Building Blocks for Data-Driven Theories of Language Understanding

Julian Michael

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2023

Reading Committee:

Luke Zettlemoyer, Chair

Noah A. Smith

Emily M. Bender

Program Authorized to Offer Degree:

Paul G. Allen School of Computer Science & Engineering

University of Washington

Abstract

Building Blocks for Data-Driven Theories
of Language Understanding

Julian Michael

Chair of the Supervisory Committee:
Luke Zettlemoyer
Paul G. Allen School of Computer Science & Engineering

I propose a paradigm for scientific progress in natural language processing, centered around the development of *data-driven theories* of language understanding. The central idea is to collect data in tightly scoped, carefully defined ways which allow for exhaustive annotation of a behavioral phenomenon of interest. With such data, we can use machine learning to construct explanatory theories of these phenomena which can be used as building blocks for intelligible AI systems. After laying some conceptual groundwork for the idea, I describe a series of investigations into the development of data and theory for representations of shallow semantic structure in natural language — in particular, using Question-Answer driven Semantic Role Labeling (QA-SRL), a simple schema for annotating verbal predicate-argument structure using highly constrained question-answer pairs. While this just scratches the surface of the complex language behaviors of interest in AI, I outline principles for data collection and theoretical modeling which can inform future scientific progress.

TABLE OF CONTENTS

LIST OF FIGURES

Figure Number Page

LIST OF TABLES

Table Number Page

GLOSSARY

GLOSSARY: A list of opinions or beliefs about how terms should be used. A descriptive notion of glossary is also possible, but this glossary prescribes against it.

ARTIFICIAL INTELLIGENCE (AI): The study of intelligent behavior: how to perceive, reason, and act in service of discovering truth and achieving goals.

While humans provide a useful example and existence proof for certain aspects of intelligence, AI is concerned with intelligence more broadly, and not just describing or imitating human behavior.

COMPUTATIONAL LINGUISTICS: The study of human language, either viewing it as a computational system or performing analysis using computational models and tools.

MODEL: A representation of a phenomenon which produces predictions about it.

"Models," in common parlance, simplify their subjects and make them more accessible to the human mind, so humans can make these predictions. For example, the *heliocentric model* of the solar system gave Johannes Kepler a way of predicting the motion of planets in the sky. In the context of modern AI, on the other hand, the term "model" generally refers to a machine learning model which outputs predictions directly. The workings of the model don't need to be accessible to the human mind (i.e., subject to familiar patterns of reasoning) in order for the model to be useful for making predictions; instead, the human is tasked with interpreting its inputs and outputs in the context of their knowledge and goals (i.e., modeling the model). The widespread usefulness of this looser notion of "model" is a relatively recent development. It has gained prominence together with the success of what Breiman (2001) calls the "algorithmic culture of statistical learning," which seeks to model phenomena with the highest-possible fidelity by minimizing the assumptions built into the modeling process and doing model selection on the basis of predictive accuracy.

NATURAL LANGUAGE PROCESSING (NLP): The study of behaviors that exhibit or require competency with human language.

As an engineering practice, NLP comprises the set of tools and techniques for building computational systems that operate on linguistic symbols (e.g., text) or signals (e.g., audio data of speech), with a focus on performing tasks involving language competency. As a science, NLP is concerned with developing an understanding of the nature of this competency, in the form of computational theories of language-based tasks. There is considerable overlap between computational linguistics and NLP, particularly on the scientific side. The primary difference is that the object of study in computational linguistics is the human language system, whereas in NLP it is the set of intelligent behaviors involving human language. These behaviors do not need to be performed reliably (or at all) by humans, placing NLP under the umbrella of artificial intelligence.

THEORY: Common ground between a set of humans or humans and machines, allowing for systematic agreement between all parties on the expected outcome of a prediction problem.

In common parlance, a "theory" of a phenomenon is often understood to be an explanation of *why* that phenomenon happens the way it does. In my view, explaining a phenomenon fundamentally comes down providing a way to predict it correctly, especially in unseen conditions. So a theory is much the same thing as a model (see MODEL above). But I think there are two important distinctions between models and theories, insofar as the words are used in practice.

First: Viewing models and theories each as things which allow for making predictions, we can ask: *who* or *what* do they allow to make predictions? Models in the modern statistical sense (*i.e.*, machine learning models) make predictions about their data distribution, but — especially in the case of black-box neural networks — their outputs cannot generally be predicted by humans doing anything short of running the model on a computer. In contrast, I think the term *theory* is generally used to refer to models *which are amenable to manipulation by the human mind*, where humans can use them to systematically reason about a domain and predict outcomes.

Second: Theories carry the sense of being claims at *truth*, which is why we talk about theories being "falsified" but not models. Given this, a theory must provide objective prescriptions. Humans using a theory might make reasoning mistakes, inconsistent decisions, or judgments on the basis of unformalized and unarticulated intuition. To ensure that a theory is well-defined, systematic, and objective, we can require that multiple humans are able to reproduce each other's predictions, or formalize their understanding to the point that it is implementable in a machine (where in the term "machine" I include abstract machines, such as formal systems like mathematics; a machine can be implemented in a human mind). In practice, verifying that this understanding is shared between people (and/or machines) requires successful communication between all parties about the prediction problem of concern and the all details of the theory — *i.e.*, establishing it in the *common ground* (see Chapter 2). Furthermore, such communication is also what makes theories *useful*, allowing humans and machines to jointly reason and coordinate action in the service of shared goals and values. For these reasons, I choose to ground my definition of "theory" in the notion of common ground.

ACKNOWLEDGMENTS

At the moment that I am writing this, my PhD accounts for over 25% of my time on Earth. I will fall far short of capturing the profound influence that so many people had on me in the lives I lived during this time. This work is the product the warmth, joy, support and strength brought to me by my friends, family, and colleagues. And, maybe a bit of their sweat and tears too.

