STRING DIAGRAMS FOR TEXT

VINCENT WANG-MAŚCIANICA

ST. CATHERINE'S COLLEGE
THE UNIVERSITY OF OXFORD
DEPARTMENT OF COMPUTER SCIENCE

A THESIS SUBMITTED FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
2023

Contents

(Acknowledgements will go in a margin note here.)

Novel contributions:

- Section REF is a pedestrian introduction to weak n-category theory (via `homotopy.io`, underpinned by the theory of associative n-categories) from the perspective of generalising familiar string-rewrite systems to higher dimensions. The chief development of this section is a demonstration that context-free grammars and tree-adjoining grammars may be formalised in the n-categorical setting.
- Section REF spells out a generative grammar for text using an n-categorical signature as a rewrite system, which additionally provides a unified framework from which the Text Circuit Theorem first proved in CITE is recovered.
- Section REF introduces the category **ContRel** of continuous relations. I detail the relationships (or lack thereof) of **ContRel** to its cousins **Top** and **Rel**. Though **ContRel** is constructed naïvely, its definition and an exposition of its expressivity from the monoidal perspective appears to be novel.
- Section REF string-diagrammatically characterises set-indexed collections of disjoint open subsets of spaces in **ContRel** as *sticky spiders* – special frobenius algebras that satisfy certain interaction relations with an idempotent. The diagrammatic outcome is that reasoning with such set-indexed collections remains as graphically intuitive as with spiders.
- Section REF – with the aid of a theorem by Friedman [**?**] – string-diagrammatically characterises a vocabulary of linguistic topological relations in **ContRel** such as simple connectedness, touching, insideness, and rigid motion.
- Section REF argues for the centrality of explaining communication as a criterion for formal approaches to syntax, and explores the relationship between productive and parsing grammars as organised by a monoidal cofunctor. A diagrammatic treatment of monoidal cofunctor boxes is introduced for this purpose.
- REF is a standalone introduction to the mathematical setup of Montague's *Universal Grammar*, aimed at modern algebraists. An outcome of this section is the placement of text circuits as a natural mathematical development in the broadly conceived programme of Montague Semantics.
- Appendix REF constructs a subcategory of sticky spiders in **ContRel** on the unit square that behaves as a model of **FinRel** equipped with a *Turing object* – a single object in a category with respect to which all other objects and morphisms between them may be encoded. This is of particular relevance to the linguistic phenomenon of *entification* – that essentially all grammatical categories of words and even phrases have noun-equivalents, e.g. `to run` $\simeq$ `running`, `Bob drinks` $\simeq$ `the fact that Bob drinks`. The presence of a Turing object in a symmetric monoidal category additionally provides a semantic model for higher-order processes of generic input type, and for *lasso diagrams*.
- In Section REF , by parsing text as circuits (Section REF) and using monoidal cofunctor boxes (Section REF) to interpret those circuits in **ContRel** as iconic representations equipped with a vocabulary of lingustic-topological concepts (Section REF), I compute some metaphors by hand.

0
Context and synopsis

There are potentially practical and theoretical benefits to a fresh mathematical take on basic linguistics. String diagrams are formal, intuitive, expressive, fun, and pretty. I review the relevant research context.

0.1 What this thesis is about

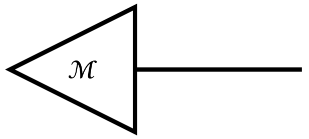

Figure 1: Let's say that the meaning of text is how it updates a model. So we start with some model of the way things are, modelled as data on a wire.

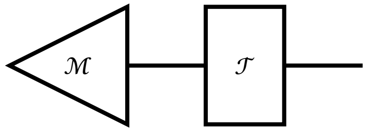

Figure 2: Text updates that model; like a gate updates the data on a wire.

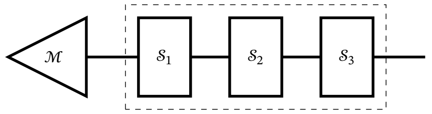

Figure 3: Text is made of sentences; like a circuit is made of gates and wires.

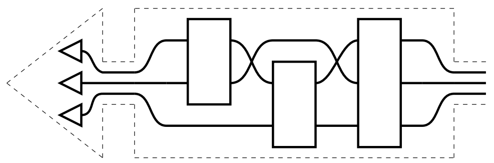

Figure 4: Let's say that ***The meaning of a sentence is how it updates the meanings of its parts.*** As a first approximation, let's say that the *parts* of a sentence are the nouns it contains or refers to. Noun data is carried by wires. Collections of nouns are related by gates, which play the roles of verbs and adjectives.

This thesis is about studying language using string diagrams.

I am interested in using contemporary mathematical tools as a fresh approach to modelling some features of natural language considered as a formal object. Specifically, I am concerned with the compositional aspect of language, which I seek to model with the compositionality of string diagrams. Insofar as compositionality is the centrepiece of "knowledge of language", I share a common interest with linguists, but I will not hold myself hostage to their methods, literature, nor their concern with empirical capture. I will make all the usual simplifying assumptions that are available to theoreticians, such that an oracular machine will decide on lexical disambiguation and the appropriate parse using whatever resources it wants, so that I am left to work with lexically disambiguated words decorating some formal grammatical structure. It is with this remaining disambiguated mathematical structure that I seek to state a general framework for *meaningful compositional representations of text*, in the same way we humans construct rich and interactable representations of things-going-on in our minds when we read a storybook. So if you are interested in understanding language, this thesis is an invitation to a conception of formal linguistics that's maybe worth a damn in a world where large language models exist.

Objection: Isn't that reinventing the wheel?

Yes, to an extent. I am not interested in the human language faculty *per se*, so my aims differ. There are several potential practical and theoretical benefits that a fresh mathematical perspective on language enables. First, the mathematics of applied category theory allows us to unify different views of syntax, and conservatively generalise formal semantics to aspects of language that may have seemed beyond the reach of rigour, such as metaphor. Practically, the same mathematics allows us to construct interfaces between syntax/structure and semantics/implementation in such a way that we can control the former and delegate the latter by providing specifications without explicit implementation, which (for historical reasons I will explain shortly) is possibly the least-bad idea for getting at natural language understanding in computers from the bottom-up. Second, there are probably benefits to expressing linguistics in the same mathematical and diagrammatic lingua franca that can be used to represent and reason – often soundly and completely – about linear and affine algebra [Sob15, BSZ17, BPSZ19], first order logic [HS20, BDGHS24], causal networks [LT23, JKZ19], signal flow graphs [BSZ14], electrical circuits [BS22], game theory [Hed15], petri nets [BM20], probability theory [FGP21], machine learning [CGG$^+$22, KLLW24, RFL$^+$24], and quantum theory [CK17, CG23], to name a few applications. At the moment, the practical achievements of language algorithms de-emphasise the structure of language, and there is no chance of reintroducing the study of structure with dated mathematics.

Point of information: What do you mean by natural language?

Natural language is a human superpower, and the foundation of our collective achievements and mistakes as a species. By *natural language* I mean a non-artificial human language that some child has grown up speaking. English is a natural language, while Esperanto and Python are constructed languages. If you are still reading then you probably know a thing or two already about natural language. Insofar as there are rules for natural languages, it is probable that like most natural language users, you obey the rules of language intuitively without knowing what they are formally. For example, while you may not know what adpositions are, you know where to place words like `to`, `for`, `of` in a sentence and how to understand those sentences. At a more complex level, you understand idioms, how to read between the lines, how to flatter, insult, teach, promise, wager, and so on. There is a dismissive half-joke that "engineering is just applied physics", which we might analogise to absurdity as "law is just applied linguistics"; in its broadest possible conception, linguistics is the foundational study of everything that can possibly be expressed.

Point of information: What are string diagrams?

String diagrams are a heuristically natural yet mathematically formal pictorial syntax for representing complex, composite systems. I say *mathematically formal* to emphasise that string diagrams are not merely heuristic tools backed by a handbook of standards decided by committee: they are unambiguous mathematical objects that you can bet your life on [JS91, Joy, Mac63, Lan10, Sel10].

