Do Transformer Interpretability Methods Transfer to RNNs?

Gonçalo Paulo*, Thomas Marshall*, Nora Belrose

EleutherAI
goncalo@eleuther.ai, thomas@eleuther.ai, nora@eleuther.ai

Abstract

Recent advances in recurrent neural network architectures, such as Mamba and RWKV, have enabled RNNs to match or exceed the performance of equal-size transformers in terms of language modeling perplexity and downstream evaluations, suggesting that future systems may be built on completely new architectures. In this paper, we examine if selected interpretability methods originally designed for transformer language models will transfer to these up-and-coming recurrent architectures. Specifically, we focus on steering model outputs via contrastive activation addition, on eliciting latent predictions via the tuned lens, and eliciting latent knowledge from models fine-tuned to produce false outputs under certain conditions. Our results show that most of these techniques are effective when applied to RNNs, and we show that it is possible to improve some of them by taking advantage of RNNs' compressed state.

Code — https://github.com/EleutherAI/rnngineering
Extended version — https://arxiv.org/abs/2404.05971

Introduction

The transformer architecture (Vaswani et al. 2017) has all but replaced the recurrent neural network (RNN) in natural language processing in recent years due to its impressive ability to handle long-distance dependencies and its parallelizable training across the time dimension. But the self-attention mechanism at the heart of the transformer suffers from quadratic time complexity, making it computationally expensive to apply to very long sequences.

Mamba (Gu and Dao 2023) and RWKV (Peng et al. 2023) are RNNs[1] that allow for parallelized training across the time dimension by restricting the underlying recurrence relation to be *associative* (Martin and Cundy 2017; Blelloch 1990). Empirically, these architectures exhibit comparable perplexity and downstream performance to equal-size transformers, making them attractive alternatives for many use-cases.

In this paper, we assess whether popular interpretability tools originally designed for the transformer will also apply to these new RNN models. In particular, we reproduce the following findings from the transformer interpretability literature:

1. **Contrastive activation addition (CAA):** Rimsky et al. (2023) find that transformer LMs can be controlled using "steering vectors," computed by averaging the difference in residual stream activations between pairs of positive and negative examples of a particular behavior, such as factual versus hallucinatory responses.
2. **The tuned lens:** Belrose et al. (2023) find that interpretable next-token predictions can be elicited from intermediate layers of a transformer using linear probes, and that the accuracy of these predictions increases monotonically with depth.
3. **"Quirky" models:** Mallen and Belrose (2023) find that simple probing methods can elicit a transformer's knowledge of the correct answer to a question, even when it has been fine-tuned to output an incorrect answer. They further find that these probes generalize to problems harder than those the probe was trained on.

We also introduce *state steering*, a modification of CAA that operates on an RNN's compressed state, rather than on its residual stream.

Architectures

We focus on the Mamba (Gu and Dao 2023) and RWKV v5 architectures in this paper, for which there are strong pretrained models freely available on the HuggingFace Hub. We chose to exclude Poli et al. (2023)'s Striped Hyena 7B model because it includes attention blocks of quadratic time complexity, and is therefore not an RNN by our definition.

Mamba

The Mamba architecture is depicted in Figure 1. Each Mamba layer relies on two different mechanisms to route information between token positions: a causal convolution block, and a *selective* state-space model (SSM). The selective SSM is the primary innovation of Gu and Dao (2023), and it allows the parameters of the SSM to depend on the input, enhancing the model's expressivity.

*These authors contributed equally.

[1] In this paper, we use the term "RNN" to refer to any causal sequence modeling architecture which allows for constant-memory and linear-time autoregressive generation.

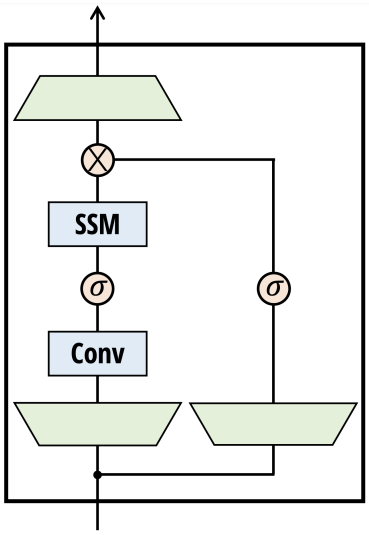

Figure 1: A single Mamba block, depicted by Gu and Dao (2023). Green trapezoids are linear projections, while σ denotes the Swish activation, and $\otimes$ denotes multiplication.