First, my research mentors and collaborators: my advisor Luke Zettlemoyer has been utterly steadfast, was always available when I needed him, and taught me a great deal about how to do good science. My early PhD mentors Mike Lewis and Omer Levy were wonderful guides in how to think about NLP, and I am grateful to Omer seeing the potential for me to help with GLUE and connecting me to a rich network of collaborators, including Sam Bowman who I work with today. I could not have asked for a better collaborator than Luheng He to show me the ropes on my first project, and I would not have survived my first three years without Kenton Lee always entertaining my crazy ideas, deeply engaging me and challenging me at every turn. It has been a pleasure collaborating with Gabi Stanovsky, Ayal Klein, Paul Roit, Valentina Pyatkin, and others on QA-SRL related projects, and of course Ido Dagan who has advised much of this work, as well as Ian Tenney and Jan Botha who mentored me at Google. Thanks as well to my thesis committee members Noah Smith, Emily Bender, and Shane Steinert-Threlkeld.

Later in my PhD I had the privilege of learning from amazingly talented students who were junior to me, including Sewon Min, Bhargavi Paranjape, Weijia Shi, Margaret Li, Alisa Liu, and many others. I will never forget when I asked Sewon about whether we were ready to start collecting data for AmbigQA, only for her to tell me it was already halfway done; or the countless conversations with Alisa about crazy, nuanced, ambiguous, and occasionally hilarious entailment examples generated by GPT-3. And while I did not collaborate with Ofir Press, I will never forget his earth-shaking laugh, thanks to him sitting three inches behind me for my last couple of years in Seattle.

Of course life is about more than just research, and I am forever grateful for the support and connection of friends like Leon Gatys — the simple happiness and peace that he brings everywhere he goes has been an incredible grounding force for me, not to mention our handstand/ice cream sessions. Ari Holtzman has enriched my intellectual and emotional life in ways I never would have expected. Yanai Elazar and Valentina Pyatkin somehow made *me* feel welcome in Seattle when *they* were the ones moving in. Amandalynne Paullada has taught me so much, and been a wonderful friend and emotional support at some of the times I needed it most (including in the writing of this thesis). I will always fondly remember learning to ski with Lucy Wang and injuring Ana Marasović on the slopes, and I look forward to many more adventures (but hopefully no more accidents or lost skis). The serendipity of meeting Beatriz Saldaña was a gift, as was the time spent climbing with her, Leon, Haraldur Hallgrimsson, and others. I am also grateful to the continuing friendship of so many people from before my PhD, including Grace Shiau, Asya Bergal, Hang Lu Su, Angela Deng, Angi Shen, Keerthana Kumar, Michael Schnebly, Sebastian Mejia, and many others who have taught me and made my life full.

Dustin Carlino, my roommate for more than five years, was a never-ending source of joy, intrigue, inspiration, motivation, common sense, uncommon silliness, outright confusion, laughter and friendship. From introducing me to parkour, to going on countless adventures, to sharing the travails of A/B Street, Dustin's presence was a rock in my life — or maybe instead of a rock, an ever-present phantom staring at me from the dark, wearing a mysterious mask and murmuring about jumps and geometry and urban infrastructure and occasionally Charles Bukowski. Kiet Phong, Cody Stetzel, Danny Schlitt, Mandy Gabriel, and Savino Petrignani brightened my life in so many ways, and Filip Tuhy and Sheep were truly pillars of my home and community, welcoming me warmly while pushing me out of my comfort zone and widening my world immeasurably. I'm thankful to the many coaches, mentors, and movers from the Seattle parkour community who helped me find my way in my body, especially Bryan Riggins who first shepherded me into the practice in his classes. And I am deeply, deeply grateful to my yoga therapist Robin Rothenberg, as well as my bodyworkers Morgan Houghton and Dave Engstrom, for helping me repair the connection between my mind and body when they were shorn apart.

One of the greatest gifts that life brought me during the PhD was a newfound closeness with my brother Jonathan. His unconditional love, his unrelenting curiosity and intellectual drive, and our hours on the phone sharing our journeys with each other have both served as a bedrock of emotional strength for me and fundamentally shaped my research. I am forever grateful to my late father, who gives me strength and humor every day, as well as my brother Sean, and finally my mother, whose wisdom, love, and support underly everything I do.

DEDICATION

To my father.

Chapter 1

INTRODUCTION: A CRISIS OF THEORY

Intelligence is whatever machines haven't done yet.

— Larry Tesler, ca. 1970[1]

Artificial intelligence, more than many fields of study, has a habit of eluding precise definition. In what has been called the "AI Effect" (Hofstadter, 1979), once we understand how to build a machine to do something, it is common to no longer see that behavior as requiring true "intelligence." Minsky (1990) writes (emphasis mine):

> Even though we don't yet understand how brains perform many mental skills, we can still work toward making machines that do the same or similar things. "Artificial intelligence" is simply the name we give to that research. But as I already pointed out, this means that the focus of that research will keep changing, since as soon as we think we understand one mystery, we have to move on to the next. In fact, AI research has made enormous progress in only a few decades, and because of that rapidity, the field has acquired a somewhat shady reputation! This paradox resulted from the fact that whenever an AI research project made a useful new discovery, that product usually quickly spun off to form a new scientific or commercial specialty with its own distinctive name. **These changes in name led outsiders to ask, Why do we see so little progress in the central field of artificial intelligence?** Here are a few specialties that originated at least in part from AI research but later split into separate fields and, in some instances, commercial enterprises: robotics, pattern recognition, expert systems,

[1] http://www.nomodes.com/Larry_Tesler_Consulting/Adages_and_Coinages.html

> automatic theorem proving, cognitive psychology, word processing, machine vision, knowledge engineering, symbolic applied mathematics, and computational linguistics.

Today, it is safe to say that very few people are asking: *why do we see so little progress in artificial intelligence?* Not only is progress in AI widely regarded as happening at breakneck speeds, but some of the subfields Minsky mentioned as non-central to AI, particularly machine vision and word processing (or more broadly, natural language processing), are now seen as core areas of study within AI and are often branded as such.

So, what changed? In the case of natural language processing (NLP), I think there are two main reasons the AI Effect has weakened:

- **Machine Learning.** The primary factor is almost certainly the dominance of machine learning in AI. Machine learning, especially in the case of deep learning, has the relatively unique property that it allows us to build machines which perform certain behaviors *without us fully understanding those behaviors*. This allows us to claim success in building AI systems without losing the "magic" of intelligence.[2]

- **The Eliza Effect.** Discussed more in Chapter 2, people tend to attribute human qualities to machines that produce language, even if they know that language is generated by simple rules or programs. This counters the AI Effect and helps NLP systems retain their magic, especially in the case of language generation.

