# Title

Identifying and Causally Probing Rhyming Behaviour in LLMs via Sparse Autoencoders

# Abstract

We aim to identify and exert causal control over the internal mechanisms responsible for rhyming behaviour in large language models (LLMs), using sparse autoencoders (SAEs) to isolate interpretable features. By designing a dataset of rhyme-constrained generations and correlating rhyme metrics with SAE features, we seek to locate candidates responsible for rhyming. We then evaluate the causal role of these features through patching interventions. Our stretch goal is to develop interactive mechanisms for steering model outputs in poetic domains, treating LLM internals as programmable control surfaces.

# Motivation

Interpretability tools like SAEs are increasingly capable of isolating semantically meaningful components of neural computation. While most prior work focuses on classification or reasoning behaviours, we explore a more creative task, rhyming, which is stylistically rich and computationally non-trivial. Understanding and controlling rhyming can pave the way for broader capabilities in manipulating creative behaviors within LLMs.

# Research Questions

1. Which SAE features in a language model are most predictive of rhyming behavior in its outputs?
2. Does causal patching of such features influence rhyme generation in a statistically significant way?
3. Are these features general across prompts, styles, and fine-tuning variants?

# Experimental Design

## Phase 1: Dataset Creation & Rhyme Annotation

- **Prompt Design**: Construct \~3,000 paired prompts designed to produce rhyming and non-rhyming outputs using controlled templated completions.
- **Model**: Use `google/gemma-2b` and corresponding `gemma-scope` SAE features from `sae-lens`.
- **Rhyme Detection**:
- **Automated Metrics**: Compute rhyme similarity using multiple methods:
    - `pronouncing` library for perfect end-rhyme via phonemes
    - stress pattern alignment
    - syllable match
    - orthographic distance (e.g., Levenshtein)
- **Human Validation**: Annotate a subset (e.g. 300 samples) for rhyme type and quality
- **Metadata Schema**:

```
{
  "text": "line one\nline two",
  "rhyme_type": "perfect|slant|eye|none",
```

```
  "stress_pattern": "matched|mismatched",
  "syllable_count": [3, 2],
  "human_validated": true
}
```

### Phase 2: Feature Discovery

- **Activation Extraction**: For each prompt, extract SAE feature activations at the final token of each line.
- **Statistical Association**:
- Use regularized logistic regression or random forest classifiers with rhyme labels as targets.
- Run 10-fold cross-validation to avoid overfitting.
- Evaluate feature importance via SHAP or permutation tests.
- **Control Features**:
- Include features known to correlate with unrelated behaviours (e.g., syntax)
- Randomly sampled SAE features as null distribution

### Phase 3: Causal Probing

- **Feature Patching**:
- Perform feature interventions by activating selected SAE features at generation time using forward hooks
- Vary strength of intervention (e.g., 1σ, 2σ, 3σ, 4σ)
- **Temporal Specificity**:
- Patch features at different generation steps (e.g. only at line-ending tokens vs entire prompt)
- **Evaluation**:
- Compare rhyme rates across intervention levels
- Use significance testing (e.g., paired t-tests, bootstrap confidence intervals)

### Phase 4: Generalisation & Validation

- **Out-of-Distribution Tests**:
- Evaluate rhyme control on different poetic styles or multilingual prompts
- **Negative Controls**:
- Patch unrelated features and verify absence of rhyme induction
- **Additional Behaviours** (Optional):
- Extend method to meter, repetition, alliteration or other poetic structure

## Metrics

```
metrics = {
  'rhyme_rate': float,               # % of outputs that end with a rhyme
  'phonetic_similarity': float,     # cosine similarity of phoneme vectors
  'semantic_coherence': float,      # SBERT cosine score between lines
  'syntactic_validity': float,      # POS tag entropy or grammar score
  'human_preference': float         # blinded human eval
}
```

## Risks & Limitations

- **Interpretation Ambiguity**: Correlation ≠ Causation
- **Overfitting Risk**: With 16K features, false positives are likely without correction (Bonferroni/FDR)
- **Rhyme Subjectivity**: Cross-dialect issues, slant rhyme ambiguity
- **Sparse Labeling**: Human annotation costly—limits supervised validation

## Reproducibility

- All code, prompts, annotations, and patching logic to be made public
- Exact SAE model checkpoints and Gemma weights will be versioned
- Random seeds and intervention configs stored in metadata logs

## Expected Outcomes

- A ranked set of rhyme-associated SAE features
- Demonstration of increased rhyme rate via patching
- Tools for controlled rhyming in poetry generation

## Timeline

**Week 1**: Prompt engineering, dataset curation, annotation schema \ **Week 2**: Feature extraction + statistical modelling \ **Week 3**: Causal patching experiments, evaluation and generalisation \ **Week 4**: Paper writing + reproducibility sweep

## References


1. Cunningham, A., et al. (2023). *Sparse Autoencoders Discover Interpretable Directions in Language Model Representations.* https://arxiv.org/abs/2310.00603
2. Cunningham, A., et al. (2024). *sae-lens: A Library for Sparse Feature Supervision in Transformer Models.* https://github.com/AlignmentResearch/sae_lens
3. Elhage, N., et al. (2022). *Toy Models of Superposition.* https://transformer-circuits.pub/2022/toy_model/index.html
4. Belrose, B., et al. (2023). *The Scaling Hypothesis: Neural Networks Learn Interpretable Circuits.* https://arxiv.org/abs/2304.14997
5. Galichin, A., Dontsov, A., Druzhinina, P., Razzhigaev, A., Rogov, O. Y., Tutubalina, E., & Oseledets, I. (2024). *Probing and Steering Language Models with Sparse Superposition Features.* https://arxiv.org/abs/2404.19713