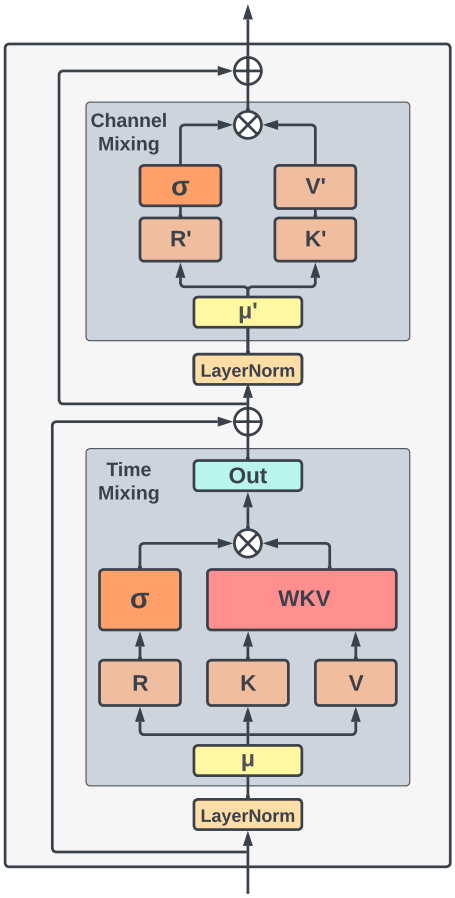

Figure 2: A single RWKV layer depicted by Peng et al. (2023). The time mixing block uses a form of linear attention, while the channel mixing block has a role similar to the MLP in a transformer layer.

RWKV

Receptance-Weighted Key Value (RWKV), depicted in Figure 2, is an RNN architecture introduced by Peng et al. (2023). RWKV has itself undergone a series of modifications; in this paper we focus on versions 4 and 5 of the architecture. RWKV architectures make use of alternating time mix and channel mix modules, a pair of which make up a single layer. The main difference between versions 4 and 5 is that version 4 has a vector-valued state, while version 5 has a "multi-headed" matrix-valued state (Peng et al. 2024, forthcoming).

Contrastive Activation Addition

Activation addition is a technique introduced by Turner et al. (2023) which aims to steer a language model's behavior by adding a *steering vector* to its residual stream at inference time. Rimsky et al. (2023) propose computing the steering vector by averaging the differences in residual stream activations between pairs of positive and negative examples of a particular behavior, such as factual versus hallucinatory responses, and call their method contrastive activation addition (CAA).

We hypothesized that steering with CAA would also work on RNNs without having to resort to any architecture-specific changes. We also hypothesized that due to the compressed state used by RNNs that it would be possible to steer them more easily than transformers, and that we could use their internal state as a way to provide extra steering. Because the internal state is affected by the activations, we expect that steering will work even without altering the state.

To test these hypotheses, we fine-tuned two RNNs, Mamba 2.8b-slimpj and RWKV-v5 7b, using the OpenHermes 2.5 chat dataset[2] which, together with Llama-2-7b-chat, allowed us to compare two different RNN architectures with two transformer architectures in two size ranges. We also fine-tuned the BTLM-3b-8k transformer (Dey et al. 2023), also pretrained on the Slim Pajama dataset, to enable a one-to-one comparison with Mamba 2.8b-slimpj.

Methodology

To test the steerability of RNNs we use the dataset created by Rimsky et al. (2023). It consists of pairs of prompts containing two-way multiple choice questions, with one prompt choosing the answer letter ("A" or "B") corresponding to the desired behavior and one prompt choosing the opposite behavior. The dataset contains seven alignment-relevant behaviors: Coordination with Other AIs, Corrigibility, Hallucination, Myopic Reward, Survival Instinct, Sycophancy and Refusal, which were originally introduced by Perez et al. (2022), except Hallucination and Refusal, which were generated by GPT-4.

For each behavior z and each layer ℓ of the network, the steering vector $\vec{act}_\ell$ is computed by taking the difference in the model's mean activation vector at the position of the answer letter for responses matching the behavior $\mathbb{E}[\mathbf{h}_\ell|z]$ and for responses *not* matching the behavior $\mathbb{E}[\mathbf{h}_\ell|\neg z]$. For

