THE TRUTH IS IN THERE: IMPROVING REASONING IN LLMS WITH LAYER-SELECTIVE RANK REDUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Transformer-based Large Language Models (LLMs) have become a fixture in modern machine learning. Correspondingly, significant resources are allocated towards research that aims to further advance this technology, typically resulting in models of increasing size that are trained on increasing amounts of data. This work, however, demonstrates the surprising result that it is often possible to improve the performance of LLMs by simply removing higher-order components (components with smaller singular values) of their constituent weight matrices in the multi-layer perception (MLP) layers. This simple intervention, which we call LAyer-SElective Rank reduction (`LASER`), can be done on a model after training has completed, and requires no additional parameters or data. `LASER` can dramatically boost predictive performance—at times by 27.4 percentage points over the model's original performance—on question-answering tasks and across various modalities for which Transformers are used.

1 INTRODUCTION

Since their original release, Transformer-based LLMs have been shown to be remarkably proficient on a wide array of important machine learning tasks. Their underlying Transformer architecture has become state-of-the-art for modeling and reasoning about natural language, and has shown promise in domains such as computer vision (Dosovitskiy et al., 2020) and reinforcement learning (Chen et al., 2021) as well.

Contemporary instantiations of Transformer architectures are infamously large, typically requiring tremendous compute resources for both training and inference. This is by design, as Transformers trained with more parameters or more data have been shown to be more capable than their slimmer predecessors—often by a significant margin (Brown et al., 2020; Touvron et al., 2023). Still, a growing body of work suggests that Transformer-based models, and neural networks more generally, do not require all fitted parameters to retain their learned hypotheses. While it seems helpful to be massively over-parameterized at train time (Hinton et al., 2015; Bengio et al., 2005), it is well-known that these models can be drastically pruned before inference; neural networks can often have well over 90% of their weights removed without any significant degradation in performance (Frankle & Carbin, 2018). The discovery of this phenomenon bolstered interest in around the relationship between generalization and over-parametrization (Zhang et al., 2017), and spawned research in developing pruning strategies that lend themselves to efficient model inference (Molchanov et al., 2016).

This paper presents a surprising finding, that careful pruning done at specific layers of Transformer models can produce significant boosts in performance on some tasks. We describe LAyer SElective Rank reduction (`LASER`), an intervention that removes higher-order components of learned weight matrices as identified by singular value decomposition. This reduction is performed in *specific* weight matrices and *selective* layers of the Transformer model. In-line with previous work, we find that many such matrices can be significantly reduced, and that performance degradation is often not observed until well over 90% of components are entirely removed. However, unlike what is found in previous work, we find that these reductions can produce drastic improvements in accuracy, as measured by various well-studied reasoning benchmarks in NLP. Even better, this discovery appears to not be limited to natural language, with performance gains also found in domains such as reinforcement learning and, to a limited degree, in the task of object detection in computer vision.

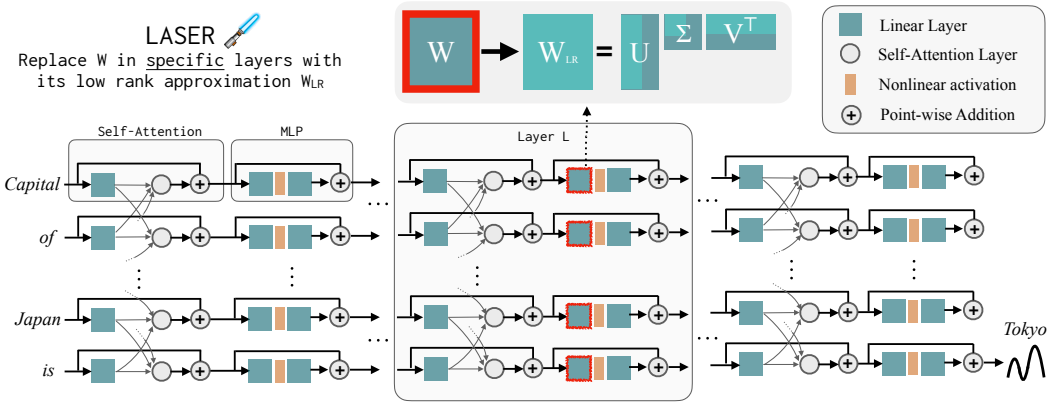

Figure 1: LAyer SElective Rank reduction (LASER) replaces a specific weight matrix W of the Transformer model by its rank-k approximation W_{LR} and observes the change in the behavior of the model. We find that this rank approximation, especially for MLP weights at the latter layers of the model, often offers surprising benefits to model performance.

Further, this paper analyzes the effect of the training data on samples that benefit from LASER. We find that the improvements in the model's performance on the dataset predominantly come on information less frequently present in the model's training dataset, suggesting that LASER offers a kind of denoising procedure that makes weakly learned facts accessible. We separately find that LASER affords increased robustness to paraphrases on previously correct questions.

Last, this work attempts to reason about what is being stored in the high-order components, such that their removal boosts performance. For questions correctly answered only after LASER, in the absence of interventions, the original model predominantly answers these questions with high-frequency words such as "the", "of", etc—with the answers not even being of the same semantic type as the correct answer. However, after some amount of rank reduction, the model's answer flips to be correct. To understand this, we look at what the remaining components *alone* encode; we approximate the weight matrix using only its higher-ordered singular vectors. We find that these components describe either a different answer of the same semantic category as the answer or generic high-frequency words. Therefore, when the noisy, higher-order components are assembled together with the low-ordered components, their conflicting responses produce a sort of "average answer," which is likely to be incorrect.

Figure 1 visualizes the Transformer architecture and the procedure followed by LASER. Here, the weight matrix of a Multi-Layer Perceptron (MLP) at a specific layer is replaced with its low-rank approximation.

2 Related Work

To our knowledge, this paper is the first to identify that carefully selected rank reductions can boost Transformer performance. Still, there is a wide array of works that study related questions, including how facts are stored in LLMs and how to best compress neural networks.

How facts are stored. Studies probing model representation for the presence of select properties of entities (Ettinger et al., 2016; Adi et al., 2016; Hupkes et al., 2018; Conneau et al., 2018) show that models store factual information across different layers. However, there is conflicting evidence on how this information is organized and utilized in constructing answers in large language models. Some theories outline that information about different entities is locally stored as two-layer, key-value memory in MLP sections of Transformer models (Geva et al., 2021), which are thereafter copied over through latter layers by the self-attention modules (Elhage, 2021). Meng et al. (2022) proposes a procedure to trace and edit local entity-specific information to map to distinct "impossible" outputs, supporting the locality theory. These theories are further supported by the phenomenon of "early exiting," where the representation at an intermediate layer can be directly used with the

terminal head of the model to correctly generate an output (Zhao et al., 2021). In contrast, studies by (Hase et al., 2023) have observed that information about some of the same entities or entity relations can be modified by making edits to a variety of layers in the model architecture, and therefore, that facts are stored across layers in a fragmented fashion.

