Words in Motion: Extracting Interpretable Control Vectors for Motion Transformers

Ömer Şahin Taş* Royden Wagner*

*equal contribution

Karlsruhe Institute of Technology

FZI

Contributions

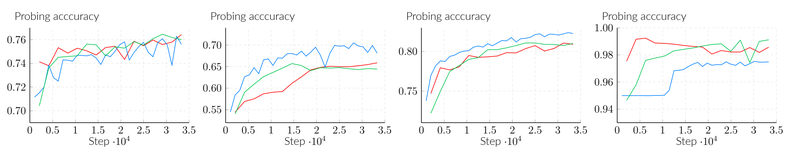
Research marked in dark blue represents ongoing work, complementing the findings presented in our workshop paper.

1. We use **linear probes** to measure neural collapse towards interpretable features in hidden states.
2. We leverage these latent space properties to fit **control vectors** for each interpretable feature.
3. We optimize our control vectors with **sparse autoencoding**. Notably, we show that enforcing sparsity leads to a more linear relationship between control vector temperatures and forecasts.
4. We apply our method to address domain shift and enable **zero-shot generalization**.

Method

1. We use neural collapse as a metric of interpretability. We use it in multimodal models for motion forecasting (i.e., regression) extending its use beyond unimodal vision classifiers [5] or language models [6].
2. We measure how close abstract hidden states are related to interpretable semantics using linear probing accuracy [1].
3. Rather than steering hidden state changes across all modules (i.e., neural trajectories) as in [2], we steer only the hidden states in the last module of the motion encoder.
4. We do not use our sparse autoencoders during inference [3], but to optimize control vectors beforehand, resulting in negligible computational overhead.

Extracting Interpretable Features



(a) Speed (b) Acceleration (c) Direction (d) Agent

Figure 1. Linear accuracies [1] for RedMotion, Wayformer, and HPTR forecasting models on the validation split of the AV2F dataset.

References

[1] G. Alain and Y. Bengio. Understanding intermediate layers using linear classifier probes. In *ICLR*, 2017.

[2] A.Zou et al. Representation Engineering: A Top-Down Approach to AI Transparency. *arXiv:2310.01405*, 2023.

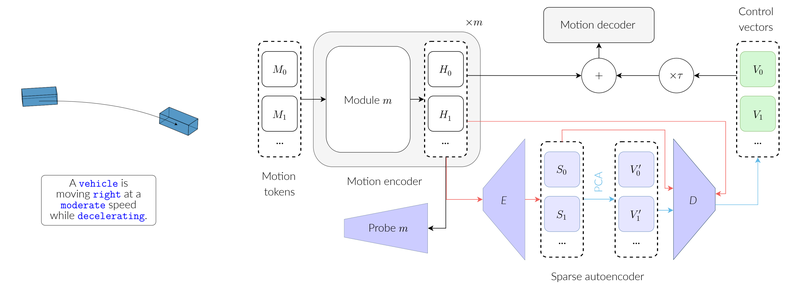
[3] H.Cunningham et al. Sparse Autoencoders Find Highly Interpretable Features in Language Models. *arXiv:2309.08600*, 2023.

[4] S.Rajamanoharan et al. Jumping Ahead: Improving Reconstruction Fidelity with JumpReLU Sparse Autoencoders. *arXiv:2407.14435*, 2024.

[5] V.Papyan et al. Prevalence of neural collapse during the terminal phase of deep learning training. *PNAS*, 2020.

[6] R. Wu and V. Papyan. Linguistic Collapse: Neural Collapse in (Large) Language Models. In *NeurIPS*, 2024.

