

How does fine-tuning change internal representations in an audio transcription model?

Stanford CS224N Custom Project

Jason Hu

Department of Computer Science
Stanford University
hujason@stanford.edu

Jenny Jin

Department of Computer Science
Stanford University
yqjin@stanford.edu

Brandon Liu

Department of Computer Science
Stanford University
liubr@stanford.edu

Abstract

Although supervised fine-tuning of foundation models can lead the model to improve performance on domain-specific tasks, the underlying mechanistic changes in the model's internal representation remain poorly understood. We investigate the internal mechanisms of the Whisper audio transcription model which has been fine-tuned on air traffic control (ATC) audio. Specifically, we target the embeddings of an intermediate decoder layer to study how fine-tuning the Whisper model on ATC domain-specific data may shift the model's internal representations, and what representations are shifted. Using sparse autoencoders, we demonstrate that the fine-tuned model exhibits statistically and causally significant differences in this intermediate layers representation, and we can identify and steer clusters which correspond to ATC-specific semantic concepts.

- Mentor: Qinan Yu
- Late Days:

Team member	Late days remaining
Brandon Liu	6
Jason Hu	5
Jenny Jin	0
Total pooled	11
Team size	3
Late days applied for the report	3 days * 3 members = 9

1 Introduction

Foundation speech models such as OpenAI's Whisper model achieve state-of-the-art accuracy on general transcription tasks, but achieving similar performance on specialized, challenging domains requires supervised fine-tuning on in-domain data. One such domain is air traffic control (ATC), which has many specialized transcription features (e.g., heavy static, all capital letters, NATO alphabet, cardinal numbers). Fine-tuning Whisper on ATC data drastically reduces word error rate (WER) from >99% to 6.5% [1], but how the model's internal mechanisms adapt during the fine-tuning process is poorly understood. For example, it is unclear if the model learns biases in late-stage layers, or if it reorganizes its end-to-end mechanism to adapt to the ATC domain.

We investigate Whisper-Large-v3’s internal activations before and after fine-tuning. We target the activations of an intermediate decoder layer (layer 16 out of 32 total), to compare the features learned by the base and the fine-tuned models. We choose this layer because it is likely to contain interesting, abstract features [2]. We hypothesize that in this intermediate layer, the base and fine-tuned models exhibit significant differences in the distribution of interpretable activations that explain output differences. The fine-tuned model may produce more semantically similar representations for the set of tokens in the specialized domain. For instance, cardinal numbers are overrepresented in the ATC dataset, so representations of numbers may emerge more frequently after fine-tuning. ATC chatter is a highly specialized domain with a very different token distribution compared to general natural speech, which could introduce larger distributional differences.

Our experiments confirm our hypothesis that the decoder layer’s distribution tightens to focus on ATC terminology. Using a Maximum Mean Discrepancy (MMD) test on the sentence embeddings of LLM-judge generated feature labels, we find a statistically significant distributional shift between the base and fine-tuned models ($p = 0.0008$), and a significantly tighter clustering of concepts in the fine-tuned embedding space ($p = 0.019$). We validate our clustering labels from the extracted SAE features by conducting a dose-response steering experiment where we add the SAE residuals for the feature which corresponds to a particular semantic concept, and we show that we can successfully steer the model to emphasize the targeted feature.

2 Related Work

Scaling Sparse Autoencoders. Gao et al. (2025) [3] scales unsupervised interpretability methods to frontier models by training sparse autoencoders (SAEs) that de-superimpose embeddings in large language models to extract semantic and human-interpretable concepts. SAEs are trained to simulate the forward pass of an intermediate layer in a neural network. While prior work generally applies small SAEs to toy models, this research established the scaling laws necessary to apply Top-K SAEs to larger model architectures. We extend this foundation by applying scaled SAEs to the multimodal representation of Whisper.

Activation Shifts from Fine-Tuning. Minder et al. (2026) [4], demonstrates that fine-tuning creates detectable activation differences between base and a fine-tuned models, which can show the specific biases introduced during domain adaptation. We use this methodology to project the residual streams of our base and fine-tuned Whisper models into the sparse latent space of our trained SAEs.

Falsifying Spurious Features. Ma et al. (2026) [5] questions whether SAE features with large activations represent conceptual understanding or are residue from overfitting to activations. Using causal token injections, they found that many features which appeared to demonstrate reasoning were spurious. Given this result, we acknowledge that emergent SAE features correlating with ATC jargon may be from activation overfit rather than inherent semantic meaning.

3 Approach

We investigate the base Whisper Large-v3 [6] and the fine-tuned Whisper Large-v3 model, which has been fine-tuned on ATC communication (fine-tuning details can be found at <https://huggingface.co/jacktol/whisper-large-v3-finetuned-for-ATC>). First, we conduct baseline evaluations and activation differencing to understand variations in model performance by dataset domain.

Second, for each model, we train 5 top-K SAEs using layer 16 decoder activations extracted from inference on the ATC dataset, in order to discover if specific human readable features are enhanced or tend to emerge more in the domain. Details of our SAE training methodology are included in A.1. We must be careful with statistical interpretation of results, as there may be significant variability in which features emerge depending on the initialization of the SAE [7].

Thirdly, we probe the SAE latents using clustering and LLM-based auto-interpretability methods to find semantically-coherent latent clusters. To interpret the latent units, we collect SAE activations associated with each output token predicted by the model and run high-dimensional clustering algorithms to identify if any latent units correspond to semantically coherent concepts.

Identification of semantically coherent concepts was done via an LLM judge which attributes semantic meaning to the set of clustered tokens. The LLM judge client code was borrowed from Assignment 4. We then embed identified labels into a sentence encoder to move them into a smooth space where we can run statistical tests, evaluate distances between sets of labels, and evaluate extreme values.

Lastly, we steer the original model with our identified intermediate feature axes and SAEs to evaluate the coherence of the fine-tuned embedding space with that of the original model.

4 Experiments

4.1 Data

We use the Air Traffic Control - Automatic Speech Recognition (ATC-ASR) dataset (<https://huggingface.co/datasets/jacktol/ATC-ASR-Dataset>) [8][9][10][11], which consists of ATC audio recordings paired with human-labeled transcriptions. This dataset was used to fine-tune the fine-tuned Whisper model. The dataset consists of 8,122 samples (848MB total), split into 6.5k train, 812 validation, 813 test.

