There's no shortage of debates and drama in the history of AI.

To give insights, we make some high level discussions.

End-to-End Good, Modular Better

General-Purpose Good, Domain-Specific Better. When fully developed, domain-specific approaches are by definition better (not worse) than general-purpose ones. However problems in AI often are at a large scale and domain specific approaches can't reach its full potential, and general-purpose ones can win thanks to scalability, i.e. the status quo.

What we are trying to do here

End-to-End vs Modular By definition, "modular" is a better property in terms of debuggability, robustness, flexibility. However, under the current deep-learning based AI frameworks, modularity is harder to achieve than end-to-end.

New Programming Language Design We propose a new programming language design that would make AI engineering much more efficient, allowing domain specific models that perform much better than general purpose end-to-end methods.

The language shall address challenges in the three major aspects:

- Inference. The domain specifically optimal inference could involve computation other than matrix algebra, the language shall be able to **implement quickly system level algorithms** that are the building blocks of the overall model; furthermore, queries could be repetitive with each other in some aspects, so the language shall be able to support **incremental computation and memoization**. Lastly, we desire modular rather than end-to-end models, so the language shall support **distribute features across modules and packages**.

- Training. Gradient descent cannot be used because the proper hypothesis class will not be smoothly parametrizable in general (say, each hypothesis is a rule database, or a Rust program). So the language shall allow us to **write sophisticated domain specific (symbolic) searching algorithms for training**. We may ease the optimization difficulty by labeling some intermediate features either manually or through pretrained deep learning models, so language need to **make it trivial to integrate with manual labeling or derived label functions**. It's also possible to adapt the training process, i.e. language need to support **higher order machine learning**.

- Development. In traditional programming, we largely have clear ideas about what we do beforehand and then write code to implement them; however, in domain specific AI engineering, we need to interact to know what to do. For example, we surmise that a certain feature based on a geometric function of curves will work for MNIST,

only to find that it performs good overall except a few cases. We then need to narrow down to those cases and see what's happening. It might be our idea is wrong, or might be the code is buggy, or these cases are just not meant for this function. We need an **interactive, fully debuggable and visualizable IDE system that display inputs, labels, features for either the whole dataset, or a specific subset or just a single point**.

We shall give a more detailed explanation in section ??. But the full specification will be given in a separated paper.

If successful, we shall be able to invent domain specific AI techniques that

- Inference is more efficient, robust and explainable. Could have language models without illusion yet 1000 times faster, runnable on local devices.
- Training. Require significantly less examples than deep learning, close to human for zero shot learning. Use significantly less computation resources due to domain adaptation.
- Development. Deep learning is losing diversity. Only a few places in the globe can train large language models from scratch. Also deep learning defies theoretical analysis, progress is largely empirical. So there's uncertainty about whether it can keep the moment of advancement today in the next decade. However, the new generation of domain specific AI models will be super cheap to train even without GPUs, then everyone can try his/her ideas easily, making the research community much more diverse; one can reason to great depths about inference and training behavior, theories can be applied, progress will be made steadily.

The specifics of these advantages for computer vision and natural langauge processing will be discussed respectively in section ?? and section ??.

By the time of writing, the new language is close to a new working version, and we have yet to prove our points through experiments. So the core argument is through a theoretical framework called **modular pattern recognition graph**. We assume that this framework characterizes the reality, and serve as the foundation for a more general machine learning theory.