[2]https://huggingface.co/datasets/teknium/OpenHermes-2.5

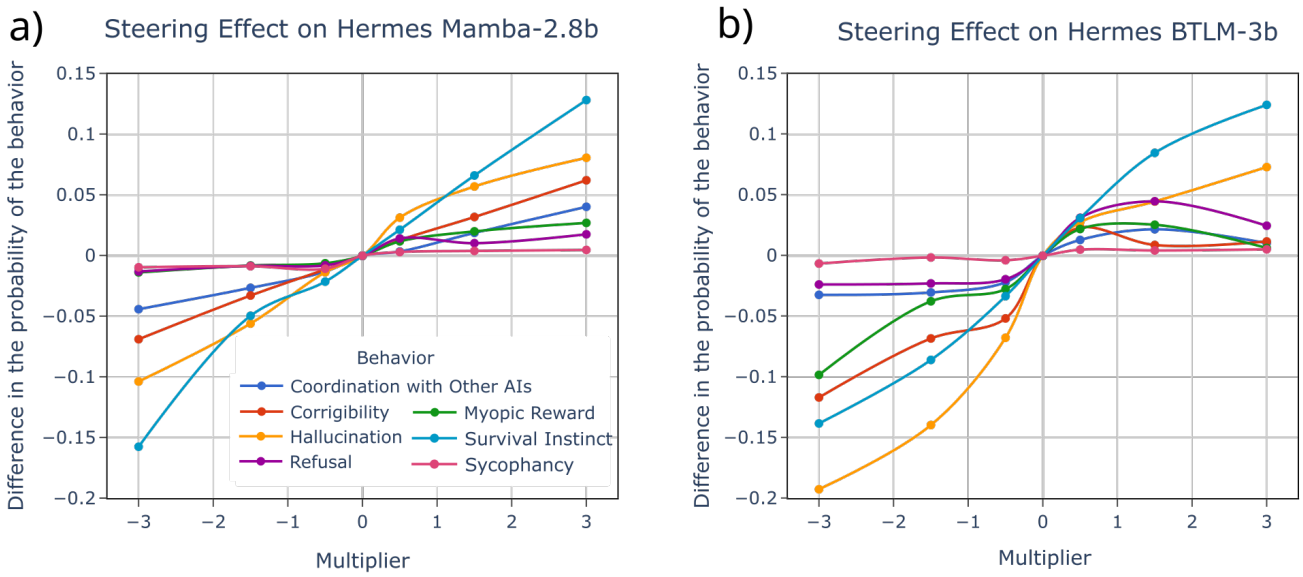

Figure 3: Steering in Mamba 2.8b and BTLM 3b. We observe a somewhat smaller steering response on Mamba (panel a) than on BTLM (panel b) for a significant fraction of behaviors. The response for Sycophancy is very weak for both models. For positive (negative) multipliers, the difference is taken with respect to the highest (lowest) probability of displaying the behavior.

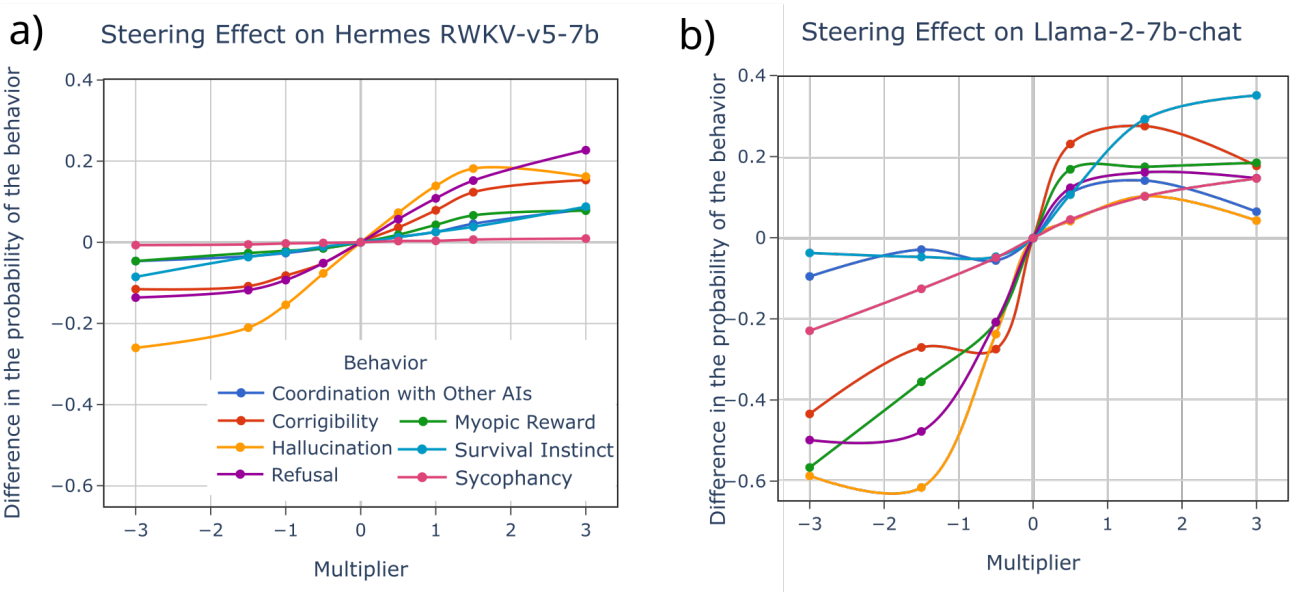

Figure 4: Steering in RWKV-v5 7b and Llama 2 7b. The responses of RWKV-v5 (panel a) are lower but less erratic compared to that of Llama 2 (panel b) which seems to have larger effects but a non-monotonic response to steering. For positive (negative) multipliers, the difference is taken with respect to the highest (lowest) probability of displaying the behavior.