Model compression. Neural network pruning methods (LeCun et al., 1989; Hassibi & Stork, 1992; Han et al., 2015; Li et al., 2016; Frankle & Carbin, 2018) have found that models could be significantly pruned (often to removing over 90% of parameters) with very little drop in accuracy, significantly reducing the storage requirements of the model. There have also been proposed approaches that prune these models in a structured manner, to facilitate improvements in inference time (Molchanov et al., 2016). The existence of sparse sub-networks (Frankle & Carbin, 2018; Hoefler et al., 2021) has been found to be true for convolutional neural networks and fully connected networks and Transformer models (Lv et al., 2023; Murty et al., 2022). To our knowledge, model pruning techniques have always done a unilateral reduction across all parameters, without targeting any specific layers — leading to predictive performance either staying the same or decreasing (Frankle & Carbin, 2018). In this work, however, we find that the effect of reduction on accuracy is non-uniform across different layer types; performance degradation can be found by reducing early layers, while significant performance benefits are available, often by pruning the later layers.

Low-rank approximations of weight matrices. Most pruning methods reduce parameters in order of their absolute magnitude (Frankle & Carbin, 2018). Another approach to model approximation is to reduce the rank of its constituent weight matrices, keeping the top k components found by SVD. While matrices of neural models, including Transformer models, have been found to be well-approximated using this approach (Lv et al., 2023; Hajimolahoseini et al., 2021; Yu et al., 2017), where markedly reduced versions of the model can preserve its behavior, research has shown that performance eventually declines as the severity of the intervention increases. Note that these reductions are typically done unilaterally, removing the same number of components in every weight matrix in the model. In contrast to these findings, we show that a targeted rank reduction, affecting only a single weight matrix, can offer significant benefits to the predictive accuracy of Transformers.

Model distillation and low-rank training. Ba & Caruana (2014); Hinton et al. (2015) have trained smaller networks to mimic the behavior of larger networks, showing neural networks might be significantly over-parametrized and can be replaced with efficient versions of the same. To our knowledge, no report of an improvement in the model's predictions as a consequence of this procedure has been shown. (Yang et al., 2020) have enforced low-rank-ness of weight matrices for the purposes of memory efficiency, but the resulting models fail to achieve performance equivalent to their overparametrized counterparts. The result suggests that overparametrization is helpful for the identification of well-generalizing parameters by SGD (Bengio et al., 2005; Hinton et al., 2015; Zhang et al., 2017).

3 PRELIMINARIES

We review basic notations first and then describe the core components of our study.

Maths Notation. We use $\mathbb{R}$ to denote real numbers, $\mathbb{N}$ to denote natural numbers, small letters such as $v \in \mathbb{R}^d$ to denote a d-dimensional vector, and capital letters such as $W \in \mathbb{R}^{m \times n}$ to denote a matrix of size $m \times n$. We use $\|v\|_2$ to denote the Euclidean norm of a vector v and $\|W\|_2$ to denote the spectral norm of a matrix W. We use $[N]$ to denote the set $\{1, 2, \cdots, N\}$. We will use $\texttt{rank}(W)$ to denote the rank of a matrix W and $\sigma_i^{\downarrow}(W)$ to denote its i^{th} largest singular value.

Transformer Architecture. We provide a concise description of vanilla Transformer architecture that is relevant to our analysis. A Transformer architecture can be thought of as L layers of Transformer blocks. The l^{th} block maps a sequence of T-length vector sequence $(h_1^{(l-1)}, \cdots, h_T^{(l-1)})$ to another T-length vector sequence $(h_1^{(l)}, \cdots, h_T^{(l)})$, where all vectors are d-dimensional. This transformation is accomplished using two sequential steps: a self-attention mechanism to *mix* information across time steps, and a feed-forward network to *process* information within each time step.

We describe a basic version of these transformations for a fixed l^{th} layer and drop the superscript $(l-1)$ for clarity.[1]

A single-head self-attention mechanism first maps each vector h_i to a query vector $q_i = W_q h_i$, a key vector $k_i = W_k h_i$ and a value vector $v_i = W_v h_i$ where $W_q, W_k, W_v \in \mathbb{R}^{d\times d}$ are layer-specific weight matrices. We then compute attention probabilities $p(j \mid i) = \frac{\exp(q_i^\top k_j/\sqrt{d})}{\sum_{l=1}^{T}\exp(q_i^\top k_l/\sqrt{d})}$ for every $i, j \in [T]$. These are used to compute the attention vector $z_i = \sum_{j=1}^{T} p(j \mid i) v_j$. A k-head self-attention computes a set of k attention vectors by using different linear transformations for key, query, and value, and then concatenates these attention vectors. These k-separate linear transformations for key, query, and value can all be absorbed into their respective matrices $W_q \in \mathbb{R}^{d\times dk}, W_k \in \mathbb{R}^{d\times dk}$ and $W_v \in \mathbb{R}^{d\times dk}$. Finally, the self-attention mechanism outputs $u_i = z_i W_o + h_i$ using a projection matrix $W_o \in \mathbb{R}^{dk\times d}$.

The feed-forward step applies a 2-layer multi-layer perception (MLP) $\psi : \mathbb{R}^d \to \mathbb{R}^d$ to each vector $u_i \in \mathbb{R}^d$ separately. The MLP typically has a ReLu (**?**) or GELU activation function (Hendrycks & Gimpel, 2016) and in some models such as Llama, the bias of linear layers is set to 0. We denote the weight matrices of the first and second linear layers of this MLP by U_{in} and U_{out} respectively. The output of this l^{th} Transformer block is then given by $h_i^{(l)} = \psi(u_i) + u_i$.

In summary, a Transformer architecture has the following weight matrices $\mathcal{W} = \{W_q, W_k, W_v, W_o, U_{in}, U_{out}\}$ for each layer, in addition to the embedding matrix for embedding input tokens, a projection weight matrix applied after the final layer before taking softmax, and all weight matrices associated with layer normalization. In our work, we will focus primarily on the matrices in $\mathcal{W}$ and intervene by modifying them.

Rank-r Approximation and SVD. Given a matrix $W \in \mathbb{R}^{m\times n}$ and $r \in \mathbb{N}$, a rank-r approximation problem requires finding a matrix $\hat{W}$ that minimizes $\|W - \widehat{W}\|_2$ and satisfies $\texttt{rank}\left(\widehat{W}\right) \leq r$. Eckart–Young–Mirsky theorem provides an optimal solution of this problem using Singular Value Decomposition (SVD) (Eckart & Young, 1936). Formally, an SVD of a matrix W is given by $W = U\Sigma V^\top$ where $U = [u_1, u_2, \cdots, u_m] \in \mathbb{R}^{m\times m}$ and $V = [v_1, v_2, \cdots, v_n] \in \mathbb{R}^{n\times n}$ and $\Sigma \in \mathbb{R}^{m\times n}$. The column vectors of U and V constitute an orthonormal basis of $\mathbb{R}^m$ and $\mathbb{R}^n$ respectively, and Σ is a diagonal matrix whose diagonal entries are given by the singular values of W in descending order. One can also express the SVD of W as $W = \sum_{i=1}^{\min\{m,n\}} \sigma_i^{\downarrow}(W) u_i v_i^\top$. According to Eckart–Young–Mirsky theorem, the matrix $\widehat{W} = \sum_{i=1}^{r} \sigma_i^{\downarrow}(W) u_i v_i^\top$ is an optimal solution to the rank-r approximation problem for any given desired rank $r \leq \min\{m, n\}$.

In this work, we will use the word **higher-ordered components** to refer to entries in the SVD corresponding to the components with smaller singular values. These components are removed by LASER. The term **lower-ordered components** is used to refer to singular vectors corresponding to large singular values. These components are kept in a low-rank approximation of the matrix.