String diagrams are also compositional blueprints that we can give semantics to – i.e. instantiate – in just about any system with a notion of sequential and parallel composition of processes. In particular, this means string diagrams may be interpreted as program specifications on classical or quantum computers, or as neural net architectures. Moreover, we can devise equations between string diagrams to govern the behaviour of interacting processes without having to spell out a bottom-up implementation.

Many fields of study have developed string diagrams as informal calculational aids, unaware of their common usage across disciplines and the rather new mathematics that justifies their use; everybody knows, but it isn't common knowledge. Why is that so? Because just as crustaceans independently converge to crab-like shapes within their own ecological niches by what is called *carcinisation*, formal notation for formal theories of "real world" problem domains undergo "string-diagrammatisation" in similar isolation. Why is that so? Because our best formal theories of the real world treat complexity as the outcome of composing simple interacting parts; perhaps nature really works that way, or we cannot help but conceptualise in compositional terms. When one has many different processes sending information to each other via channels, it becomes tricky to keep track of all the connections using one-dimensional syntax; if there are N processes, there may be on the order of $\mathcal{O}(N^2)$ connections, which quickly becomes unmanageable to write down in a line, prompting the development of indices in notation to match inputs and outputs. In time, probably by doodling a helpful line during calculation to match indices, link-ed indices become link-ing wires, and string-diagrammatisation is complete.

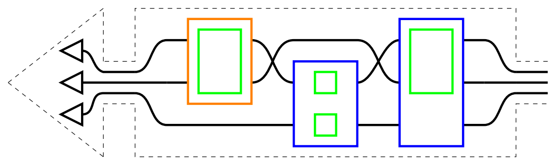
Figure 5: Gates can be related by higher order gates, which play the roles of adverbs, adpositions, and conjunctions; anything that modifies the data of first order gates like verbs.

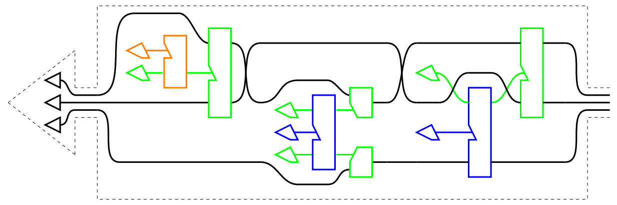
Figure 6: In practice, higher order gates may be implemented as gates that modify parameters of other gates. Grammar, and *function words* – words that operate on meanings – are in principle absorbed by the geometry of the diagram. These diagrams are natural vehicles for *dynamic semantics* [NBvV22], broadly construed, where states are prior contexts and sentences-as-processes update prior contexts.

Definition 0.1.1 (Text Circuits)**.** *Text circuits* are made up of three ingredients:

- wires
- boxes, or gates
- boxes with holes that fit a box, or 2nd order gates

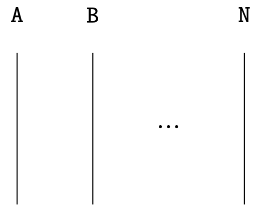

Figure 7: Nouns are represented by wires, each 'distinct' noun having its own wire.

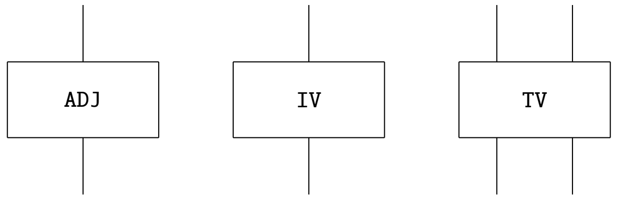

Figure 8: We represent adjectives, intransitive verbs, and transitive verbs by gates acting on noun-wires. Since a transitive verb has both a subject and an object noun, that will then be two noun-wires, while adjectives and intransitive verbs only have one.

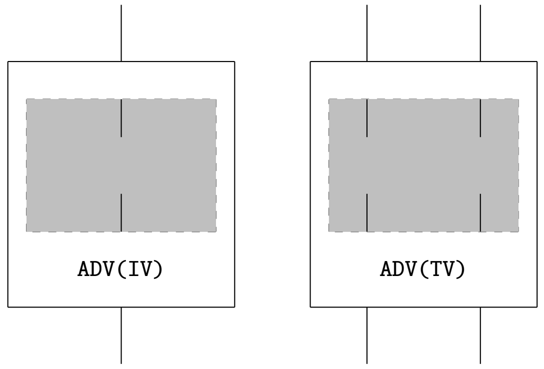

Figure 9: Adverbs, which modify verbs, we represent as boxes with holes in them, with a number of dangling wires in the hole indicating the shape of gate expected, and these should match the input- and output-wires of the box with the whole.

0.2 ***Question:*** *What is the practical value of studying language when Large Language Models exist?*

This is the devastating question. Although this thesis is pure theory, I wish to address the question of practical value early because I imagine practical people are impatient. I will summarise the stakes: LLMs raise questions of existential concern for the field of linguistics. More narrowly, they demand justification as to why I am writing a thesis about theoretical approaches to basic linguistics as a computer scientist in current year. I will note in passing that I have an ugly duckling problem, in that I am not strictly aligned with machine learning, nor linguists broadly construed, nor mathematical linguists. I feel enough affinity to have defensive instincts for each camp, but I am distanced enough from each that I fear attacks from all sides. Perhaps a more constructive metaphor than war is that I am writing in a cooperative spirit between domains, or that I am an arbitrageur of ideas between them. With that in mind, I am for the moment advocating on behalf of pen-and-paper-and-principles linguists in formulating a two-part reply to the devastating question, and I will switch sides later for balance. First a response that answers with practical values in mind, and then a response that asserts and rests upon the distinct values of linguists.

Point of information: What are Large language models? Assume that everything about LLMs is prefaced with "at the time of writing", because the field is developing so quickly. Large Language Models are programs trained using a lot of data and a lot of compute time to predict the next word in text, a task for which computational techniques have evolved from Markov n-grams to transformers [VSP+17]. This sounds unimpressive, but in tandem with methods such as fine-tuning from human feedback in the case of chatGPT [**?**] it is enough to tell and explain jokes [Bas22], pass the SAT [ted22] and score within human ranges on IQ tests [Tho22]. There is an aspect of genuine scientific and historical surprise that text-prediction can do this kind of magic. On the account of [MN21], computational linguistics began in a time when compute was too scarce to properly attempt rationalist, knowledge-based and theoretically-principled approaches to modelling language. Text-prediction as a task arose from a deliberate pursuit of "low-hanging fruit" as a productive and knowledge-lean alternative to doing nothing in an increasingly data-rich environment. Some observers [Chu11] expressed concern that the fruit would be quickly picked bare but those concerns are now evidently unfounded.

I'm sure there will be many further notable developments, and to be safe I won't make any claims about what machines can't do if we keep making them bigger and feed them more data or have them interact with one another in clever ways. Nonetheless there remain limitations that seem persistent for the foreseeable future, not in terms of *capabilities*, but in terms of *interpretability, explainability and safety*. These models have a tendency to hallucinate facts and are (ironically, for a computer) bad at arithmetic [HBK+21]. I imagine that the cycle of discovering limitations and overcoming them will continue. Despite whatever limitations exist in the state-of-the-art, it is evident to all sane observers that this is an important technology, for several reasons.

1. LLMs are a civilisational milestone technology. A force-multiplication tool for natural language – the universal interface – built from abundant data and compute in the information age may have comparably broad, deep, and lasting impact to the conversion of abundant chemical fuel to physical energy by steam engines in the industrial revolution.

2. LLMs represent a paradigm shift for humanity because they threaten our collective self-esteem, in a more pointed manner than losing at chess or Go to a computer; modifying a line of thinking from [Flo14], LLMs demonstrate that language and (the appearance of) complex thought that language facilitates is not a species-property for humans, and this stings on par with Darwin telling us we are ordinary animals like the rest, or Galileo telling us our place in the universe is unremarkable.

