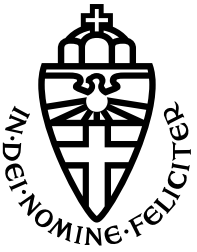
Radboud University Nijmegen



Faculty of Social Science

FlowCoder

A Model of Symbolic Thought Composition via Program Synthesis using Generative Flow

Thesis MSc Artificial Intelligence

Student Information

Surname:	Hommelsheim
First name:	Ron
Student number:	s1000522
E-mail address:	ron.hommelsheim@ru.nl
Course code:	SOW-MKI94 (45 EC)
AI Specialisation:	Cognitive Computing

Supervisor Information

Role:	Supervisor
Surname:	Thill
First name:	Serge
Institute:	Radboud University
E-mail address:	serge.thill@donders.ru.nl
Supervision type:	Internal

Notes

- All agents have in common the ability to pursue goals. - The size of the goal is a way to classify and compare diverse intelligences. For example, a single-celled organism may have the goal of finding food and avoiding predators, while a human may have the goal of building a family or writing a thesis about cognition. - ML uses the term **cognitive light cone** (a reference to physics). The cognitive light cone is a metaphor for the limits of our knowledge and ability to act. It is defined by the set of all events that we can perceive, measure, and model. - The self gradually emerges from being an oocyte, which is "just physics," into a thing that can reason about itself. - This process of self-emergence is closely linked to the scaling of cell and agent organization. - When you scratch a blastoderm, it self-organizes and you get twins or triplets and so forth. - This shows that the number of selves in an embryo is not set up by genetics, but is rather a physiological self-organizational process. - How many selves are in an embryo is not set up by genetics. its a physiological self organisational process - This problem of where do I end and where does the world begin needs to be solved "online" - This means that the self is constantly being updated and renegotiated in light of new information and experiences. - How bodies self-organize is fundamentally the same question as how minds self-organize. All bodies use a multi-scale competency architecture. This means that they are made up of many different levels of organization, from the individual cell to the entire organism. Each level of organization has its own goals and objectives, and these goals and objectives are coordinated to achieve the overall goals of the organism. - Embodiment is the state of having a physical body. It is the grounding of our cognitive processes in our bodily experiences. Our bodies

Formalizing the problem as constructing thoughts out of concepts in a vast state space. This is the general notion of building thoughts out of more primitive components. We don't define the nature of these components here. They could be programs or hyperdimensional vectors, or other stuff. 20

Here we make the case for a probabilistic programming paradigm. 21

Formalizing the problem as program synthesis. 21

At a high-level the approaches we develop in this work follow a 2-stage pipeline: in the first stage a learned model predicts probabilistic weights, and in the second stage a symbolic search algorithm uses those weights to explore the space of source code. Our contributions target the second stage of this pipeline, and we focus on theoretical analysis of sampling-based search algorithms, new search algorithms based on neurally-informed enumeration, and empirical evaluations showing that recent neural program synthesizers can compose well with our methods. 26

the DSL is given by a set of primitives together with their (possibly polymorphic) types and semantics. We describe the machine learning pipeline for program synthesis, illustrated in Figure 1 on a toy DSL describing integer list manipulating programs. The compilation phase constructs a context-free grammar (CFG) from the DSL together with a set of syntactic constraints. The CFG may incorporate important information about the program being generated, such as the n last primitives (encompassing n-gram models) or semantic information (e.g. non- zero integer, sorted list). A prediction model (typically a neural network) takes as inputs a set of I/O and outputs a probabilistic labelling of the CFG, inducing a probabilistic context-free grammar (PCFG). The network is trained so that most likely pro- grams (with respect to the PCFG) are the most likely to be solutions, meaning map the inputs to corresponding outputs. We refer to Appendix A for an in-depth technical discussion on program representations and on the compilation phase. In this work we focus on the search phase and start with defining a theoretical framework for analysing search algorithms. The PCFG obtained through the predictions of the neural network defines a probabilistic distribution D over programs. We make the theoretical assumption that the program we are looking for is actually sampled from D, and construct algorithms searching through programs which find programs sampled from D as quickly as possible. Formally, the goal is to minimise the expected number of programs the algorithm outputs before finding the right program. . 26

the following is from GFlowNet-EM 30

Acknowledgement

Abstract

I posit that the conceptual architecture we create, i.e. the generative model, obides by the same principles as any hierarchical self-organising system, namely discernment about what it is and what it is not.

Contents

List of Figures

List of Tables

Cool words to use:

- explanadum

Go through text and see which words should be operationalised into the glossary

Put all terms in gpt and ask for similar terms

Use plagiarism checker

make sure that the tone of the thesis is the same everywhere. Write it with wit. Make it fun to read.

1 Introduction

This thesis aims to delve into the intricacies of cognitive systems, specifically focusing on [model-based and model-free] approaches, and their potential role in the emergence of the 'self'.

structure to keep in mind:

- 3 (or more, see van Gerven) Levels of Marr
- Computational

  (Building a compositional hierarchical structure)
- Algorithmic

  (Free energy principle, GFlowNet)
- Implementational How can such a system be built in hardware? How can neurons carry out the computations?
- Continuously asking deeper questions as we encounter them. (what is a thought, understanding, etc.)
- draw out the shape of the thesis. kind of a v shape?
- golden circle (why, what, how)
- But & therefore rule of matt stone and trey parker

1.1 Main Contributions

The main contributions of this thesis are:

-

computational Why do things work the way they do? What is the goal of the computation? What are the unifying principles? here we describe the problem as a self-organizing system, building hierarchical generative models.

algorithmic What representations can implement such computations? How does the choice of representations determine the algorithm? Here we describe the FEP and how the discrimination between me and world gives rise to self-organization Here we take the stance of a LOT and talk about the benefits of a symbolic paradigm although this could also be done in other ways, e.g. GLOM

implementational here we talk about program synthesis and combinatorial search, MCMC, RL, DreamCoder, etc.

Other things to clarify

- What is a symbol (also relate to semiotics but also in the cognitive sense. Something that relates to something else, a pointer. Think of memories that are stored in the hippocampus.) what about bioelectrical prepatterns? are all representations symbolic? How would symbols be implemented in the brain?
- Analytic vs. Geometric Solutions as an analogy to approximation vs symbol? imitating understanding vs actual and think about whether there is actually any difference. perhaps a limitation of the symbolic programming approach is that it is rigid and less flexible?

Chapter 1

Computational Description

1 Cognitive Systems

change size

What is a cognitive system?

1.1 The origin of cognition

From Levin Talk on consciousness

- All agents have in common the ability to pursue goals. - The size of the goal is a way to classify and compare diverse intelligences. For example, a single-celled organism may have the goal of finding food and avoiding predators, while a human may have the goal of building a family or writing a thesis about cognition. - ML uses the term **cognitive light cone** (a reference to physics). The cognitive light cone is a metaphor for the limits of our knowledge and ability to act. It is defined by the set of all events that we can perceive, measure, and model. - The self gradually emerges from being an oocyte, which is "just physics," into a thing that can reason about itself. - This process of self-emergence is closely linked to the scaling of cell and agent organization. - When you scratch a blastoderm, it self-organizes and you get twins or triplets and so forth. - This shows that the number of selves in an embryo is not set up by genetics, but is rather a physiological self-organizational process. - How many selves are in an embryo is not set up by genetics. its a physiological self organisational process - This problem of where do I end and where does the world begin needs to be solved "online" - This means that the self is constantly being updated and renegotiated in light of new information and experiences. - How bodies self-organize is fundamentally the same question as how minds self-organize. All bodies use a multi-scale competency architecture. This means that they are made up of many different levels of organization, from the individual cell to the entire organism. Each level of organization has its own goals and objectives, and these goals and objectives are coordinated to achieve the overall goals of the organism. - Embodiment is the state of having a physical body. It is the grounding of our cognitive processes in our bodily experiences. Our bodies are not just passive vessels for our minds; they are active participants in our cognitive lives. - Although embodiment (may be?) is a necessary component of cognition, per-

Perhaps moving from bioelectrical pattern representation of the brain, which is in a sense the first counterfactual, this may be the first primitive version of symbolic? representation, and allude to the brain being a machine to store more complex patterns.

1.2 Embodied cognition and circular causality: on the role of constitutive autonomy in the reciprocal coupling of perception and action

[3] In Varela's words, cognition is *effective action*: action that preserves the agent's autonomy, maintaining the agent and its ontogeny, i.e., its continued development.

There are two hallmarks of a cognitive agent: 1. Prospection, i.e., prediction or anticipation 2. The ability to learn new knowledge by making sense of its interactions with the world around it and, in the process, enlarging its repertoire of effective actions.

1.3 Active Inference

From Maxwell Ramstead Tutorial on active inference

- The brain structure in a sense recapitulates the structure of its environment in which it is encapsulated (the paper Evidence of a predictive coding hierarchy in the human brain listening to speech is related to this) - Paper by Park & Friston shows nested networks: dendritic structure, neurons, brain regions, etc. - These multi-scale systems work in the spatial (microcosm to macrocosm) as well as in the temporal dimension (phylo-, epi-, ontogenetic) - how would we address all these different spatial and temporal scales in a principled way what kind of framework would be able to enable - Most self-organizing systems in nature tend to dissipate. - This means that they consume the gradients around which they organize. For example, a lightning bolt self-organizes around a charge gradient, and in striking, it consumes the gradient around which it's self-organized, effectively leaving the entire system at equilibrium. The same could be said for tornadoes, although in this case the gradient is temperature instead of charge. - The main takeaway is that for almost all systems in nature, self-organization serves to increase entropy. In this context, entropy is a measure of spread. You can think about it roughly as a quantification of how many different configurations the system could be in. High entropy means a high number of available configurations, while low entropy means a low number of available configurations. - roughly 80% of the brain's connections are doing this feedback thing. they are descending so in a sense it would sort of be surprising that 80% of the brain's energy budget goes to something that's just feedback - How does meaning emerge from this? Somehow, meaning emerges from the aggregation of geometric features of stimuli. It's not clear how that's supposed to happen and we call this the *binding problem* - The predictive processing view flips this on its head and says that maybe the brain is mainly engaged in top-down prediction. From this point of view, the main activity of the brain is to produce predictions about what it should expect to sense next. What flows up is the prediction error, which is the difference between what was predicted at a given moment and what is actually sensed. - it's sort of like moving from a conception of the brain is a passive collector of data and a combiner of geometric or statistical information to a view of the brain as a kind of query machine. - In the predictive processing view, perception is like a Google search. You ask a question and you get an answer. A visual cicada is as asking the world a question essentially, so perception is an answer to that question. - Basically, the brain, according to active inference, is busy reversing this arrow. The arrow of causality goes from hidden States to data, and what we want to do is move from the data that we have to an inference of what the most likely causal factors are that caused the data. - At layers above and at the same layer, any unit is receiving predictions about what it should sense. What's going on is a constant comparison between what the brain expects to perceive and what it actually does perceive, and the discrepancy between these two signals is what gets shuffled up the hierarchy. - There are two ways to minimize this discrepancy: 1. Change your model. 2. Change the world. - We have data and we're trying to infer the most probable latent States that cause the data that we're trying to explain. Causality flows in this direction, and inference flows in that direction. - How can we quantify how well a model explains the data? - The way that you typically do this is by constructing several alternative models that each encode a different hypothesis about how the data might have been caused. Then, you evaluate how well that model accounts for the variance in your data. - If you want to compress this algorithm into a quantity, what you get is