Before the dominance of machine learning in AI, one might regard the role of AI scientists as being *to automate themselves away*: to take seemingly ineffable aspects of intelligent behavior and characterize them well enough to relegate them to formal study in (potentially new) non-AI fields. For example, logic-based AI researchers may describe their work as "discovery, refinement, and formalization of the basic categories of our language and thought" (Gelfond, 2021). But

[2] By "fully understanding," I mean the ability to manually code such behaviors in a traditional programming language and to reason about emergent patterns in the results on a high level. What it means for us to "understand" our systems will be discussed in more detail in Chapter 2.

machine learning, and especially deep learning, leads to a different dominant *modus operandi*: building systems which we don't understand, but which exhibit behaviors which we associate with intelligence.

This is, in many ways, a positive development. It has allowed for the development and proliferation of myriad technologies and products, and we don't always need to understand exactly how something works in order for it to be useful to us. For example, it is likely that no individual person deeply understands how every component of a computer works — and yet many people can productively and enjoyably use computers. In addition, there is reason to believe that even if our goal is to understand the world better, or build machines that understand the world better, we must start with only the most general principles of learning and abstraction and let machines figure out the details on their own. Based on a career of experience in AI, Sutton (2019) writes:

> We have to learn the bitter lesson that building in how we think we think does not work in the long run... the actual contents of minds are tremendously, irredeemably complex; we should stop trying to find simple ways to think about the contents of minds, such as simple ways to think about space, objects, multiple agents, or symmetries. All these are part of the arbitrary, intrinsically complex, outside world. They are not what should be built in, as their complexity is endless; instead we should build in only the meta-methods that can find and capture this arbitrary complexity.

This lesson has, to some extent, been borne out by recent progress in training large models on as much data as possible (Devlin et al., 2019; Liu et al., 2019b; Brown et al., 2020; Chowdhery et al., 2022; Radford et al., 2021; Ramesh et al., 2022).

However, the Bitter Lesson leaves something behind: *intelligibility*. Large-scale machine learning systems doubtlessly can catch on to a great deal of the structure that appears in their training data. But if we don't understand what that structure is — if we don't have any sort of handle on the "irredeemably complex" functions of intelligent behavior, then we don't understand what those systems are doing. As well as they may optimize their objective on their training data, in many cases it's unclear *a priori* whether that will translate into the desired outcomes when the machine is put out into the world. In order to trust AI systems, be able to anticipate and correct their failures, and integrate them with other software, we need to be able to reason effectively about their behavior. Building machines which are “more accurate” only benefits us to the point that we have a *theory* of *what they’re more accurate at doing*, *i.e.*, of the task we wish to accomplish. The notion of theory will be discussed in more detail in Chapter 2.

There is a fundamental tension at work here. My view is that producing intelligible machines is, at its core, about building machines which we can successfully model (*i.e.*, we have good theories of their behavior). However, if we had a correct and complete model of the intelligent behavior we wished to implement, then we wouldn’t need deep learning to do it. And as of today, deep learning on big data has brought unprecedented and fast progress in building machines, while trying to construct theories of intelligent behavior hasn’t brought the same sort of success. I don’t think it’s an overstatement to say that the field of natural language processing is facing a *crisis of theory*. As progress in emulating human behavior has seemed to rapidly outpace our ability to understand our models, should we just throw up our hands and embrace the chaos? What do we make of the original scientific challenge of AI? To develop *theories of intelligent behavior*, or of language understanding? This thesis explores these questions, with a particular focus on representations of sentence-level linguistic structure and meaning.

Stated broadly, my position is this: We must take Sutton’s Bitter Lesson to heart. Intelligence and language are irredeemably complex, and we cannot expect good results from simply programming them the way we intuitively think they should work. Rather, we must be empirical in our approach and be driven by data. However, being empirical does not mean being *atheoretical*: on the contrary, data has no meaning — no connection to anything — without a theory that allows for its interpretation (see Part I). Instead of programming machines according to “how we think we think” *a priori*, we should try to develop what I will call **data-driven theories**, using machine learning to help us construct comprehensible theories that capture tractable subsets of the complexities of language and the world, as they appear in our data. These theories can help situate our machines in the world (Chapter 2), and provide conceptual and programmatic handles which allow us to engineer new capabilities into our systems (Part III). In this thesis, I lay out the basic tenets of data-driven theory as they apply to NLP. The paradigm requires thinking carefully about our approaches both to data collection and modeling to facilitate the production of systems which capture progressively more of the true complexity of natural language, in a way which scales with data while remaining comprehensible and engineerable.

The rest of this document is organized as follows:

- **Part I — NLP as a Science: Data-Driven Theories.** First, I lay some conceptual groundwork. In Chapter 2, I explicate the importance of *theories* of task performance and model behavior as the basis for AI systems which use language in a way that can be said to share common understanding with humans. Then in Chapter 3, I present an epistemological framework under which these theories can be developed on the basis of data at large scale.

- **Part II — Data: Annotating Natural Language with Natural Language.** I lay out *Four Principles of Scientific Data for NLP*, a guiding framework for data collection for the purpose of developing data-driven theories. I explore these principles through case studies in data collection and modeling for sentence-level predicate-argument structure (Chapters 4 and 5) and discuss the advantages of Question-Answer driven Semantic Role Labeling (QA-SRL) as a representation for this purpose.

- **Part III — Theory: From Language, Structure.** I discuss some of the challenges of developing useful theories in a complex domain like language AI, and how it may be done on the basis of the right data. As an example, I show how to induce a theory of semantic roles from QA-SRL, and discuss recent work employing QA-SRL to add capabilities to a question generation system which were not directly available in its training data.

- **Part IV — Concluding Thoughts.** I summarize the main tenets of data-driven theory and briefly discuss future directions.

Part I

NLP AS A SCIENCE: DATA-DRIVEN THEORIES

In Chapter 1, I presented the basic idea of data-driven theory: our goal is to use machine learning and data curation to develop a better understanding of intelligent behavior and language use. In this part of the thesis, I explore what this means and propose some basic ideas for how and why to do it.

Chapter 2 presents a foundational discussion of the role of theory in language understanding systems. Core to my proposal is to model the use of AI systems as a *signaling game*, where in order to understand how to make decisions on the basis of a model's output, a human must reason about it on the basis of *common ground* shared with that model. For our discussion, this common ground takes the form of *theories* of two kinds:

- **Model Theory:** A theory of how a model behaves and what it reflects about the world. These are what allow us to draw conclusions about the world and take action on the basis of a model's outputs.