3. LLMs embody the latest and greatest case study of the bitter lesson [Sut19]. The tragedy goes like this: there's a group of people who investigate language – from syntax and semantics to pragmatics and analogies and storytelling and slang – who treat their subject with formal rigour and have been at it for many centuries. Their role in the story of LLMs is remarkable because it doesn't exist. They were the only qualified contestants in a "let's build a general-purpose language machine" competition, and they were a no-show. Now the farce: despite the fact that all of their accumulated understanding and theories of language were left out of the process, the machine is not only built but also far exceeds anything we know how to build in a principled way out of all their hard-earned insight. That is the bitter lesson: dumb methods that use a lot of data and compute outperform clever design and principled understanding.

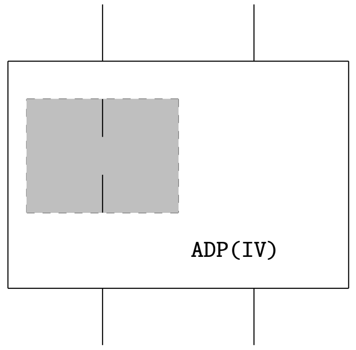

Figure 10: Similarly, adpositions also modify verbs, by moreover adding another noun-wire to the right.

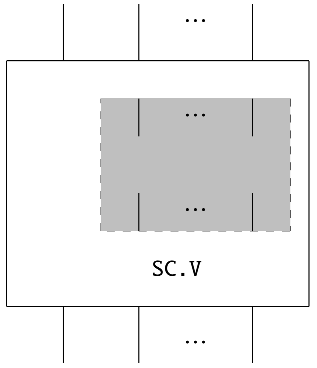

Figure 11: For verbs that take sentential complements and conjunctions, we have families of boxes to accommodate input circuits of all sizes. They add another noun-wire to the left of a circuit.

0.3 ***First Reply:*** *Interpretability, maybe.*

Expressing grammar as composition of processes might yield practical benefits. Moreover, we want economy, generality, and safety for language models, and we can potentially do that with few tradeoffs if we use the right framework. Simplified, half of the problem of learning language is learning the meaning of words. The meanings change over time and are context-dependent, and the words are always increasing in number. Encoding these meanings by hand is a sisyphean task. Data-driven learning methods are a good fit: the patterns to be learnt are complex and nebulous, and there is a lot of data. However, data-driven methods may be weaker at the second half of the problem: learning and executing the composition of meanings according to syntax. We can see just how much weaker when we consider the figures involved in 'the poverty of the stimulus'.

POINT OF INFORMATION: WHAT IS THE POVERTY OF THE STIMULUS? In short, this famous problem is the observation that humans learn language despite having very little training data, in comparison to the complexity of the learned structure. It is on the basis of this observation – alongside many others surrounding language acquisition and use – that Chomsky posits [Cho00] that language is an innate human faculty, the

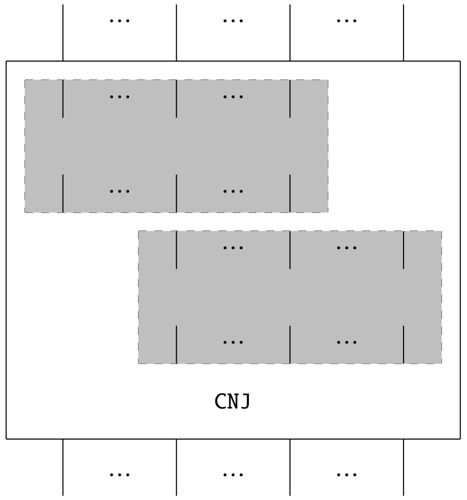

Figure 12: Conjunctions are boxes that take two circuits which might share labels on some wires.

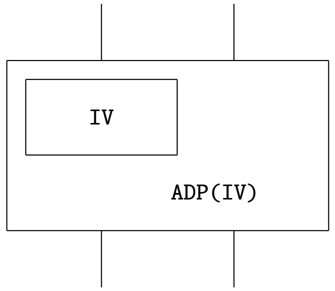

Figure 13: Of course filled up boxes are just gates.

development of which is less like effortfully going to the gym and more like effortlessly growing arms you were meant to have. The explanation goes like this: we can explain how a complex structure like grammar gets learnt from a small amount of data if everyone shares an innate Universal Grammar with a small number of free parameters to be learned. Whether or not the intermediate mechanism is a species-property of humans, the point is that we humans get a very small amount of input data, that data interacts with the mechanism in some way, and then we know a language. So, now that there are language-entities that are human-comparable in competence, we can make a back-of-the-envelope approximation of how much work the intermediate mechanism is doing or saving by comparing the difference in how much data and compute is required for both the human and for the machine to achieve language-competence. Humans get about 1.5 megabytes of data [MP19], 90 billion neurons [Her12], and an adult human consumes around 500 calories per day for thinking, for let's say 20 years of language learning. Rounding all values *up* to the closest order of magnitude, this comes to a cost metric of 10^{29} bits $\times$ joules $\times$ neurons. PaLM – an old model which is by its creators' account the first language model to be able to reason and joke purely on the basis of linguistic ability and without special training [CND+22, NC22] – required 780 billion training tokens of natural language (let's discount the 198 gigabytes of source code training data), which we generously evaluate at a rate of 4 characters per token [Kha23] and 5 bits per character. The architecture has 540 billion neurons, and required 3.2 million kilowatt hours of energy for training [Tom22]. Rounding values for the three units down *down* to the nearest order of magnitude comes to a cost metric of 10^{41} bit-joule-neurons. Whatever the human mechanism is, it is responsible for an order of magnitude in efficiency *give or take an order of magnitude of orders of magnitude*. It's possible that over time we can explain this difference away by various factors such as the efficiency of meat over minerals, separating knowledge of the world from knowledge of language, more efficient model architectures, or the development of efficient techniques to train new language models using old ones [TGZ+23]. One thing is clear: if it is worth hunting a fraction of a percent of improvement on a benchmark, forget your hares, a 10^{10} factor is a stag worth cooperating for.

Point of information: What progress have linguists made on this problem? The linguistic strategy for hunting the stag starts with what we know about how the mechanism between our ears works with language. The good news is that the chief methodology of armchair introspection is egalitarian and democratic. The bad news is that it is also anarchistic and hard-by-proximity; we are like fish in water, and it is hard for fish to characterise the nature of water. So the happy observations are difficult to produce and easily verified, and that means there are just a few that we know of that are are unobjectionably worth taking into account. One, or *the* such observation is *systematicity*. The intuition is best summarised by a quote. "Just as you don't find linguistic capacities that consist of the ability to understand sixty-seven unrelated sentences, so too you don't find cognitive capacities that consist of the ability to think seventy-four unrelated thoughts." (Fodor and Pylyshyn [FP88]).

POINT OF INFORMATION: SYSTEMATICITY? Systematicity refers to when a system can (generate/process) infinitely many (inputs/outputs/expressions) using finite (means/rules/pieces) in a "consistent" (or "systematic") manner. In short, how systems (like our capacity for language) achieve infinite ends by finite means. Like pornography, examples are easier than definitions. For example(s); we observe that anyone capable of understanding `Alice likes Bob` seems also to be capable of understanding `Bob likes Alice`; we know finitely many words but we can produce and understand potentially infinitely many texts; we can make infinitely many lego sculptures out of finitely many types of pieces; we can describe infinite groups and other mathematical structures using finitely many generators and relations; in the practical domain of computers, systematicity is synonymous with programmability and expressibility.

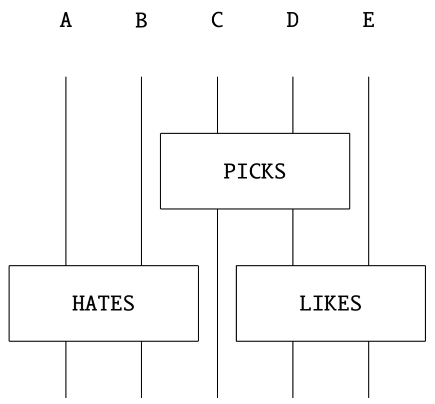

Figure 14: Gates compose sequentially by matching labels on some of their noun-wires and in parallel when they share no noun-wires, to give text circuits.