variational free energy. This is a measure of how much evidence is provided by the data for a given model of the process. - It's important to stress that free energy is not the same as energy in the thermodynamic sense. It's a measure of how well your model explains the variance in your data. - The reason it's called variational free energy is because it is analogous to the thermodynamic quantity free energy in thermodynamics, where free energy is the amount of energy left in a system that can perform work. In the information theoretic context the variational free energy is basically the amount of wiggle room you still have on your parameters. - A Markov blanket is a way of saying that conditioned on the existence of a set of states, the sensory and active States of the system are independent of the external states of the world. - Active inference then is a story about how internal states (which encode our model) and active States (which are like our skeletal muscles) change to minimize free energy, and the end effect is to allow this inference process to happen. - All of the components of this system are also systems. This is the observation that we started with: I'm a system, I'm an organism, but I'm made of networks of organs that are themselves systems, and the organs themselves are systems of cells, and so forth. - So every component of a Markov blanket is itself Markov blanketed. Cells that share a generative model and over time reach a target configuration. This is basically a little creature with a head and a tail. Everyone sees that, and so what you have plotted here is basically the beliefs of each cell about what kind of cell they are. - They're able to infer their place relative to other cells so long as they're able to commute with other cells, because we all have the same expectations. We all expect to sense the same kinds of things. - This is how you effectively connect levels of organization: units at one level sharing a generative model are able to enact a target morphology.

1.4 Bioelectrical patterns and predictive processing

Bioelectrical patterns are generated by ion channels and electrical synapses, which are found in all cells, including neurons. These patterns can be used to control cell movement, differentiation, and communication.

In the brain, bioelectrical patterns are thought to play a role in a variety of cognitive functions, including perception, memory, and learning. For example, bioelectrical patterns in the visual cortex have been shown to correlate with the perception of visual stimuli.

The predictive processing view of cognition suggests that the brain uses top-down predictions to infer the causes of sensory input. These predictions are generated based on the brain's internal model of the world. The brain then compares its predictions to the actual sensory input it receives. If there is a discrepancy between the two, the brain updates its model of the world to reduce the error.

One way to think about the relationship between bioelectrical patterns and predictive processing is to consider how bioelectrical patterns may be used to generate predictions. For example, the bioelectrical patterns associated with a particular posture could be used to predict the visual input that would be received if the body were to move in a certain way.

The brain could then compare this prediction to the actual visual input it receives. If there is a discrepancy between the two, the brain could update its model of the world to account for the difference. This process of prediction and error correction could allow the brain to learn and adapt to its environment.

1.5 Self-organization and predictive processing

Self-organization is a process by which complex systems emerge from the interaction of simpler parts. The brain is a self-organizing system, and its predictive processing capabilities may have emerged from the self-organization of its neurons.

As the brain develops, its neurons self-organize into networks that are able to generate and test predictions about the world. This process of self-organization allows the brain to develop a model of the world that it can use to make inferences about sensory input.

One way to think about the relationship between self-organization and predictive processing is to consider how self-organization may allow the brain to develop a model of the world. As the brain develops, its neurons self-organize into networks that are able to represent different aspects of the world. For example, some networks may represent the visual world, while others may represent the auditory world or the somatosensory world.

These networks are constantly interacting with each other, and this interaction allows the brain to develop a unified model of the world. This model of the world

can then be used to generate predictions about sensory input.

The identity of a multi-scale system can be thought of as the emergent properties that arise from the interactions of the different levels of organization. For example, the identity of a cell is not simply the sum of the identities of its individual molecules. Rather, the cell's identity is emergent from the way in which the molecules interact with each other to form complex structures and networks.

This is reminiscient of embedding spaces in which words are self-organizing by getting input from the world.

LLMs also create a generative model, but lack the causal structure.

establish the argument for a generative model. Then we talk about the kind of generative model. Maybe introduce Transformers here already.

From [1]: "Self-organization is typically defined as the spontaneous emergence of spatiotemporal order or pattern-formation processes in physical and biological systems resulting from interactions of its components with the environment (Camazine et al. 2001; Seeley 2002, Rosas et al. 2018). Properties of a global higher level system emerge from—and are dependent upon—interactions of its components at the lower level. For example, when the wind blows over a uniform surface of sand, a pattern of regularly spaced ranges emerges (cf. FIG. 1) as a combination of gravity forces and wind speed act on the sand particles (Forrest and Haff 1992). In turn, the surface of the sand determines the flow of air, shaping the reports. Self-organisation therefore entails both bottom-up and top-down causation (Ellis et al., 2011) that is reflected in the circular" "causality implicit in the enslaving principle (Haken & Portugali, 2016) in synergetics or the centre manifold theorem in dynamical systems theory (Carr, 1981)." "However, biological organization is more than emergent complexity. It is fundamentally composed of nested goal-seeking (homeostatic and allostatic) agents, ranging from molecular pathways to whole organ systems and beyond. These not only exhibit complex behaviour, but also specifically act to minimize and maximize various quantities and solve problems by navigating (with diverse degrees of competency) a variety of spaces including transcriptional, physiological, and anatomical spaces in addition to the familiar space of behaviour (Fields & Levin, 2022)."

1.6 Identity and Essence

Here we discuss homeostasis, allostasis — me against the world

Categories

- Prototype Theory (+ prototype, stereotype, archetype) (types and relation to programming?)
- intension extension?
- LLMs
- mereology
- ontology
- substance theory

- process theory
- krakauer information individuality

Computational models of categorization

Prototype models: These models represent categories as a prototype or central tendency of the features associated with a category. They assume that category membership is based on the similarity of an object to the prototype. Exemplar models: These models represent categories as a set of stored examples or exemplars. They assume that category membership is based on the similarity of an object to the stored exemplars. Feature-based models: These models represent categories as a set of defining features. They assume that category membership is based on the presence or absence of specific features. Connectionist models: These models represent categories as patterns of activation in a neural network. They assume that category membership is based on the pattern of activation that results from processing sensory input. Bayesian models: These models represent categories as a probability distribution over features. They assume that category membership is based on the likelihood that an object belongs to a particular category given its features.

What is the point here? what do I think? I think categories, similar to any self organizing system maintain a boundary of what they are and what they are not. they themselves can be thought as agents. We have the concepts but our perceptions are shaped ,mostly by evolution and society. Our subjective interpretation is actually minimal. Our perception of objects strengthens in concordance with the crowd, the society. So these concepts, which are sort of memes can be seen as agents themselves. They self-organise, and have a mapping to the actual underlying brain structure which is a mapping of the environment, to solve certain problems. a concept is the minimization or optimization of free variables in an active inference frame. This relates to gibsons affordances. so concepts might be formed also by function, or the relation of how i could interact with the concept. but concepts dont always have to be functional. this is also related to meaning. the function of a concept is its meaning. in that way it is the set of inferences one could make from said concept. of course it doesnt mean that it is meaningful, which is subjective to the observer and dependent on personal preference and goals. concepts are agents that serve a goal. notice how concepts have different levels of granularity, according to our interaction with them.

1.7 What is a thought?

- Explain the idea of a thought as a trajectory through semantic space.
- How does this trajectory look like in LLMs? (Wolfram)
- How does this trajectory look like in GFlowNet? (perhaps operationalised as a Markovian trajectory?)
- How does a thought look like in enactivist/ purely reactive or reflexive cognitive frameworks.
- Thought as constructing probabilistic programs
- What does it mean to have a model?
- What does it mean to compute?
- Can any concept be definitively defined as a particular concept (either symbolic or symbolic vector e.g. SPA, hyperdimensional vectors) or are all concepts approximations?

1.8 What does it mean to understand?

What does it mean to truly understand something?

I would say that understanding requires causality, not just correlation.

Is it possible for artificial intelligence to grasp the essence of a concept like “body” or “mass”? It can approximate to an arbitrary degree but you can’t square the circle, i.e. approximations at the limit do not become the thing approximated, they are just treated as such. This is related to joscha bach’s interpretation of Gödel’s theorem, that it cannot be implemented since there are no infinities in physical reality. Anyways, approximations might be enough, and might be what we’re doing too. This is important for the symbolic vs connectionist debate.

GPT learns through the context in which words and phrases appear, essentially creating a concept network of words. The AI analyses this abstract relational structure to generate responses. Yet, we might question whether this process equates to understanding.

The conundrum is reminiscent of the mary’s room thought experiment, where Mary, despite having all the data about colour, has never truly experienced it. If she were to experience colour one day, would she glean

any new information that was not already available in the data she had?

The question seems to probe whether phenomenological experience can be simulated through abstract data. While it's possible to argue that a blind man will never truly see just by reading about it, we must also consider that our experiences are essentially the result of neurons firing in our brains. If we possess the necessary cognitive mechanisms, it's plausible that phenomenological experience could be replicated by alternative means.

- are human's capabilities (or LLMs) just a bunch of bag of tricks, masquerading in a trenchcoat.