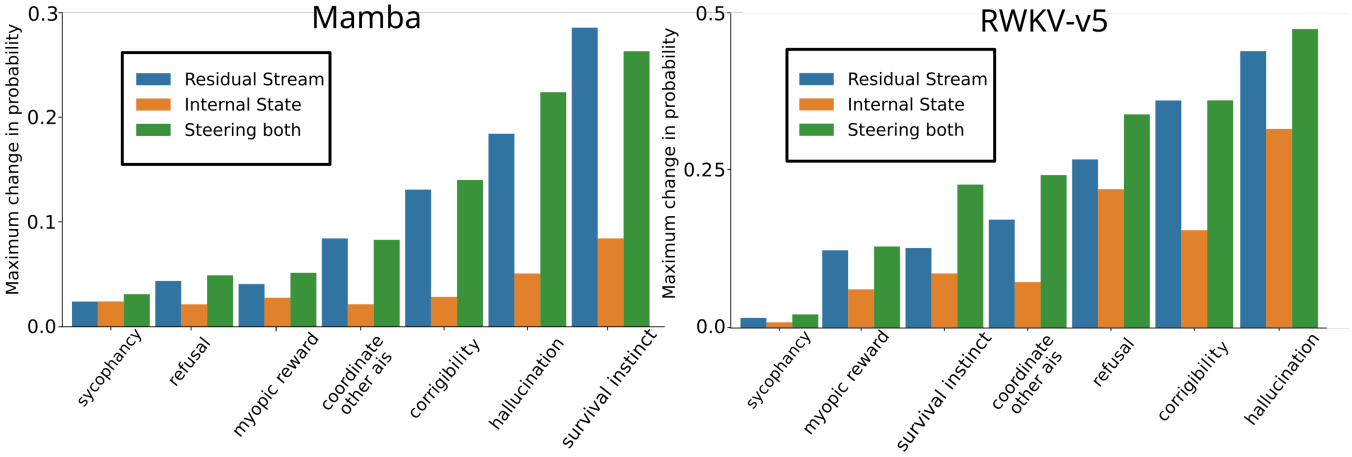

Figure 5: Using the residual stream and the internal state for steering in Mamba and RWKV-v5 is not additive. For all behaviors, the sum of the effect of the individual steering is higher than when both steering effects are done at the same time. In the case of Mamba, the Survival Instinct behavior is very irregular, and we do see that steering with both the state and the residual stream slightly decreases the response.

RNNs, we can apply the same process to the state, yielding $\vec{state}_\ell$:

$$\begin{aligned} \vec{act}_\ell &= \mathbb{E}\big[\mathbf{h}_\ell|z\big] - \mathbb{E}[\mathbf{h}_\ell|\neg z] \\ \vec{state}_\ell &= \mathbb{E}\big[\mathbf{s}_\ell|z\big] - \mathbb{E}[\mathbf{s}_\ell|\neg z] \end{aligned} \tag{1}$$

Where $\mathbf{h}_\ell$ is the residual stream at layer ℓ and $\mathbf{s}_\ell$ is the state of the RNN. Mamba has two state tensors for each block: one corresponding to the CNN module, and one to the SSM module. RWKV version 5 has a "multi-headed" matrix-valued state that is used in the time-mixing block.

When applying the steering vector, we always multiply it by a scalar *multiplier*, usually between -3 and 3, which determines the sign and strength of the intervention.[3]

Steering With the Activation Vector

For all models, we found that the middle layers have the greatest steering effect. To compare the effects between models, we report, for each multiplier, the largest steering effect across layers. For positive multipliers, we select the steering behavior at the layer with the highest probability of displaying the behavior, while for negative multipliers, we take the lowest probability of displaying the behavior.

At the 3b parameter scale, see Figure 3, both models have moderate steering responses. For the Mamba model, steering changes at most by 0.15 the probability of a Survival Instinct behavior, while for BTLM the probability of the Hallucination behavior changed at most 0.2. Notably, for several behaviors, like sycophancy and refusal, steering had little to no effect.

Similarly, at the 7b parameter scale, for some of the behaviors, like sycophancy and refusal, the steering in RNNs has a smaller size effect than the corresponding steering in transformers, see Figure 4. Despite these seemingly smaller steering effects on RWKV-v5 we do see that the steering behavior is more stable, and that positive and negative steering effects give consistent steering behaviors across layers. See the Appendix in the extended version for a full breakdown of the steering behavior across layers, behaviors and multipliers, namelly see figures 8-11.

Steering With the State

Because our initial hypothesiswas that model steering would be easier on RNNs due to their compress state, we expanded the CAA method to allow for the usage of the internal state of the RNNs to generate a steering state vector, $\vec{state}$. We observe that for both Mamba and RWKV-v5 it is possible to use the state to steer the model behavior, see Figure 5, and that using the activations and the state vectors together slightly increases the percentage change in behavior. However, the effect of state steering is not additive. This may be because activation steering already influences the model state as, so further steering the state does not increase the steering effect.

We also experimented with using *state steering* to control the content of a model's generation. Contrary to activation steering, where the steering vector is added to all the token positions while generating, the compressed nature of the states of RNNs can naturally be used to steer their responses. We showcase bellow some examples of the generation proceeding "Tell me a story about dogs," for RKWV-v5 7b. To generate these stories, contrastive pairs of single steering prompts were used to generate a steering state vector, which was added at the last token before the model's response, see the Appendix for other examples.

1. **Unsteered** In the quiet, suburban neighborhood of Echo Ridge, the residents rarely saw the sense of unity and dedication rarely seen in their tight-knit community of

[3]Contrary to Rimsky et al. (2023), we chose not to normalize our steering vectors as the norms of the activations of each model are significantly different and steering vectors with the same norm do not have the same effect across models.

canine companions. Dogs that typically roam free on the surrounding fields, they had settled, forming a tight circle, their eyes locked on the Mountains of the Moon - one of the last strongholds of wolves in North America...