POINT OF INFORMATION: DO WE HAVE MATHS FOR SYSTEMATICITY? Yes, and I will consider it to be whatever it is that applied category theorists study. The concepts of systematicity and compositionality are deeply linked, because the only way we know how to achieve systematicity in practice is by a compositional systems, which can achieve infinite ends by finite means. Frege's initial conception of compositionality [Fre84] was borne of meditations on language, and states that a whole is the sum of its parts. Later conceptions of compositionality, the most notable deviation arising from meditations on quantum theory, generalises Frege's set-function conception of compositionality by varying the formal definitions of parts and the method of summation, and weakening the identification of the wholes with its parts to methods of keeping track of the relationships between wholes and parts [Coe21].

RETURNING TO THE STAG: So our starting point is that language is systematic and systematicity is the empirical surface of compositionality as far as we know, so compositionality is probably part of the solution to the poverty of the stimulus, if not most of it. The reasoning above should clarify why some folks don't think LLMs have anything to do with language as we humans do it. Their issue with purely data-driven architectures is either that we know immediately that they cannot be operating upon their inputs in a compositional way, or perhaps they appear to but their innards are too large and their workings too opaque to tell with confidence. Insofar as the task of learning language splits between learning meanings and learning the compositional rules of syntax that give rise to systematicity, the framework I present in this thesis is a proposal to split the cake sensibly between the two halves of the problem: meanings for the machines, and we'll supply the compositional rules. Syntax is still difficult and vast, but the rules are finite and relatively static. We can crack the black-box by treating syntax as directions for composition of smaller black-boxes that handle semantics. We all stand to benefit: we may give machines an easier time – now they only have to learn the meanings of words well – and we might gain confidence that the internal representations of the machine – their "mind's eye" – contains something we can probe and understand.

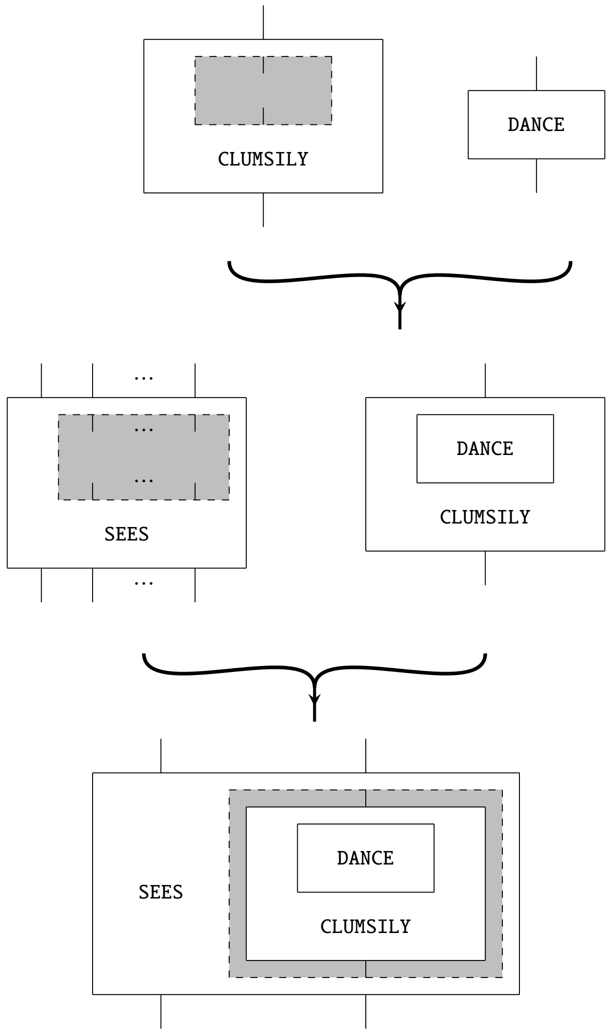

Figure 15: To summarise: composition by nesting corresponds to grammatical structure within sentences. Sentences correspond to filled gates, boxes with fixed arity correspond to first-order modifiers such as adverbs and adpositions, and boxes with variable arity correspond to sentential-level modifiers such as conjunctions and verbs with sentential complements.

0.3.1 **Objection:** *You're forgetting the bitter lesson.*

The bitter lesson is so harsh and often-enough repeated that this viewpoint is worth addressing proactively. The caveat that saves us is that the curse of expertise applies only to the object-language of the problem to be solved, not model architectures. We agree that qualitative improvements in problem-solving ability rarely if ever arise from encoding expert knowledge of the problem domain. Instead, these improvements come from *architectural* innovations, which means altering the parts and internal interactions of a model: changing *how* it thinks rather than *what* it thinks, to paraphrase Sutton's original prescription. We have good historical evidence that this prescription works, which we see by tracing the evolutionary path for data-driven language models from markov chains to deep learning [LBH15], RNNs [RM87], LSTMs [HS97], and now transformers [VSP+17]. Such structural changes are motivated by understandings (at varying degrees of formality) of the "geometry of the problem" [BBCV21]. The value proposition here is that with an appropriate mathematical lingua franca for structure, composition, and interaction, we can mindfully design rather than stumble upon the "meta-methods" Sutton calls for, allowing experts to encode *how* machines think and discover rather than *what*. Importing compositional and structural understanding from linguistics to machine learning via string diagrams might allow us to cheat the bitter lesson in spirit while adhering to the letter, and there is some preliminary empirical evidence for this, which I report on in Section **??**.

0.3.2 **Objection:** *GOFAI? GO-F-yourself!*

Hostility (or at least indifference) to symbolic approaches is a stance espoused by virtually all of modern machine learning, and for good reasons. This stance is worth elaborating and steelmanning for pen-and-paper-people in the context of engineering language applications.

First, many linguistic phenomena are nebulous [Chapm]: the boundary of a simile is like that of a cloud, not sharp like the boundary of a billiard ball. Second, linguistic phenomena are complex, dynamic, and multifactorial: there are so many interacting mechanisms and forces in the production and comprehension of language that it is plausibly "computationally irreducible" [Wol02], or a "type 2" problem [Mar77], both terms referring to a kind of computational difficulty where the only explanation of a system amounts to a total computational simulation of it. Third, nebulousity and irreducibility together weakly characterise the kinds of problem domains where machine learning shines, so add to all this that we can achieve better results by caring less, c.f. Jelinek on speech-recognition: "Every time I fire a linguist, the performance of the speech recognizer goes up". So for the practical person, these are very good reasons to not bother with trying to understand or "break down" the phenomenon in a principled way as part of the process of engineering an application.

So what good are pen-and-paper theories as far as practical applications are concerned? To borrow terms from concurrency, there is already plenty of liveness, what is needed is more safety; liveness is when the program does something good, and safety is a guarantee it won't do something bad. For example, there is ongoing work in integrating LLMs with stuctured databases for uses where facts and figures and ontologies matter; there is still a need for safeguards to prevent harmful outputs and adversarial attacks like prompt injection; while LLMs give a very convincing impression of reasoned thought, we would like to be sure if ever we decide to use such a machine for anything more than entertainment, such as assisting a caregiver in the course of healthcare decisions.

The good news is that symbolic-compositional theories are the right shape for safety concerns, because they can be picked apart and reasoned about. It is clear however that symbolic-compositional approaches by themselves are nowhere near achieving the kind of liveness LLMs have. Therefore, the direction of progress is synthesis.

0.3.3 **Objection:** *How does any of this improve capabilities?*

It's not meant to. The core value proposition for synthesis is interpretable AI, which operates in a manner we can analyse, and if appropriate, constrain. When lives are on the line (or more gravely, when capital is at risk), we would like to be certain that outputs are backed by guarantees. For this purpose, merely knowing *what* a deep-learning model is thinking is not enough: i.e. solving something like symbol-grounding alone is a necessary but insufficient component. For instance, merely knowing what the weights of subnetworks of an image classification model represent does not meet our requirement of an understanding of the computations that manipulate those representations. It would be nice to simply tell the AI *how to behave* in such-and-such a way according to common sense, but having it do as you mean and not as you say is such a difficult problem that it has a name: *alignment*, and it's worth noting that category theory underpins some of the most promising approaches to this problem [dav]. This isn't to say that techniques such as reinforcement learning from human feedback cannot in principle succeed at doing precisely what we want for alignment, it's just that a constructive methodology of verifying or guaranteeing success to the bulletproof epistemic standards of mathematics remains wanting. Our best bet is some kind of symbolic-compositional structure for us to begin reasoning about the innards of the machines.