1.9 Part-Whole Hierarchies/ Holarchies

- argue thoroughly why the conceptual framework is built as a holarchy
- We agree on an assumption that we construct a model of the world.
- What exactly would an explicit model or an implicit model or no model at all look like?
- We agree that there are certain hierarchical relationships. [example book, music, ontology, etc.]
- Here we can talk about conceptual structures, holarchy, part-whole hierarchies.
- How concepts are split with more attention, etc.
- Do the upper levels of a holarchy control the lower levels? I.e. Does the holarchy become a heterarchy?
- Heterarchy, sand example in nested organism pregnancy friston paper, DNA, GEB
- Multi-scale systems
- Ontology
- Mereology? and relation to Category theory?
- This is essentially learning a
- hierarchical LVM
- hierarchical graphical model
- Parse tree
- Can be formalised as Context Free Grammar.
- The grammar here is the generative model.
- What is a generative model?
- A generative model is simply a probabilistic description of how causes (i.e., latent states) generate consequences (i.e., data or sensations). [4]

Shared competencies, or similar strategy between compositional hierarchical structures in organisms and in the construction of concepts/ thoughts

From "Building machines that learn and think like people

Compositionality pertains to the formation of new representations through the combination of simpler, primitive elements [**?**].

Compositionality is central to cognitive productivity and learning. It allows for the creation of infinitely varied representations from a finite set of basic elements, similar to the mind's capacity to think an endless array of thoughts or generate countless sentences. This is particularly useful in forming hierarchical structures that simplify complex relationships, thus making inductive reasoning more efficient.

1.10 Correlation Does Not Imply Causation

Generative models assign probability distributions, representing observations in such a way that they can generate new data points similar to the observations. The mapping between the probability distributions and the data is correlational. They can therefore be described as *correlational models*. Causal models in contrast, represent hypotheses of how observations were generated. They do not just represent the observations by correlation, but aim to accurately reflect the mechanisms by which the data originated.[source] Significant challenges persist in deducing latent causal variables. [source]

Another crucial aspect of human cognition is the ability to acquire new knowledge and skills more quickly and easily by leveraging prior knowledge and experience. We can decompose situations quickly into basic components, e.g. when approximately understanding the anatomical composition of humans, we can quickly find the essence and apply it to mammals. We can parse scenes and situations into parts and generalize to other circumstances. This is related to surfaces and essences. Similarly to organisms which are

composed in a nested hierarchy, each level of the nested hierarchy solves problems in their own domain without the need to being micromanaged from the level above, i.e. the process of self organisation happens at each level independently, so similarity it is crucial that learning to learn happens at multiple levels of the holarchy. For example, a machine that learns a compositional representation of a car can understand that a car is made up of parts such as wheels, an engine, and a chassis. This knowledge can then be used to learn new things about cars, such as how to drive a car or how to fix a car.

Using GFlowNet we can make inferences about intermediate variables. This is because we approximate probability distributions and not point estimates.

argument for having two systems 1, 2

1.11 Thinking fast

[**?**] "4.3. Thinking Fast The previous section focused on learning rich models from sparse data and proposed ingredients for achieving these human-like learning abilities. These cognitive abilities are even more striking when considering the speed of perception and thought: the amount of time required to understand a scene, think a thought, or choose an action. In general, richer and more structured models require more complex and slower inference algorithms, similar to how complex models require more data, making the speed of perception and thought all the more remarkable. The combination of rich models with efficient inference suggests another way psychology and neuroscience may usefully inform AI. It also suggests an additional way to build on the successes of deep learning, where efficient inference and scalable learning are important strengths of the approach. This section discusses possible paths toward resolving the conflict between fast inference and structured representations, including Helmholtz machine–style approximate inference in generative models (Dayan et al. 1995; Hinton et al. 1995) and cooperation between model-free and model-based reinforcement learning systems."

- Efficient inference over hierarchical Bayesian models is still difficult.
- Approximate inference
- Here we can go into models monte carlo e.g. etc.
- Einstein : intuition, rationality
-

argument for hemispheric lateralization

- Broad vs narrow attention. One sees the trees, the other the forest.
- the left wants to jump to conclusions and is much more quick and dirty, while the right says hang on
- left deals with things that are known, when something isn't, its better to leave it to the right until it can be categorised by the left
- things in the left are more isolated, while in the right everything is connected to everything else
- left is static, right is flowing and changing
- the left abstracts, the left is more interested in categories, the right in the unique case
- the left sees things as inanimate, the right sees them as animate
- as we age we use the representations of what we know increasingly, neglecting the world. We become more solipsistic. This makes sense because perceiving things for the first time is computationally heavy and expensive, so the things we know from experience will usually be good enough.

Often, the most obvious things, once unpacked, reveal the most [examples?]. Our most basic perceptions and assumptions are so deeply integrated into our cognitive processes that they become invisible to us, essentially automatic. When driving the same route everyday, people may arrive at their destination without conscious awareness of it. Tasks become automated. [show studies, other examples like piano.]

From a computational standpoint, one could say that these deeply embedded perceptions function much like 'compiled code' in a computer program — efficient and fast, but not easily inspected or altered. These perceptions are model-based representations optimized for computational efficiency, trading off flexibility for speed. They exist as pre-computed "shortcuts" that enable us to interact with the world without incurring high computational costs each time we encounter a familiar situation.

In machine learning, this relates to the trade-off between model-based and model-free learning. Model-free approaches require higher computational costs upfront but are more adaptable. Model-based strategies

offer efficient but rigid ways to interact with the world. Both strategies have their pros and cons, reflecting a trade-off between computational efficiency and cognitive flexibility. [source]

As we age, we rely increasingly on these internal representations instead of engaging directly with external reality, making our view of the world more solipsistic. [source]

This phenomenon can be explained by the increasing reliance on "cached" or "memoized" cognitive models, which offer a computationally cheaper way to navigate reality. It's a form of cognitive "greedy algorithm," opting for local optimizations based on past experience rather than recalculating the optimal path each time. [source + relation to predictive coding paradigm]

This concept parallels the notion of "overfitting" in machine learning, where a model becomes so tailored to the training data that it loses generalizability. In human cognition, an over-reliance on established mental models could result in decreased adaptability and an insular worldview, effectively isolating the individual from the ever-changing external environment.

As we automate more cognitive functions—either through learned habits or technological aids—we engage less in "active sampling" of the world, reducing the richness and nuance in our perceptions. This leads to a form of "lossy compression" of reality, where only the most salient features, according to our models, are retained.

- Problems should be solved at the lowest possible level of the hierarchy. Larger goals require more sophisticated organization. - How does a system recognize that it cannot solve it alone and that it needs to recruit other cells? Think of the computational boundary. Perhaps we must be able to estimate computational complexity. - This compression of sensory information into concepts may be the origin of symbolic processing.

- Things we don't have concepts for don't exist. We can't describe them in our language. [related to joscha about truth.]

1.12 From Language to Consciousness: JB (Check Craft note)

- Concepts are the address space of mental representations. They are merely pointers. - There's a hierarchy (holarchy) of abstraction of mental representations - The least abstract thing are impulses that come from sensory neurons. - Basically, discernible difference, which is how we define information, is the closest to physical reality. - We organize them into features. features describe how reality changes in the aggregate from moment to moment. - Features are arranged into objects - Objects remain stable if the perspective changes - Concepts abstract over all objects that belong to a group - the extension of a category - These concepts are organized in an embedding space

Conceptual space (relate this to Gordenford)

- relationship path
- mutual information (how well do concepts predict each other, co-occurence in reality)
- change distance
- parameter distance (related to hofstadter mathemagival themas)

Meaning: You establish the relationship between pattern and the function that describes the universe, your conceptual model, and that is meaning. Meaning is your entire mental universe. It's the unified model of reality that your mind is constructing. So for any new thing, if we can establish a relationship between the thing and our model, it has meaning. If we can place it in our conceptual framework, it gets meaning.

difference between what is meaning and what is meaningful, maps of meaning. what is meaningful is what guides you. its a gradient, a heuristic. Show studies that deeper understanding improves memory and meaning.

1.13 Free-Energy Principle

- Discernment of Concepts
- Me against the World
- Homeostasis
- Allostasis
- Operational Closure
- Organizational Closure
- Intentional Stance
- Active inference
- Explain Bayes theorem and how it relates

$$\underbrace{F(s,\mu)}_{\text{Free Energy}} = \underbrace{D_{KL}\left(q(\psi|\mu)||p(\psi|a)\right)}_{\text{Complexity}} - \underbrace{E_q\left[\log p(s|\psi,a)\right]}_{\text{Accuracy}} \tag{1.1}$$

where the complexity is the Kullback-Leibler divergence between the true posterior and its approximation. The Accuracy is essentially the expected negative log likelihood (?)

The FEP states that systems minimize their free energy by updating their internal models and taking actions that are consistent with those models. This can be done by:

Updating the internal model: Systems can update their internal models based on sensory feedback. This is done by comparing the predicted sensory input to the actual sensory input and adjusting the internal model accordingly. Taking actions: Systems can also minimize their free energy by taking actions that are consistent with their internal models. This is because actions that are consistent with the internal model are more likely to lead to sensory input that is consistent with the internal model.