- **Task Theory:** A theory of what it means to do a task, and what relates that task's inputs and outputs. These are what allow us to coordinate around the design and use of datasets and models so we can employ them in service of our collective goals.

Both kinds of theories can be developed through exploration of either datasets or models, and I present some examples of how to do so. I propose such theories as a missing link beyond benchmark performance, allowing us to reason about our models beyond what ostensive measures on datasets allow.

However, we quickly run up against the irredeemable complexity of language and the world: full, comprehensible theories of language meaning and use require the representation of a great deal of linguistic and world knowledge. Manual development of such theories is impractical, if not impossible. This fact has led many, in the spirit of Sutton's Bitter Lesson, to advocate for abandoning the so-called "Rationalist" program of specifying theories of specific intelligent behaviors in favor of the "Empiricist" program of encoding only the most general learning principles into AI systems and letting them learn on their own.

In Chapter 3, I present an alternative framework based in the epistemology of *Pragmatism*, which acknowledges the advantages of both the Rationalist (theoretical) and Empiricist (data-driven) approaches to AI. As a pragmatist program of progress, I outline an approach to the development of *scalable theories*, embracing the idea that a theory should be the best explanation of data, and the fact that machine learning is an ideal tool both for simulating and explaining data. The essential idea is to collect relatively theory-agnostic data at large scale, and then to model it using carefully chosen inductive biases that allow for the automatic construction of computational theories of the phenomena that data captures.

Parts II and III will put the ideas from this part into practice, developing the basic components of a scalable approach to shallow semantic representations. This thesis necessarily covers only a very small part of the story. The Empiricist paradigm has undoubtedly advanced our engineering ability, but it has offered us less when it comes to deeply understanding what we are doing, and how to better characterize our objectives. This problem is increasingly important as NLP systems see widespread deployment. Altogether, I hope the chapters in this part present a thought-provoking alternative to the Empiricist mainstream line of thought in NLP.

Chapter 2

COMMON GROUND: A BASIS FOR ASSIGNING MEANING TO MACHINE LANGUAGE

I present a philosophical framework for human–machine communication and language understanding. My key assumption is to view language not as an independent object which may be "understood," but as a medium which agents may use to understand *each other* and the world. Drawing on the philosophy of language, I argue that the key prerequisite to successful communication between humans — as strategic, cooperative communicators — is the development of *common ground*: working assumptions which are mutually adhered to, and assumed to be in the common ground, by all parties. Generalizing this, I argue that successful communication between humans and machines relies on *theories of behavior* which are implemented by the machines and understood by the humans. I outline two paradigms for advances in language understanding systems in this spirit: analyzing black-box models to derive theories of their behavior, constructing models on the basis of prescriptive task specifications to allow for formal analysis.

2.1 Introduction

The *ELIZA Effect*, named after a 1966 chat bot by Joseph Weizenbaum, is the human tendency to attribute to software greater abilities than it actually has, particularly when it produces fluent natural language. Weizenbaum noticed that users of his system quickly became emotionally involved with it, "conversing with the computer as if it were a person who could be appropriately and usefully addressed in intimate terms" (Weizenbaum, 1976, p. 7). He continues:

> Another widespread, and to me surprising, reaction to the ELIZA program was the spread of a belief that it demonstrated a general solution to the problem of computer understanding of natural language... This reaction to ELIZA showed me more vividly

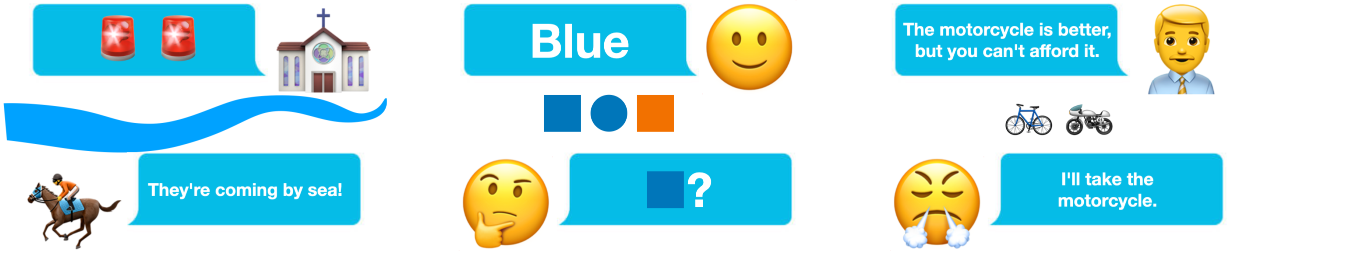
> than anything I had seen hitherto the enormously exaggerated attributions an even well-educated audience is capable of making, even strives to make, to a technology it does not understand. (Ibid, p. 7)

Today more than ever, AI systems perform a wide range of classification tasks with high accuracy, produce natural language text which is highly fluent and diverse, and about them we understand preciously little. As Bender and Koller (2020) argue, we should be careful not to impute an understanding of language meaning, particularly in terms of communicative intent, on systems which do not possess intent and have not been exposed the external world with which that meaning is concerned.

Yet, the imputation of "meaning" in a broader sense is inevitable: anyone who is using an AI system for something is interested in what its outputs *mean* for the task they are trying to accomplish. When a machine produces high-accuracy classification results, or seemingly fluent, relevant, and correct natural language text, we are faced with the question of what to do with it: *what kinds of meaning can a machine convey, and how?* This question is crucial for guiding the development of AI systems, as it is the basis upon which we use these systems to make decisions in the world.

In this chapter, I propose a framework in which to answer this question. My key assumption is to view symbolic outputs of a machine (such as classification labels or natural language text) as utterances in a signaling game (Lewis, 1969) aimed at communicating underlying meanings between agents (Section 2.2.1). In light of this, asking whether a system "understands language" misses something fundamental to language, which is that it is a *medium* through which agents may understand *each other* — that is, language understanding is not a property of a system, but a relation between systems (and/or humans).

