-decoder language models, which we did not have enough time to explore. Another potential approach could be to first use captioning to summarize the image into text, a task at which ReFT excels, and then concatenate the prompt with the image's summary. This might lead to better VQA

[1] We did not apply dropout during ReFT training because, although it reduces overfitting, it significantly slowed down optimization.

performance for ReFT compared to learning directly from the image features.

7. Contributions and Acknowledgement

This research is an extension of the ReFT (30) project. In addition, this research adopted the codebase of VL-Adapter (27), DoRA (19), and Pyvene (31). This project is also part of the Stanford NLP Research Lab's work, which integrated with Stanford NLP Lab's computing resources. We thank the guidance of Zhengxuan Wu and Aryaman Arora as mentors of this project.

References

[1] A. Arora, D. Jurafsky, and C. Potts. CausalGym: Benchmarking causal interpretability methods on linguistic tasks. *arXiv:2402.12560*, 2024.

[2] N. Belrose, D. Schneider-Joseph, S. Ravfogel, R. Cotterell, E. Raff, and S. Biderman. LEACE: Perfect linear concept erasure in closed form. *Advances in Neural Information Processing Systems*, 36, 2023.

[3] X. Chen, H. Fang, T.-Y. Lin, R. Vedantam, S. Gupta, P. Dollar, and C. L. Zitnick. Microsoft coco captions: Data collection and evaluation server, 2015.

[4] A. M. Dai and Q. V. Le. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[5] A. Geiger, H. Lu, T. Icard, and C. Potts. Causal abstractions of neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 9574–9586. Curran Associates, Inc., 2021.

[6] A. Geiger, Z. Wu, C. Potts, T. Icard, and N. D. Goodman. Finding alignments between interpretable causal variables and distributed neural representations. *arXiv:2303.02536*, 2023.

[7] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering, 2017.

[8] C. Guerner, A. Svete, T. Liu, A. Warstadt, and R. Cotterell. A geometric notion of causal probing. *arXiv:2307.15054*, 2023.

[9] J. He, C. Zhou, X. Ma, T. Berg-Kirkpatrick, and G. Neubig. Towards a unified view of parameter-efficient transfer learning. In *The Tenth International Conference on Learning Representations, ICLR 2022*, Virtual Event, 2022.

[10] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. d. Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. Parameter-efficient transfer learning for NLP. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR, 2019.

[11] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models, 2021.

[12] Z. Hu, L. Wang, Y. Lan, W. Xu, E.-P. Lim, L. Bing, X. Xu, S. Poria, and R. Lee. LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In H. Bouamor, J. Pino, and K. Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5254–5276, Singapore, Dec. 2023. Association for Computational Linguistics.

[13] D. A. Hudson and C. D. Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering, 2019.

[14] K. Lasri, T. Pimentel, A. Lenci, T. Poibeau, and R. Cotterell. Probing for the usage of grammatical number. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2022.

[15] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics.

[16] X. L. Li and P. Liang. Prefix-tuning: Optimizing continuous prompts for generation. In C. Zong, F. Xia, W. Li, and R. Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online, Aug. 2021. Association for Computational Linguistics.

[17] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. In *NeurIPS*, 2023.

[18] S. Liu, H. Ye, L. Xing, and J. Zou. In-context vectors: Making in context learning more effective and controllable through latent space steering. *arXiv:2311.06668*, 2024.

[19] S.-Y. Liu, C.-Y. Wang, H. Yin, P. Molchanov, Y.-C. F. Wang, K.-T. Cheng, and M.-H. Chen. Dora: Weight-decomposed low-rank adaptation, 2024.

[20] J. Lu, V. Goswami, M. Rohrbach, D. Parikh, and S. Lee. 12-in-1: Multi-task vision and language representation learning, 2020.

[21] T. Mikolov, W.-t. Yih, and G. Zweig. Linguistic regularities in continuous space word representations. In L. Vanderwende, H. Daumé III, and K. Kirchhoff, editors, *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[22] J. Pfeiffer, I. Vulić, I. Gurevych, and S. Ruder. Mad-x: An adapter-based framework for multi-task cross-lingual transfer, 2020.

[23] E. M. Ponti, A. Sordoni, Y. Bengio, and S. Reddy. Combining modular skills in multitask learning, 2022.

[24] D. E. Rumelhart, J. L. McClelland, and P. R. Group. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1: Foundations. MIT Press, 1986.

[25] P. Smolensky. Neural and conceptual interpretation of PDP models. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 2: Psychological and Biological Models, pages 390–431. MIT Press/Bradford

Hyperparameters	**Bart-base w/ VQA for ReFT**
text prefix+suffix position p_t + s_t	{$p1$+$s1$, $p3$+$s3$, <u>$p6$+$s6$</u>}
image prefix+suffix position p_i + s_i	{$p1$+$s1$, $p3$+$s3$, <u>$p6$+$s6$</u>, $p9$+$s9$, $p12$+$s12$, $p18$+$s18$}
Tied weight p, s	{True, <u>False</u>}
Text Intervention Rank r_t	{1, <u>4</u>, 16, 64}
Image Intervention Rank r_i	{1, 4, 16, <u>64</u>, 128}
Layer L (sep. w/ ';')	<u>all</u>
Dropout	{<u>0.00</u>, 0.05}
Optimizer	AdamW
LR	{2×10^{-4}, 6×10^{-4}, 8×10^{-4}, <u>1×10^{-3}</u>, 2×10^{-3}}
Weight decay	{0, 5×10^{-3}, <u>1×10^{-2}</u>}
LR scheduler	Linear
Batch size	<u>300</u>
Warmup ratio	{0.00, 0.05, <u>0.10</u>}
Epochs	20
Clip gradient norm	{ No clip, 2, <u>5</u>}

Table 2. Hyperparameter search space of Bart-base models with ReFT on the VQAv2 development set with the best settings <u>underlined</u>. We use greedy decoding without sampling during hyperparameter tuning.

Books, Cambridge, MA, 1986.

[26] A. Suhr, S. Zhou, A. Zhang, I. Zhang, H. Bai, and Y. Artzi. A corpus for reasoning about natural language grounded in photographs, 2019.

[27] Y.-L. Sung, J. Cho, and M. Bansal. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks, 2022.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc., 2017.

[29] R. Vedantam, C. L. Zitnick, and D. Parikh. Cider: Consensus-based image description evaluation, 2015.

[30] Z. Wu, A. Arora, Z. Wang, A. Geiger, D. Jurafsky, C. D. Manning, and C. Potts. Reft: Representation finetuning for language models. 2024.

[31] Z. Wu, A. Geiger, A. Arora, J. Huang, Z. Wang, N. D. Goodman, C. D. Manning, and C. Potts. pyvene: A library for understanding and improving PyTorch models via interventions. In *arXiv:2403.07809*, 2024.

[32] Z. Wu, A. Geiger, C. Potts, and N. D. Goodman. Interpretability at scale: Identifying causal mechanisms in Alpaca. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[33] M. Zhong, Y. Shen, S. Wang, Y. Lu, Y. Jiao, S. Ouyang, D. Yu, J. Han, and W. Chen. Multi-lora composition for image generation, 2024.

[34] A. Zou, L. Phan, S. Chen, J. Campbell, P. Guo, R. Ren, A. Pan, X. Yin, M. Mazeika, A.-K. Dombrowski, S. Goel, N. Li, M. J. Byun, Z. Wang, A. Mallen, S. Basart, S. Koyejo, D. Song, M. Fredrikson, J. Z. Kolter, and D. Hendrycks. Representation engineering: A top-down approach to AI transparency. *arXiv:2310.01405*, 2023.