[4] Crucially, free energy is a functional (i.e., a function of a function) of two quantities. First, sensory data and a probability distribution over the unobservable states generating those data. This variational density is taken to be encoded, represented, or parameterised by the internal states of any system, ranging from a particle to a person. On this view, perception corresponds to changing internal states to minimise the divergence between the variational density and the posterior density over latent states, given observations. Conversely, action can change the way that data or sensations are sampled—to ensure that they provide the greatest evidence for the generative model entailed by an agent. This dual aspect—to optimising free energy—gracefully accounts for action and perception, where both are in the service of maximising (a variational bound on) marginal likelihood. In general, there are three levels of optimisation under the free energy principle. These correspond to the unknowns (i.e., latent causes) in the generative model. These unknowns comprise (i) latent states generating outcomes, (ii) model parameters encoding contingencies and statistical regularities and, finally (iii) the form or structure of the generative model. Each is equipped with variational density (i.e., a Bayesian belief) that is parameterised by the (i) states, (ii) weights, and (iii) structure of the agent at hand. At the fastest timescale, inference can then be read as optimising the states (e.g., synaptic activity) to optimise variational free energy. This is usually cast in terms of a gradient flow on free energy. Crucially, the gradients of free energy can almost universally be cast as prediction errors. The second set of unknowns are the parameters of the generative model, encoded in slowly changing weights (e.g., synaptic efficacy).Finally, we have the structure or form of the model, e.g., cortical hierarchies in the brain (Mumford, 1992). The structure of the model can be regarded as being optimised with respect to free energy or model evidence via a process of Bayesian model selection; namely, selecting those models with the greatest marginal likelihood, as assessed over an extended period of time. This level of optimisation manifests at different scales. For example, one can construe natural selection as nature's way of performing "Bayesian model selection—i.e., accumulating evidence about an econiche by selecting phenotypes that have high adaptive fitness or marginal likelihood (Campbell, 2016; Frank, 2012). At a somatic timescale, in biology, this could be regarded as neurodevelopment with (epigenetic) hyperpriors over model structure. In cognitive science, this kind of optimisation process is often referred to as structure learning" Above, we divided optimisation into inference, learning and model selection. However, a finer grained analysis of inference calls for a consideration of the representation of uncertainty. If one subscribes to the free energy principle, then optimisation corresponds to optimising posterior or Bayesian beliefs (or their sufficient statistics). This means that it is not sufficient to use point estimates of various quantities, the precision or inverse dispersion (i.e., negentropy) of these beliefs also has to be optimised. Sometimes this is a more difficult problem that estimating the average or expectation of an unknown

here he says that point estimates are not enough.
so MLE over MAP

this relates to gflownets representing the posterior

- log model evidence can be decomposed into accuracy and complexity. The complexity of a generative model corresponds to the kullback leibler divergence between posterior and prior. in other words the effective number of parameters or degrees of freedom that are required to accurately account for some data and its sampling. "Optimising free energy, therefore, puts pressure on finding the simplest explanations and mod-

els" This is exactly the same idea that underwrites the minimisation of algorithmic complexity in the setting of minimum description or message length schemes discrete actions solicit a sensory outcome that informs approximate posterior beliefs about hidden or external states of the world – via minimisation of variational free energy under a set of plausible policies (i.e., perceptual inference). The approximate posterior beliefs are then used to evaluate expected free energy and subsequent beliefs about action (i.e., policy selection). Note a subtle but important move in this construction: the expected free energy furnishes prior beliefs about policies. it means that agents infer policies and, implicitly, active states. In other words, beliefs about policies – encoded by internal states – are distinct from the active states of the agent's Markov blanket. In more sophisticated schemes, agents infer hidden states under plausible policies with a generative model based on a Markov decision process. This means the agent predicts how it will behave and then verifies those predictions based on sensory samples. In other words, agents garner evidence for their own behaviour and actively self-evidence. Another technological idea is the 'game engine in the head', i.e., the use of very fast approximate simulators for graphics physics and planning, which can simulate complex physical situations in realistic ways and use those as a prototype for the model in the agent's head. It is something like an approximat ion to the common-sense systems of understanding the world, that evolution has built into our brains, and that even young babies can use to explore the world.

[5] "The free energy principle originated from the work of von Helmholtz on 'unconscious inference' [57], postulating that humans inevitably perform inference in order to perform perception. This implies that the human perceptual system continuously adjusts beliefs about the hidden states of the world in an unconscious way." ([Mazzaglia et al., 2022, p. 3]

"According to the free energy principle, in order to minimize free energy, the agent learns an internal model of potential states of the environment." "Crucially, these internal states do not need to be isomorphic to the external ones, as their purpose is explaining sensorial states in accordance with active states, rather than replicating the exact dynamics of the environment. Isomorphism, in this context, refers to considering a structure-preserving mapping of the state space."

Because of these assumptions, the concept of 'reward' in active inference is very different from rewards in RL, as rewards are not signals used to attract trajectories, but rather sensory states that the agents aims to frequently visit in order to minimize its free energy

Training the model is then typically alternated with collecting new data by interacting with the environment, using the model for planning [49], or using an amortized (habitual) policy [67, 33]. In practice, one can also train a model upfront using a dataset of collected trajectories from a random agent [68] or an expert [32]. The latter is especially relevant in contexts where collecting experience online with the agent can be expensive or unsafe [69].

1.14 generative models

- show representations in human brains and argue why the brain may largely be a generative model.

1.15 hierarchical generative model

- amortized sampling tutorial

1.16 Organisms across scale and multi-scale cognition

- Cellular Automata
- Lenia
- (Pure) Enactivism?

1.17 Language in Thought

- Syntax & Semantix
- Jabberwocky sentences (would be equivalent to a sentence that doesn't minimize surprise well (given the task, because it could also be a goal))
- Sapir-Whorf
- Why do we need language
- Does meaning emerge in LLMs (is meaning the network or do we need grounding)
- Grounding
- Binding problem

1.18 LLMs

- Self-Attention
- Successes
- Limitations
- relation to part-whole hierarchies, FEP, grounding, etc.

Limitations

from JB: LLMs learn how to complete sequences, they do statistics over patterns patterns → syntax → style → semantics they deep-fake so well that the outcome becomes indistinguishable from what we do. thats why its said to brute force. Minds learn how to understand we do it the other way around semantics → patterns → syntax → style

1.19 Human's capabilities

Humans naturally exhibit the aptitude to innately evaluate the reliability of our knowledge. Understanding the limits of one's knowledge directs the acquisition of new information, thus optimizing the learning process.

Moreover, human cognition is characterized by modularity and reusability. Our mental architectures are adept at disentangling knowledge into modular components, which can be recalled, combined, and repurposed to address novel challenges. By continuously crafting and refining abstract concepts, humans can process, assimilate, and communicate multifaceted knowledge structures with remarkable efficiency.

Further, we intuitively discern causal relationships, allowing for predictive and counterfactual reasoning. Causal models, seen as extensive distribution families influenced by interventions, provide the scaffolding for this intuitive understanding. As highlighted by Kemp et al. (2015), these models enable humans to extrapolate beyond observed data, envisage hypothetical scenarios, and navigate the world with a goal-oriented mindset. In computational terms, integrating causality can tremendously amplify the generalization capabilities of algorithms, allowing them to predict outcomes under novel interventions and facilitating deeper understanding of intricate systems.

Often there are multiple solutions to a query, which is why the inference network should model a multimodal distribution rather than a point estimate. Difference between MAP and MLE.

- Throw in some definitions for intelligence. (+ Analogy)
- Out-of distribution data, imagination, simulation
- Types of questions [check note: categorizing questions] (equivalent to inference types?)

Operationalising "Understanding"

- Reverse Engineering: Is understanding being able to reconstruct the thing we want to understand? If you can break down an object or concept and then reconstruct it, perhaps it implies you understand it.
- Modelling: Understanding might be proportional to the model we create of a concept. The more accurate and comprehensive our model, the greater our understanding.
- Utilization: It could also be the degree to which we can use a concept. If we can apply it effectively, we could claim to understand it.
- Maybe understanding is the size/ strength of a concept? Imagine it as blobs in an abstract space, a large blob with many connections would be understood, whereas a new little blob with little connections is not.
- Children use language without knowing what the words actually means. The meaning is the context.
- People can even teach without understanding.
- Fake it 'til you make it?
- Is the limit of imitation a clone? ()

1.20 System 1 and 2

- Slow, Fast
- Reflexive vs Reasoning (reflexive is the direct response of the generative model, while reasoning is a higher level model, essentially a model over the reflexive part (this is also why we are not conscious over reflexive actions. Think e.g. how we don't notice driving a familiar road))
- Transformers are correlational, we need to build causation. this is the same as reflexive vs reasoning.

- does, or when does, correlation approximate causation? Are humans correletional (reflexive but really complex) or causal? Wouldn't causality mean that we have free will?

Poverty of the Stimulus

The "poverty of the stimulus" argument is an argument that is often made in the field of linguistics and cognitive science. It argues that children are able to learn language despite being exposed to a relatively limited amount of data, which suggests that they must have some innate knowledge or ability that helps them learn language.

The argument is based on the observation that the linguistic input that children receive is often incomplete, ambiguous, or inconsistent. For example, children may hear sentences that contain errors or that are not grammatical. Despite this, children are able to learn the rules of their language and use them to produce and understand sentences that they have never heard before.

One possible explanation for this is that children have innate knowledge or abilities that help them learn language. For example, it has been proposed that children have an innate knowledge of universal grammar, which is a set of grammatical principles that are common to all human languages. This knowledge enables children to quickly learn the rules of their language, even when the input they receive is incomplete or ambiguous.

The poverty of the stimulus argument is controversial, and it has been challenged by some researchers who argue that children are able to learn language by relying on general cognitive abilities, such as statistical learning or pattern recognition. However, the argument continues to be an important topic of debate in the field of linguistics and cognitive science.

1.21 Hemispheric Lateralisation

- notion that if we don't have a concept for it, it does not exist for us. I.e. even if our sensory equipment can in principle take it in, and does take it in, it is not there for us. It seems we are unable to attend to it. Perhaps attribute meaning to it and therefore it doesn't enter our conscious awareness.

1.22 Language of Thought

One of the most fascinating undertakings of Artificial Intelligence (AI) is its objective to reveal and learn from the mechanisms of human cognition. – The idea that the thoughts are constructed by some kind of Language of Thought (LOT), in order to understand and represent our reality has a long history. Gottfried Leibniz imagined a *Characteristica Universalis*, a formal language capable of expressing metaphysical concepts [6]. Jerry Fodor, among many other scholars, developed this hypothesis, and proposed that thoughts are composed according to some form of internal grammar [7]. If our thoughts are indeed composed by some language, we can let our intuitions about its structure and limitations be guided by the advances made in the past century in regards to formal languages and computationalism. Gödel showed that any formal system strong enough to express Peano Arithmetic is incomplete [8]. Turing showed that any possible computation can be done by a Turing machine and concluded that the human mind must be computational [9]. [Piccinini disagrees] Chomsky's Hierarchy classifies formal languages of different strengths and the automata that recognize them [10]. These results and many more are consequential when thinking about a language of thought. Many important questions about concepts are still left unanswered, e.g. regarding their formation, the structure of the framework they reside in, their representation, how they connect to other concepts and so on. Many contemporary versions of the LOT hypothesis (LOTH) argue that our conceptual framework is constructed bottom-up using some initial primitives. More so, the language we construct concepts in is executable, akin to a programming language [11]. The idea is that we model the world by creating programs that generate our observations [12]. Roumi et al.'s experiment indeed suggest that humans may interpret and compress sequence regularities in a type of program of minimum description length (MDL) [13]. Dehaene et al. further test this hypothesis and find similar results [11]. The question then remains, where primitives actually come from. Piantadosi asserts that symbols have no inherent meaning, instead, meaning emerges from the dynamic relations between symbols. I.e., only the conceptual role and the dynamics of the symbols define meaning [14]. By defining meaning as an emergent phenomenon of conceptual roles, Piantadosi and Hill do not rule out that large language mod-

els (LLM) already have some foundation of meaningful concepts [15]. Recent work tries to relate these ideas to biological systems, e.g. Quan et al. propose how role-filler interactions could be implemented in the brain via temporal synchrony to permit symbolic processing and dynamic binding of variables [16]. Santoro et al. argue for a semiotic approach, assert that symbols are subjective, and propose that meaning of symbols in artificial systems will arise when they are subjected to socio-cultural interactions [17].