Within this framework, the key prerequisite to mutual understanding between humans is the existence of *common ground*: working assumptions of communication which are mutually adhered to, and assumed to be in the common ground, by all parties (Section 2.2.2). Generalizing this, I argue that successful communication between humans and *machines* relies on *theories of behavior* which are implemented by the machines and understood by the humans. I illustrate this argument


They're coming by sea!
Blue
?
The motorcycle is better, but you can't afford it.
I'll take the motorcycle.

(a) **Paul Revere's Ride** (Lewis, 1969). According to legend, American Revolutionary Paul Revere used a signaling protocol to learn from across the Charles river when a British invasion began: one light in the Church tower if they came by land, two if by sea.

(b) **Reference games** (Frank and Goodman, 2012). Interlocutors reason about each other's speech situations to deduce meanings. Here, the listener understands "blue" to refer to the blue square, since the circle has a better referring expression available.

(c) **Psychological games** (Berne, 1964). The salesman's implicit meaning that the customer is of low status is (also implicitly) refuted by the customer. The sale succeeds because of mutually held assumptions about human psychology and society.

Figure 2.1: Signaling games of varying complexity.

with a thought experiment, *The Dinner Non-Dilemma* (Section 2.4), demonstrating the role of these theories being 1) correctly implemented by the machines, and 2) in the common ground between the users and designers of the systems. I then outline two ways to use these theories: as *descriptive* theories of black-box models (Section 2.5), or as *prescriptive* theories according which to construct models (Section 2.6.1).

2.2 Background: a Primer on Communication

To motivate my approach, I will first examine the problem of communication in the abstract. This will provide us with general concepts and principles which we will later apply to the problem of human communication with machines (Section 2.3).

2.2.1 Signaling Games

A straightforward framework for understanding communication is Lewis (1969)'s *signaling games*. The simplest form of a signaling game consists of a *speaker* S and a *listener* L. S has in mind a *meaning* $m \in M$ (corresponding to, e.g., a world state, desired action, or other information relevant to the speaker's goals). S can only communicate to the listener by sending them a single utterance $u \in U$, where $|M| \leq |U|$ and these sets have no other structure. The listener L knows the set of possible meanings M but does not have direct access to m. The goal of both agents is the following outcome:

Goal (successful communication): *L correctly guesses S's meaning m on the basis of the their utterance u.*

This goal is shared by both agents, so we say they are *cooperative* and *strategic*. Real signaling games may be as simple as this one or much more complex, with a variety of notions of "meaning" (see Figure 2.1 for some examples). We may understand the goals of each agent as follows:

Speaker's goal: Given a meaning m, produce an utterance u from which the listener will derive the meaning m.

Listener's goal: Given an utterance u, derive the meaning m from which the speaker would have produced u.

Not only does the ability of a speaker and listener to communicate successfully depend on their models of each other, but these models are mutually recursive: the listener must model the speaker's model of the listener's model of the speaker, and so on, *ad infinitum*. While this circularity may seem like a problem, the mutual modeling approach has been fruitfully applied in Gricean pragmatics (Grice, 1975; Frank and Goodman, 2012) and game-theoretic approaches to natural language semantics (Clark, 2012). The key to overcoming the circularity inherent to this problem is the idea of **common knowledge**.

2.2.2 Common Knowledge and Common Ground

A group of agents has *common knowledge* of a proposition P when everyone in the group knows P, everyone in the group knows that everyone knows P, and so on, *ad infinitum* (Lewis, 1969). Despite requiring an infinite regress to state mathematically, this kind of shared knowledge is ubiquitous in practice. Friedell (1969) gives the example of two mathematicians A and B who know each other well. We may write:

(1) A knows that $2 + 2 = 4$.

(2) B knows that A knows that $2 + 2 = 4$.

(3) A knows that B knows that A knows that $2 + 2 = 4$.

$\ldots$

And so on. It is easy to imagine both A and B agreeing that every statement in this infinite sequence is true.

Now consider an injective function $f\colon\ M \hookrightarrow U$ from meanings to utterances in the context of the simple signaling game in Section 2.2.1. The speaker S will only use this function to produce utterances if they know that the listener L knows that they will use it, which will only be the case if L knows that S knows that L knows, etc.; these conditions are satisfied when S and L are able to coordinate on f beforehand and establish it as common knowledge. This then allows them to communicate optimally.

Adapting this theoretical solution to realistic scenarios requires a slight change. In practice, agents often do not have the opportunity to coordinate on common knowledge about their communication protocol in advance, so even if it is shared between them, it can not be *known* to be shared (or known to be known to be shared, etc.). So instead of speaking of *common knowledge*, the best an agent can do is *presuppose* working assumptions about communicative conventions, use these presuppositions a practical basis for producing and interpreting messages, and revise or build on top of them as they interact with others. Roughly following Stalnaker (2002), let us say that an agent *presupposes* a proposition P among a group of interlocutors if they act as if they take P for granted, and also presuppose that everyone in the group presupposes P. This allows us to define the following notion:

Definition (common ground). *The common ground among a set of agents is the information which is presupposed by all agents in the set.*

Common ground underlies the use of communicative convention in light of the “arbitrariness of the sign” noted by de Saussure (1916), particularly in the setting of strategic, cooperative agents. The game-theoretic framework of signaling games assumes that each agent’s status as strategic and cooperative is in the common ground as well; this assumption is operationalized in the theory of Rational Speech Acts (Frank and Goodman, 2012) to explain pragmatic inferences. Approaches to pragmatics such as that of Grice (1975, 1989) add additional assumptions (called *maxims of communication*) to the common ground to explain pragmatic inference in a similar way. Even in the context of conventional semantics, many approaches to language meaning explicitly model discourse as a process of updating or adding to the common ground (Stalnaker, 2002) or other bodies of information shared by interlocutors (Portner, 2004), an approach broadly termed *dynamic semantics* (Kamp, 1981; Heim, 1983; Stalnaker, 1978).

Studies also suggest that acquiring and building common ground is integral to human language acquisition. Baldwin (1995) argue for the importance of joint attention in language learning, where there is mutual awareness (*i.e.*, common ground) that a child and their caregiver are attending to the same thing. Not only has joint attention been experimentally demonstrated to facilitate language learning in children (Tomasello and Farrar, 1986; Brooks and Meltzoff, 2005),[1] but Tomasello et al. (2007) argue that infants actively seek out shared intentionality (leading to joint attention) when pointing to objects. Computationally speaking, the importance of joint attention — a lived experience of mutual knowledge — makes sense as a base case of common ground upon which further communication can grow it into a full language system.

