per sample. An auxiliary loss $\mathcal{L}_{aux} = ||e - \hat{e}||^2$ is used to avoid dead latents, where $\hat{e} = W^{dec}z$ is the reconstruction using only the top-k_{aux} dead latents (usually 512), this loss is scaled by a small coefficient α (usually 1/32).

BatchTopK SAEs Bussmann et al. (2024) relaxes the top-k constraint of TopK SAEs to the batch level during training, i.e. retaining the top $b * k$ activations per batch where b is the size of the batch. This introduces a dependency between samples in the batch, so at test time a global threshold parameter θ, estimated on the training data, is used instead, with only activations above this threshold being retained.

$$\theta = \mathbb{E}_{\mathbf{X}}[\min\{z_{i,j}(\mathbf{X})|z_{i,j}(\mathbf{X}) > 0\}] \tag{8}$$

JumpReLU SAEs (Rajamanoharan et al., 2024b) replace the standard ReLU activation function with the JumpReLU activation, defined as

$$\text{JumpReLU}_\theta(z) := zH(z - \theta) \tag{9}$$

where H is the Heaviside step function, and θ is a learned parameter for each SAE latent, below which the activation is set to zero. JumpReLU SAEs are trained using a loss function that combines L2 reconstruction error with an L0 sparsity penalty, using straight-through estimators to train despite the discontinuous activation function. A major drawback of the sparsity penalty used in JumpReLU SAEs compared to (Batch)TopK SAEs is that it is not possible to set an explicit sparsity and targeting a specific sparsity involves costly hyperparameter tuning. While evaluating JumpReLU SAEs, Rajamanoharan et al. (2024b) chose the SAEs from their sweep that were closest to the desired sparsity level, but this resulted in SAEs with significantly different sparsity levels being directly compared. JumpReLU SAEs use no auxiliary loss function.

A.3 Example latents

Figure 7 shows a histogram of the maximum decoder cosine similarity for each latent in GPT2-1536 over all latents in GPT2-768. On the right-hand-side, there is a cluster of latents with high cosine similarity.

Bricken et al. (2023) use the cosine similarity between latent activations as a measure for latent similarity. However, we use decoder cosine similarity due its lower computational cost and because it captures the latent's effect on the reconstruction. Empirically, we find a high correlation between these two metrics at values of decoder similarity relevant to our stitching experiments (see Figure 8).

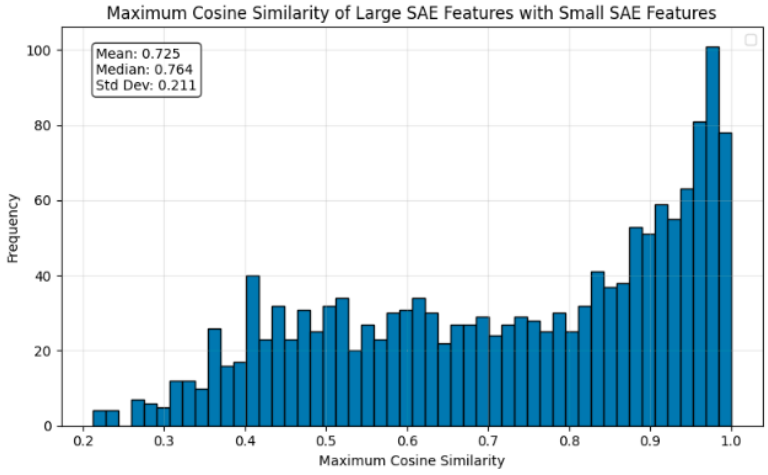

Figure 7: Distribution of maximum cosine similarities between decoder weights of latents in GPT2-1536 and GPT2-768. Many latents in the larger SAE have high similarity to latents in the smaller SAE, but there is also a long tail of novel latents.

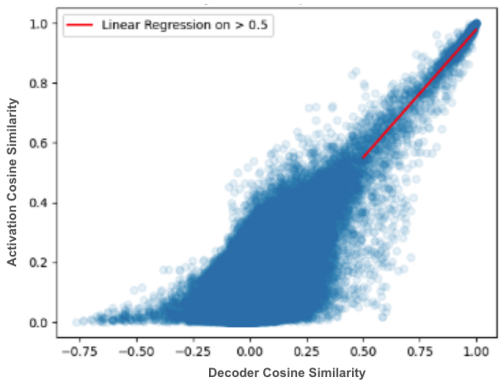

Figure 8: The activation similarity by the decoder cosine similarity on the GPT2-768 and GPT2-1536 SAEs. At high values of cosine similarity (> 0.5) the coefficient of determination between these is 0.87.

Figure 9 and Figure 10 show feature dashboards from Neuronpedia (Lin, 2023). Feature dashboards are a popular method to visualize and interpret SAE latents, see also Bricken et al. (2023). These dashboards show tokens in input sentences that maximally activate a certain latent in green, and the tokens they boost (in blue) and suppress most (in red).

Figure 9 shows an example of a latent from GPT-1536 and a latent from GPT-768 that have a cosine similarity of 0.99. We see that both of these latents activate strongly on the same inputs, and boost similar logits.

Figure 9: Neuronpedia dashboard of example latents with high cosine similarity. (Redacted URL)

However, GPT2-1536 has a latent for “make sure” that has no counterpart in GPT-768. The nearest latents have a decoder cosine similarity of around 0.3, and are shown in

Figure 10: Example GPT2-1536 latent with no similar latent in GPT-768, with the three most similar latents shown. (Redacted URL)

We evaluate the reconstruction performance of the two SAEs on inputs where this latent is active and inactive. The reconstruction performance of the smaller SAE is considerably worse on inputs where this larger SAE latent is active, compared to inputs where the latent is not active.

	Latent inactive	**Latent active**	**Difference**
GPT2-1536	2.225	2.518	0.293
GPT2-768	2.703	3.292	0.589

Averaging this metric across all 657 latents in GPT-1536 that have low maximum cosine similarity with all latents in GPT-768, we see a similar pattern (Figure 11)

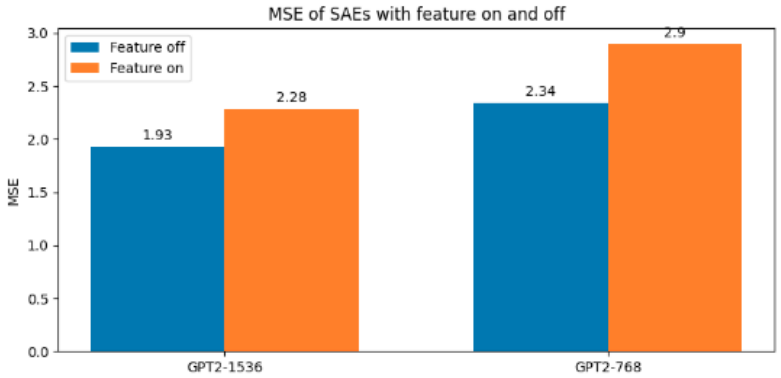

Figure 11: Reconstruction MSE of SAEs on inputs where novel latents in the larger SAE are active and inactive