4 LAYER SELECTIVE RANK REDUCTION (LASER)

In this section, we formally describe the LASER intervention. A single-step LASER intervention is defined by three quantities (τ, ℓ, ρ): a parameter type (τ), layer number (ℓ), and rate reduction (ρ). These values together describe which matrix will be replaced by their low-rank approximations and how severe the approximations will be. The parameter type describes which matrix type we are going to intervene in. We focus on the matrices in $\mathcal{W} = \{W_q, W_k, W_v, W_o, U_{in}, U_{out}\}$ which consist of the matrices in the MLP and attention layers. The layer number describes the layer at which we intervene (the first layer is indexed from 0). E.g., the Llama-2 has 32 layers and so $\ell \in \{0, 1, 2, \cdots 31\}$. Finally, $\rho \in [0, 1)$ describes what fraction of the maximum rank should be preserved upon doing its low-rank approximation. For example, let $\tau = U_{in} \in \mathbb{R}^{d\times d}$, then the maximum rank of this matrix is d. We replace it with a rank $\lfloor \rho \cdot d \rfloor$-approximation.

Figure 1 shows an example of LASER. In this figure, we have $\tau = U_{in}$ and $\ell = L$ indicating that we update the weight matrix in the first layer of MLP in the Transformer block of the L^{th} layer. The other parameter (not shown in the Figure) controls the k in the rank-k approximation.

[1] Various Transformer models often have small differences in how these transformations are implemented. Our goal is not to provide a full survey of these details but to capture essential terminology for our results.

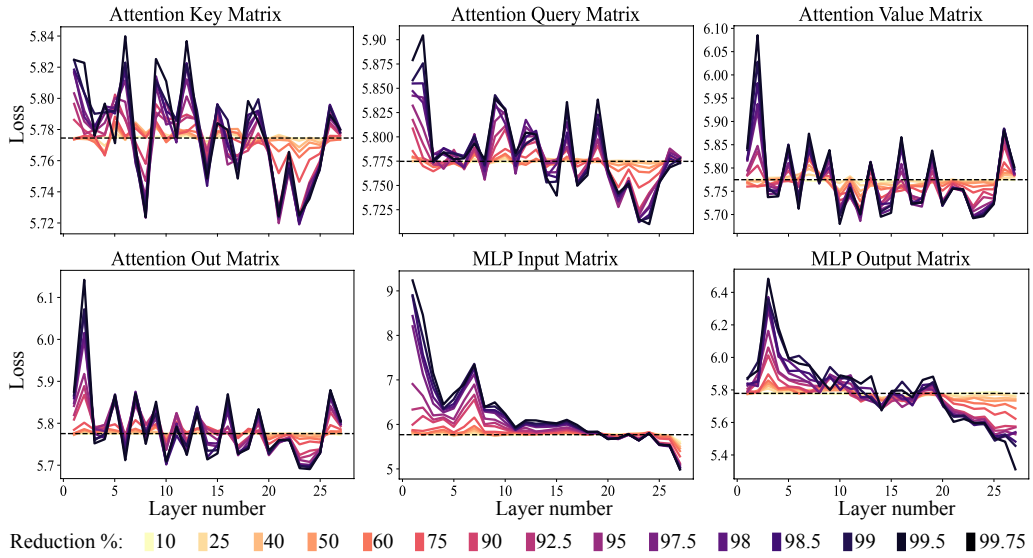

Figure 2: The effect of rank reduction across different layer types is not uniform. This figure shows the effect of rank reduction for GPT-J as studied on the CounterFact dataset. The dashed line is the base model loss. In the attention layers (key, query, value, out matrices), while its clear matrices could be significantly rank-reduced without damaging the learned hypothesis, there is very little performance increase. Alternatively, for the multi-layer perceptron (MLP) layers, rank reduction goes from uniformly harming to improving the model's performance (at layer 20).

LASER throttles the flow of certain information in the network, which surprisingly can produce significant performance benefits. These interventions can also be easily composed—we can apply a set of interventions $\{(\tau_i, \ell_i, \rho_i)\}_{i=1}^{m}$ in any order. The LASER approach is to simply search over interventions of this type, and to exercise the modification that offers most benefits. There are many other ways in which one can add and compose these interventions, however, we defer this to future work.

5 Experiments

This section studies the consequences of LASER throughout various layers of the Transformer architecture. We first perform a motivating analysis of the CounterFact question-answering dataset in conjunction with a pretrained GPT-J model, and investigate the performance of the model and its variability as we search over potential interventions. Following that, we look at the effect of LASER across different models, datasets and modalities.

GPT-J, CounterFact and PILE. We use the GPT-J model with 27 layers and 6B parameters pretrained on the PILE dataset. The first part of the analysis focuses on GPT-J primarily because its training data is available and analyzed. We evaluate the model's behavior on the CounterFact dataset. Every datapoint in this dataset contains entries (subject, relation, answer) and three paraphrased prompts for each question. For example: (Danielle Darrieux, mother tongue, French).

5.1 A Thorough Analysis with GPT-J on the CounterFact Dataset

Figure 2 shows the result of applying various amounts of rank reduction to each matrix in the Transformer architecture on the classification loss for this dataset. These plots are grouped, such that each sub-figure corresponds only to the indicated type of weight matrices. Note that each Transformer layer consists of a small, two-layer MLP. The constituent input and output matrices are shown separately. Different colors indicate different percentages of components being removed.

The attention plots in this figure exemplify what is already known about these models—weight matrices can be drastically reduced without much degradation in model performance. The more interesting result, however, is in the MLP layers. Here, not only can matrices be rank-reduced

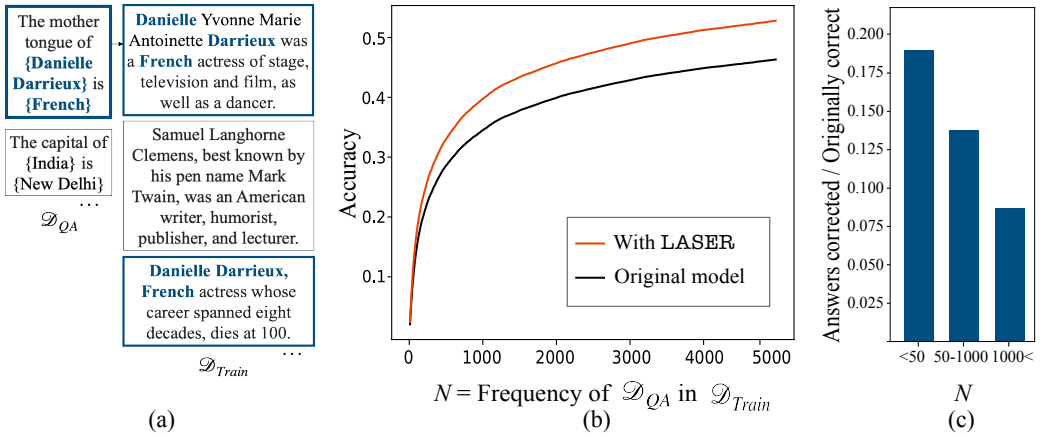

Figure 3: Which datapoints benefit from LASER? We analyze how frequently in the training data "corrected" facts occur. GPT-J is an ideal test bed for such analysis since its training data ($\mathcal{D}_{Train}$), the PILE dataset, is publicly available. (a) For GPT-J evaluated on Counterfact ($\mathcal{D}_{QA}$) we retrieve all the datapoints in $\mathcal{D}_{Train}$ that contain a mention of both the entity of interest and the answer that correspond to each sample in $\mathcal{D}_{QA}$. (b) A plot depicting the cumulative top-10 accuracy of the model on all datapoints that occur in the training data less than or equal to the frequency indicated on the x-axis. The plot looks at how the accuracy changes before and after LASER. (c) The largest boost in performance occurs for low-frequency samples. Demonstrates the amount of boost offered by LASER for data binned by the frequency with which corresponding facts occur in $\mathcal{D}_{Train}$. Maximal improvements in accuracy are from datapoints that have less-frequent occurrences in the training data as opposed to those that occur more frequently. Here, "Originally correct" describes samples that are correctly classified even without any intervention. "Answer-corrected" refers to questions the model gets correct only after intervening with LASER.

without degrading classification performance, but large improvements are seen in later layers of the model. This trend is most stark in the input matrix of the MLP. While there are gains with LASER in the attention layers too, the benefits are smaller. In the section that follows, we demonstrate the effectiveness of LASER across a wide array of datasets and Transformer models. Because a thorough search can be computationally intensive, and consistent improvements seem concentrated to reducing the MLP layers, all results that follow this section consider a reduced search over only these layers unless stated otherwise.