To distinguish the difference in approach here, I have to draw a distinction in neurosymbolic approaches that does not seem well-supported in the literature. There are many approaches concerned with using connectionist architectures to simulate or aid or be-aided-by symbolic composition, which we can see the beginnings of in LLMs by examples such as chain-of-thought reasoning [WWS+23], and by probing their behaviour with respect to understood symbolic models [KW23]. The second kind of approach I would like to articulate is the inverse, where connectionist architectures are organised and reasoned with by symbolic-compositional means. Some examples of the first kind include implementing data structures as operations on high-dimensional vectors, taking advantage of the idiosyncrasies of linear algebra in very high dimension [Kan19], or work that explores how the structure of word-embeddings in latent space encode semantic relationships between tokens. Some examples of the second kind include reasoning about the capability of graph

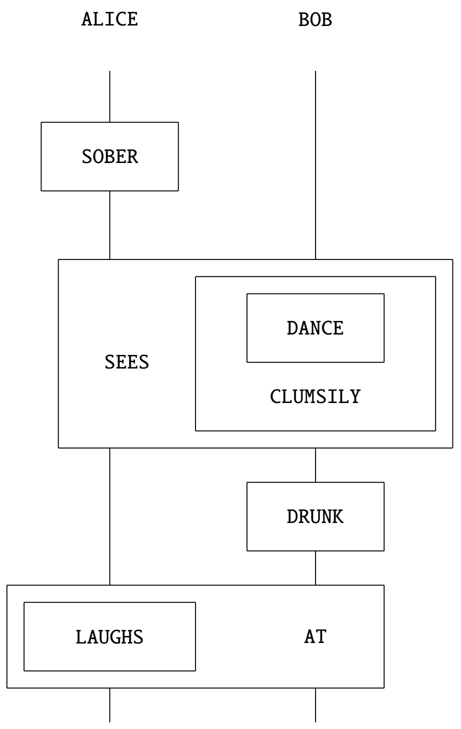

Figure 16: Composition by connecting wires corresponds to identifying coreferences in discourse. We obtain the same circuit for multiple text presentations of the same content, e.g. `Sober Alice who sees drunk Bob clumsily dance laughs at him.` yields the same circuit as the text `Alice is sober. She sees Bob clumsily dance. Bob is drunk. She laughs at him.`

neural networks by identifying or isolating their underlying compositional structure [LGT23], or architectures whose behaviour arises from compositional structure using neural nets as constituent parts, such as GANs [GPM+14] and gradient boosted decision trees [CG16]. The work in this thesis builds upon a research programme – DisCoCat [CSC10], elaborated in Section **??** – which lies somewhere in the middle of a duality of approaches to merging connectionism and symbolic-composition. It is, to the best of my knowledge, the only approach that explicitly incorporates mathematically rigourous compositional structures from the top-down alongside data-driven learning methods from the bottom-up. Fortifying this bridge across the aisle requires a little give from both sides; I ask only that reader entertain some pretty string diagrams.

I recount the following from [Søg23], which argues that symbol-grounding is solvable from data alone, and in the process surveys the front of the symbol-grounding problem in AI: the issue of whether LLMs encode what words refer to and mean. On the account of [BK20], the performance of current LLMs is a form of Chinese Room [Sea80] phenomenon, so no amount of linguistic competence can be evidence that LLMs solve the symbol-grounding problem. However, the available evidence appears to suggest otherwise. For example, large models converge on word embeddings for geographical place names that are isomorphic to their physical locations [LAS21]. Since we know that brain activity patterns encode abstract conceptual space with the same mechanisms as they do physical spaces [KS16], extrapolating the ability of LLMs to encode spatially-analogical representations would in the limit suggest that LLMs encode meanings in a way isomorphic to how we do, modulo the token-word distinction and so long as we take seriously some version of Gärdenfors' [**?**] thesis that meaning is encoded geometrically.

0.4 ***Second Reply:*** *LLMs don't help us understand language; how might string diagrams help?*

Another way to deal with the devastating question of LLMs is to reject it, on the basis that using or understanding LLMs is completely different from understanding language, and language is worth understanding in its own right. To illustrate this point by a thought experiment, what would linguistics look like if it began today? LLMs would appear to us as oracles; wise, superhumanly capable at language, but inscrutable. Similarly, most people effortlessly use language without a formal understanding which they can express. So the fundamental mystery would remain unchanged. Understanding how an LLM works at the algorithmic level cannot help. Borrowing and bastardising a thought from Marr, suppose you knew the insides of a mechanical calculator by heart. Does that mean you understand arithmetic? At best, obliquely, and maybe not at all: the calculator is full of inessentialities and tricks to fit platonic arithmetic against the constraints of physics, and you would not know where the tricks begin and the essence ends. Similarly, suppose you knew every line of code within and every piece of data used to train an LLM; does that mean you understand how language works? How does one delineate what is essential to language, and what is accidental? So let's forget about LLMs. The value proposition to establish now is how string diagrams and some category theory comes into the picture for the formal linguist who is concerned with understanding how language works, and that's the whole rest of the thesis. I sense one more objection from the practical reader, and one from the theoretical reader, so I'll address them in that order before moving on.

0.4.1 ***Objection:*** *Isn't the better theory the one with better predictions?*

LLMs are a theory of language in the same way a particular human brain is a theory of cognition; at best, debatablely. There are various criteria – not all independent – that are arguably necessary for something to qualify as an explanatory theory, and while LLMs satisfice (or even excel) at some, they fail at others. Empirical adequacy – the ability of theory to account for the available empirical data and make good predictions about future observations – is one such criterion, and here LLMs excel. In constast to the idealised and partial nature of formal theories, the nature of LLMs is that they are trained on empirical data about language that captures the friction of the real world. So, in terms of raw predictive power, we should naturally expect the LLMs to have an advantage over principled theories. They are so good at empirical capture that to some degree they automatically satisfy the related criteria of coherence – consistency with other established linguistic theories – and scope – the ability to capture a wide range of phenomena. But while empirical capture is necessary for explanatory theories, it is insufficient.

There are several criteria where the adequacy of LLMs is unclear or debatable. Fruitfulness is a sociological criterion for goodness of explanatory theories, in that they should generate new predictions and lead to further discoveries and research. While they are certainly a potent catalyst for research in many fields even beyond machine learning, it is unclear for now how they relate to the subject matter of linguistics. Whether they satisfy Popper's criterion of falsifiability is as of yet not determined, because it is not settled how to go about falsifying the linguistic predictions of LLMs, or even express what the content of a theory embodied by an LLM is. The closest examples to falsifiability that come to mind are tests of LLM fallibility for reasoning and compositional phenomena [DLS$^+$23], or their weakness to adversarial prompt-injections [Ril22], but these weaknesses do not shed light on their linguistic competence and "understanding" directly.

Now the disappointments. As far as we can tell; LLMs are far from simple, and simplicity (Occam's Razor) is an ancient criterion for the goodness of explanation; while they exhibit, they do not explain the structure, use, and acquisition of language; they do not unify or subsume our prior understanding of linguistics. The first two points are basically unobjectionable, so I will briefly elaborate on the criterion of unification and subsumption of prior understandings, borrowing a framework from cognitive neuroscience. A common methodology for investigating cognitive systems is Marr's 3 levels [Mar10] (poorly named, since they are not hierarchical, but more like interacting domains.) Level 1 is the computational theory, an extensional perspective that concerns tasks and functions: at this level on asks what the contents and aims of a system are, to evaluate what the system is computing and why, respectively. Level 2 is representation and algorithm, an intensional perspective that concerns the representational format of the contents within the system, and the procedures by which they are manipulated to arrive at outcomes and outputs. Level 3 is hardware, which concerns the mechanical execution of the system, as gears in a mechanical calculutor or as values, reads, and writes in computer memory. In the case of LLMs, we understand well the nature of computational theory level, at least in their current incarnation as next-token-predictors, which is a narrow and clear task. Furthermore, we understand the hardware level well, from the silicon going up through the ladder of abstraction to software libraries and the componentwise activity of neural nets. Yet somehow, we know everything and nothing at once about the representation and algorithm level; we can explain how transformer models work in terms of attention mechanisms and lookback, and how it is that these models are trained using data to produce the outputs they do. In spite of understandings which should jointly cover all of level 2, we cannot relate their operations on language to our own.