Ellis et al. use λ-calculus, which is Turing complete, to build something akin to a LOT [18]. In response, Piantadosi shows that combinatorial logic is equivalent to λ-calculus and why combinatorial-logic-like (LCL) language is preferable, for one because it doesn't assume primitives, thereby avoiding the problem of their meaning and origin. One of the questions regarding a LOT is whether thoughts can be produced by simple syntactic symbol manipulation. I (and many others) suspect that semantics are not just an emergent phenomenon but that it actively shapes our perception and concept formation [17, 19]. Moreover, purely symbolic attempts, albeit being the original approach of artificial intelligence, have proven to be insufficient [20]. Instead, the successes of modern AI are owed to the advances of connectionist models. These however, have their own deficits, namely, being incredibly data-hungry, lacking causality, and lacking out-of-distribution generalization, to name a few. Therefore, new approaches try to combine these two AI strands into what is known as *neurosymbolic AI* [20].

- What is the LOT in relation to the generative model?
- Is the LOT the policy over the gen model?
- do we need them to be distinct? (PCFG vs Q), Bengio
- Relation to self-organization, active inference, FEP, autopoiesis, etc.
- show JBs argument of LLMs learning from
- From Donald hoffmann and joscha bach: consciousness , etc. .. : proof cannot reach truth. godel found that there is truth that cannot be found with mathematics. but the opposite is true. there is no deeper notion of truth than proof. perception cannot be true or false. it just is. physical events cannot be true or false, a pattern you observe isnt true or false. the pattern itself is an interpretation. its hermeneutics. in order for something to be true or false you need a language which needs to be defined such that truth can be established. And the process of establishing truth is a computation. There are two types of languages in which truth can be defined: classical mathematics is stateless and thus timeless. it lacks the temporal dimension. this allows you to create functions that take infinitely many args in a single step. but in a computational system you cannot assign a single digit to pi. here, in a language with steps, we are losing the ability to treat pi as a value. it is now a function we can only approximate to a certain degree. we get a fundamental difference between a value and a function. (relate this to the universal function approximator: can we approximate downstream functions? perhaps thats what we need symbols, or grounding for). this means that also truth changes. its no longer this platonic thing that exceeds mathematics . it has to be contingent on the language that uses it. if the language has internal contradictions, truth becomes impossible to determine and you can never prove statements that cannot be described in your language (extrapolation).

  In essence it means that we can define languages which do not align with the real world, but are above it. we need to define languages of the same level as actual reality.

- what is a symbol and what is a symbolic architecture. symbolic here literally means symbol manipulation. context free grammar, etc. logical predicates. DC makes it neurosymbolic by using nns. I make it even more connectionist by embedding the symbols.

Free will: if i can out model you, i.e. i can predict every action you do,

Neurosymbolic AI

Our conclusion from the above discussion is that in neurosymbolic AI: • Knowledge should be grounded onto vector representations for efficient learning from data based on message passing in neural networks as an efficient computational model. • Symbols should become available as a result of querying and knowledge extraction from trained networks, and offer a rich description language at an adequate level of abstraction, enabling infinite uses of finite means, but also compositional discrete reasoning at the symbolic level allowing for extrapolation beyond the data distribution. • The combination of learning and reasoning should offer an important alternative to the problem of combinatorial reasoning by learning to reduce the number of effective combinations, thus producing simpler symbolic descriptions as part of the neurosymbolic cycle. [2]

Formal Grammar

- Syntax vs. Semantix
- Semiotics

1.23 Computational Mind

- RNA computation
- Church-Turing Thesis
- Difference in computation between brain (Piccinini), ANNs and symbolic processing

1.24 Some approaches

- GLOM also related to NCA and to holarchies
- Kissner
- JEPA
- etc.

Constructing compositional structures

1.25 Compositional world model and inference machine

In a language of thought, we make the assumption that thoughts are compositional. In one way or another, thoughts are hierarchical (?). Both in ontologies, i.e. the way concepts are structured, (animal - bird - fink) but also in the sequential nature of thought construction (as in natural language, reasoning tasks, etc.). We are creating parse trees, or abstract syntax trees.

- but there is a difference between ontology, i.e. the structure of all concepts vs the sequential construction of thoughts.
- so what are we simulating here?
- One is the generative model. the world model. the other is the inference machine.

It seems that we have essentially two problems

1. Constructing a conceptual framework, with relationships of all concepts.
2. Navigating this conceptual framework.
3. maybe the generative model is correlational, while the inference model turns that into causality. (see beyond the fep internalism externalism, ramstead)
4. I think having a sense of self, i.e. a self-referential story, involves this inference model that builds causal structure, a narrative (see [21]).
5. Is this related to interpolation vs extrapolation? (is it the same in high dimensions? chollet?lecun?)

We want to find out

1. Where the self emerges.
2. What is meaning and what is meaningful

- Do we have one generative model or multiple competing models? Or do we just have competing representations at the frontier and it is more difficult to change beliefs deeper in the conceptual space, where concepts are foundational and others rely on?
- Jerome Busemeyer uses Quantum formulations to show how we are in a super-position of beliefs until we collapse them upon evidence.

The renowned professor Yoshua Bengio explains that thoughts are composed of multiple intermediate steps and explains that reasoning, or thinking, is the combination of knowledge and inference. He theorises that we construct a world model which may take the form of an energy based model.

It is interesting to think about whether this world model is explicit or implicit. In DreamCoder and other

SOTA methods in program synthesis the world model is essentially an explicit PCFG from which one samples programs. In neural program synthesis, the new approach of neurosymbolic programming, the role of the neural network is to act as a recognition model, helping to search programs from this PCFG. I imagine it a little different, and see the thought construction process more like an agent, following a policy.

On one side, in alignment with Occam's razor, our world model aims to identify the essential knowledge fragments required to interpret observed data. This model should be concise, optimizing the encoding bits and enhancing the reutilization of knowledge segments, thereby enhancing its generalization capability to unfamiliar scenarios. This aligns with the principle of parsimoniousness or, as Dehaene describes as finding programs of minimum description length.

A trained inference machine amortizes the cost of searching through a vast solution space which resource-intensive search methods like MCMC, traditional AI techniques, or optimization algorithms cannot handle. In conclusion, approximate inference could be the solution here, since a larger model doesn't lead to overfitting but better approximations, while not being dependent on lots of data.

Yoshua warns that a notable vulnerability of current large language models is the conflation of the world model with the inference mechanism. Although LLMs may construct an implicit world model, the distinction between world model and inference may be the underlying cause of their difficulty for out-of-distribution prediction [22, 23].

- RL
- MCMC
- variational inference
- amortized variational inference
- the transcendentals as the axioms of ontology
- proto-concepts
- concepts vs no concepts
- concepts as pointers
- normalizing over sensory data -¿ features -¿ objects -¿ concepts
- Concepts could be "chunked" into hyperdimensional vectors

1.26 Problem Statement

How do parts combine to solve shared goals?

Formalizing the problem as constructing thoughts out of concepts in a vast state space. This is the general notion of building thoughts out of more primitive components. We don't define the nature of these components here. They could be programs or hyperdimensional vectors, or other stuff.

- Combinatorial search problem in which compositional latent hierarchical variables need to be found
- Amortized inference/ sampling
- GFlowNet?
- We want to show that the overall objective is to learn a model that samples proportionally to a given reward distribution
- Formalise the GFN approach
- Talk about induction, deduction, abduction and how sampling from a reward distribution is similar to abduction aka inference to the best explanation
- add markpoels 7 criteria

Chapter 2

Algorithmic Description

1 Probabilistic Programming Paradigm

1.1 Game Engine in the Head

Here we make the case for a probabilistic programming paradigm.

> "All I ever wanted was to pick apart the day, put the pieces back together my way." - Aesop Rock

- Reverse engineering the world
- Intuitive physics etc.
- Representing the world via probabilistic programs
- Dehaene
- Simulation and Simulacrum
- Is the GEitH the same as a generative model?
-

Formalizing the problem as program synthesis.

Chapter 3

Implementational Description

1 Program Synthesis

CFG

1. Background and Introduction

The field of programming has always been underpinned by the intricacies of formal grammars. Context-Free Grammars (CFGs), a subset of formal grammar, are essential in defining the syntactical structures of many programming languages. However, given the generative nature of CFGs, the potential program space defined by even a modestly complex grammar can be immensely vast. Searching for a specific program within this space, or ensuring that a particular space is sufficiently explored, poses significant computational challenges.

2. Problem Definition

2.1 CFG and Program Space

Let $G = (N, \Sigma, P, S)$ be a Context-Free Grammar, where:

- N is a finite set of non-terminal symbols.
- Σ is a finite set of terminal symbols with $N \cap \Sigma = \emptyset$
- P is a finite set of production rules, where each rule is of the form $N \to (N \cup \Sigma)^*$
- S is the start symbol, with $S \in N$

Given such a CFG, the derived program space $\Pi(G)$ is the set of all possible strings (or sequences of symbols) derivable from S.

2.2 Problem Statement

Given a Context-Free Grammar G and a defined objective function f that maps any program $p \in \Pi(G)$ to a real value representing its desirability or fitness:

Find p^* such that:

$$p^* = \arg\max_{p \in \Pi(G)} f(p)$$

In other words, the problem is to locate a program p^* within the vast program space $\Pi(G)$ defined by G that maximizes (or, alternatively, minimizes) the objective function f.

3. Challenges and Complications

3.1 Size of the Search Space

The generative capacity of CFGs means that even grammars of moderate complexity can define immensely vast program spaces. The sheer size of these spaces poses computational and search challenges.

3.2 Non-Linearity and Discontinuities

The mapping between programs and their fitness as defined by f might be non-linear with multiple local maxima, making search strategies based on gradient ascent or other linear heuristics suboptimal.

