MATS 6.0 Lee Sharkey application
Daniel Tan

Abstract

- We propose to construct a model organism of superposition, in order to serve as a testbed for interpretable feature extraction
- We propose desiderata on suitable domains - tractability, complexity, and sparsity. We propose (i) code and (ii) games as domains which might meet these desiderata.
- We argue that zero-shot generalization properties are a good indicator of whether the model organism represents ground-truth features robustly.
- We outline key failure modes associated with this research - most notably, the difficulty in establishing correct ground-truth features.

Motivation

Superposition is a key challenge which any effective interpretability method must overcome.
- Large models do not represent features in a monosemantic way, but instead learn representations in superposition. Toy Models of Superposition
- The effectiveness of interpretability methods to address superposition is difficult to evaluate if we do not also know the ground truth (un-superposed) features.

A model organism of superposition would serve as a benchmark to evaluate interpretability methods.

This seems especially timely given that there is currently significant interest in sparse autoencoders, which seem empirically promising but lack rigorous theoretical justification
- Sparse autoencoders have been proposed as a promising method to address superposition Towards Monosemanticity: Decomposing Language Models With Dictionary Learning.
- However, when training SAEs on real LLMs, it remains unclear whether SAEs recover the ground-truth abstractions represented by the model.
    - We discuss this more in the Appendix.
- Being able to provide empirical evidence as to whether SAEs do in fact recover ground truth features will be very informative for future interpretability research on SAEs or otherwise.

Current model organisms of superposition are flawed..
- The 1-layer MLP networks considered in TMS turn out to be too toyish and disanalogous to real LLMs to tell us anything useful. [Full Post] Progress Update #1 from the GDM Mech Interp Team

- An alternative is TracR, which compiles a symbolic program into a transformer that executes the same program. However, the compilation process is contrived and induces disanalogies to real models. In particular, the compilation scheme is designed such that there is no superposition, and artificially inducing superposition doesn't seem to work well. [Full Post] Progress Update #1 from the GDM Mech Interp Team

Concrete Research Questions

1. Can the model organism validate predictions made by existing theories of computation in superposition? Toward A Mathematical Framework for Computation in Superposition — LessWrong
2. Can the model organism prove (or disprove) that sparse autoencoders recover ground-truth features used by the model?

Desiderata

The above failed attempts suggest that it is important to construct a model organism that is tractable to obtain ground-truth features for, while being sufficiently similar to a real LLM to yield meaningful insights. We propose the following desiderata.

- **Domain Tractability.** It must be possible to analytically read-off the ground truth features that are both present in a given input and predictive of output.
- **Feature Sparsity.** The ground truth features must activate in a sparse pattern, i.e. only a relatively small number are present in any single input, thus allowing superposition to emerge.
- **Domain Complexity.** The domain on which the model organism operates needs to be sufficiently complex that the only way the model could generalise perfectly to all inputs is by grokking the ground-truth features.
- **Zero-shot Generality**. The model organism must be capable of solving any task in the given domain, as opposed to the TracR models which are single-task.

Possible Domains for Model Organisms

Restricted Subsets of Language

Although general language modelling could be too broad, it may be possible to study in isolation certain restricted classes of prompt-response pairs. For example, question-answering tasks based on retrieval from a context. [2312.10091] Look Before You Leap: A Universal Emergent Decomposition of Retrieval Tasks in Language Models

Initial guesses for ground-truth features in the retrieval task:

- Names and values of relevant variables

Tasks considered in existing work:
- IOI https://openreview.net/pdf?id=MHIX9H8aYF
- Factual recall https://arxiv.org/abs/2402.07321
- Context-based question answering https://arxiv.org/abs/2312.10091

Code / Other Formal Languages

We can train an LLM to read code written in a simple scripting language (e.g. Lisp, Python) and then predict the output of the script. More generally, formal languages are associated with a grammar (i.e. a set of rules that generate the language), which might be amenable to identifying ground-truth features.

Initial guesses for “ground truth features” in such a domain:
- The names and values of variables
- The names and definitions of functions

We note that existing work may have already trained suitable models in code e.g. CodeBERT CodeBERT: A Pre-Trained Model for Programming and Natural Languages | Papers With Code

Games

We can train an LLM to read a sequence of moves played in a perfect-information game (e.g. Othello, Chess) and predict the next move.

Initial guesses for “ground-truth features” in such a domain:
- Entities in the game (e.g. pieces in chess, othello, plus their positions)
- Rules of the game (e.g. in chess, rules governing the movement of pieces)

We note that existing work may have already trained suitable models, e.g. in Othello [2210.13382] Emergent World Representations: Exploring a Sequence Model Trained on a Synthetic Task and in chess GitHub - sgrvinod/chess-transformers: Teaching transformers to play chess.

Finding Ground Truth Features

Given a choice of domain, how do we identify what the ground-truth features will be? The initial guesses presented above are probably wrong. Finding the actual ground-truth features will be nebulous, difficult, and involve a lot of guesswork and iteration. We envision an iterative procedure consisting of repeating the following loop:

1. Generating candidate ground-truth features (mostly via domain knowledge)

2. Training a sufficiently good model on the data (which will also serve as the model organism). (Note, this may not need to be repeated)
3. Filtering the candidates to the subset that is represented by the model (using e.g. probing and attribution patching)

We note that other filtering criteria have been explored, such as “transferability” to a different model [2310.16410] Bridging the Human-AI Knowledge Gap: Concept Discovery and Transfer in AlphaZero. This would select for features which tend to be represented universally.