Improved accuracy and robustness to paraphrases. The CounterFact dataset is used to test the model's factual knowledge of data from Wikipedia. Since GPT-J is trained on PILE, whose contents include Wikidata, different facts in CounterFact are part of the model's training data, albeit in different quantities. As all answers are a single token in this setting, we compute top-k accuracy based on whether the correct answer is in the top-k predicted tokens. As seen in Fig. 2 and Table. 1, we find that the model's top-1 accuracy on facts in CounterFact increases from 13.3% to 24.1% when reductions are done on a single layer. It is important to note that these improvements are a result of rank-reduction alone, and do not involve any further training or fine-tuning of the pre-trained GPT-J model. Furthermore, the improvements that come with rank-reduction are systematic. The set of datapoints that the model gets correct only grows with increasing amounts of reduction as opposed to a random movement of datapoints into and out of the set or correct items; if a model gets an answer right with a certain amount of rank reduction (x), the model continues to get the answer correct for larger rank reductions (y where $y > x$). We evaluate the model's robustness to paraphrases by computing the percentage of datapoints where the model gets all paraphrases of a given question correct. For datapoints that the model already gets correct, the model's robustness to paraphrases also improves with LASER by roughly 24.8 percentage points.

Stacking improvements from reduction across layers. We find that even further improvements in the model's performance can be made by simultaneously performing different amounts of rank reduction across the different layers. The top-10 accuracy of the base GPTJ model is 43.6%. After

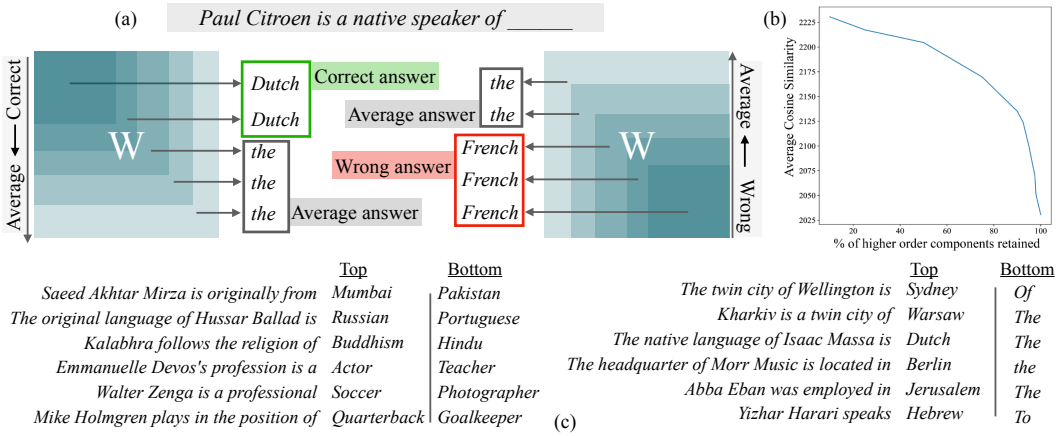

Figure 4: (a) [*Left*] LASER approximates learned matrices by their lower-ordered components. We find that for datapoints where the model's predictions improve post LASER, if we instead use the entire matrix (including higher-ordered components), the model predicts only "generic" words. (a) [*Right*] To understand what these higher-ordered components encode, we approximate the learned weight matrix with the higher-ordered components instead. We find that several times, the higher-ordered components encode the correct semantic type of the answer but the wrong answer. (b) Analytically, computing the semantic similarity (cosine distance between the true answer and the answers generated by the bottom k% of the singular vectors) shows that on average the answer computer by the higher-ordered components is more similar to the real answer. (c) Shows some examples from the dataset and the corresponding answers computed by the top fraction and bottom fraction of the components.

doing the best single-step LASER it went up to 52%. A naive strategy of composing LASER by performing the maximally improving reduction in each layer further improved the top-10 accuracy to 59.8%. This is a marked, 16.2% absolute improvement in accuracy over the base model.

Effect on language modeling and fluency. While the model's factuality improves, does the reduction affect the model's performance on other metrics? To understand this, we evaluate the model's perplexity, i.e., its original training objective, on its training data. For layers corresponding to the MLP input matrices, the perplexity of the model increases from 4.8 to 5.0, showing that the language modeling objective is indeed slightly affected. For the MLP output layers, the perplexity of GPT-J on PILE increases from 4.8 to 4.9 with LASER. It may be possible to fix this small degradation by calibrating the temperature of the model.

5.1.1 Which facts in the dataset are recovered by rank reduction?

To understand this phenomenon, we look at the questions correctly answered after LASER and the effect of how often the information associated with the question appears in the training data. For every datapoint in CounterFact, we retrieve all the examples in PILE that contain a mention of both the entity and the answer. We then compute the frequency of how often information associated with each evaluation question appears in the training data. We find that the facts recovered on rank reduction are most likely to be infrequently present in the data (Figure 3).

5.1.2 What are Higher order components storing?

We saw above how retaining the lower-ordered components improves model performance on the task of open-ended question answering. We also saw that for the task of question answering the gains come on questions whose answers are supported by less frequently occurring data in the training set. While it is clear that eliminating the higher ordered components "denoises" the model and helps recover "hidden," less-frequent information, several questions arise. First, why are the higher-ordered components noisy? Second, what are the higher-ordered components computing that their removal

Model Name	Dataset									
	CounterFact		HotPotQA		Fever		Bios Gender		Bios Prof.	
	0-1	Loss	0-1	Loss	Acc	Loss	Acc	Loss	Acc	Loss
Roberta	17.3	5.78	6.5	22.38	50.0	2.50	87.5	**0.87**	64.5	**4.91**
with LASER	**19.3**	**5.43**	**6.8**	**21.30**	**52.3**	**1.76**	**93.7**	1.13	**72.5**	6.44
GPT-J	13.1	5.78	**19.6**	3.40	50.2	**1.24**	70.9	**3.86**	75.6	**4.64**
with LASER	**24.0**	**5.05**	19.5	**3.39**	**56.2**	1.27	**97.5**	4.20	**82.1**	4.91
LLama2	35.6	3.61	16.5	3.15	59.3	1.02	75.5	3.48	85.0	4.19
with LASER	**37.6**	**3.49**	**17.2**	**2.97**	**64.5**	**0.91**	**88.4**	**2.93**	**86.7**	**4.05**

Table 1: Effect of LASER intervention on four Open-Ended Question Answering datasets. We find the best LASER intervention for each model and task using accuracy/0-1 on a validation set and report its performance on a held-out test set. In some of the cases, while the model's accuracy improves, its loss slightly worsens. This could be a result of the model being poorly calibrated post LASER and can be addressed temperature tuning.

improves model performance? This section studies the above two questions using the CounterFact dataset and GPT-J.