2.3 Communication with Machines

From the arguments in the previous section, the importance of common ground for communication between humans seems clear. But we are interested in the question of communication and language

[1] See Bender and Koller (2020) for more discussion of these references. They argue that that language models, lacking intersubjectivity or experience of things like joint attention, cannot learn meaning. While I agree about the importance of joint attention in humans, I embrace a broad picture of the acquisition of meaning. Joint attention may be one way of bootstrapping common ground; in Section 2.3, I present alternatives which are more directly tailored to communication using machines.

understanding with *machines*. Can common ground exist between a human and machine?

As described in Section 2.2.1, successful communication between strategic, cooperative agents requires each agent's model of the other's behavior to converge on a shared mental model of what connects symbols to their meanings or referents in the world. Communication between humans succeeds to the extent that human experience grounds our language in similar concepts, and language communities can coordinate on common conventions to express these concepts (Lewis, 1969). Machines, however, do not necessarily leverage the same concepts or have access to the same conventions, and they do not necessarily behave as strategic, cooperative agents modeling their interlocutors.

This puts us in a relaxed version of Lewis's signaling games. Consider the case where the speaker S is a machine and listener L is a human. Again, we can view S as a function $f\colon\ W \to U$ from world states $w \in W$ to utterances $u \in U$.[2] However, in this case f is not necessarily strategically chosen — it may be fixed, *e.g.*, if f is directly programmed into S by its designers. The goal of L remains the same: to infer the current world state w, or some information about it, from the machine's utterance u. Success in this game simply requires L to have a model of f^{-1}. (A symmetric argument applies to the case where S is a human and L is a machine.)

Said another way, successful communication involving humans and machines relies on humans having a good understanding of *how machine behavior reflects and impacts the world*. This implies that the problem of producing *language understanding systems* as such is the problem of building *systems which use language in human-modelable ways*. Building modelable machines in this sense means building machines whose behavior we can understand and reliably connect to the external world in which they're operating to gain insights about it.

I see two main scenarios for this, though I do not claim that this breakdown is necessarily systematic or exhaustive:

1. **Descriptive theories** of the emergent behavior of black-box systems (Section 2.5). Large-scale deep learning models, such as large language models, encode a lot of information about

[2] Formally, this is the same as the speaker in Section 2.2.1, but I refer to a function of "world states" rather than "meanings" to avoid the implication that the machine necessarily has an "intended meaning" in mind.

their training corpora, but it is not clear *a priori* what we can conclude about their training process, or the world more broadly, from their behavior. Developing theories of how deep learning models work will allow us to draw better conclusions about *how to act* on the basis of their outputs.

2. **Prescriptive theories**, or *specifications* of machine behavior according to which we intentionally design systems to reliably behave (Section 2.6.1). Instead of building purely black-box models and then studying how they work, we can directly develop theories of the tasks that we want to achieve — *e.g.*, strategies of how to decompose a problem — which we can use to structure our models, providing strong guarantees about their behavior. In this case, successful communication with machines relies on the existence of common ground between a system's users and its *designers* who specify its behavior.

In addition to providing guiding principles for NLU research, I think this perspective invites a simple resolution to the highly-debated question of whether language models (or any machine learning systems) can "understand language." Instead of debating what kind of behaviors or prior conditions must hold in order for a system to be objectively said to "understand language," we can regard language understanding as a contextually-dependent relation between humans and machines, where the nature and degree of understanding about the world that humans can gain from machines (or the effectiveness with which the human can delegate action in the world to the machine) is built on the fidelity of the human's model of the machine's behavior. Put another way, this is the combined extent to which 1) the system models something about the world *and* 2) we can reliably model this model and interpret what it means. Note that this does not necessarily privilege machines which *sound human* — rather, it privileges machines which which have *clearly specified, interpretable language behavior*, which may or may not be human-like.

Before diving into each of the three above scenarios in turn, I will illustrate the main idea in more detail with a thought experiment.

2.4 A Thought Experiment: The Dinner Non-Dilemma

> It is just before 5pm and you are at your workplace. Your phone produces a familiar noise: a high pitched *ding*. You look at the screen and see a panel appear with the name of a friend who works near you. It also contains the text: “Dinner?” You conclude that your friend is thinking about having dinner with you tonight and wants to know whether you’re available and interested. You are correct.

Communication succeeded — but how? This feat relies on **three** interrelated communication systems, which together will form our understanding of the domains of common ground which must be established in order to integrate machines into the communication process.

2.4.1 Human↔Human: Language

On the surface level, our thought experiment is an instance of language communication between humans. Decoding your friend’s mental state from their very brief message leverages a great deal of common ground. First, there must be a mutually understood concept of “dinner” associated with the word. Second, understanding that the dinner in question will be *tonight* is an example of the use of a game-theoretic *focal point* (Schelling, 1960): it is the only reasonable default in the absence of a specified time; since your friend knows that you will default to it, they don’t need to say it. And finally, decoding your friend’s idea of *having dinner with you* follows from Grice (1975)’s maxim of relevance: as there is no other mutually apparent reason a question of theirs about dinner would be relevant to you. That your friend is also practically aware of this collection of factors is also why they chose to produce the message “Dinner?” in the first place, confident that you would discern the intended meaning.

By comparison, if our utterances are ultimately being interpreted by machines and not other humans, our choice of utterance depends on our understanding of how the machine processes language, and a better understanding will help us better achieve our goals. For example, in web search, the ability of a user to find the information they’re looking for depends not only on the capabilities of the search engine, but the user’s skill at leveraging them. This includes an understanding of the system's limitations (*e.g.*, not bothering with complete sentences for keyword search), any of its special syntax or operations (*e.g.*, + and - to force results to include or exclude a word), and tricks for obtaining more reliable results, *e.g.*, including `site:reddit.com` in a Google query (DKB, 2022). The same applies when the user is acting as listener: when a friend shares information out loud that they are reading off of a web page, you might expect that they are paraphrasing the main points; whereas, when a search engine provides a snippet of language content from the page, you might treat it as a verbatim quote. Communicating a model of how a system processes language, so the user can accordingly adjust how they produce and interpet language with it, is an important component of building language understanding systems.