Case Study: OthelloGPT

An example where this has been done well is in OthelloGPT, which was found to contain 192 linear representation of board state (“empty” vs “my piece” vs “opponent piece” for 64 board positions).

However, SAEs trained on OthelloGPT did not find these features: Research Report: Sparse Autoencoders find only 9/180 board state features in OthelloGPT. These features may also not activate sparsely enough for the model to learn a superposed representation.

Evaluating a Model Organism

The model organism should learn to robustly represent the ground-truth features. We propose “**rule-based zero-shot generalization**” as a criterion for evaluating whether the model has learned a sufficiently robust representation of the ground-truth features.

- If the model organism truly represents the ground-truth features, then it should generalize perfectly to new input-output pairs. C.f. [2201.02177] Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets
- In particular, the generalization should be in a rule-based way, where the “rules” are the ground-truth “rules” of the domain. DISTINGUISHING RULE- AND EXEMPLAR-BASED GEN- ERALIZATION IN LEARNING SYSTEMS. (See figure below, reproduced from paper)

Figure: Rule-Based vs Exemplar-Based Generalization

- Rule-based generalization involves constructing a symbolic predictor consistent with all inputs and outputs, then transferring that to novel examples.
- Exemplar-based generalization involves memorizing all input-output pairs, and then performing nearest-neighbour lookup at inference time to predict the output.

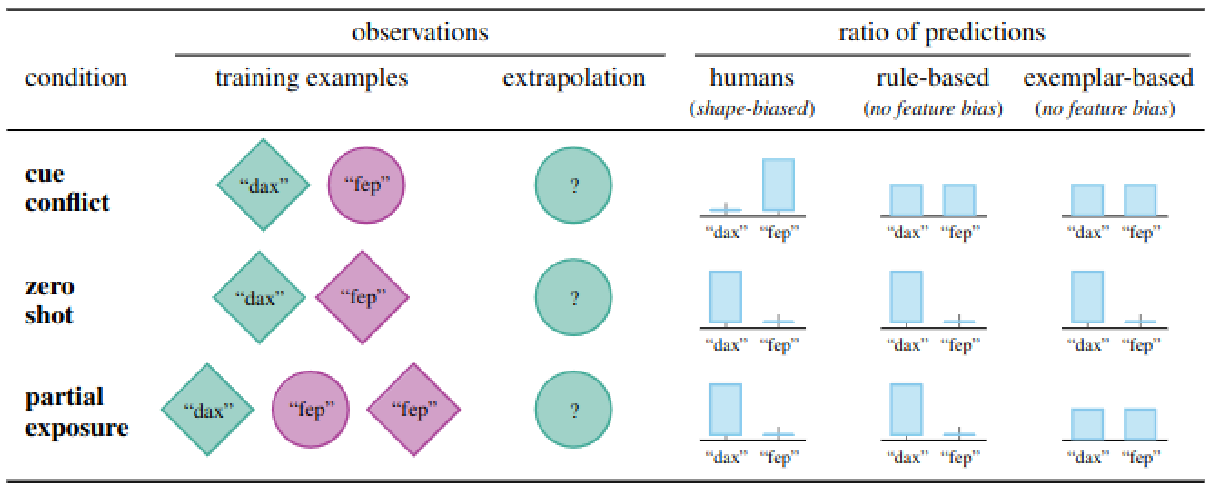

Figure 1: Illustrative category learning experiment: Training examples from the 3 independent training conditions, the extrapolation test, and characteristic behavior for learners with different inductive biases.

Using the Model Organism

Once we have a model organism, we can evaluate an interpretability method by comparing the recovred features with the ground-truth features to compute an error term. If these differ, there may be two explanations

- The model organism may only represent a subset of the ground-truth features in the domain.
- The interpretability method has failed to recover the ground-truth features.

The former case means that we cannot use the absolute error as a metric of an interp method. However, because it affects all interpretability methods equally, we can still use relative error to compare different interpretability methods.

Failure Modes

Here we discuss reasons why this work might fail.

- **It might be too difficult to enumerate all ground-truth features.** One key difficulty will be to fully enumerate (or mostly enumerate) all ground-truth features present in the input space and this will likely entail significant domain knowledge.
    - As argued here, what a "feature" is, is still a very fuzzy concept. Against Almost Every Theory of Impact of Interpretability

    - **Counterpoint: Enumerating a subset of important features might be sufficient** for obtaining a good model organism.
- **Insight gained from the model organism may not transfer to LLMs**
    - If we are considering domains other than language, it is hard to say whether the insights will transfer.
    - Even if we do use language, the subtasks we define may not actually be what the model uses in its representations.
- **Model organisms may be beyond the capability of existing interpretability methods to decipher.**
    - Even if we end up training a model organism that groks its domain, the resulting model might be too large for existing interpretability methods to work, making it less beneficial in the short term.
    - **Counterpoint: There is a lot of research on scaling SAEs right now.** So we might expect this to get solved relatively soon.

Related Work

[2402.04362] Neural Networks Learn Statistics of Increasing Complexity - might offer insight into what "atomic features" could be

[2303.13506] The Quantization Model of Neural Scaling - suggested to me by Alice Rigg but I haven't read it yet

Appendix

Example Failure Mode of an SAE

Suppose we have two binary features X,Y and the data distribution consists of three elements (0,0), (0,1), (1,0). An SAE might learn one feature for each of the three elements, and achieve perfect reconstruction but this does not recover the underlying binary features X, Y.

(Credit to Jake Mendel who initially proposed this example to me.)