We also use the Librispeech corpus as a reference dataset. This is a general-purpose dataset of roughly 1000 hours of spoken English audiobook recordings with associated transcriptions. For evaluation purposes, we use the test.clean split, which consists of 2.62k samples (~350MB).

4.2 Evaluation method

We evaluate our model’s baseline performance using the Word Error Rate (WER) metric, which measures the error percentage (insertions, substitutions, or deletions) out of all N words generated.

$$\text{WER} = \frac{\text{Insertions} + \text{Substitutions} + \text{Deletions}}{\text{Words Generated}}$$

To quantitatively evaluate our SAE, we use the metric of reconstruction loss (formulated with mean square error [MSE] regression). Poor reconstruction loss may result in significant degradation of model performance, limiting what features can be inferred from the SAE. However, improved reconstruction loss comes with a trade-off of reduced sparsity.

Once trained, we will use SAEs to evaluate how features in embeddings space might change after fine tuning. In annotating our base-model and fine-tuned-model SAEs, different semantic features may emerge. Because SAEs can learn different sets of features on the same model and data depending on its initialization [7], we will evaluate multiple SAEs on the same models to create a statistical test. Our null hypothesis is that the emergent features come from the same distribution, and our alternate hypothesis is that they come from different distributions. To perform the test, we first embed all labels that emerge from the auto-interpretability pipeline with a sentence encoder. This allows us to analyze the interpretable labels in both distributions in a metric space. We can then run a Maximum Mean Discrepancy (MMD) [12] test to evaluate whether we can detect a significant difference in the underlying distribution of labels.

Finally, we evaluate steering the model by identifying clusters of activations which correspond to a set of SAE latents and injecting the latent activations as a steering vector in the base model. If we can successfully steer these features in the base Whisper model, this would indicate that fine-tuning did not create entirely new embedding directions in embedding space or significantly modify critical circuits. We evaluate our steering using the metrics of Average Target Word Frequency, which is the average number of times the targeted word appears per steered sample, and Target Word Coverage, which is the percentage of all steered samples in the dataset that contain the targeted word.

4.3 Experimental details

4.3.1 Baseline performance

To measure baseline performance of the base and fine-tuned model, we evaluate each model’s WER (lower is better) on both the ATC-ASR dataset and the Librispeech clean.test dataset (figure 1). We

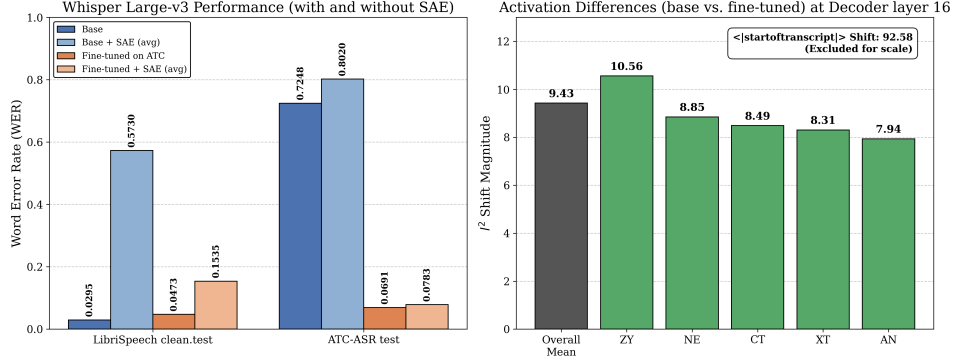


Figure 1: Left: Word Error Rate (WER) baselines for the base and fine-tuned models with (averaged) and without SAE stitching. Right: Activation Diffing (l^2 norm) at Decoder Layer 16 between the base and fine-tuned models. Largest 5 activation differences plotted with associated token.

also compare the activation differences for Layer 16 of the base and fine-tuned models using the l^2 norm.

We start by training 5 SAEs per model (base Whisper, fine-tuned) over air traffic control data to account for initialization variance [7], for a total of 10 SAEs. We save activations for both the base and fine-tuned model before and after decoder layer 16 over the ATC-ASR train split (400MB per model). We validate over the ATC dataset validation split. Because our data is limited, we train our TopK SAE for 50 epochs with learning rate $3e-4$. Training on a single A100 40GB GPU took ~20 minutes per SAE, for a total of ~4 hours of training time. Training details can be found in A.1.

4.3.2 Clustering

Running an automatic-interpretability workflow on multiple SAEs with 10,000 latent units can become prohibitively expensive both in time and cost for the context of this class. To reduce the cost of our analysis, we ran the interpretability pipeline on a cluster of latent units rather than on a per-unit basis.

We performed two complementary clustering analyses to summarize activation structure in the SAE representations. First, we cluster tokens based on their SAE activation patterns. For each token in the dataset, the trained SAE produces a sparse activation vector $z \in \mathbb{R}^{m_{\text{latents}}}$ containing exactly $k = 32$ non-zero entries due to the Top- K sparsification constraint. These activation vectors are L2-normalized and clustered using MiniBatchKMeans with $K = 32$ clusters, which provides computational efficiency for the large number of tokens processed. Tokens grouped into the same cluster therefore share similar patterns of co-activating latent features, allowing each cluster to approximately capture a shared semantic concept.

Second, we clustered SAE latent features themselves based on their learned decoder directions. Specifically, we applied k -means clustering to the rows of the decoder matrix $W_{\text{dec}} \in \mathbb{R}^{m_{\text{latents}} \times d}$, where each row corresponds to the direction in residual space associated with a latent unit. Clustering these vectors groups together latent features that represent geometrically similar directions in the model’s embedding space. This partitions the full set of 10,240 latent units into $K = 32$ clusters, each containing roughly 320 features.

For each resulting cluster, we construct a compact summary consisting of the most frequent tokens assigned to that cluster and the latent features with the highest activation frequencies within the cluster. These summaries are passed to the LLM-based interpretability pipeline. By operating on clusters rather than individual latent units, this approach substantially reduces the computational cost of automatic interpretation while preserving meaningful structure in the learned feature space. The full clustering pipeline ran on all five SAE initializations for both the base and fine-tuned Whisper models, producing independent cluster sets for each model. The cluster labels produced by this process form the distributions used in the downstream Maximum Mean Discrepancy (MMD) statistical test described in section 4.3.4.