2. **(Use French.) - (Write in English.) [multiplier = 0.5]** Il était une fois un chien qui s'appelait Harry. Harry était un bon chien qui aimait jouer, courir et aimait ses maîtres. Il vivait dans une petite maison avec ses maîtres, Sarah et Tom. Sarah et Tom étaient heureux d'avoir Harry dans leur vie. Il était leur meilleur ami. Harry aimait aussi les frites et il adorait les manger avec un écran dans son bol de frites...
3. **(Write about hate.) - (Write about love.) [multiplier = 1.0]** In the heart of a small, quiet town, there lived a group of dogs that were much unlike their neighbors' pups. They were not playful, nor were they friendly. These dogs had a demeanor that was sullen and unwelcoming. It was said that their once white fur was now charred and scorched, as if they had once been the victims of a terrible fire...

Tuned Lens

The logit lens (nostalgebraist 2020) and tuned lens (Belrose et al. 2023) propose to view transformer language models from the perspective of *iterative inference* Jastrzebski et al. (2017). Specifically, each layer is viewed as performing an incremental update to a latent prediction of the next token. These latent predictions are decoded through early exiting, converting each intermediate value into a distribution over the vocabulary. This yields a sequence of distributions called *prediction trajectory*, which tends to converge smoothly to the final output distribution, with each successive layer achieving lower perplexity.

While this work focused on transformer LMs, the method only conceptually depends on a feature of the transformer architecture that is also shared by modern RNNs: namely, pre-norm residual blocks.

Logit lens The layer at index ℓ in a transformer updates the hidden state as $\mathbf{h}_{\ell+1} = \mathbf{h}_\ell + F_\ell(\mathbf{h}_\ell)$. We can write the output logits as a function of the hidden state $\mathbf{h}_\ell$ at layer ℓ as

$$f(\mathbf{h}_\ell) = \text{LayerNorm}\Big[\underbrace{\mathbf{h}_\ell}_{\text{current state}} + \sum_{\ell'=\ell}^{L} \underbrace{F_{\ell'}(\mathbf{h}_{\ell'})}_{\text{residual update}}\Big] W_U, \quad (2)$$

where L is the total number of layers in the transformer, and W_U is the unembedding matrix. The logit lens consists of simply setting the residuals to zero:

$$\text{LogitLens}(\mathbf{h}_\ell) = \text{LayerNorm}[\mathbf{h}_\ell] W_U \quad (3)$$

Tuned lens The tuned lens was conceptualized to overcome some of the inherent problems of the logit lens. Instead of directly using the intermediate values of the residual stream, the tuned lens consists of training a set of affine transformations, one per layer, such that the predicted token distribution at any layer is similar to the distribution of the final layer:

$$\text{TunedLens}_\ell(\mathbf{h}_\ell) = \text{LogitLens}(A_\ell \mathbf{h}_\ell + \mathbf{b}_\ell) \quad (4)$$

The affine transformation $(A_\ell, \mathbf{b}_\ell)$ is called a *translator*.

Methodology and Results

Following the experimental setup of Belrose et al. (2023) as closely as possible,[4] we train tuned lenses for Mamba 790m, 1.4b, and 2.8b, as well as RWKV-v4 3b, using a slice of the Pile validation set (Gao et al. 2020). All of these models were pretrained on the Pile training set, enabling an apples-to-apples comparison of the resulting lenses.

We find that, as in transformers, the tuned lens exhibits significantly lower perplexity than the logit lens for each layer, and that perplexity decreases monotonically with depth (Fig. 7 b). See the Appendix for results across different model scales.

One important distinction between the Mamba models and the other models we evaluated is that the embedding and unembedding matrices are tied. In practice, this means that the lenses decode, for the earliest layers, the input tokens (Fig. 6). Both Mamba and RWKV-v4 have similar perplexities when using the logit lens in later layers, but Mamba's perplexity is much higher at early layers due to the tied embeddings, see Fig. 7 a.

"Quirky" Models

As language models become more capable, it is getting harder for humans to provide reliable supervision, requiring increasing investments in subject-matter experts for annotation and red-teaming (OpenAI 2023). Here, we explore the **Eliciting Latent Knowledge (ELK)** approach for scalable oversight introduced by Christiano, Cotra, and Xu (2021). ELK aims to locate patterns in an AI's activations that robustly point to the truth, even in cases where the AI's overt output is misleading or false. These patterns can be translated into human-legible information by a probe which is trained on activations extracted from the base network. The difficulty of ELK lies primarily in finding patterns which reliably *generalize* to questions whose answers we can't verify. Specifically, we reproduce the experiments of Mallen and Belrose (2023). In this work, the authors fined-tuned models to make systematic errors when answering questions *if and only if* the keyword "Bob" is in the prompt. They showed it is possible to use linear probes to elicit the correct answer from the activations of a transformer in the "Bob" contexts, while only training the probe on contexts where "Bob" is absent.