3.3 Generalization vs Specification

While CFGs provide a generalized representation of possible programs, the objective function might lead to highly specialized solutions. Balancing between the two is non-trivial.

3.4 Syntactic vs Semantic Validity

A CFG ensures syntactic validity but does not guarantee semantic correctness. Ensuring that a program derived from a CFG is semantically meaningful or error-free in a given context is an additional layer of complexity.

- How are programs represented (ASTs, GNNs, other ideas from different papers)
- Tree-Transformers
-

Rewrite the following: Program synthesis is a notoriously challenging problem. Its inherent challenge lies in two main components of the problem: intractability of the program space and diversity of user intent.

Statistical Various kinds of statistical techniques have been proposed including machine learning of

probabilistic grammars, genetic programming, MCMC sampling, and probabilistic inference. Machine learning techniques can be used to augment other search methodologies based on enumerative search or deduction by providing likelihood of various choices at any choice point. One such choice point is selection of a production for a non-terminal in a grammar that specifies the underlying program space. The likelihood probabilities can be function of certain cues found in the input-ouput examples provided by the user or the additionally available inputs [89]. These functions are learned in an offline phase from training data. Genetic programming is a program synthesis method inspired by biological evolution [72]. It involves maintaining a population of individual programs, and using that to produce program variants by leveraging computational analogs of biological mutation and crossover. Mutation introduces random changes, while crossover facilitates sharing of useful pieces of code between programs being evolved. Each variant's suitability is evaluated using a user-defined fitness function, and successful variants are selected for continued evolution. The success of a genetic programming based system crucially depends on the fitness function. Genetic programming has been used to discover mutual exclusion algorithms [68] and to fix bugs in imperative programs [146] MCMC sampling has been used to search for a desired program starting from a given candidate. The success crucially depends on defining a smooth cost metric for Boolean constraints. STOKE [124], a superoptimization tool, uses Hamming distance to measure closeness of generated bit-values to the target on a representative test input set, and rewards generation of (almost) correct values in incorrect locations. Probabilistic inference has been used to evolve a given program by making local changes, one at a time. This relies on modeling a program as a graph of instructions and states, connected by constraint nodes. Each constraint node establishes the semantics of some instruction by relating the instruction with the state immediately before the instruction and the state immediately after the instruction [45]. Belief propagation has been used to synthesize imperative program fragments that execute polynomial computations and list manipulations [62].

[24]

1.1 EM

In the exact EM, the distribution is stored as a matrix of logits, in the variational EM it is represented as parameters of a simpler distribution, and in the amortized variational EM, it is represented as the weights of a neural network.

1.2 DreamCoder

DreamCoder (DC) is a model that synthesises programs from initial primitives in a domain specific language (DSL) and a set of tasks [18]. The model utilises a modified version of a wake-sleep algorithm [25] to learn a generative model and a recognition network in tandem. The generative model learns a probability distribution over programs while the recognition network learns to map from tasks to programs and serves as neurally guided search over the program space. DreamCoder builds its library of programs from primitives in its DSL, and creates higher-order functions. Subroutines that are commonly used, are chunked and abstracted into concepts which become more accessible. This narrows the search tree immensely and enables scalability.

- hypothesis space - learning its own dsl

Tasks can be generative (e.g. creating an image) or conditional, (input-output relation, e.g. sorting a list).

The model operates in three distinct phases.

Wake In the wake phase, the objective is to find the best program ρ for each task $x \in X$ DreamCoder is presented with. Each task consist of one or a few (up to ~ 10) examples. The system synthesises the programs using a neural network as the recognition model Q to search through its library D.

Sleep: Abstraction A key component of the success of DreamCoder is the refactoring of programs. In this phase, DreamCoder consolidates new abstractions from the programs it has found during the wake phase. Experiences are replayed, and commonalities between programs are found so that they can be abstracted into new concepts to use in the future. The objective is to find programs of minimum description length (MDL).

Sleep: Dreaming The objective during the dreaming phase is to train the recognition model Q to perform MAP inference by both fantasies, and replays. When

fantasising, programs are drawn from the learned library as a generative model. The model is then trained on the produced tasks. This ensures a large and varied data-set to learn from and encourages out-of-distribution generalisation. Replays, simply training on previously seen task-program pairs, ensures that the model improves its predictions on the actual tasks it needs to learn, by using its programs.

A Bayesian View - What exactly happens? - The generative model is a CFG, parameterized my a Include the parameterization of the generative model as a bigram transition matrix (three dimensional. Parent, arg index, depth?)

- Frontier of primitives and abstracted concepts essentially form a markov blanket.

2 Transformer

At the heart of the transformer is the self-attention mechanism, which enables the model to weigh the significance of different tokens in a sentence relative to a given token.

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

where

- Q: Query matrix
- K: Key matrix
- V: Value matrix
- d_k: Dimensions of keys

The self-attention mechanism allows the model to focus on different tokens with varying intensities, enabling it to capture long-range dependencies in the data.

Since the transformer lacks inherent notions of sequence order (unlike e.g. RNNs), positional encodings are added to the embeddings at the bottoms of the encoder and decoder stacks. This ensures that the model can attend to the position of words in the sequence. These encodings are added to the embeddings and have the same dimension, enabling the summed values to be processed in the self-attention layers.

Embeddings convert tokens into dense vectors of fixed size, capturing semantic meanings and relationships. This enables the model to understand relationships between different tokens.

- Why is attention all you need?
- Why is the transformer architecture so relevant in the past few years?
- Why does it take away the need for archictecture search
- Multi-Head Attention: Instead of a single set of attention weights, the model uses multiple sets, enabling it to focus on different parts of the input for different tasks simultaneously.
- Feed-forward Networks: Each layer of the transformer contains a feed-forward neural network, which is applied independently to each position.
- Normalization and Residual Connections: These components help in stabilizing the activations and enabling deeper stacking of layers.
- Stephen Wolfram explains in his book about ChatGPT and the accompanying blog article how a thought can be seen as a trajectory through semantic space.

Recent advancements in program synthesis have seen the rise of novel techniques aimed at generating code that not only satisfies a given specification but also matches human-written code patterns. One of the most notable methods in this sphere is DreamCoder. This model, along with several other state-of-the-art (SOTA) methodologies, has shown exceptional proficiency in understanding and generating complex code structures based on given specifications.

However, a critical examination of such methods, including DreamCoder, reveals a foundational limitation: their heavy reliance on syntactical constraints. While these constraints are undoubtedly vital for ensuring the correctness of generated programs, they do not necessarily guarantee a deep understanding or utilization of semantic relationships within the code. This is akin to learning the grammar of a language but missing out on the nuances and contexts that give meaning to words and phrases.

One could argue that the attention mechanism, as seen in transformers, provides a potential solution to this limitation. The attention mechanism inherently captures dependencies between various parts of an input, no matter how distant they might be. In the context of program synthesis, such a mechanism could

be instrumental in deriving and maintaining meaningful semantic relationships between different code segments. The beauty of attention is that it provides a dynamic way to weigh different parts of the input based on context, potentially allowing a transformer-based model to capture intricate semantic details of a program.

Nevertheless, a transformer's capabilities come with its own set of challenges. While the attention mechanism offers a promising avenue for capturing semantic nuances, transformers, especially when trained from scratch, require colossal amounts of data to achieve satisfactory performance. The fundamental architecture of transformers necessitates this data-driven learning, often demanding diverse and extensive training datasets that might not always be available or feasible, especially in niche areas of program synthesis. Moreover, when the goal is to create a model of human cognitive abilities, it is obvious that humans do not have access to vast amounts of data, instead, we are able to infer causal relationships, from only a limited number of examples. E.g. consider the problem `[1,2,3]` $\rightarrow$ `[3,2,1]`. An astute reader might promptly conclude that the list has been reversed. Dougs analogies

This is where GFlowNets (GFNs) carve their niche. Contrary to the data-hungry nature of transformers, GFNs do not necessitate gargantuan datasets for efficient training. Instead, they leverage a more intuitive approach: defining a reward distribution that's directly proportional to the objects (or programs) we intend to generate. This not only streamlines the learning process but also ensures that the synthesized programs are aligned with our desired outcomes.

In the realm of program synthesis, our objectives are clearly defined. We aim to construct objects, or more specifically, pieces of code that precisely meet the given specification. This isn't just about producing syntactically correct code; it's about generating semantically rich and efficient programs that can effectively solve the specified problem without superfluous or redundant components.

- What is the problem? why can't we marginalise over the CFG? or can we?

- How does GFN or DC deal with it?

3 GFlowNet

The problem can be formalized as finding a latent hierarchical structure from a limited set of specifications.

We can specify our GFlowNet in the realm of program synthesis as building a directed acyclic graph over abstract syntax trees. We formalise it. The gfn is conditioned on the task, and so we have a conditional reward distribution $R(z|x)$, as well as a conditional forward policy $P_F(s|x)$ and partition function $Z(x)$.

I am building on the DeepSynth framework where the goal is to find programs in a domain specific language. The task is list editing, i.e. i get some inputs and outputs relationship and the network has to find a program that solves it. we want diversity, so perhaps even multiple programs that solve it.

4 Theory

The trained GFlowNet gives us a the stochastic policy $\pi(a|s)$, where a is an action from the action space A and s is a state from the state space S.

- Relation to MDPs, POMDPs
- Relation to Reinforcement learning
- Relation to MCMC sampling
- Relation to Variational inference.

Since their initial publication [SOURCE], many extended and modified variants have been published. See e.g. [SOURCE, awesome GFlowNets for an overview.]

5 Methods

- Reasons for using a GFlowNet
- Transformer allows for semantic relationships (although we still don't use control or data flow, but we could in principle)
- We want to approximate a multimodal distribution, unlike DreamCoder in which the MAP estimate refrains from finding semantically equal but syntactically different solutions.

6 DeepSynth

In their 2021 paper, Fijalkow et al. proposed a framework called "distribution-based search", in which they

tackle the difficult problem of searching through a DSL to find programs matching a specification in a vast hypothesis space.

The authors introduce DeepSynth [1], a general-purpose program synthesizer which constructs programs from input-output examples [26]. They make use of a 2-step pipeline in which a neural network learns to predict weights for a context free grammar (CFG), making it a probabilistic CFG (PCFG) and then a search algorithm seeks programs matching the query.