4.3.3 Automatic Interpretability (Auto-Interp)

The LLM-judge used to label clusters of latent units was Gemini 2.0 Flash. We pooled the most common tokens that activated this cluster of latents, and used the prompt template in section A.3. To account for stochasticity in the LLM judge’s output, we ran the pipeline 3 times per cluster, and pooled all labels together. This naturally provides more weight to confident responses, as they will appear more frequently in the label pool. We also tested a variant where the domain of air traffic control was provided to the judge (section A.4), however, since we are comparing distributions, we want to be careful not to bias the judge into limiting interpretations too much, as this could compress interpretations into a space which does not capture the full dynamics of the model.

4.3.4 Label Embeddings and Statistical Tests

Once labels were generated for all SAEs in both of the models, we embed the labels with a sentence encoder. We chose all-mpnet-base-v2, which embedded our discrete labels into a smooth 768 dimensional space [13]. Separating the label embeddings by their source (fine-tuned or base model), we run an MMD test. MMD tests whether two sets of points were drawn from the same distribution by comparing their mean embeddings in a kernel feature space. We used a gaussian RBF kernel. The difference in mean embeddings is determined to be significant by evaluating against the null hypothesis, which consists of random permutations that combine the two classes into a single empirical distribution. The test is bootstrapped with 10,000 random permutations, sampling points with replacement. This experiment tests our hypothesis that the distribution of interpretable features is significantly different between the baseline and fine-tuned models.

We hypothesized that the underlying distributions are different, and also that interpretable features resulting from the fine-tuning have a tighter semantic distribution. We evaluated semantic tightness qualitatively by projecting the label embeddings along principle components and observing clustering patterns. We also quantitatively evaluated tightness by calculating pairwise distances between members of each class in label embedding space. We ran another permutation test to evaluate if the mean pairwise distances in the fine-tuned model labels are significantly smaller than the mean pairwise distances in the base model labels to reject the null hypothesis, that both classes have the same underlying pairwise distances.

We also hypothesized that a component of the fine-tuned model’s tighter distributions is more robust representations for numbers and numerical-related concepts, which are overrepresented in ATC chatter compared to general natural speech. We counted auto-interp labels related to numbers in the base and fine-tuned labels, and ran a t-test to evaluate if there is a significant difference in mean counts. We ran the labeling pipeline eight times on all ten of our SAEs for this statistical test.

4.3.5 Base Model Steering

We steer the base model using the outputs of our cluster labels to determine whether these clusters truly represent semantic concepts. We run the base Whisper model and add residuals from the three SAE latents with the greatest average activation for the tokens in that cluster. As a case study, we target the latent features corresponding to the tokens ONE (Cluster 16 in A2) and TWO (Cluster 12 in A2). During autoregressive generation, we inject a steering vector into the Layer 16 residual stream h

$$h_{\text{steered}} = h + \alpha \sum_{i \in F} W_{\text{dec}}^{(i)}$$

where α is a scalar steering intensity coefficient, F is the set of target steering latents in the SAE, and W_{dec} is the activations of these features from the SAE decoder matrix.

4.4 Results

4.4.1 Baseline Performance

Figure 1 indicates that the fine-tuned Whisper model greatly outperforms the base model on the ATC-ASR dataset (+65.1% WER), but the base model slightly outperforms the fine-tuned model on the general-domain Librispeech dataset (+1.8% WER).

We also see that the fine-tuned SAE is largely able to represent the ATC domain, where intervening only degrades accuracy slightly (+0.9% WER). However, using the base SAE with the reference Librispeech dataset leads to a drastic performance reduction (+54.4% WER), far worse than the fine-tuned SAE’s performance (+10.6% WER) which could indicate that the fine-tuned model’s internal representations have become more compact and more easily compressible in the ATC domain.

On the right side of Figure 1, we notice that the largest activation difference is in the `<|startoftranscript|>` token (92.58), while all other tokens have order-of-magnitude smaller activation differences.

4.4.2 Clustering

To better understand the semantic structure learned by the sparse autoencoder, we examined representative token clusters produced by the clustering pipeline. Several clusters exhibit near-perfect token purity, with a single token type dominating the entire cluster, while others contain mixed subword fragments. Full visualizations of the top tokens per cluster for both the base and fine-tuned models are provided in Appendix A (Figures A1 and A2); here we highlight a few illustrative examples (Figure 2).

In the base model, the token ONE forms a dedicated cluster (Cluster 16) containing only the space-prefixed variant ‘ ONE’, while TWO similarly occupies its own isolated cluster (Cluster 12) with ‘ TWO’. These highly pure clusters indicate that the SAE already captures distinct activation patterns for individual number tokens. More generally, the base model contains many clusters dominated by a single token, suggesting that token fragments often activate highly specific latent features rather than grouping with related fragments. In the fine-tuned model, the representation of ONE becomes slightly more consolidated: Cluster 6 groups both the space-prefixed ‘ ONE’ and the unprefixed ‘ONE’ variant. This suggests that fine-tuning strengthens the association between positional variants of the same word, grouping together Byte Pair Encoding fragments that correspond to the same semantic concept.

Another difference appears in the treatment of number-ending fragments. In the base model, fragments such as ‘IVE’ (FIVE) and ‘OUR’ (FOUR) appear together in Cluster 10. In the fine-tuned model, however, ‘OUR’ instead appears in a cluster (Cluster 20) with other number endings (‘VEN’, ‘IGHT’, and ‘IX’), while ‘IVE’ moves to a cluster (Cluster 4) containing tokens such as ‘FL’ (FLIGHT) and ‘MIKE’ from the NATO phonetic alphabet. This grouping reflects common ATC communication patterns and shows that fine-tuning reorganizes SAE features around domain-relevant token structures. Overall, the fine-tuned model exhibits fewer single-token clusters and more clusters containing semantically related fragments, suggesting a tighter and more structured SAE latent space.

4.4.3 MMD Test

In our results, we found a significant difference in the latent unit labels, with $p = 0.0008$. Note that when we inform the LLM judge to bias toward ATC related chatter, our results become marginally insignificant, with $p = 0.086$.

4.4.4 Embeddings Distances

In Figure A4, we plot the embeddings along two principle axes. One hypothesis we can test is that when the model is uncertain, the fine-tuned model learns to predict a tighter distribution of tokens that are particularly relevant for domain. While the first two principle components only explain a combined 17% of the variance, it is not implausible to see tighter clusters along these two components in the fine-tuned model. For instance the tightest cluster in the plot is in the fine-tuned set roughly at coordinates $(-0.4, -0.1)$ on Figure A4.