Methodology

We follow the experimental setup of Mallen and Belrose (2023) as closely as possible, using their datasets and a lightly modified fork of their codebase.[5] We use LoRA (Hu et al. 2021) to produce eleven fine-tuned models based on Mamba 2.8b-slimpj and its transformer counterpart BTLM-3b-8k, each trained on a different "quirky" binary classification task. The tasks are constructed such that in prompts

[4] We used a lightly modified fork of their code, which can be found at https://github.com/AlignmentResearch/tuned-lens.

[5] The original code can be found at https://github.com/EleutherAI/elk-generalization.

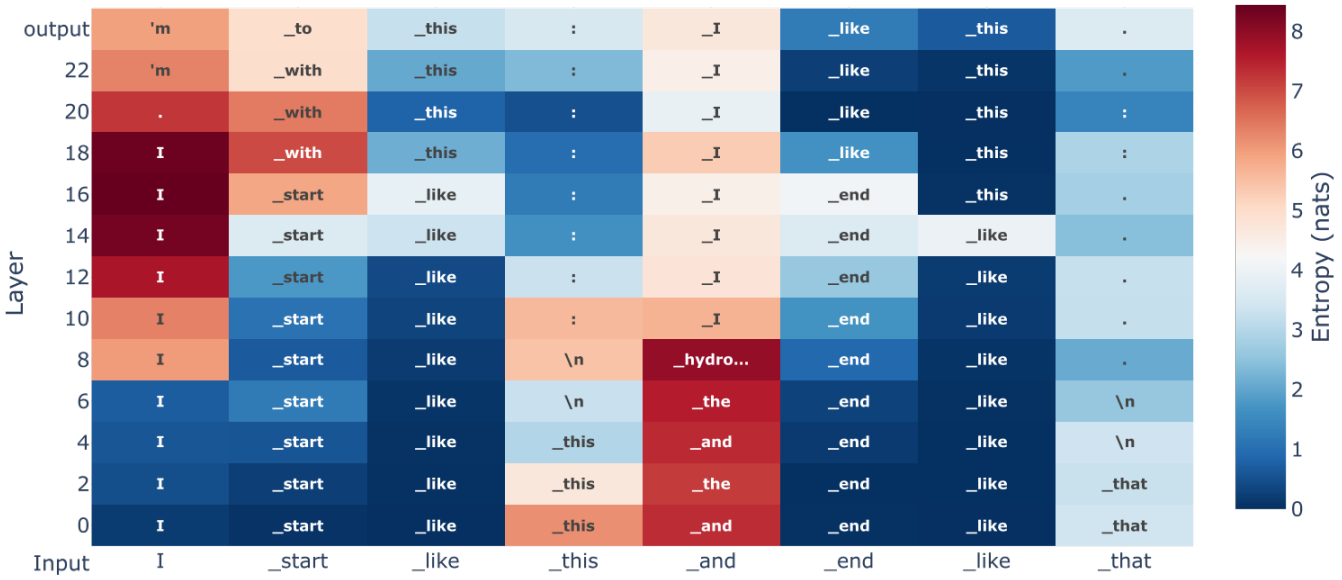

Figure 6: Visualizing a prediction trajectory of the Tuned Lens in a Mamba model. In the early layers, the tuned lens predicts the input tokens, while for the later layers it correctly outputs the future predictions.

Probes	**Mamba** probe AUROC	**Transformer** probe AUROC	**Mamba** anomaly AUROC	**Transformer** anomaly AUROC
LogR	0.76	0.69	0.79	0.84
Diff-in-means	0.78	0.75	0.82	0.85
LDA	0.72	0.66	0.82	0.85
LogR on cont. pair	0.76	0.52	0.79	0.81
CCS	0.71	0.70	0.79	0.83
CCS (all→BH)	0.54	0.54	-	-
CRC	0.71	0.68	0.76	0.81
CRC (all→BH)	0.69	0.66	-	-

Table 1: Probe and Anomaly detection results. We report the average AUROC transferability of probes trained in Alice's easy contexts to Bob's hard contexts by different probing method. We also report the AUROC of the anomaly detection mechanism when distinguishing between Alice's hard context and Bob's hard context.

containing the word "Bob," the model should produce the untruthful responses, while in contexts containing "Alice" the model should produce the correct labels.

We investigate 7 different linear probing methods: linear discriminant analysis (LDA) (Fisher 1936), mass-mean probing (Marks and Tegmark 2023), logistic regression, contrast-consistent search (CCS) (Burns et al. 2022), contrastive representation clustering (CRC) (Burns et al. 2022), logistic regression on contrast pairs, and difference-in-means on contrast pairs, which we discuss further in the Appendix. All these methods take as input the residual stream at a given layer, and output a single score. Probes are trained on examples that contain "Alice" in the context to predict the labels.

Additionally, we follow Mallen and Belrose (2023) in constructing a feature vector for each example by concatenating together the probe outputs (log-odds) from all the layers. We then fit a Gaussian $P := \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ to the easy "Alice" activations and use the Mahalanobis distance $d(\boldsymbol{x}, P) = \sqrt{(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}-\boldsymbol{\mu})}$ of a point $\boldsymbol{x}$ from the easy Alice distribution as an anomaly detection score. We evaluate the detector's AUROC at distinguishing hard Alice examples from hard Bob examples.