But there is a worthwhile observation we can make from an understanding of the computational aims of LLMs. Insofar as the computational aim of a finished LLM is purely to predict the most plausible next token (modulo RLHF and with respect to a massive corpus), it is now an empirical fact that the artefact of language as it exists outside of human users carries sufficient structure to reconstruct the appearance of novel complex thought processes. I cannot understand why linguists are not all deeply excited at the possibilities. If it is the case that we learn such complex thought processes in the first place from language, we might elevate our consideration of language from a technology or tool to an equal and symbiotic partnership with its users as a living repository of disembodied cognition; linguists stand to be promoted from archaeologists to keybearers of *thinking*. The existence of competent non-human language users tantalises the exploration of language as a phenomenon in its own right, outside of the cognitive turn and the human perspective – consider that if aliens were discovered tomorrow, xenobiologists would simply be called biologists; why should the study of language remain parochial when the aliens landed yesterday? Plus our aliens don't mind vivisection! However, such radical reconceptions of language have not yet been articulated, so it remains that LLMs do not help linguists do linguistics in its current conception.

To illustrate the insufficiency of empirical capture to make a theory, consider the historical case study of models of what we now call the solar system. The Ptolemaic geocentric model of the solar system was more empirically precise than the heliocentric Copernican, even though the latter was empirically "more correct". This should not be surprising, because Ptolemaic epicycles can overfit to approximate any observed trajectory of planets. It took until Einstein's relativity to explain the precession of perihelion of mercury, which at last aligned theoretical understanding with empirical observation. But Newton's theory of gravity was undeniably worthwhile science, even if it was empirically outperformed by its contemporaries. Consider just how divorced from reality Newton was: Aristotelian physics is actually correct on earth, where objects don't continue moving unless force is continually supplied, because friction exists. It took a radical departure from empirical concerns to the frictionless environment of space in order obtain the simplified and idealised model of gravity that is the foundation of our understanding of the solar system and beyond. The lessons as I see them are as follows. First, aimed towards some advocates of theory-free approaches, we should belay the order to evacuate linguistics departments because performance is to some degree orthogonal to understanding. In fact, the scientific route of understanding involves simplified and idealised models that ignore friction, and will necessarily suffer in performance while maturing, so one must be patient. Second, aimed towards some theoreticians, haphazard gluing together of different theories and decorating them with bells-and-whistles for the sake of fitting empirical observation is no different than adding epicycles; one must either declare a foundational or philosophical justification apart from empirical capture (which machines are better at anyway), or state outright that it's just a fun and meaningful hobby, like painting. Third, interpretability done well requires a suitable representation and level of abstraction; imagine an epicyclist explaining the precession of mercury's perihelion by pointing at a collection of epicycles and calling it a "distributed representation", and compare to prodding subnetworks.

0.4.2 *Why Category Theory?*

The short answer: no reason. Implicitly, what's wrong with λ-calculus and whatever else? The short answer is that there's no reason to use category theory if you don't feel like it's worth the effort. It's definitely not an issue of expressivity: after all, whatever we can do with a modern programming language we can also do with punchcards in principle, and one can think of category theory as just a high-level math language that abstracts away a lot of details some may consider unimportant.

The longer answer: why not? The modeller mediates the gap between mathematics and reality by a necessarily subjective process. If formal linguistics is a hobby, then the choice of mathematics used is merely a matter of taste, and there is no need for further discussion. If however formal – explicitly *mathematical* – linguistics aspires to something universal and canonical, then it may be a good idea to start with a mathematical metalanguage where structural similarities, compositionality, and modularity are primitives. Now let me sketch why using some more complicated mathematics might in this case be a good idea.

Our capacity for language is one of the oldest and sophisticated pieces of compositional technology, maybe even the foundation of compositional thought. So, linguists are veteran students of compositionality and modularity. How does syntax compose meaning? How do the constraints and affordances of language interact? Concern number one for the formal study of language is having a metalanguage in which to build models and theories, and here for the moment we find our λs and sequents and whatever else.

Linguistics embodies a encyclopaedic record of how compositionality works "in the field", just as botanists record flowers, early astronomers the planetary motions, or stamp-collectors stamps. But a disparate collection of observations encoded in different formats does not a theory make; we will inevitably wish to bring it all together. Accordingly, concern number two for the formal study of language is having a metametalanguage with which to relate the various metalanguages. Obviously, the metametalanguage is set theory, which is the gold standard that backs everything else.

The set theoretic standard was forced by a historical lack of alternatives, and as a result, serious formal linguists are applied set theorists. However, set theory is not well-suited for complex and interacting moving parts, because it demands bottom-up specifications. So for instance if one wishes to specify a function, one has to spell out how it behaves on the domain and codomain, which means spelling out what the innards of the domain and codomain are; to specify a set theoretic model necessitates providing complete detail of how every part looks on the inside. This is an innate feature of set theory. Consider the case of the cartesian product of sets, one of the basic constructions. $A \times B$ is the "set of ordered pairs" (a, b) of elements from the respective sets, but there are many ways of encoding ordered pairs that are equivalent in spirit but not in syntax; a sign that the syntax is a hindrance, or obfuscating something important. Here is a small sampling of different ways to encode an ordered pair. Kuratowski's definition is

$$A \times B := \Big\{\{\{a\},\{a,b\}\} \mid a \in A\ ,\ b \in B\Big\}$$

Which could have just as easily been:

$$A \times B := \Big\{\{\{a,b\},b\} \mid a \in A\ ,\ b \in B\Big\}$$

And here is Wiener's definition:

$$A \times B := \Big\{\{\{a,\varnothing\},b\} \mid a \in A\ ,\ b \in B\Big\}$$

But we don't care what the precise implementation is so long as the property that $(a,b) = (c,d)$ just when $a = c$ and $b = d$ holds. The same kind of problem keeps occurring at all levels of complexity: suppose you have a set-indexed set of things $\{T_i \mid i \in I\}$, which you can choose to implement as a function $I \to \mathbf{T}$. Then somebody else wants to make the indexing set dynamically updatable with novel elements, so they have to rephrase the indexing mechanism as a set of tuples $\{(T_{i^1}, i^1), (T_{i^2}, i^2), \cdots\}$ so that they can add or remove elements, and then someone else comes along and decides that the indexes have structure that disallow certain things to be indexed... All this means that if one wants to use set theory to relate different theories at a "structural" level, one must first analyse both in terms of their constituent sets and functions in order to construct more functions between sets and functions. As you may already know if you're in the business of articulating formal systems, representation-dependency makes this process a bureaucratic nightmare.

If you are a formal linguist doing serious work with set-theoretic foundations, take this quick test to see if categories and diagrams might be right for you. For each of the statements below, note whether you agree or disagree.

- It is not fun read, write, or think with set-builder notation.
- It is difficult to relate my work to what other people are interested in.
- It is costly to tinker with and modify my framework.
- It is hard to communicate my framework to others.
- It would be nice to integrate my work with methods used in other fields, like computer science.

If you agreed with any of the above, consult your nearest category theorist to see if string diagrams are right for you. If not, have a nice day.

0.4.3 ***Objection:*** *Aren't string diagrams just graphs?*

Yes and no! This point is best communicated by a mathematical koan. Consider the following game between two players, you and me. There are 9 cards labelled 1 through 9 face up on the table. We take turns taking one of the cards. The winner is whoever first has three cards in hand that sum to 15, and the game is a draw if we have taken all the cards on the table and neither of us have three cards in hand that sum to 15. I will let you go first. Can you guarantee that you won't lose? Can you spell out a winning strategy? If you have never heard this story, give it an honest minute's thought before reading on.

The usual response is that you don't know a winning strategy. I claim that you probably do. I claim that even a child knows how to play adeptly. I'll even wager that you have played this game before. The game is Tic-Tac-Toe, also known as Naughts-and-Crosses: it is possible to arrange the numbers 1 to 9 in a 3-by-3 magic square, such that every column, row, and diagonal sums to 15.