Table 1: Distribution Tightness of SAE Feature Label Embeddings

Judge	Model	Mean Dist. to Centroid	Std. Dist. to Centroid	Mean Dist. to Median	Mean Pairwise Dist.
Unbiased	Base Model	0.8926	0.0683	0.8941	1.2595
	Fine-tuned Model	0.8816	0.0793	0.8831	1.2438
Biased	Base Model	0.9098	0.0472	0.9117	1.2831
	Fine-tuned Model	0.9080	0.0441	0.9103	1.2799

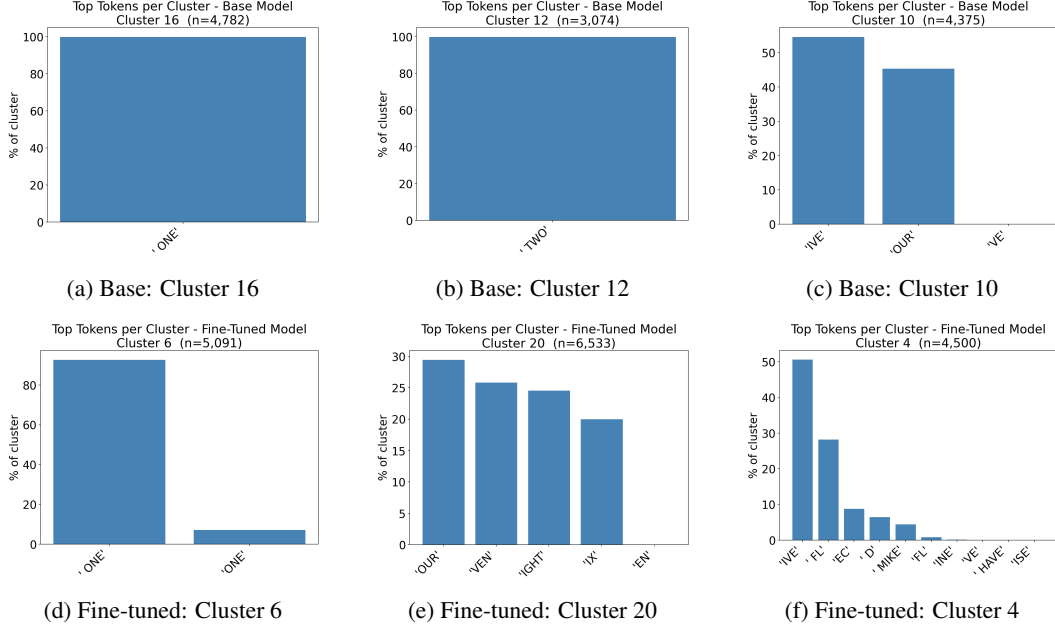


Figure 2: Representative token clusters from the base and fine-tuned Whisper models. The top row shows illustrative clusters from the base model, while the bottom row shows corresponding examples from the fine-tuned model.

In Table 1, we see that across both judges, the resulting label embeddings are tighter in the fine-tuned model than in the base model, though the actual magnitudes of differences are small. To determine significance, we run a permutation test, this time on the mean pairwise distances in each set of the unbiased judge. We find that our results are significant with $p = 0.019$, providing evidence that the distribution of labels in the fine-tuned model are tighter than that of the base-model.

4.4.5 Base Model Steering

We successfully steered the model towards meaningful semantic concepts in a dose-response experiment. We focus on steering for clusters 16 and 12 in the base model, corresponding to ONE and TWO, respectively, because they are natural and semantically-relevant features. At baseline ($\alpha = 0.0$), ONE appears naturally in only 2.0% of the transcripts. When steered with $\alpha = 10.0$, the model hallucinates ONE in 80.0% of the transcripts, averaging 5.96 occurrences per sample. We observed similar causal behavior when steering the feature for TWO (increasing from 3.0% to 26.0% coverage), indicating that these specific latent directions govern semantic token generation. See Table 3 for full statistics.

5 Analysis

Our baseline evaluations indicates that the base model struggles to generalize to the ATC domain, while the fine-tuned model only slightly loses general transcription capabilities. This behavior indicates that the fine-tuned model does not undergo catastrophic forgetting in the fine-tuning process, and similar features exist in both models. Additionally, our activation differencing appears to indicate a fundamental distribution shift between the structure of the base and fine-tuned model transcripts, since the first token exhibits a large l^2 shift compared to text tokens.

Our statistical tests showed statistically significant differences in the underlying distributions of the auto-interp labels. The MMD test is non-parametric, and allowed us to claim that the distributions of semantic labels are different between base and fine-tuned models. Increased performance on the ATC domain appears to drive a shift in the semantic concepts represented in the intermediate hidden layer. Interestingly, when biasing the judge to look for ATC related concepts, our results became marginally insignificant. This may be because the LLM judge attributes noisier units to concepts directly related to ATC more often, resulting in a more limited distribution of labels that are more difficult to separate.

Indeed, the underlying model is trained on more than just ATC data, even if our evaluations are primarily focusing on this domain. The remaining statistical tests provide evidence for our two other hypotheses that the distribution of semantic concepts in the fine-tuned model is tighter, and that the fine-tuned model has a more differentiated representation of numbers. This pattern is also reflected in the clustering results: the fine-tuned model more frequently groups semantically related tokens together, while the base model contains more single-token clusters. Together, this may reflect the fine-tuned model learning a strategy to increase the probability of outputting number-related tokens.

Our steering experiments demonstrate that we can semantically enhance certain identified features in the output. When comparing the results for clusters 16 (corresponding to the ONE token) and 12 (corresponding to the TWO token), we need a larger steering coefficient to achieve maximum steering for cluster 12 3, and the effect is not as strong. This may suggest that the cluster 16 is more aligned with the ONE feature than cluster 12 is with the TWO feature, which could be because there are more ONE tokens (4,782) than TWO tokens (3,074) in the ATC dataset.

6 Conclusion

We investigated the differences in internal representations in the base Whisper Large-v3 model and a fine-tuned Whisper Large-v3 model for ATC transcription. We successfully trained SAEs to identify semantic clusters in both a base Whisper model and a Whisper model fine-tuned on ATC transcriptions, then established causality in the latent units by steering them to produce effects in model output. We found statistically-significant evidence that there is a significant difference in the underlying distribution of semantic content between the base Whisper model and a Whisper model fine-tuned on ATC transcriptions. This difference is characterized by tighter semantic distributions in intermediate layer representations, which may reflect a model strategy of increasing the probability of outputting domain-related tokens such as numbers.