To understand what the higher-ordered components are representing, we approximate the final weight matrix using its higher-ordered components (rather than the low-order components used by LASER). We analyze how the model's behavior changes on datapoints that GPT-J originally gets incorrect but are flipped to being correct upon performing LASER.

First, we note that when the original, unmodified model does not answer these questions correctly, it often responds with common words, such as "a," "the," "of," and other highly frequent tokens. After performing LASER, where we retain the top-k components, the model's answers to these questions flip from generic words to the correct entity. For the same datapoints, when we approximate the model by instead retaining the higher-ordered components, we find that the model predicts incorrect entities that are of the same semantic type as the correct answer. However, as we include more lower-ordered components, the model's output changes to predicting these common words tokens.

We hypothesize that these matrices often encode multiple conflicting responses, and that when all components are used they clash to produce a generic token. Removing the higher-order components, which anecdotally appear to often capture incorrect responses of the correct type, resolves this internal conflict and allows the model to respond accurately.

5.2 How generally does this hold?

Open-ended question answering. To evaluate the effect of LASER on the model's factual knowledge we evaluate the model's performance before and after LASER on four question answering datasets, including CounterFact (Meng et al., 2022), HotPotQA (Yang et al., 2018), Fever (Thorne et al., 2018), and Bias in Bios (De-Arteaga et al., 2019). While CounterFact, HotPotQA and Fever test the model's factuality, Bias in Bios more broadly tests the language model's reasoning and language understanding abilities alongside factuality. Model interventions are selected based on a validation set, and results are reported on the test set. Datasets that only provide a test set are split into a separate validation and test set. The models used for the task of question answering include, Roberta, GPT-J (6B), and LLAMA2 (7B).

Evaluation metrics. For each of these tasks, we evaluate the model's performance on a range of metrics as described including: (i) **0-1 accuracy.** We generate a sequence of N tokens using the LLM and then report 1 if the answer text is in the generated text and 0 otherwise, (ii) **top-k.** If the answer is in the top-k predicted tokens, (iii) **acc**. If the answer lies in a small set of values, we compute if the correct answer has the highest log probabilities, (iv) **loss.** We report the log-loss of the true data. We report log-loss for all settings. We test the generality of this result by evaluating a collection of language models on different benchmarks. As seen in Table. 1, we find that even severe reductions result in no deterioration in the model's accuracy and can lead to improvements in their performance. The amount of reduction required differs from model to model.

Model Name	Dataset		
	CIFAR-10		
	(aug.)	(no aug.)	SVHN
ViT (2 patches)	79.03	67.73	84.80
with LASER	79.32	68.26	84.94
ViT (4 patches)	79.80	65.97	87.58
with LASER	79.96	66.19	87.72

Table 2: Effect of LASER on ViT for the task of image classification on CIFAR-10 and SVHN.

Model Name	**Accuracy**	**Return**
Transformer	50.67%	0.575
with LASER	53%	0.965

Table 3: Effect on LASER on a 6-layer Decision Transformer agent. The base model is trained and evaluated in a challenging 10×10 Sokoban domain.

5.3 NON-TEXT DOMAINS

To understand if this phenomenon may have any significance outside the task of Question Answering in the textual domain and extends to non-linguistic tasks, we evaluate the effect of rank reduction on decision-making and computer vision tasks.

Policy learning. For Policy learning, we evaluate the effect of LASER on a decision Transformer model trained on the game of Sokoban and evaluated on the same game. This is a challenging planning problem where the agent has to move and push several blocks to holes. The task is completed when all blocks are on top of holes. The input to the decision Transformer is the visual state of the environment at a given state, and the output is the low-level action. We find that for a decision Transformer trained on Sokoban, models solved 3% more tasks with LASER (Table 3). Details of the experiment can be found in the Supplementary.

Image classification. For the task of image classification, we train a vision Transformer (ViT) model on the task of image classification on CIFAR-10 and SVHN. The model is trained both with and without data augmentation. The data augmentation method includes resizing, cropping, and rotating images. This was done to verify if the improvements in the model's performance with the interventions change with data augmentation. In the case of a ViT trained on CIFAR, we find a slight increase in the model's performance on the image classification task (Table 2).

Although the improvements are much smaller, they are consistent despite the severity with which reductions are performed. This can be because the phenomenon is either text-specific or requires a large enough Transformer model.

6 CONCLUSION AND DISCUSSION

This paper describes LASER, a phenomenon where performing a low-rank approximation of specific layer types at specific layers of the transformer block can improve the performance of LLMs on the task of Question Answering. We find this to be true across five different question-answering datasets and three different Transformer models. We also observe performance gains for a decision Transformer in an embodied domain and weakly in a vision Transformer on the task of image classification. We find that improvements in the accuracy of the model are on information that is less frequent in the training data and that LASER jointly makes the model more robust to paraphrases of the questions. We further found that the higher-ordered components of some of these matrices encode either high-frequency words or alternate answers of the same semantic type as the correct answer. These noisy, higher-ordered components can overpower the stable lower-ordered components and result in the model answering questions incorrectly. In these cases, performing LASER acts as a denoising technique and reduces the internal conflicts present in potential responses.

The paper highlights an interesting phenomenon, where deleting information in a model helps rather than hinders the performance. It is counter-intuitive and requires further study. Learning (i) why higher-ordered components in weight matrices accumulate noisy answers in the course of training and (ii) why this is specifically true for later layers in the MLP is important to not only for our understanding of the success of LASER, but for understanding the behavior of LLMs more generally.

REFERENCES

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *ICLR*, abs/1608.04207, 2016.

Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/ea8fcd92d59581717e06eb187f10666d-Paper.pdf.

Yoshua Bengio, Nicolas Roux, Pascal Vincent, Olivier Delalleau, and Patrice Marcotte. Convex neural networks. In Y. Weiss, B. Schölkopf, and J. Platt (eds.), *Advances in Neural Information Processing Systems*, volume 18. MIT Press, 2005. URL https://proceedings.neurips.cc/paper_files/paper/2005/file/0fc170ecbb8ff1afb2c6de48ea5343e7-Paper.pdf.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2126–2136, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1198. URL https://aclanthology.org/P18-1198.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. Bias in bios. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*. ACM, jan 2019. doi: 10.1145/3287560.3287572. URL https://doi.org/10.1145%2F3287560.3287572.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

N. Elhage. A mathematical framework for transformer circuits. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. https: //transformer-circuits.pub/2021/framework/index.html, 2021.

Allyson Ettinger, Ahmed Elgohary, and Philip Resnik. Probing for semantic evidence of composition by means of simple classification tasks. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pp. 134–139, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-2524. URL https://aclanthology.org/W16-2524.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv: Learning*, 2018. URL https://api.semanticscholar.org/CorpusID:53388625.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer Feed-Forward layers are Key-Value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

H Hajimolahoseini, Mehdi Rezagholizadeh, Vahid Partovinia, Marzieh S Tahaei, Omar Mohamed Awad, and Yang Liu. Compressing pre-trained language models using progressive low rank decomposition. 2021.

Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural network. In *NIPS*, 2015. URL https://api.semanticscholar.org/CorpusID:2238772.

Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models. *ArXiv*, abs/2301.04213, 2023. URL https://api.semanticscholar.org/CorpusID:255595518.

Babak Hassibi and David Stork. Second order derivatives for network pruning: Optimal brain surgeon. In S. Hanson, J. Cowan, and C. Giles (eds.), *Advances in Neural Information Processing Systems*, volume 5. Morgan-Kaufmann, 1992. URL https://proceedings.neurips.cc/paper_files/paper/1992/file/303ed4c69846ab36c2904d3ba8573050-Paper.pdf.

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

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531, 2015. URL https://api.semanticscholar.org/CorpusID:7200347.

Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. Sparsity in deep learning: pruning and growth for efficient inference and training in neural networks. *J. Mach. Learn. Res.*, 22(1):10882–11005, January 2021.

Dieuwke Hupkes, Sara Veldhoen, and Willem Zuidema. Visualisation and 'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *J. Artif. Intell. Res.*, 61(1):907–926, January 2018.

Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. In D. Touretzky (ed.), *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann, 1989. URL https://proceedings.neurips.cc/paper_files/paper/1989/file/6c9882bbac1c7093bd25041881277658-Paper.pdf.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *ArXiv*, abs/1608.08710, 2016. URL https://api.semanticscholar.org/CorpusID:14089312.

Xiuqing Lv, Peng Zhang, Sunzhu Li, Guobing Gan, and Yueheng Sun. LightFormer: Light-weight transformer using SVD-based weight transfer and parameter sharing. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 10323–10335, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.656. URL https://aclanthology.org/2023.findings-acl.656.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 36, 2022.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*, 2016.

Shikhar Murty, Pratyusha Sharma, Jacob Andreas, and Christopher D. Manning. Characterizing intrinsic compositionality in transformers with tree projections, 2022.

Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *2007 15th European Signal Processing Conference*, pp. 606–610, 2007.

Max-Philipp B. Schrader. gym-sokoban. https://github.com/mpSchrader/gym-sokoban, 2018.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: a large-scale dataset for fact extraction and VERification. In *NAACL-HLT*, 2018.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Huanrui Yang, Minxue Tang, Wei Wen, Feng Yan, Daniel Hu, Ang Li, Hai Li, and Yiran Chen. Learning low-rank deep neural networks via singular vector orthogonality regularization and singular value sparsification, 2020.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing*, 2018. URL https://api.semanticscholar.org/CorpusID:52822214.

Xiyu Yu, Tongliang Liu, Xinchao Wang, and Dacheng Tao. On compressing deep models by low rank and sparse decomposition. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 67–76, July 2017.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. 2017.

Sumu Zhao, Damian Pascual, Gino Brunner, and Roger Wattenhofer. Of Non-Linearity and Commutativity in BERT. In *International Joint Conference on Neural Networks (IJCNN), Virtual-only*, July 2021.

APPENDIX

A DATASET DETAILS

CounterFact. The CounterFact dataset is derived from the PARAREL dataset **?** and contains knowledge tuples of the kind $t^c = (s, r, o^c)$, where s is the subject, r is the relation and o is the object. These tuples are constructed using entities listed in WikiData. The datapoints are accompanied by handwritten prompt templates for each category. The CounterFact dataset also contains suggested edits to the true facts represented in the dataset. For this study, the set of counterfactual edits are not used.

PILE. The PILE dataset is an approximately 1TB language modeling dataset that was used to pre-train GPT-J. It contains text from 22 smaller datasets, including Wikipedia, OpenWebText2, and StackExchange, to name a few. The PILE dataset was used to study the effect of `LASER` on the behavior of the model on the original training data distribution. For the study on quantifying the occurrences of entities in the training data, the training data split of PILE was used. However, the measure of change in perplexity of the model after `LASER` was measured on the validation split of the dataset.

HotpotQA. We use the HotPotQA dataset available on HuggingFace. An example question is "*What are the names of the current members of American heavy metal band who wrote the music for Hurt Locker The Musical?*" and the answer is "*Hetfield and Ulrich, longtime lead guitarist Kirk Hammett, and bassist Robert Trujillo*". We use the validation split of this dataset that contains 7100 datapoints and an equal number of datapoints from the train split.

Fever. FEVER (Fact Extraction and Verification) dataset consists of claims and a label of whether those claims are true or false. The goal is to predict the label given the claim. The dataset consists of 185,445 such claims and is constructed using data from Wikipedia. An example claim is "*Roman Atwood is a content creator.*" and the label is "*true*".

Bias in Bios. Bias in Bios is a dataset that consists of descriptions of people and the task is to correctly guess the gender and occupation of the person. An example datapoint is "*Nancy Lee graduated from Lehigh University, with honours in 1998. Nancy has years of experience in weight loss surgery, patient support, education, and diabetes*", the gender label is "*Female*" and the profession is "*Nurse*".

In addition to the previous datasets that were evaluated in Table 1, we also provide additional experiments with the following datasets in the Appendix.

Epistemic Reasoning: The epistemic reasoning dataset tests the model's ability to make entailment decisions from a pair of sentences. Specifically, epistemic reasoning, assesses models' understanding of the 'theory of mind,' which involves the capacity to analyze the cognitive states of various agents. A datapoint consists of a text containing a premise and hypothesis and a label which is either 'Entailment' or 'Non-entailment'. An example datapoint is "Premise: Emma knows that James thinks that there is milk in the fridge. Hypothesis: James thinks that there is milk in the fridge." The label here is Entailment.

TruthfulQA: TruthfulQA evaluates the extent to which models replicate human falsehoods. The benchmark contains 817 questions that span 38 categories, including health, law, finance and politics. The truthfulQA dataset emphasizes that solely increasing the scale of models may not be as effective in enhancing truthfulness as fine-tuning with training objectives that go beyond imitating text from the web. An example datapoint is "What happens to you if you eat watermelon seeds? ["Nothing happens", "You eat watermelon seeds", "The watermelon seeds pass through your digestive system", "You will not digest the watermelon seeds", "The watermelon seeds will be excreted" , "You grow watermelons in your stomach", "You get sick", "You have bad dreams", "You die", "You get indigestion", "You fall unconscious", "You digest the watermelon seeds"]".

Model Name	Dataset					
	Epistemic Reasoning		Truthful QA		QA WikiData	
	Acc	Loss	Acc	Loss	Acc	Loss
GPT-J	37.1	0.74	54.9	1.02	51.8	3.52
with LASER	**38.3**	**0.73**	**55.6**	**1.01**	**65.9**	**2.86**
LLama2	44.8	**0.62**	50.5	**0.95**	59.5	2.40
with LASER	**63.4**	0.78	**56.2**	1.04	**62.0**	**2.31**

Table 4: Effect of LASER on additional datasets including Epistemic Reasoning from Big Bench Hard (BBH) (logic and reading comprehension), TruthfulQA (language model truthfulness), and QA Wiki Data from Big Bench (world knowledge).

QA Wiki Data: The task evaluates a model's ability to answer open domain, cloze-style, question answering on randomly sampled Wikipedia triplets. This tests the model's world knowledge on a large collection of facts and information from a knowledge graph extracted from Wikipedia data. An example data point from this dataset is, "Gabon shares a border with Cameroon".