However, rather than trying to improve models for program synthesis, they focus on how to best utilize this neural predictor in order for the system to be useful and scale beyond trivial programs.

At a high-level the approaches we develop in this work follow a 2-stage pipeline: in the first stage a learned model predicts probabilistic weights, and in the second stage a symbolic search algorithm uses those weights to explore the space of source code. Our contributions target the second stage of this pipeline, and we focus on theoretical analysis of sampling-based search algorithms, new search algorithms based on neurally-informed enumeration, and empirical evaluations showing that recent neural program synthesizers can compose well with our methods.

- Loss optimality
- Learning the weights of a CFG making it a PCFG, then finding programs using this PCFG.
- There is a trade-off between finding many programs rapidly, or fewer programs that are more likely to solve the problem.
- enumeration vs sampling
- They show that all sampling algorithms are either loss optimal or have infinite loss?

In DeepSynth, we are given a list of primitives, the initial DSL, with a number of syntactic constraints which compile into a context free grammar. they then train a prediction model to predict the weights of the cfg making it a pcfg. lastly they search through the pcfg to find programs meeting the given specification.

[1] `https://github.com/nathanael-fijalkow/DeepSynth`

the DSL is given by a set of primitives together with their (possibly polymorphic) types and semantics. We describe the machine learning pipeline for program synthesis, illustrated in Figure 1 on a toy DSL describing integer list manipulating programs. The compilation phase constructs a context-free grammar (CFG) from the DSL together with a set of syntactic constraints. The CFG may incorporate important information about the program being generated, such as the n last primitives (encompassing n-gram models) or semantic information (e.g. non- zero integer, sorted list). A prediction model (typically a neural network) takes as inputs a set of I/O and outputs a probabilistic labelling of the CFG, inducing a probabilistic context-free grammar (PCFG). The network is trained so that most likely pro- grams (with respect to the PCFG) are the most likely to be solutions, meaning map the inputs to corresponding outputs. We refer to Appendix A for an in-depth technical discussion on program representations and on the compilation phase. In this work we focus on the search phase and start with defining a theoretical framework for analysing search algorithms. The PCFG obtained through the predictions of the neural network defines a probabilistic distribution D over programs. We make the theoretical assumption that the program we are looking for is actually sampled from D, and construct algorithms searching through programs which find programs sampled from D as quickly as possible. Formally, the goal is to minimise the expected number of programs the algorithm outputs before finding the right program.

7 FlowCoder

What do we need? Differently from the other models which do not embed programs (they do have a programencoder, so what is that then? the output). The output tensor is encoded as a program. The programs themselves are never embedded. What is the difference?

In this work, I am implementing a novel program synthesizer, which integrates with the DeepSynth framework.

I am separating the generative model (world model) from the inference machine. And am using a full transformer. Tasks are encoded using the transformer en-

coder. [explain in detail], which states are encoded using the decoder. A state is represented as a rule of the cfg. at each step of the trajectory, the combined inputs are used for the forward logit network to predict logits over possible actions which are then added to the state.

7.1 Introduction

A Generative Flow Network, or GFlowNet, operates as a generative model driven by a trained stochastic policy. It constructs objects $z \in Z$, where Z is the space of completed states, sequentially, with each sampling probability being proportional to a reward function $R(z)$, where $R(z)$ is non-negative and integrable. GFlowNets excel in sampling diverse solutions, eliciting a high reward [27].

The state space can be visualized as a directed acyclic graph (DAG), where vertices correspond to states and edges denote transitions.

A trajectory τ is a series of state transitions commencing from an initial state s_0 and culminating at a terminal state, $s_n \in Z$.

The forward policy, denoted as $P_F(s'|s)$, encompasses the children of all non-terminal states in $S \backslash Z$. It inherently generates a distribution over complete trajectories:

$$P(\tau) = \prod_{i=1}^{n} P_F(s_i|s_{i-1})$$

Using sequential sampling from P_F, one can deduce a distribution $P_F^{\top}$ over terminal states:

$$P_F^{\top}(z) = \sum_{\tau \text{ leading to } z} P_F(\tau)$$

For any reward function $R : Z \to \mathbb{R}_{\geq 0}$, GFlowNets aims to determine a parametric policy where object sampling likelihood is proportional to its reward.

The main theorem describes that if the flow function F is trained such that it matches the flow-matching constraint, i.e. for any state the flow going into the state matches the flow going out of it, and the flow at terminated states x is defined by the reward function $R(x)$, GFlowNet will sample terminated states with probability $\frac{R(x)}{\sum_{x'} R(x\prime)}$.

Since its original publication many extended and modified variants have been published.

1. **TB Objective**: Trajectory Balance (TB) approach, as elaborated by Malkin et al. (2022), necessitates simultaneous learning of a forward policy, a backward policy $P_B(s|s';\theta)$, and a scalar Z_θ. The TB objective is:

$$L_{TB}(\tau;\theta) = \left[\log \frac{Z_\theta \prod_{i=1}^{n} P_F(s_i|s_{i-1};\theta)}{R(z)\prod_{i=1}^{n} P_B(s_{i-1}|s_i;\theta)}\right]^2 \tag{3.1}$$

However, since I am essentially predicting a linearized tree, each node can only have one parent. Therefore, P_B will always be 1 and can be disregarded from the equation. We can then simplify the TB Objective, making use of log rules, to

$$L_{TB}(\tau;\theta) = \left(\log Z_\theta + \sum_{i=1}^{n} \log P_F(s_i|s_{i-1};\theta) - \log R(z)\right)^2 \tag{3.2}$$

2. **Training Dynamics**: Nullifying L_{TB} ensures proportional sampling to object reward. The loss can be minimized using gradient descent with on-policy and off-policy strategies, reminiscent of reinforcement learning techniques.

- amortized marginalization
- variational approximation, KL divergence, ELBO
- Do we want multiple solutions? MAP vs MLE + how could we convert one to another (Talk also about DC, and why they chose MAP)
- Surfaces and Essences analogy example abc:xyz :: abd: (wyz/ xya)
- Neural PCFG (Compare to Neural PCFG from Rush and GFN-EM)
- what is the difference between amortized EM and GFlowNet EM?
- I think GFlowNet-EM is using a neural PCFG as a generative model.
- write about tractability
- should there be one or multiple world models? what is the world model here? the PCFG? i.e. the generative model?
- Write about context free grammars. number of possible derivations.
- learning your own dsl
- should there be one dsl or multiple?

- it could be e.g. having multiple representations/ models of the world and then a model on top of that which tells you which model is useful. This reminds me of society of mind and also of neural darwinism. World models may be competing with one another.
- Sticking with a task for how long, before switching
- Evaluation of variables
- Now we are predicting rules, and encoding them individually. We could also use their features like depth, arg idx etc. to encode them
- Why we dont need backward logits
- when we only predict rules, it may be ambiguous. The same rule could expand different parts of the tree. That's why DFS
- we want to show that using the transformer for semantics makes sense, therefore we need to show that it learnt program embedding. relating tasks to programs and also relationships between tasks and relationships between programs. Also relate that to the embeddings of tasks in DC, which shows that similar tasks are approximately in the same space.
- Explain what exactly this task shows or is supposed to represent. Algorithmic thinking, or any type? Dehaene, (also sensory data pattern prediction machine paper)
- explain what the batches do

How are programs encoded? How are tasks encoded?

- If we are using encoder + decoder, are we violating the Markovian Flow assumption? Look at the smiley example. If the NN would get [[left brow], [left brow, right brow], [left brow, right brow, smile]] as input, i.e. the sequence of the states, it would violate the assumption. but it only gets the current state, e.g. [left brow, right brow], and from that it has to infer the next step. When using a decoder, we would indeed give it the whole trajectory of states, so it would violate the assumption. But even now i am encoding the sequence and giving the whole trajectory as input. and since the CFG is essentially a tree, there is only one parent for each state.

It would probably be faster to do it bottom up like in HEAP search from Nathanael and from GFN-EM, then we could also use sub-trajectory balance, i.e. calculate intermediate Rewards.

Another thing we could do is predict a bunch of terminals at once, and then combine in each step.

Framework

- Explain CFG, parameters like program depth, sizemax etc.
- How many programs can actually be created?
- Exploration
- What is the generative model? The CFG or the Transformer?
- Minimum description length
- In the paper they say they produce programs up to depth 6
- We want to find a probability distribution that is proportional to a reward.
- Explain the parameterization. I.e. we could parameterize the rules vs the primitives.
- Write about the marginalization. Can we marginalize the CFG? How does DC do it? (they only marginalize over a beam)

Model

- RuleEncoder
- we predict rules.
- Talk about the expansion order
- We do it DFS now.
- Gives structure for Transformer to understand
- no need for Backward logits because its a tree
- Otherwise we also need to decide expansion order since rules are context free, i.e. if we only predict the rule there may be multiple parts of the tree where this could be applied to. However there may be benefits in expanding in different orders. We could apply a second GFN to predict the next best node to expand.
- Talk about GNNs and why you did not end up using them

- E.g. Left hand expansion might give more information than right hand in certain situations.
- Evaluation.
- We could also expand bottom up, which would let us evaluate partial expression and include that information for the next expansion.
- Talk about decision to embed rules, vs primitives. and perhaps other strategies.
- We could use a GNN and do both node and edge prediction.
- A lot of overhead because i had to learn the problem itself, how to represent programs, etc. and also new neural networks i never worked with. (Don't bitch)
- IOEncoder
- Data augmentation
- Self-supervised learning
- Sleep phase
- Replay
- Fantasy
- Overfitting
- Loss optimality. Quality vs Quantity

Wake-sleep

- Replay
- Fantasy

Reward

- Hamming distance
- Compare embeddings, cosine similarity
- Energy based model
- Mutual Gain
- Binary
- Edit distance, Levenshtein distance

7.2 Results

7.3 Analysis

- Ablation
- Speed
- Programs solved
- visualize task embeddings (from DC) and the according programs. Maybe take the programs with the highest reward even if it didn't solve it
- How many programs are uniquely created?
- Make a histogram of all the programs. Can we make it over time? I.e. for each program we should see how often it is created over time.
- The program is not loss optimal, i.e. we may get stuck in a local minimum
- Compare to baseline (Uniform PCFG?)
- Ablation study
- Show the partition function
- losses graph
- my hopes would be that similar programs are embedded close in space. - how can we check that?
- How many programs can we solve
- How long does it take to solve a program?