6.1 Limitations

In this project, we were heavily limited by compute resources. Our conclusions are only based on an analysis of one decoder layer in Whisper, and they may not hold for other model layers. To limit the number of latent units we needed to interpret, we clustered them together by similar activation patterns on tokens. With more time and compute, we could repeat our experiments for each individual SAE latent. Additionally, we trained our SAEs only on activations from the ATC dataset. This allows us to determine if the base model discovers features relevant to the ATC dataset *less* often than the fine-tuned model. Future work could train SAEs including data from the reference Librispeech dataset, which could evaluate if the fine-tuned model is attributing features relevant to the specialized ATC dataset *more* often than the base model on the reference dataset. That is, we can evaluate if the model is over-attributing ATC-relevant tokens to instances where it should not. Future work could also ablate the underlying sentence embedding as well as the specific LLM judge model to increase the power of our tests.

Team contributions

JJ conducted clustering experiments and analyzed the resulting cluster structures. BL conducted statistical tests and auto-interp of clusters. JH conducted baseline evaluations, trained the SAEs, and evaluated model steering. All members contributed equally to the final report.

References

- [1] Jack Tol. Fine-tuning Whisper for Air Traffic Control (ATC). https://jacktol.net/posts/fine-tuning_whisper_for_atc/, 2023. Accessed: 2026-03-14.
- [2] Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, Alex Tamkin, Esin Durmus, Tristan Hume, Francesco Mosconi, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermy, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, May 2024.

- [3] Leo Gao, Tom Dupre la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *The Thirteenth International Conference on Learning Representations*, 2025.
- [4] Julian Minder, Clément Dumas, Stewart Slocum, Helena Casademunt, Cameron Holmes, Robert West, and Neel Nanda. Narrow finetuning leaves clearly readable traces in the activation differences. In *The Fourteenth International Conference on Learning Representations*, 2026.
- [5] George Ma, Zhongyuan Liang, Irene Y. Chen, and Somayeh Sojoudi. Falsifying sparse autoencoder reasoning features in language models, 2026.
- [6] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine Mcleavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 28492–28518. PMLR, 23–29 Jul 2023.
- [7] Gonalo Paulo and Nora Belrose. Sparse Autoencoders Trained on the Same Data Learn Different Features, January 2025. arXiv:2501.16615 [cs].
- [8] Juan Zuluaga-Gomez, Karel Veselý, Igor Szőke, Petr Motlicek, et al. Atco2 corpus: A large-scale dataset for research on automatic speech recognition and natural language understanding of air traffic control communications. *arXiv preprint arXiv:2211.04054*, 2022.
- [9] Juan Zuluaga-Gomez, Seyyed Saeed Sarfjoo, Amrutha Prasad, et al. Bertraffic: Bert-based joint speaker role and speaker change detection for air traffic control communications. *IEEE Spoken Language Technology Workshop (SLT), Doha, Qatar*, 2022.
- [10] Juan Zuluaga-Gomez, Amrutha Prasad, Iuliia Nigmatulina, Saeed Sarfjoo, et al. How does pre-trained wav2vec2. 0 perform on domain shifted asr? an extensive benchmark on air traffic control communications. *IEEE Spoken Language Technology Workshop (SLT), Doha, Qatar*, 2022.
- [11] Luboš Šmíd. Air traffic control communication, 2011. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL).
- [12] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A Kernel Two-Sample Test. *Journal of Machine Learning Research*, 13(25):723–773, 2012.
- [13] sentence-transformers/all-mpnet-base-v2 · Hugging Face, January 2024.

A Appendix

A.1 Top-K SAE training details

The SAE is trained on decoder block activations with shape $[B, T, d]$, where the dimension $d = 1280$. Each token-level embedding is treated as an independent input to the SAE. Our Top-K SAE is inspired by the formulation in Gao et al. (2025) [3], but all model components and training logic were implemented by us in PyTorch.

Given an input activation $x \in \mathbb{R}^d$, the SAE first applies a linear encoder with a shared bias term: $z_{\text{pre}} = W_{\text{enc}}(x - b_{\text{dec}}) + b_{\text{enc}}$ where W_{enc} expands the representation by an expansion factor r , and b_{enc} and b_{dec} are learned bias parameters. To promote interpretability and prevent dense feature superposition, we apply a Top-K sparsification operator, which retains only the k strongest latent activations per token and suppresses the rest: $z_i = \max(z_{\text{pre},i}, 0)$ if $i \in \text{TopK}(z_{\text{pre}})$, otherwise $z_i = 0$. This ensures that each token activates exactly k latent features. The sparse latent representation is decoded back into the original embedding space via a linear decoder $\hat{x} = W_{\text{dec}}z + b_{\text{dec}}$ and the SAE is trained to minimize the mean squared reconstruction error $\mathcal{L}_{\text{rec}} = \|x - \hat{x}\|_2^2$. Decoder weight vectors are normalized to unit L_2 norm to stabilize training and encourage each latent dimension to correspond to a distinct direction in the embedding space.

A common challenge in training sparse autoencoders is the emergence of dead latents, which rarely activate and fail to capture meaningful structure. To monitor latent utilization, we track activation frequencies across token positions, $f_j = \mathbb{E}[\mathbf{1}(z_j > 0)]$, and identify latents with consistently low activation rates. For these underutilized features, we introduce an auxiliary reconstruction objective that encourages dead latents to model residual reconstruction error, improving coverage of the latent space while preserving the fixed Top-K sparsity constraint.

Table 2: Final-epoch SAE Training Metrics across five initializations. Training was stable with steadily decreasing MSE and increasing EV.

SAE Initialization	Base Whisper Model		Fine-Tuned Whisper Model	
	MSE	Explained Variance (EV)	MSE	Explained Variance (EV)
Run 0	0.0060	0.9983	0.0047	0.9957
Run 1	0.0061	0.9982	0.0051	0.9953
Run 2	0.0102	0.9971	0.0071	0.9936
Run 3	0.0100	0.9971	0.0070	0.9936
Run 4	0.0101	0.9971	0.0071	0.9936

A.2 SAE Swapping Results

The first experiment we ran involved training two SAEs, one on the base model, and one on the fine-tuned model. These SAEs are trained using the fine-tuned ATC data. We then swapped the SAEs between the models (see figure A1). One interesting observation is that the fine-tuned model performs better on the reference dataset with the base SAE, which suggests that the base SAE learns a more generalizable decoder layer in the base model during training, even if it is on ATC activations.