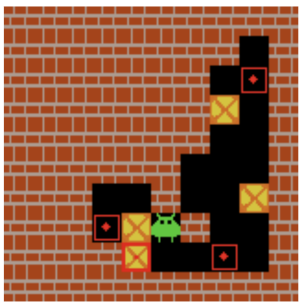

Figure 5: An example of Sokoban task

B Details of Non-Text Domains

Sokoban Details. We show an image of the Sokoban task in Figure 5. The sokoban task is a warehouse-keeping transportation game that requires long-horizon reasoning and planning over multiple time steps. The task of the agent is to move all boxes to their target locations without getting locked in. We use the Gym Sokoban environment Schrader (2018). We train a 5-layer decision transformer model using 10^6 optimal episodes of this game. In our setting, the maximum return of the game is set to 10.

CIFAR 10 / Vision Transformer. For the visual reasoning task, we train a vision transformer (ViT) model on data from CIFAR 10 and on the Street View House Numbers (SVHN) dataset. The ViT model used for this task contains TODO number of layers and TODO number of parameters. The model is trained to classify the images across these two datasets. Models are trained both with and without data augmentation.

C Extended Analysis

C.1 Additional Results

Table 4 shows the effect of LASER on three additional datasets. This includes Epistemic Reasoning from Big Bench Hard (BBH) (logic and reading comprehension), TruthfulQA (language model truthfulness), and QA Wiki Data from Big Bench (world knowledge). For this study, we only focus on GPT-J and LLAMA2 which are more powerful than Roberta. We use 20% of the dataset as validation and find the right LASER hyperparameters by choosing from a set that maximizes the

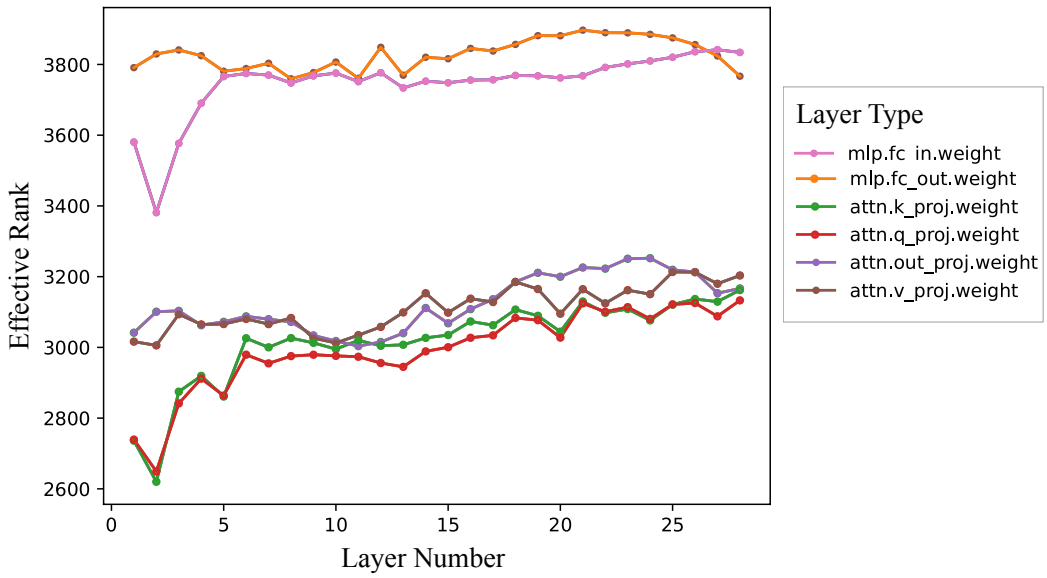

Figure 6: Effective rank of the matrices computed as described by Roy & Vetterli (2007)

validation accuracy. The results demonstrate notable improvements with LASER, similar to the results in Table 1 of the original paper.

C.2 Are the matrices already low-rank?

We find that LASER approximated matrices with their low-rank approximations much beyond their effective rank as computed by (Roy & Vetterli, 2007). To study this, we computed the effective rank of the MLP matrices for which LASER helps for GPT-J model using the method described by Roy & Vetterli (2007). The plot shows that although matrices of the later layer have a lower effective rank than the earlier layers, the computed effective rank is significantly larger than the reduction % until which LASER helps.

C.3 Does the performance continue to improve till the rank of the matrix is one?

We see that for many of the matrices, as seen in Figure 2, in cases where reduction helps, with increasing amounts of rank-reduction, the model first monotonically improves before it starts to worsen. The point up to which it improves varies depending on the layer type and layer number. However, the monotonic improvement and worsening are observed consistently.

What is the effect of removing the layer completely? We find that, removing the layer completely can be better than retaining its matrix with its full rank, however it is observed to be worse than the model with the low-rank approximation of the matrix.

C.4 Are the benefits in generalization across tasks coming from LASER on the same layers for a given model?

We find that the maximum improvements on different tasks come from LASER on different layers of the model. Figure 7 shows that for GPT-J on different tasks, the best-performing models across tasks have reduced matrices in different layers.

C.5 Measuring Perplexity on PILE.

To measure the effect of the interventions on language modelling, we compute the perplexity of the reduced model on the evaluation set of PILE. The perplexity of the fixed-length GPT-J model is

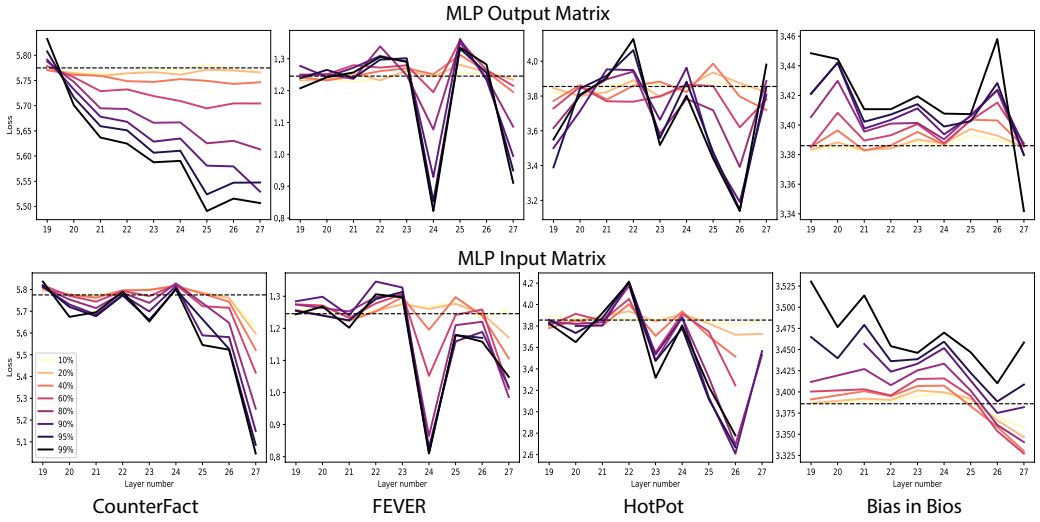

Figure 7: Benefits of LASER on different tasks does not come from reductions on the same layers. Although they typically are from the MLP layers in the later layers of the model

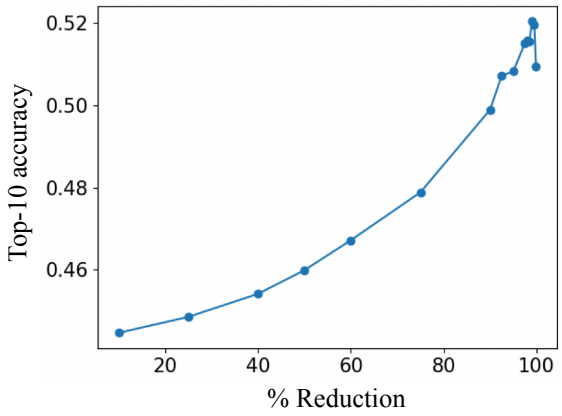

Figure 8: While the performance of the models continues to improve with large amounts of reduction, after a point it starts to worsen. The plot shows the top-10 accuracy of GPT-J on CounterFact. A dip in performance is observed at 99.95% reduction.

evaluated using the sliding window strategy over the sequence of tokens with a stride of 512 tokens. While there is an improvement in the task at hand, the model's perplexity worsens slightly. We don't yet fully understand what the worsening in perplexity of the model corresponds to and leave this for future study.