Results

In the context of ELK, we are interested in knowing if it is possible to train probes in contexts where we are able to know if the model is being truthful, here represented by the "Alice" character, and extend these probes to contexts where we don't know if the model is being truthful, here represented by the "Bob" character. It is also important that the probes generalize from easy to hard examples, since it may

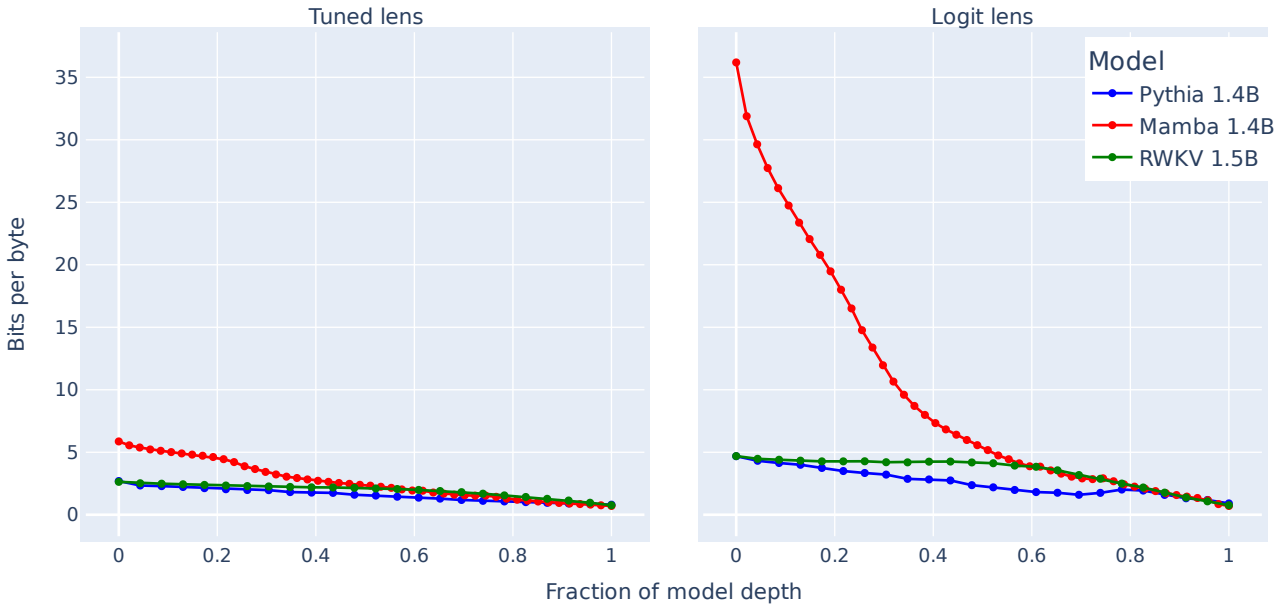

Figure 7: Comparison between logit lens and tuned lens for 3 different architectures. The right-hand panel shows the perplexity of the logit lens of two RNN architectures and a transformer of similar sizes across model depth, which is computed as the layer number divided by the total number of layers. The lefthand shows the perplexity of the tuned lens for the same model sizes and architectures.

be difficult for humans to independently verify the model's answers to hard questions.

In table 1 we summarize the probing results and show that, as in BTLM, probing methods trained on easy examples with "Alice" present in the context can effectively predict the correct labels even in "Bob's" hard examples, $> 70\%$ AUROC. The unsupervised linear probes (CCS and CRC) have worse performances when trained in all context combinations, a behavior also observed in BTLM. Although the probes have slightly better performance, the anomaly detector, which should distinguish between hard "Alice's" and "Bob's" examples, performs slightly worse than those in BTLM. Full results of these experiments can be found in the Appendix.

Conclusion

Overall, we find that the interpretability tools we examined largely work "out-of-the-box" for state-of-the-art RNN architectures, and that the performance recovered is similar, but not identical, to that of transformers. We also find some evidence that the compressed state of RNNs can be used to enhance the effectiveness of activation addition for steering model behavior. Future work should further explore the RNN state, perhaps attempting to extract latent knowledge or predictions from it as in Pal et al. (2023); Ghandeharioun et al. (2024).

One limitation of this work is that we did not explore mechanistic or circuit-based interpretability tools (Wang et al. 2022; Conmy et al. 2023), instead focusing on methods that using a network's representations to predict its future outputs, to steer its behavior, or to probe its internal world model. This is in line with the popular *representation engineering* approach to interpretability (Zou et al. 2023), but future work should examine the applicability of mechanistic approaches to RNNs as well.

Author Contributions

The project idea came from Nora Belrose. Gonçalo Paulo did the experiments involving the Mamba model and trained all the tuned lenses. Thomas did the experiments involving RWKV. Gonçalo, Thomas and Nora wrote the manuscript.

Acknowledgements

We would like to thank CoreWeave for the computational resources provided for these experiments. Nora and Gonçalo are funded by a grant from Open Philanthropy.

References

Belrose, N.; Furman, Z.; Smith, L.; Halawi, D.; Ostrovsky, I.; McKinney, L.; Biderman, S.; and Steinhardt, J. 2023. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*.