Additionally, intervening the fine-tuned SAE with the baseline model, the model began outputting capital letters, which matches the distribution of the labels used to train the SAE. At a minimum, this demonstrates some space or vector representing capital letters in the baseline model has not been destroyed or significantly altered in the fine-tuned model. The SAE, trained on a distribution that contains only capital letters, is adding a component representing capital letters into the residual stream which is still interpretable by the base model.

A.3 LLM Judge Unbiased Prompt

```
PROMPT_TEMPLATE = """You are analyzing a feature cluster from a Sparse
↪ Autoencoder trained on an audio transcription model.
```

Tokens that strongly activate this cluster, with each token formatted as
↪ (token, frequency in cluster):
Tokens: {positive_tokens}

Give a 2-5 word label for the concept this cluster detects, a confidence,
↪ and coherence evaluation on how well the.

Your output will be the ONLY the following text and formatted as such:

LABEL:
CONFIDENCE: low | medium | high
COHERENCE: coherent | mixed""

A.4 LLM Judge Biased Prompt

PROMPT_TEMPLATE = ""You are analyzing a feature cluster from a Sparse
↪ Autoencoder trained on an audio transcription model in the domain of
↪ air traffic control.

Tokens that strongly activate this cluster, with each token formatted as
↪ (token, frequency in cluster):
Tokens: {positive_tokens}

Give a 2-5 word label for the concept this cluster detects, a confidence,
↪ and coherence evaluation on how well the.

Your output will be the ONLY the following text and formatted as such:

LABEL:
CONFIDENCE: low | medium | high
COHERENCE: coherent | mixed""

A.5 Example Outputs from Feature Steering

We provide example outputs from our dose-response steering intervention on the base Whisper model. In these examples, the base model is steered using the latent features corresponding to the token ONE extracted from the trained SAE.

At $\alpha = 0.0$, the model outputs its standard (often flawed) baseline transcription. At $\alpha = 10.0$, the model is successfully steered to hallucinate the target concept, frequently injecting the word "ONE" into the sequence regardless of the source audio. At extreme coefficients ($\alpha \geq 30.0$), the model fails and outputs a loop. See 3 for fuller statistics.

Example 1

Ground Truth: DIRECT RATEV LUFTHANSA EIGHT MIKE MIKE THANK YOU

Coef 0.0 (Base): That's up to the front of the mic. Thank you.

Coef 10.0: Thank, ONE MORE LIME ONE MORE LIME ONE MORE LIME ONE MORE ONE
ONE ONE ONE ONE ONE ONE ONE ONE ONE ONE ONE ONE ONE ONE ONE ONE
ONE ONE ONE ONE ONE ONE ONE ONE ONE ONE ONE ONE ONE ONE ONE

Coef 30.0: more more more more more more more more more more more more more more more
more- more- more- more- more- more- more- more- more- more- more- more- more- more-

Example 2

Ground Truth: LOT FIVE TWO FIVE TURN LEFT HEADING ONE SIX ZERO CLEARED FOR ILS APPROACH RUNWAY ONE THREE SPEED ONE SEVEN ZERO

Coef 0.0 (Base): LOT 525, the left heading 150 cleared for approach runway 13, speed 170.

Coef 10.0: Five Five Five Five Five Five F ONE M E S E S E E S E E S E E S E E S E E S
E E S E E S

Coef 60.0: more- more- more- more- more- more- more- more- more- more- more- more-
more- more- more- more- more- more- more- more- more- more- more

Example 3

Ground Truth: NUMBER ONE GOLF BRAVO

Coef 0.0 (Base): Everyone from 30,000, go for it.

Coef 10.0: one for THT END ONE FOR THT END ONE FOR THT END ONE FOR THT END
ONE FOR THT END ONE FOR THT END ONE FOR THT END ONE FOR

[illegible]

Table 3: Dose-response of SAE-based steering for base Whisper.

Steering Coefficient (α)	Target: ONE (Cluster 16)		Target: TWO (Cluster 12)	
	Average Target Word Frequency	Target Word Coverage (%)	Average Target Word Frequency	Target Word Coverage (%)
0.0 (Baseline)	0.03	2.0%	0.03	3.0%
10.0	5.96	80.0%	0.32	26.0%
30.0	0.00	0.0%	2.51	39.0%
60.0	0.00	0.0%	0.00	0.0%

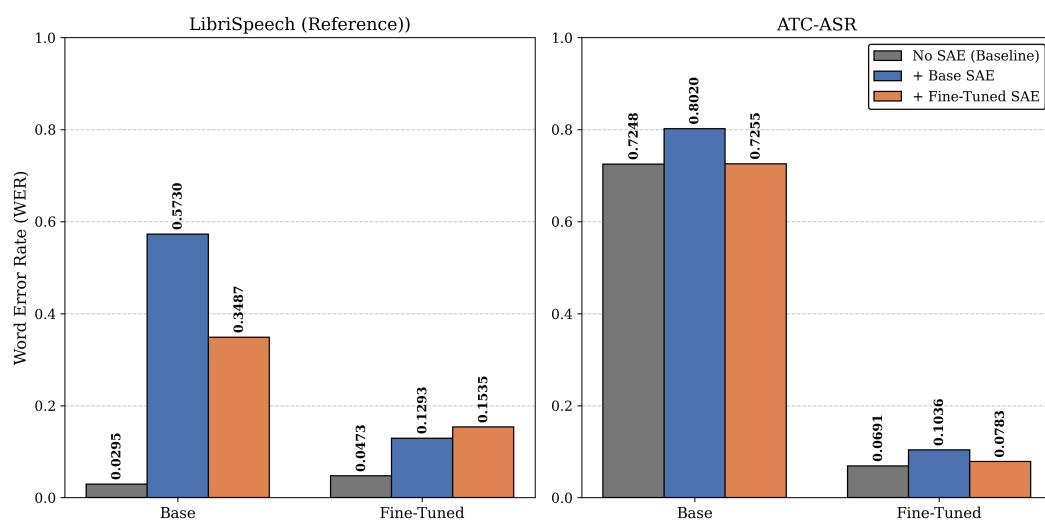


Figure A1: Lower WER is better. Cross-stitching SAEs slightly degrades performance on the ATC dataset but improves performance on the reference dataset.