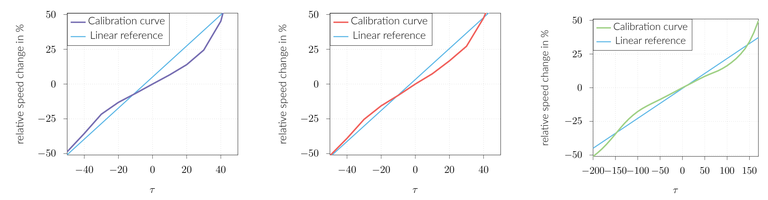
Overview



(a) Interpretable motion features (b) Controllable motion encoder

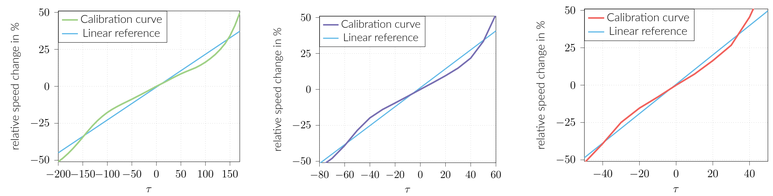
Figure 2. **Words in Motion. (a)** We classify motion features in an interpretable way, as in natural language. **(b)** We measure the degree to which these interpretable features are embedded in the hidden states H_i of transformer models with linear probes. Furthermore, we use our discrete features and sparse autoencoding to fit interpretable control vectors V_i that allow for controlling motion forecasts at inference. The training of the sparse autoencoder is shown with red arrows (→) and the fitting of control vectors with blue arrows (→).

Calibration Curves



(a) Plain PCA [2] (b) SAE [3] (c) ConvSAE k64

Figure 3. Calibration curves for activation steering with plain PCA and SAE-based speed control vectors for relative speed changes in forecasts of ±50%.



(a) ConvSAE k64 (b) ConvSAE k64 JumpReLU (c) SAE128 JumpReLU [4]

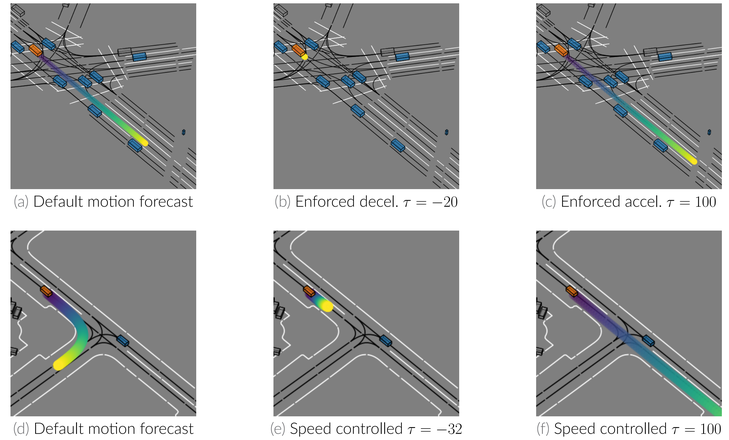
Figure 4. JumpReLU compensates feature shrinkage as reflected in a smaller range of τ values for the same range of relative speed changes.

Quantitative Comparison of Control Vectors

Table 1. Linearity measures for activation steering with control vectors: Pearson correlation coefficient, coefficient of determination (R^2), and straightness index.

Autoencoder	Activation function	Pooling	Patch/kernel size	Pearson	R^2	S-idx
–	–	PCA	–	0.988	0.969	0.981
SAE	ReLU	PCA	–	**0.993**	0.984	0.988
SAE	JumpReLU	PCA	–	**0.993**	**0.986**	0.988
Sparse MLPMixer	ReLU	PCA	64	0.992	0.980	0.986
Sparse MLPMixer	JumpReLU	PCA	64	0.992	0.981	0.986
Sparse MLPMixer	ReLU	PCA	32	0.990	0.978	0.985
Sparse MLPMixer	JumpReLU	PCA	32	0.991	0.980	0.986
ConvSAE	ReLU	PCA	64	0.986	0.383	**0.991**
ConvSAE	JumpReLU	PCA	64	0.987	0.861	0.978
ConvSAE	ReLU	PCA	32	0.988	0.622	0.986
ConvSAE	JumpReLU	PCA	32	0.989	0.623	0.986

Qualitative Results



(a) Default motion forecast (b) Enforced decel. $\tau = -20$ (c) Enforced accel. $\tau = 100$

(d) Default motion forecast (e) Speed controlled $\tau = -32$ (f) Speed controlled $\tau = 100$

Figure 5. Controlling a vehicle with various temperatures.

Zero-shot Generalization with Control Vectors

Table 2. Zero-shot generalization to a Waymo dataset version with reduced future speeds.

Control vector	Temperature τ	minADE↓	Brier minADE↓	minFDE↓	Brier minFDE↓	Overlap rate↓	Miss rate↓
None		3.271	6.547	4.617	8.933	0.220	0.580
SAE-128	-30	1.685	4.838	2.870	8.429	0.179	**0.224**
SAE-128	-50	**1.174**	**2.759**	**1.798**	**4.329**	**0.174**	0.236
SAE-128	-70	1.808	3.576	2.035	3.676	0.189	0.302

nxt AIM

The research leading to these results is funded by the German Federal Ministry for Economic Affairs and Climate Action within the project "NXT GEN AI METHODS".