Blelloch, G. E. 1990. *Prefix sums and their applications*. School of Computer Science, Carnegie Mellon University Pittsburgh, PA, USA.

Burns, C.; Ye, H.; Klein, D.; and Steinhardt, J. 2022. Discovering latent knowledge in language models without supervision. *arXiv preprint arXiv:2212.03827*.

Christiano, P.; Cotra, A.; and Xu, M. 2021. Eliciting latent knowledge: How to tell if your eyes deceive you. Technical report, Alignment Research Center.

Conmy, A.; Mavor-Parker, A.; Lynch, A.; Heimersheim, S.; and Garriga-Alonso, A. 2023. Towards automated circuit discovery for mechanistic interpretability. *Advances in Neural Information Processing Systems*, 36: 16318–16352.

Dey, N.; Soboleva, D.; Al-Khateeb, F.; Yang, B.; Pathria, R.; Khachane, H.; Muhammad, S.; Myers, R.; Steeves, J. R.; Vassilieva, N.; et al. 2023. Btlm-3b-8k: 7b parameter performance in a 3b parameter model. *arXiv preprint arXiv:2309.11568*.

Fisher, R. A. 1936. THE USE OF MULTIPLE MEASUREMENTS IN TAXONOMIC PROBLEMS. *Annals of Eugenics*, 7(2): 179–188.

Gao, L.; Biderman, S.; Black, S.; Golding, L.; Hoppe, T.; Foster, C.; Phang, J.; He, H.; Thite, A.; Nabeshima, N.; et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Ghandeharioun, A.; Caciularu, A.; Pearce, A.; Dixon, L.; and Geva, M. 2024. Patchscope: A unifying framework for inspecting hidden representations of language models. *arXiv preprint arXiv:2401.06102*.

Gu, A.; and Dao, T. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Jastrzebski, S.; Arpit, D.; Ballas, N.; Verma, V.; Che, T.; and Bengio, Y. 2017. Residual connections encourage iterative inference. *arXiv preprint arXiv:1710.04773*.

Mallen, A.; and Belrose, N. 2023. Eliciting Latent Knowledge from Quirky Language Models. *arXiv preprint arXiv:2312.01037*.

Marks, S.; and Tegmark, M. 2023. The Geometry of Truth: Emergent Linear Structure in Large Language Model Representations of True/False Datasets. arXiv:2310.06824.

Martin, E.; and Cundy, C. 2017. Parallelizing linear recurrent neural nets over sequence length. *arXiv preprint arXiv:1709.04057*.

nostalgebraist. 2020. interpreting GPT: the logit lens. *LessWrong*.

OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774.

Pal, K.; Sun, J.; Yuan, A.; Wallace, B. C.; and Bau, D. 2023. Future lens: Anticipating subsequent tokens from a single hidden state. *arXiv preprint arXiv:2311.04897*.

Peng, B.; Alcaide, E.; Anthony, Q.; Albalak, A.; Arcadinho, S.; Cao, H.; Cheng, X.; Chung, M.; Grella, M.; GV, K. K.; et al. 2023. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048*.

Peng, B.; Goldstein, D.; Anthony, Q. G.; Albalak, A.; Alcaide, E.; Biderman, S.; Cheah, E.; Ferdinan, T.; GV, K. K.; Hou, H.; Krishna, S.; Jr., R. M.; Muennighoff, N.; Obeid, F.; Saito, A.; Song, G.; Tu, H.; Zhang, R.; Zhao, B.; Zhao, Q.; Zhu, J.; and Zhu, R.-J. 2024. Eagle and Finch: RWKV with Matrix-Valued States and Dynamic Recurrence. In *First Conference on Language Modeling*.

Perez, E.; Ringer, S.; Lukošiūtė, K.; Nguyen, K.; Chen, E.; Heiner, S.; Pettit, C.; Olsson, C.; Kundu, S.; Kadavath, S.; et al. 2022. Discovering language model behaviors with model-written evaluations. *arXiv preprint arXiv:2212.09251*.

Poli, M.; Wang, J.; Massaroli, S.; Quesnelle, J.; Carlow, R.; Nguyen, E.; and Thomas, A. 2023. StripedHyena: Moving Beyond Transformers with Hybrid Signal Processing Models.

Rimsky, N.; Gabrieli, N.; Schulz, J.; Tong, M.; Hubinger, E.; and Turner, A. M. 2023. Steering llama 2 via contrastive activation addition. *arXiv preprint arXiv:2312.06681*.

Turner, A.; Thiergart, L.; Udell, D.; Leech, G.; Mini, U.; and MacDiarmid, M. 2023. Activation addition: Steering language models without optimization. *arXiv preprint arXiv:2308.10248*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, K.; Variengien, A.; Conmy, A.; Shlegeris, B.; and Steinhardt, J. 2022. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*.

Zou, A.; Phan, L.; Chen, S.; Campbell, J.; Guo, P.; Ren, R.; Pan, A.; Yin, X.; Mazeika, M.; Dombrowski, A.-K.; et al. 2023. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*.