Top Tokens per Cluster - Base Model

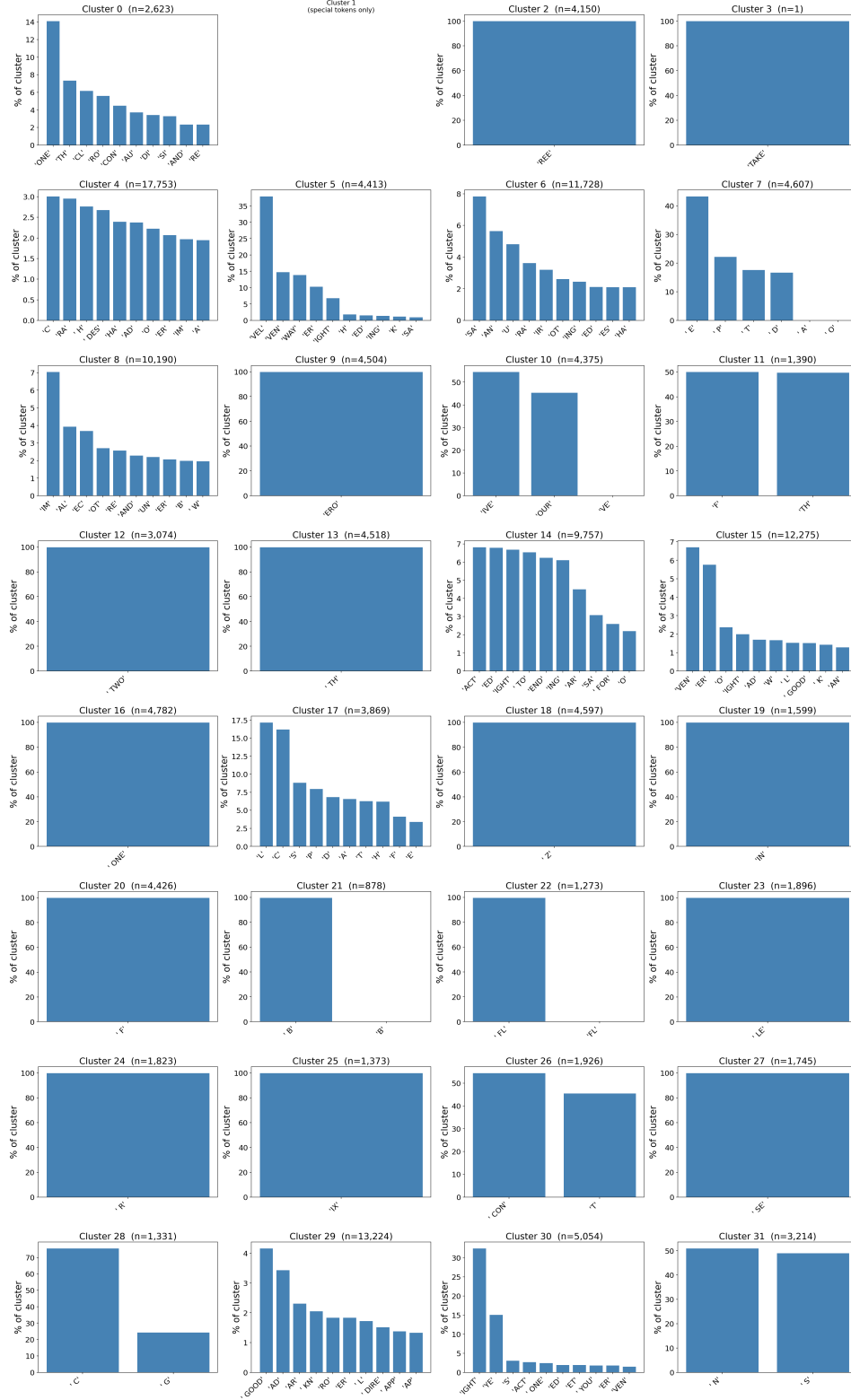


Figure A2: Most frequent tokens per cluster for the SAE trained on activations from the base Whisper model. Each subplot shows the most common tokens assigned to a cluster based on SAE activation patterns.