The lesson here is that choice of representations matter. In the mathematical context, representations matter because they generalise differently. On the surface, here is an example of two representations of the same platonic mathematical object. However, Tic-Tac-Toe is in the same family as Connect-4 or 5-in-a-row on an

The deeper objection here is that diagrams do not look like *serious* mathematics. The reasons behind this rather common prejudice are worth elaborating. This is the wound Bourbaki has inflicted. Nicolas Bourbaki is a pseudonym for a group of French mathematicians, who wrote a highly influential series of textbooks. It is difficult to overstate their influence. The group was founded in the aftermath of the First World War, around the task of writing a comprehensive and rigourous foundations of mathematics from the ground up. The immediate *raison-d'être* for this project was that extant texts at the time were outdated, because the oral tradition and living history of mathematics in institutions of learning in France were decimated by the deaths of mathematicians at war. In a broader historical context, Bourbaki was a reactionary response to the crisis in the foundations of mathematics at the beginning of the century, elicited by Russell's paradox. Accordingly, their aims were rationalist, totalitarian, and high-modernist, in line with their contemporary artistic and musical fashions; they wanted to write timelessly, to settle the issues once and for all. Consequently, Bourbaki's Definition-Proposition-Theorem style of mathematical exposition is a historical aberration: a bastardisation of Euclid that eschews intuition via illustration and specific examples in favour of abstraction and generality, requiring years of initiation to effectively read and write, and remaining *de rigeur* for rigour today in dry mathematics textbooks. The deeper objection arises from the supposition that serious mathematics ought to be arcane and difficult, as most mathematics exposition after Bourbaki is. The reply is that it need not be so, and that it was not always so! The Bourbaki format places emphasis and prestige upon the deductive activity that goes into proving a theorem, displacing other aspects of mathematical activity such as constructions, algorithms, and taxonomisation. These latter aspects are better suited for the nebulous subject matter of natural language, which doesn't lend itself well to theorems, but is a happy muse for mathematical play.

unbounded grid, while the game with numbered cards generalises to different variants of Nim. That they coincide in one instance is a fork in the path. In the same way, viewing string diagrams as "just graphs" is taking the wrong path, just as it would be true but unhelpful to consider graphs "just sets of vertices and edges". String diagrams are indeed "just" a special family of graphs, just as much as prime numbers are special integers and analytic functions are special functions.

In a broader context, representations matter for the sake of improved human-machine relations. These two representations are the same as far as a computer or a formal symbol-pusher is concerned, but they make world of difference to a human native of meatspace. We ought to swing the pendulum towards incorporating human-friendly representations in language models, so that we may audit those representations for explainability concerns. As it stands, there is something fundamentally inhuman and behavioural about treating the production of language as a string of words drawn from a probability distribution. I don't know about you, but I tend to use language to express pre-existing thoughts in my head that aren't by nature linguistic. Even if we grant that the latent space of a data-driven architecture is an analog for the space of internal mental states of a human user of language, how can we know whether the spaces are structurally analogous to the extent that human-machine partnership through the interface of natural language is safe? So here again is a possible solution: by composing architectures in the shape of language from the start, we may begin to attempt guarantees that the latent-space representations of the machine are built up in the same way we build up a mental representation when we read a book or watch a film.

0.5 *Synopsis of the thesis*

I'm going to try computing the semantics of some metaphors, via syntax, using string diagrams. It doesn't interest me whether it's been done before by other formal means, I only care to demonstrate the breadth and reach of string diagrams. All of the rest of the thesis until then is in some way preparation for that exercise, and the remainder of this chapter after this section will deal with mathematical and scientific background.

I will develop some diagrammatic technology in Chapter **??**, where I introduce monoidal cofunctors for dealing with the kind of systematic relationships we see in language. In the process, I introduce and explain internal wirings, and I also explore how productive and parsing grammars ought to relate to each other in light of the fact that communication is possible.

It will then be necessary to justify some kind of systematic relationship between text circuits and something resembling text in natural language, which will be the purpose of Chapter **??**. Here I introduce weak n-categorical signatures as generalisations of string rewrite systems to higher dimensions. I demonstrate that context-free, context-sensitive, and tree-adjoining grammars are all formalisable in this one setting, in which I then construct a generative grammar that simultaneously produces grammatical text as strings of words, and the requisite structure to obtain text circuits. This relationship between text circuits and text will be encapsulated in the Text Circuit Theorem. I close this section with some discussion of ongoing practical and theoretical developments in text circuits, and point out some avenues of generalisation.

Once we have text circuits, we will need some monoidal category in which to interpret and calculate with them. In particular, it would be nice to calculate formally with the kinds of iconic cartoon representations that are typically used typically as informal schematic illustrations of metaphors. For this purpose, in Chapter **??** I introduce **ContRel**, a symmetric monoidal category of continuous relations. I diagrammatically characterise set-indexed collections of disjoint open subsets of a space – i.e. shapes with labels – as idempotents that interact with a special frobenius algebra. I will then develop a vocabulary of linguistic topological concepts so that shapes can be connected or touching or inside one another, and I will make them move and dance as I please. All of that gets done using just equations between string diagrams, and the diagrams underpinning these iconic semantics are a natural basis upon which one can perform truth-conditional analyses.

Then we will have text circuits, a formal setting to reason with and about cartoons, and diagrammatic techniques to form a structured correspondence between a text and its representation as a cartoon, at which point I will do a couple of sketches in Chapter **??**, and close with the computation of a metaphor.

1
Bibliography

[Bas22] Matthias Bastian. Google PaLM: Giant language AI can explain jokes. https://the-decoder.com/google-palm-giant-language-ai-can-explain-jokes/, April 2022.

[BBCV21] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges, May 2021. Comment: 156 pages. Work in progress – comments welcome!

[BDGHS24] Filippo Bonchi, Alessandro Di Giorgio, Nathan Haydon, and Pawel Sobocinski. Diagrammatic Algebra of First Order Logic, January 2024.

[BK20] Emily M. Bender and Alexander Koller. Climbing towards NLU: On Meaning, Form, and Understanding in the Age of Data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online, 2020. Association for Computational Linguistics.

[BM20] John C. Baez and Jade Master. Open Petri Nets. *Math. Struct. Comp. Sci.*, 30(3):314–341, March 2020. Comment: 30 pages, TikZ figures.

[BPSZ19] Filippo Bonchi, Robin Piedeleu, Pawel Sobociński, and Fabio Zanasi. Graphical Affine Algebra. In *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12, June 2019.

[BS22] Guillaume Boisseau and Paweł Sobociński. String Diagrammatic Electrical Circuit Theory. *Electron. Proc. Theor. Comput. Sci.*, 372:178–191, November 2022. Comment: In Proceedings ACT 2021, arXiv:2211.01102.

[BSZ14] Filippo Bonchi, Pawel Sobociński, and Fabio Zanasi. A Categorical Semantics of Signal Flow Graphs. In *CONCUR 2014 - Concurrency Theory - 25th International Conference*, volume CONCUR 2014 - Concurrency Theory - 25th International Conference, September 2014.

[BSZ17] Filippo Bonchi, Pawel Sobocinski, and Fabio Zanasi. Interacting Hopf Algebras. *Journal of Pure and Applied Algebra*, 221(1):144–184, January 2017.

[CG16] Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, August 2016. Comment: KDD'16 changed all figures to type1.

[CG23] Bob Coecke and Stefano Gogioso. *Quantum in Pictures: A New Way to Understand the Quantum World*. Cambridge Quantum, February 2023.

[CGG^{+}22] Geoffrey SH Cruttwell, Bruno Gavranović, Neil Ghani, Paul Wilson, and Fabio Zanasi. Categorical foundations of gradient-based learning. In *Programming Languages and Systems: 31st European Symposium on Programming, ESOP 2022, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Munich, Germany, April 2–7, 2022, Proceedings*, pages 1–28. Springer International Publishing Cham, 2022.

[Chapm] Chapman, David. Nebulosity | Meaningness. https://meaningness.com/nebulosity, Dec. 15, 2010, 10:33 p.m.