7.4 Complexity analysis

7.5 Benefits over other approaches

- Differentiable reward (could also be used in other models)
- Does their approach also approximate the reward function?

7.6 Limitations

- Make the network choose where to expand (not DFS). Note that we are not going through the CFG DFS, but creating the AST with DFS
- Temporal dimension? (GLOM includes time)

7.7 Future Work

- Sub-trajectory balance
- Learning from dataset of good programs (akin to someone telling you a solution)
- Train from middle of states
- Train backward policy, then we can also see finished states and predict trajectories that led to them

7.8 Conclusion

7.9 Discussion

- Scaling the model
- Scaling neural search paper
- how would it work using GFNs

Biological Plausibility

- Why are/ aren't types plausible? Related to categories?
- CFG? where does it come from?
- primitives?

8 From notes

We formalise the free-energy principle as

$$\underbrace{F(s,\mu)}_{\text{Free Energy}} = \underbrace{D_{KL}\left(q(\psi|\mu)||p(\psi|a)\right)}_{\text{Complexity}} - \underbrace{E_q\left[\log p(s|\psi,a)\right]}_{\text{Accuracy}} \tag{3.3}$$

where the complexity is the Kullback-Leibler divergence between the true posterior and its approximation. The Accuracy is essentially the expected negative log likelihood (?)

the following is from GFlowNet-EM

The problem is formalized as a compositional hierarchical latent variable model. Here $z \to x$ is a directed graphical model where z has a hierarchical structure of iintermediate latent random variables. The likelihood is given by $p(x) = \sum_z p_\theta(z)p_\theta(x|z)$. We want to optimize

$$\mathcal{L} \log \prod_{i=1}^{T} p(x_i) = \sum_{i=1}^{T} \log \sum_z p_\theta(z)p_\theta(x_i|z)$$

. The Expectation-Maximization (EM) algorithm solves this by maximizing the evidence lower bound (ELBO), which is equivalent to the negative free energy.

$$\mathcal{L} \leq \mathbb{E}_{z\sim q}(z|x_i) \log \frac{p_\theta(z)p_\theta(x_i|z)}{q(z|x_i)} \tag{3.4}$$

$$= \mathcal{L} - \sum_{i=1}^{T} D_{KL}\left(q(z|x_i)||p(z|x_i)\right) \tag{3.5}$$

amortized variational inference $q(z|x_i)$ is parameterized as a neural network. The goal is to approximate q such that $q(z|x_i) \propto p(z|x_i)$, i.e. the true posterior.

In the E-step we optimize q such as to minimize $D_{KL}\left(q(z|x_i)||p(z|x_i)\right)$. In the M-step we optimize $\mathcal{L}$ w.r.t. the parameters of p

$$\mathbb{E}_i\left[\mathbb{E}_{z\sim q(z|x_i)} \log p_\theta(z)p_\theta(x_i|z)\right]$$

When z is high dimensional, we must be wary of posterior collapse. Wake-sleep [SOURCE] mitigates this by minimizing

$$\mathbb{E}_{z\sim p_\theta(z), x\sim p_\theta(x|z)}\left[-\log q_\phi(z|x)\right]$$

which is equivalent to minimizing $D_{KL}\left(p(z|x_i)||q(z|x_i)\right)$, the KL is opposite direction, which makes q seek a broad approximation to the true posterior, capturing all modes.

Derive GFlowNet from here.

In DreamCoder, they do something similar. They have a generative model in the form of a PCFG, and a recognition model which takes a task as input and outputs a bigram transition tensor, which serves as a policy over actions. Here Q is a conditional distribution, a mapping between IOs and programs. This is essentially an encoding of programs as tasks, which can also be seen in their visualization of task embeddings. The PCFG is completely syntactical. So they train the model to predict better weights and then search in the pcfg enumeratively.

for each task they have a beam, which they marginalize over.

What is the prior, likelihood, etc. how is it operationalised

An EM algorithm is used to estimate the parameters of the generative model.

Explain why they chose a bigram over transformer

(efficiency in searching. however one could use transformer without search, but usually some kind of search is necessary.) related to sample quality over search

Explain why they look for MAP over posterior (symmetry breaking)

- argument for attention and why it might be necessary to capture semantics.
- transformers learn nested relationships (see chapter wolfram) (relationships of increasing complexity)
-

In FlowCoder, I do a couple things differently:

- I embed all the rules (check this with neuralPCFG and maybe embed the cfg so that you have a better generative model)
- I am searching for the posterior (aligning with active inference), because i want diverse solutions, i.e. i want to learn the solution space for each task, multiple modes, not just the max mode. (how does this compare to MDL)
-

Chapter 4

Discussion

1 Societal implications

- Memes as exlicit cultural concepts of a super-organism
- Cognitive Light cone view of cancer and psychopaths (also in a game-theoretic view)
- Summarise main findings and especially relate them to the questions. What have we learned about identity and the question "Who am I?"
- Understanding, goals, models, etc.

Latex is very useful and can use glossary.

Glossary

glossary Acronyms and terms which are generally unknown or new to common readers. 24

latex LaTeX (short for Lamport TeX) is a document preparation system. The user has to think about only the content to put in the document and the software will take care of the formatting. 24

Bibliography

[1] Anna Ciaunica, Michael Levin, Fernando E. Rosas, and Karl Friston. Nested Selves: Self-Organisation and Shared Markov Blankets in Prenatal Development in Humans. preprint, PsyArXiv, May 2023.

[2] Artur d'Avila Garcez and Luis C. Lamb. Neurosymbolic AI: The 3rd Wave. *arXiv:2012.05876 [cs]*, December 2020. arXiv: 2012.05876.

[3] David Vernon, Robert Lowe, Serge Thill, and Tom Ziemke. Embodied cognition and circular causality: on the role of constitutive autonomy in the reciprocal coupling of perception and action. *Frontiers in Psychology*, 6, 2015.

[4] Karl Friston, Rosalyn J. Moran, Yukie Nagai, Tadahiro Taniguchi, Hiroaki Gomi, and Josh Tenenbaum. World model learning and inference. *Neural Networks*, 144:573–590, December 2021.

[5] Pietro Mazzaglia, Tim Verbelen, Ozan Çatal, and Bart Dhoedt. The Free Energy Principle for Perception and Action: A Deep Learning Perspective. *Entropy*, 24(2):301, February 2022. arXiv:2207.06415 [cs, q-bio].

[6] Volker Peckhaus. Leibniz's Influence on 19th Century Logic. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Winter 2018 edition, 2018.

[7] Michael Rescorla. The Language of Thought Hypothesis. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Summer 2019 edition, 2019.

[8] Panu Raatikainen. Gödel's Incompleteness Theorems. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Spring 2022 edition, 2022.

[9] B. J. Copeland. *The Essential Turing*. Oxford University Press UK, 2004.

[10] Noam Chomsky. On certain formal properties of grammars. *Information and control*, 2(2):137–167, 1959.

[11] Stanislas Dehaene, Fosca Al Roumi, Yair Lakretz, Samuel Planton, and Mathias Sablé-Meyer. Symbols and mental programs: a hypothesis about human singularity. *Trends in Cognitive Sciences*, 26(9):751–766, September 2022.

[12] Joshua S. Rule, Joshua B. Tenenbaum, and Steven T. Piantadosi. The Child as Hacker. *Trends in Cognitive Sciences*, 24(11):900–915, November 2020.

[13] Fosca Al Roumi, Sébastien Marti, Liping Wang, Marie Amalric, and Stanislas Dehaene. Mental compression of spatial sequences in human working memory using numerical and geometrical primitives. *Neuron*, 109(16):2627–2639.e4, August 2021.

[14] Steven T Piantadosi. The computational origin of representation. *Minds and machines*, 31(1):1–58, 2021.

[15] Steven T Piantasodi and Felix Hill. Meaning without reference in large language models. *arXiv preprint arXiv:2208.02957*, 2022.

[16] Quan Do and Michael E Hasselmo. Neural circuits and symbolic processing. *Neurobiology of Learning and Memory*, 186:107552, 2021.

[17] Adam Santoro, Andrew Lampinen, Kory Mathewson, Timothy Lillicrap, and David Raposo. Symbolic behaviour in artificial intelligence. *arXiv preprint arXiv:2102.03406*, 2021.

[18] Kevin Ellis, Catherine Wong, Maxwell Nye, Mathias Sablé-Meyer, Lucas Morales, Luke Hewitt, Luc Cary, Armando Solar-Lezama, and Joshua B.

Tenenbaum. DreamCoder: bootstrapping inductive program synthesis with wake-sleep library learning. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*, pages 835–850, Virtual Canada, June 2021. ACM.

[19] Douglas R. Hofstadter. *Gödel, Escher, Bach: an Eternal Golden Braid*. Basic Books Inc., 1979.

[20] Artur d'Avila Garcez and Luis C Lamb. Neurosymbolic ai: the 3rd wave. *arXiv preprint arXiv:2012.05876*, 2020.

[21] Nabil Bouizegarene, Maxwell Ramstead, Axel Constant, Karl Friston, and Laurence Kirmayer. Narrative as active inference, July 2020.

[22] Anirudh Goyal and Yoshua Bengio. Inductive biases for deep learning of higher-level cognition. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 478(2266):20210068, October 2022. Publisher: Royal Society.

[23] Yoshua Bengio. Scaling in the service of reasoning & model-based ML. `https://yoshuabengio.org/2023/03/21/scaling-in-the-service-of-reasoning-model-based-ml/`, 2022. Accessed 21-08-2023.

[24] Sumit Gulwani, Oleksandr Polozov, and Rishabh Singh. *Program synthesis*. Number 4.2017, 1-2 in Foundations and trends in programming languages. Now Publishers, Hanover, MA Delft, 2017.

[25] Geoffrey E Hinton, Peter Dayan, Brendan J Frey, and Radford M Neal. The" wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.

[26] Nathanaël Fijalkow, Guillaume Lagarde, Théo Matricon, Kevin Ellis, Pierre Ohlmann, and Akarsh Potta. Scaling Neural Program Synthesis with Distribution-based Search, October 2021. arXiv:2110.12485 [cs].

[27] Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow Network based Generative Models for Non-Iterative Diverse Candidate Generation. In *Advances in Neural Information Processing Systems*, volume 34, pages 27381–27394. Curran Associates, Inc., 2021.