2.4.2 Human↔Machine: Specifications

The Dinner Non-Dilemma thought experiment involves more than just linguistic communication. We may also ask: how, upon seeing the notification on your phone, were you able to draw a conclusion about your friend, who wasn't with you at the time? It is because when you saw the notification with your friend's name, you inferred that your friend of that name, a few seconds prior, engaged their phone (or other messaging app), selected your contact information, typed out "Dinner?", and hit the "send" button — or otherwise did something that they expected would result in the message appearing on your phone. This conclusion is based on your mental model of what connects the physical world to the digital one.

Where this model breaks down, people can be led to inaccurate conclusions: for example, understanding that phone numbers can be spoofed is important for avoiding phishing, and understanding that strangers may find your email or phone number, or automatically send messages to huge numbers of people, can be important for identifying scams. Secure messaging apps like Signal use "safety numbers" exchanged via external channels to establish the identity of both interlocutors in the common ground and avoid man-in-the-middle attacks.[3]

Furthermore, the information in the common ground about the function of texting technology

[3] https://web.archive.org/web/20230410175617/https://support.signal.org/hc/en-us/articles/360007060632-What-is-a-safety-number-and-why-do-I-see-that-it-changed-

determines what types of communication are possible. The use of "delivered" notifications in texting apps functions explicitly to establish that a message may now be treated as being in the common ground between interlocutors, making communication easier and more efficient (no need to re-send messages to ensure they arrived).[4] The presence of *read receipts*, which show a message's sender whether the recipient has read their message, adds even more information and enables entirely new communicative capabilities: "Leaving someone on read" — reading the message but not responding for a long time — has become commonly understood, in some contexts, as a rude or even performative way of ignoring someone.[5]

The more general term for kind of model that we're talking about here is a *program specification*. *Given that a computational procedure p produces symbols y, what can I conclude about the world?* This question, in the context of writing traditional (*i.e.*, non–machine-learning) software, is answered by the API contract of p. A function's API contract allows you to determine when you should use it, what you should pass in as its input, and what you should do with its output. A user's model of how a machine's outputs reflect the world is what informs the user of how to *act* on the basis of the system's outputs, and their model of how a machine acts in the world in response to input lets the user determine what to *say* or input to the machine.

2.4.3 Machine↔Machine: Protocols

The last layer of communication in the Dinner Non-Dilemma is electronic communication between your friend's device and your phone. The text "Dinner?" was encoded on your friend's device as a string of bits using an encoding scheme such as UTF-8 (Yergeau, 2003). After further processing, this string of bits was converted into an electromagnetic signal according to a wireless communication standard such as IEEE 802.11 (IEEE, 2021), trasmitted to a nearby wireless receiver connected

[4] As another perhaps more common example of such an inefficiency, consider the ubiquity of the question "can everyone see my screen?" when screen sharing in a video call. Despite most video conferencing software these days prominently notifying the user in the UI that others can see their screen, even slight doubt that such an important element of the common ground is shared must be eliminated in order for communication to continue smoothly.

[5] Bear in mind again the importance of common ground: leaving someone on read can only be understood as *performative* ignoring if the reader knows that read receipts exist, the sender knows they know this, and so on. It may not apply when communicating with less tech-savvy texters.

to the Internet or a cellular network, and routed through many more layers (and protocols) before arriving at your phone's antenna, at which point your phone inverted the encoding procedure done at your friend's device, extracted the UTF-8–encoded message, and displayed it on your screen.

The communication problem between phones and other digital devices closely reflects the pure signaling game laid out in Section 2.2.1. For machines to reliably transfer bit strings successfully, they must implement a *common protocol* for encoding and decoding those bits to and from the physical medium. From the perspective of a programmer, if you wish to write networking code that works correctly in a device, you must presuppose something about the protocol implemented by the machine on the other end: otherwise, there is no reliable way to facilitate successful communication. Establishing protocols in the common ground, giving them names, and creating canonical ways for machines to identify themselves as implementing specific protocols is one of the main functions of the RFC (Request for Comments) system through which communication protocols are established as standards. Treating RFCs as authoritative sources of common ground has allowed for the development of highly scalable computer networking systems which communicate with impressive reliability. For users communicating with electronic systems, justified confidence in this reliability, and the non-mutation of messages sent across electronic media, is important for communicative success.

2.5 Descriptive Theories: Understanding Black-Box Models

The *Dinner Non-Dilemma* in the previous section illustrates the role of theories of the behavior of intentionally-designed technologies, like phones or other electronic communication systems. Modern deep learning systems such as language models raise new questions: they clearly encode a lot of useful information from their training corpora and exhibit some familiar language behaviors, but how exactly their behavior derives from their training process is not clear, and it is unclear how to draw conclusions about the world from their behavior. In this section, I will discuss the structure and role of *descriptive theories* in making sense of language models.

Consider the following communication scenario, where Marcus and Davis (2020) use GPT-3 (Brown et al., 2020) to generate the bold text:

> You poured yourself a glass of cranberry juice, but then you absentmindedly poured about a teaspoon of grape juice into it. It looks okay. You try sniffing it, but you have a bad cold, so you can't smell anything. You are very thirsty. So **you drink it.**
>
> **You are now dead.**

Their interpretation is that "GPT-3 seems to assume that grape juice is a poison." Together with the fact that grape juice is, of course, not a actually a poison, they take GPT-3's continuation to reflect a lack of understanding of the world and "lack of comprehension" of language. But behind their interpretation and this conclusion is the assumption that text continuations from a language model will reflect the *actual* likelihood of events according to a consistent set of underlying beliefs about the world.[6]

I would interpret this interaction differently — instead, it implies that we should drop the presupposition that a language model's output, when interpreted in conventional English, will reflect likely real-world outcomes or consistent beliefs about the world. So what *does* its output reflect? Answering this question requires a *theory of language models* and how they connect to the world. Any such theory starts with this: *language models are approximate maximum-likelihood models of their training data.*[7]

Language models are "just" statistical models of text, and we cannot step into discourse with them and expect to hold all (indeed, hardly any) of the presuppositions we are used to when conversing with humans. For the cranberry juice example, instead of announcing that we have "fooled" the language model into producing an output that is "incorrect" or "nonsensical," we may ask: *how can we learn something from the model's output?* We may only answer this question using our knowledge about how the model was constructed, and a theory of how this relates it to the

[6] One may suggest that Marcus and Davis give their interpretation of this GPT-3 continuation tongue-in-cheek, as a way of illustrating that we should rethink the presuppositions we bring into the discourse. However, later in the piece, they write: "A genuinely intelligent agent would do something entirely different: draw inferences about the potential safety of mixing cranberry juice with grape juice." This risks conflating the exact distinction I highlight in this chapter, between putatively objective "intelligence" or "understanding" and subjective, presupposed concepts of how meanings should be assigned to a model's output.