[Cho00] Noam Chomsky. *New Horizons in the Study of Language and Mind*. Cambridge University Press, Cambridge, 2000.

[Chu11] Kenneth Church. A Pendulum Swung Too Far. *LiLT*, 6, October 2011.

[CK17] Bob Coecke and Aleks Kissinger. *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, Cambridge, 2017.

[CND^{+}22] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica

Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. PaLM: Scaling Language Modeling with Pathways, October 2022.

[Coe21] Bob Coecke. Compositionality as we see it, everywhere around us, October 2021. Comment: 22 pages, lots of refs, lots of pictures, as usual.

[CSC10] Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. Mathematical Foundations for a Compositional Distributional Model of Meaning, March 2010. Comment: to appear.

[dav] davidad. An Open Agency Architecture for Safe Transformative AI.

[DLS+23] Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Sean Welleck, Xiang Ren, Allyson Ettinger, Zaid Harchaoui, and Yejin Choi. Faith and Fate: Limits of Transformers on Compositionality, June 2023. Comment: 10 pages + appendix (21 pages).

[FGP21] Tobias Fritz, Tomáš Gonda, and Paolo Perrone. De Finetti's Theorem in Categorical Probability. *josa*, 2(4), November 2021. Comment: 26 pages. v3: referee's suggestions incorporated.

[Flo14] Luciano Floridi. *The Fourth Revolution: How the Infosphere Is Reshaping Human Reality*. OUP Oxford, New York ; Oxford, June 2014.

[FP88] Jerry A. Fodor and Zenon W. Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28:3–71, 1988.

[Fre84] Frege, Gottlob. Selbst concrete Dinge sind nicht immer vorstellbar. Man muss die Wörter im Satze betrachten, wenn man nach ihrer Bedeutung fragt. In *Die Grundlagen Der Arithmetik*. 1884.

[GPM+14] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks, June 2014.

[HBK+21] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring Mathematical Problem Solving With the MATH Dataset, November 2021. Comment: NeurIPS 2021. Code and the MATH dataset is available at https://github.com/hendrycks/math/.

[Hed15] Jules Hedges. String diagrams for game theory, March 2015.

[Her12] Suzana Herculano-Houzel. The remarkable, yet not extraordinary, human brain as a scaled-up primate brain and its associated cost. *Proc Natl Acad Sci U S A*, 109 Suppl 1(Suppl 1):10661–10668, June 2012.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[HS20] Nathan Haydon and Paweł Sobociński. Compositional Diagrammatic First-Order Logic. In Ahti-Veikko Pietarinen, Peter Chapman, Leonie Bosveld-de Smet, Valeria Giardino, James Corter, and Sven Linker, editors, *Diagrammatic Representation and Inference*, Lecture Notes in Computer Science, pages 402–418, Cham, 2020. Springer International Publishing.

[JKZ19] Bart Jacobs, Aleks Kissinger, and Fabio Zanasi. Causal Inference by String Diagram Surgery, July 2019. Comment: 17 pages.

[Joy] André Joyal. THE GEOMETRY OF TENSOR CALCULUS II.

[JS91] André Joyal and Ross Street. The geometry of tensor calculus, I. *Advances in Mathematics*, 88(1):55–112, July 1991.

[Kan19] Pentti Kanerva. Computing with High-Dimensional Vectors. *IEEE Des. Test*, 36(3):7–14, June 2019.

[Kha23] Tabarak Khan. What are tokens and how to count them?, 2023.

[KLLW24] Nikhil Khatri, Tuomas Laakkonen, Jonathon Liu, and Vincent Wang-Maścianica. On the Anatomy of Attention, July 2024. Comment: Replaced to fix typos.

[KS16] Nikolaus Kriegeskorte and Katherine R. Storrs. Grid Cells for Conceptual Spaces? *Neuron*, 92(2):280–284, October 2016.

[KW23] Philipp Koralus and Vincent Wang-Maścianica. Humans in Humans Out: On GPT Converging Toward Common Sense in both Success and Failure, March 2023. Comment: 10 pages.

[Lan10] Saunders Mac Lane. *Categories for the Working Mathematician: 5*. Springer, New York, NY, 2nd ed. 1978. softcover reprint of the original 2nd ed. 1978 edition edition, November 2010.

[LAS21] Bastien Liétard, Mostafa Abdou, and Anders Søgaard. Do Language Models Know the Way to Rome? In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 510–517, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015.

[LGT23] Ziming Liu, Eric Gan, and Max Tegmark. Seeing is Believing: Brain-Inspired Modular Training for Mechanistic Interpretability, May 2023. Comment: 20 pages, 19 figures.

[LT23] Robin Lorenz and Sean Tull. Causal models in string diagrams, April 2023. Comment: 105 pages, lots of diagrams; comments very welcome.

[Mac63] Saunders MacLane. Natural Associativity and Commutativity. *Rice Institute Pamphlet - Rice University Studies*, 49(4), October 1963.

[Mar77] D. Marr. Artificial intelligence—A personal view. *Artificial Intelligence*, 9(1):37–48, August 1977.

[Mar10] David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. MIT Press, July 2010.

[MN21] Marjorie McShane and Sergei Nirenburg. *Linguistics for the Age of AI*. March 2021.

[MP19] Francis Mollica and Steven T. Piantadosi. Humans store about 1.5 megabytes of information during language acquisition. *R Soc Open Sci*, 6(3):181393, March 2019.

[NBvV22] Rick Nouwen, Adrian Brasoveanu, Jan van Eijck, and Albert Visser. Dynamic Semantics. In Edward N. Zalta and Uri Nodelman, editors, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, fall 2022 edition, 2022.

[NC22] Sharan Narang and Aakanksha Chowdhery. Pathways Language Model (PaLM): Scaling to 540 Billion Parameters for Breakthrough Performance. https://ai.googleblog.com/2022/04/pathways-language-model-palm-scaling-to.html, 2022.

[RFL$^+$24] Benjamin Rodatz, Ian Fan, Tuomas Laakkonen, Neil John Ortega, Thomas Hoffman, and Vincent Wang-Mascianica. A Pattern Language for Machine Learning Tasks, July 2024.

[Ril22] Riley Goodside (@goodside) / Twitter. https://twitter.com/goodside, December 2022.

[RM87] David E. Rumelhart and James L. McClelland. Learning Internal Representations by Error Propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*, pages 318–362. MIT Press, 1987.

[Sea80] John R. Searle. Minds, brains, and programs. *Behavioral and Brain Sciences*, 3(3):417–424, September 1980.

[Sel10] Peter Selinger. A survey of graphical languages for monoidal categories. volume 813, pages 289–355. 2010.

[Sob15] Paweł Sobociński. Graphical Linear Algebra. https://graphicallinearalgebra.net/, June 2015.

[Søg23] Anders Søgaard. Grounding the Vector Space of an Octopus: Word Meaning from Raw Text. *Minds & Machines*, 33(1):33–54, March 2023.

[Sut19] Richard Sutton. The Bitter Lesson, 2019.

[ted22] teddy [@teddynpc]. I made ChatGPT take a full SAT test. Here's how it did: https://t.co/734sPFU3HY, December 2022.

[TGZ+23] Taori, Rohan, Gulrajani, Ishaan, Zhang, Tianyi, Dubois, Yann, Li, Xuechen, Guestrin, Carlos, Liang, Percy, and Hashimoto, Tatsunori B. Stanford CRFM. https://crfm.stanford.edu/2023/03/13/alpaca.html, 2023.

[Tho22] Alan D. Thompson. GPT-3.5 IQ testing using Raven's Progressive Matrices, 2022.

[Tom22] Tom Goldstein [@tomgoldsteincs]. Training PaLM takes 3.2 million kilowatt hours of power. If you powered TPUs by riding a bicycle, and you pedaled hard (nearly 400 watts), it would take you 1000 years to train PaLM, not including bathroom breaks. In that time, you'd make 320 trips around the globe!, July 2022.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, December 2017. Comment: 15 pages, 5 figures.

[Wol02] Stephen Wolfram. *A New Kind of Science*. Wolfram Media Inc, Champaign, IL, illustrated edition edition, August 2002.

[WWS+23] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models, January 2023.