Top Tokens per Cluster - Fine-tuned Model

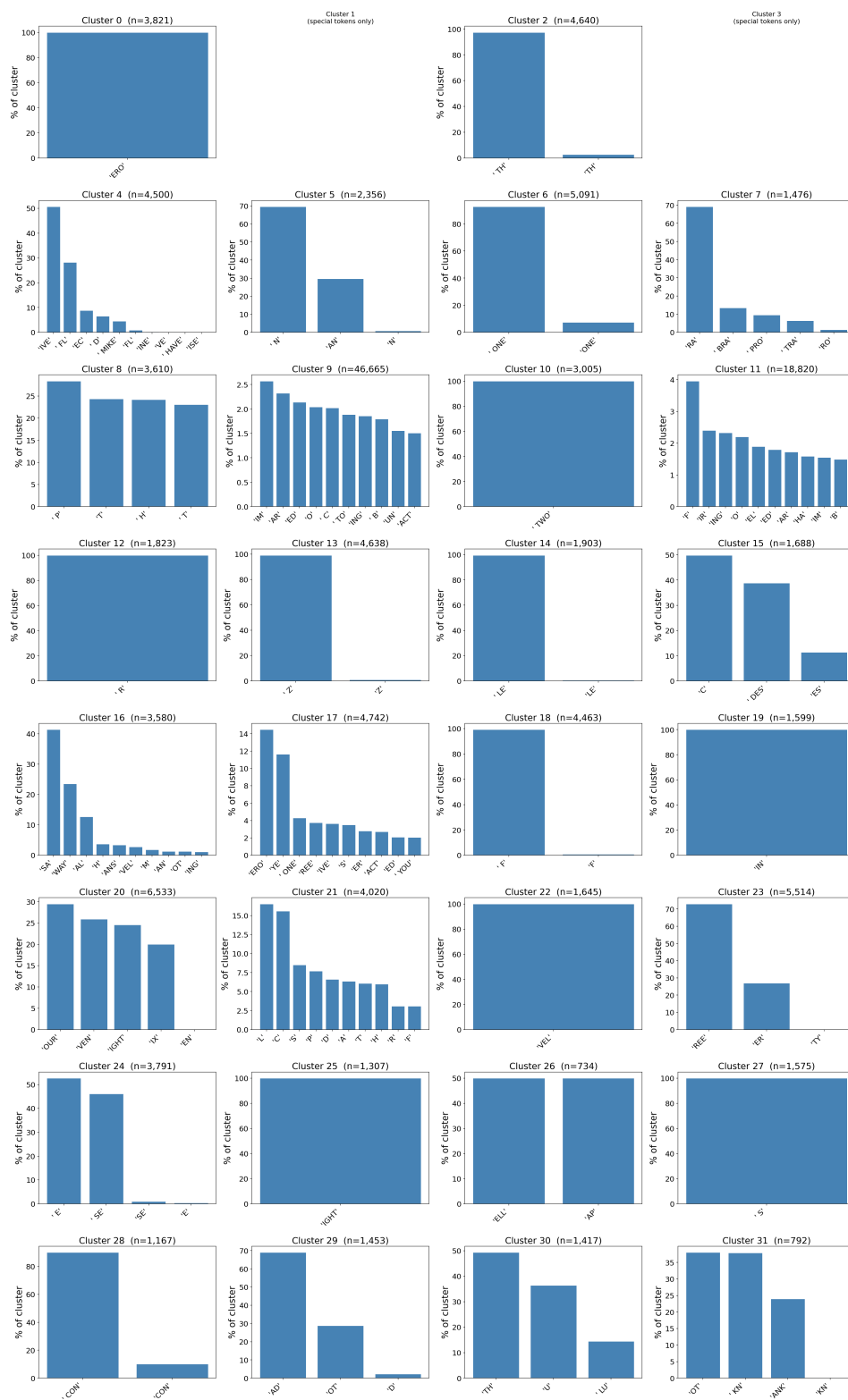


Figure A3: Most frequent tokens per cluster for the SAE trained on activations from the ATC fine-tuned Whisper model. Each subplot shows the most common tokens assigned to a cluster based on SAE activation patterns.

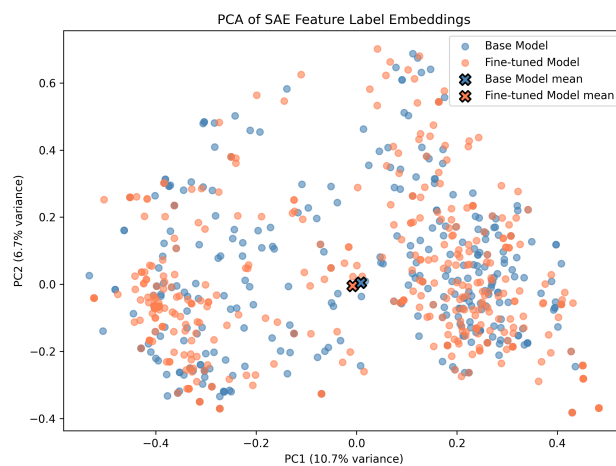


Figure A4: Fine tuned and base model SAE automatically interpreted labels in embedding space

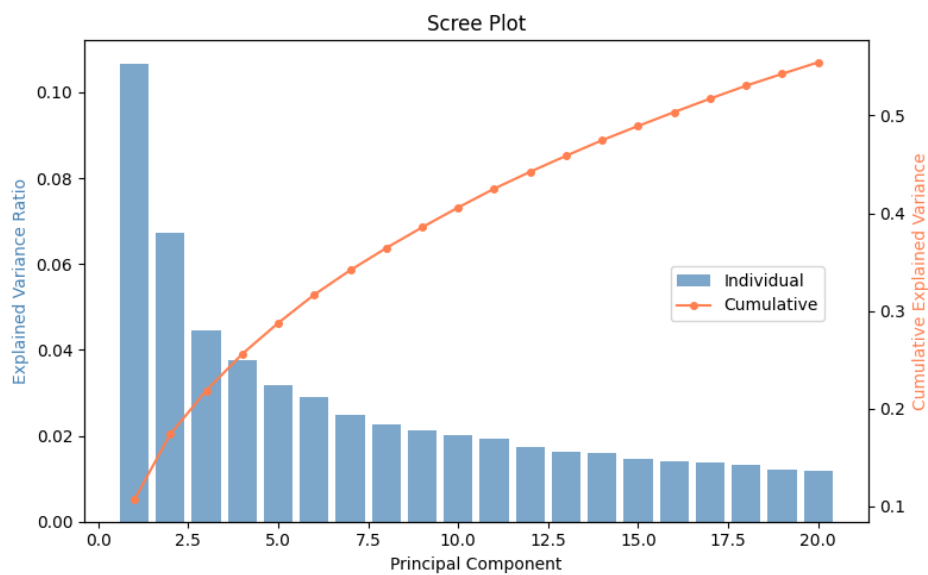


Figure A5: Scree plot of PCA components

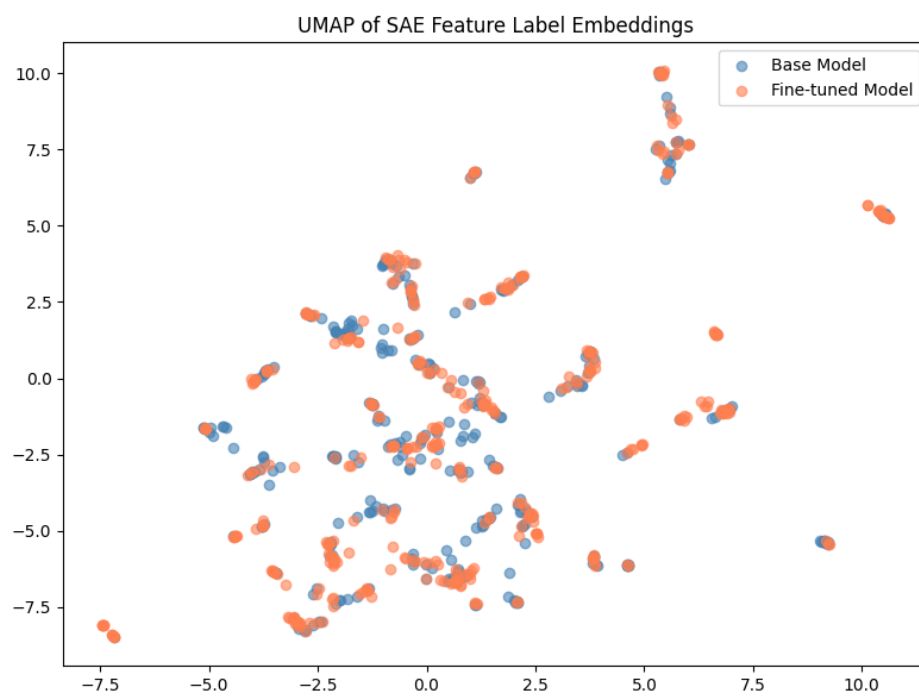


Figure A6: Label embeddings on first 2 UMAP components