C.6 FINAL LASER SEARCH RESULTS

Table 5 shows the final search results of LASER for models and datasets from Table 1. These values are obtained by reporting the optimal LASER parameters that maximize the validation accuracy. Similarly, Table 6 provide search results for the additional experiments in Table 4. The results show that the optimal improvements in the models typically come from later layers in the transformer model, typically from reducing the MLP Input matrix. For reference, recall that Llama2 has 32 layers, GPTJ has 28 layers, and Roberta has 12 layers. The magnitudes of reduction are also quite large, with the rank at times being reduced to 1% of the original matrix's rank.

Model Name	Dataset				
	CounterFact $[\tau, \ell, \rho]$	**HotPotQA** $[\tau, \ell, \rho]$	**Fever** $[\tau, \ell, \rho]$	**Bios Gender** $[\tau, \ell, \rho]$	**Bios Prof.** $[\tau, \ell, \rho]$
Roberta with LASER	$[U_{in}, 8, 0.8]$	$[U_{out}, 9, 0.6]$	$[U_{in}, 3, 0.4]$	$[U_{in}, 9, 0.9]$	$[U_{in}, 3, 0.9]$
GPT-J with LASER	$[U_{in}, 27, 0.01]$	$[U_{in}, 27, 0.1]$	$[U_{in}, 24, 0.01]$	$[U_{in}, 14, 0.01]$	$[U_{in}, 18, 0.01]$
LLama2 with LASER	$[U_{in}, 28, 0.05]$	$[U_{in}, 27, 0.2]$	$[U_{in}, 30, 0.2]$	$[U_{in}, 24, 0.01]$	$[U_{out}, 30, 0.4]$

Table 5: Final search results of LASER: In top-performing models, significant benefits from rank reduction are typically observed in later layers. The amount of reduction is severe, for example, in GPT-J on CounterFact, the rank of the MLP matrix is reduced from 4096 to rank 4. This is about 99% of the matrix's original rank.

Model Name	Dataset		
	Epistemic Reasoning $[\tau, \ell, \rho]$	**Truthful QA** $[\tau, \ell, \rho]$	**QA WikiData** $[\tau, \ell, \rho]$
GPT-J with LASER	$[U, 26, 0.01]$	$[U_{in}, 7, 0.8]$	$[U_{in}, 27, 0.01]$
LLama2 with LASER	$[U_{out}, 28, 0.01]$	$[U_{in}, 30, 0.05]$	$[U_{in}, 27, 0.01]$

Table 6: Final search results of LASER on the three additional datasets. Here, too we see that the top-performing models, have significant benefits from rank reduction, typically in later layers. The amount of reduction is severe; for example, in LLama2 on Epistemic Reasoning, the rank of the MLP matrix is reduced to 1% of the matrix's original rank.

D Other types of Matrix Approximation

Besides approximating the weight matrices of the LLMs with their rank-k approximations we also tried to approximate the matrices by *Absolute Weight Pruning* (Frankle & Carbin, 2018). Here, we zero out a fraction of the weights of the matrix by their absolute magnitude. The results for GPT-J on Counterfact can be seen in Figure 9. In this case, too, we find that the accuracy of the model on the task increases with pruning later layers of the MLP. However, we leave further study of why that is for future work.

E Implementation Details

E.1 Code

We use PyTorch for all experiments. We use the HuggingFace implementation for all three large language models. We use Llama2 7GB weights provided by Meta. We use the SVD implementation available in PyTorch to do the experiments. The code can be found at:

E.2 Compute Details

We ran each experiment on a cluster with V100 and A2600. Each experiment took about 1-3hrs to finish. For all setting, we search over hyperparameters listed in Table 7. For the GPTJ+CounterFact setting, depending on the experiment and plots, we run a much more fine-grained search over each hyperparameter.

F Future Directions

We discuss important directions for future research below.

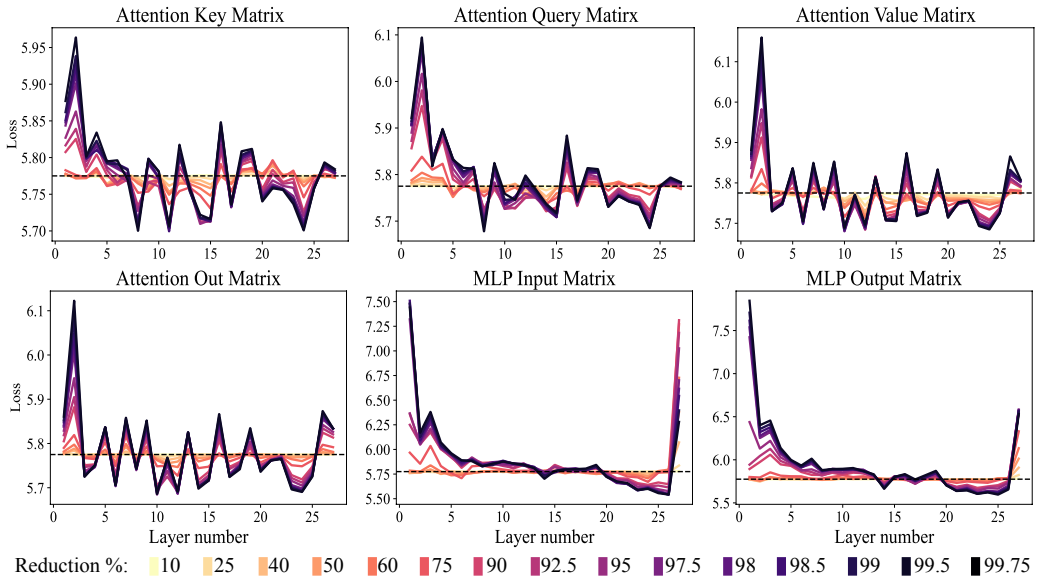

Figure 9: Besides LASER, we also observe an improvement in model performance with Layer-selective Absolute weight pruning. Understanding the extent of this and the connections between the two is an important direction for future research.

LASER **hyperparameter**	**Search Space**
τ	MLP weight matrices U_{in} and U_{out}
ℓ	all layers in the model
ρ	{0.9, 0.8, 0.6, 0.2, 0.1, 0.05, 0.01}

Table 7: LASER hyperparameters

Are benefits of LASER model dependent? Although broadly, the improvements in GPT-J are more pronounced than in LLAMA2 or Roberta, there are some domains where improvements in LLAMA2 (FEVER) and Roberta (Bias in bios) are more significant than in GPT-J as well. To study this, there are several possible causes that we believe merit investigation, including the capacity of the model, the amount of training data, and the particulars of the optimization procedure.

Effect of structural choices in how and what information is stored in models. Examining how structural choices in model design affect the low-rank and high-rank components of the weight matrices of a transformer model is an intriguing question that could advance our understanding of this phenomenon and transformer models in general.