[7] Here, I am using the term "language model" to refer specifically to maximum-likelihood models of text corpora, and not, for example, models fine-tuned with reinforcement learning.

world. One way of breaking this down, which I do not claim is systematic or exhaustive, is into the following parts:

- **Corpus:** the corpus the model is trained on.

- **Features:** the elements of the prefix or prompt that the model represents and is capable of tying back to its training corpus.

- **Algorithms:** the mechanisms by which a language model reproduces the distribution of its training data, including how it translates its input features into output tokens.

Examining a language model's behavior can tell us about its training corpus, learned features, and algorithmic mechanisms, and vice versa. Furthermore, an understanding of any of the above three inputs (*e.g.*, features and algorithm), together with the language model's behavior, tells us about the other inputs (*e.g.*, corpus). For example, suppose an unsmoothed 5-gram language model outputs *Hawaii* with probability 0.55 following the prefix *In 1961 Barack Obama was born in.* In this case, we can draw a clear conclusion: the token *Hawaii* follows 55% of the appearances of the 4-gram *Obama was born in* in the corpus. We know this because we have a full understanding of the features (4-gram prefixes) and algorithm (counting) used by n-gram models. (To then draw conclusions about the broader world, we then need a theory of how the corpus is related to that world — an important issue which I will not address here.)

The same logic applies to neural language models, but the challenges of corpus, feature, and algorithm understanding are much greater. Data like "You are now dead" in the cranberry juice example, rather than representing a challenge "failed" by the model, provides behavioral data which can lead to hypotheses like the following:

- *Corpus:* the text in the model's training data does not report outcomes in proportion to their real-world likelihood.

- *Features:* the language model may featurize its inputs less according to the real-world properties of the entities being mentioned, and more according to the narrative structure of the text.

- *Algorithm:* text continuations are determined less according to physical simulation of the described situation forward in time, and more according to lexical and structural cues or foreshadowing in the input text.

These hypotheses may then be more rigorously tested against the model's behavior in a wide range of settings. To the extent to which any of the hypotheses is backed up in further investigation, it gives us new tools for interpreting the outputs of language models — for example, if they are found to be highly sensitive to narrative structure, then their output distributions might help teach us something about that structure (particularly about how it's expressed in the training corpus). Examples of this kind of interpretive work include the following:

- *Corpus:* Brown et al. (2020) suggest on the basis of few-shot benchmarks that language models are moving towards robust numerical reasoning abilities, as they generalize surprisingly effectively and increasingly with scale. However, Razeghi et al. (2022) find that the performance of large language models on arithmetic tasks can be largely predicted by the frequency of the relevant number terms in their training data — suggesting these results do not follow from robust numerical reasoning. This analysis was enabled by the public availability of the Pile dataset (Gao et al., 2021) on which the GPT-J family of language models were trained, and would not have been possible on GPT-3 using publicly available data. Here, the training data provides a source of truth which allows us to more accurately assess what it means when the model produces an answer to an arithmetic problem — that it reflects specific correlations in the training data rather than, potentially, the result of a crudely-implemented addition algorithm.

- *Features:* Brown et al. (2020) proposed that GPT-3 was performing "few-shot learning," having picked up on some kind of meta-learning capabilities from the task of next-token prediction in context. However, an investigation by Min et al. (2022) found that performance in Brown et al.'s few-shot setting is insensitive to the pairing of inputs and outputs in the prompt, rather only depending on the input space, output space, and overall format of the input text. Knowing that the model is insensitive to this feature — the input/output mapping — is crucial for understanding how the model constructs the output that it does, and that "meta-learning" capabilities may not be an important component.

- *Algorithm:* Elhage et al. (2021); Olsson et al. (2022) provide evidence, through a large set of experiments, for the existence of "induction heads" — attention heads which implement copying-like behavior. Insights like this into the algorithmic operations of a language model can provide frameworks for us to assess whether these models implement (or fail to implement) cognitive and linguistic tasks of interest, and investigate and debug their behavior on specific inputs.

Such analysis work comprises the incremental construction of theories of language models which allow us connect their behavior to abstract concepts that we understand and properties of their training data. At the end of the day, "language understanding" with a language model is about *us understanding the language model* at least as much as it is about the language model understanding us.

Language models are not a "general solution to the problem of computer understanding of natural language," to echo the subjects of Weizenbaum (1976)'s skepticism, because language understanding does not exist inside a model. It exists *between agents*, and its nature is contingent on the common ground between those agents. But, this does not mean that language models haven't learned anything about the world, nor that they can never be relied on to perform certain tasks. The challenge is to discern the signal from the noise, developing theories of what language models *have* learned so that we can interpret and use their outputs appropriately.

2.6 Prescriptive Theories: Intentional Design

Although black-box neural network language models are the dominant paradigm in NLP, we are not constrained to acting only on the basis of reverse-engineered theories of machine behavior. We can instead begin with *prescriptive theories* of how systems should behave, and then intentionally design systems to conform with those theories. Such theories are necessary for designers to communicate with users and other stakeholders of a system in order for the use of the system to be effective and safe. There are many ways to approach this; I will discuss two: refining task specifications, and building decomposable and modular systems.

2.6.1 Refining Task Specifications

A first step is establishing common ground between humans (including designers and users of a system) about what it is a system *should* do. Unfortunately, developing common ground around task specifications proves difficult. First of all, it's well-recognized as difficult to make sure our data and metrics accurately reflect the capabilities we want to instill in models (Liao et al., 2021). Problems range from spurious correlations making data artificially easy, *e.g.*, in natural language inference (Gururangan et al., 2018) or multi-hop question answering (Min et al., 2019), to evaluation challenges on open-ended language generation tasks (Krishna et al., 2021).

Part of solving these problems is improving our data curation practices (Bowman and Dahl, 2021), but another part is fundamental to statistical learning: every dataset has artifacts (Gardner et al., 2021). So, getting data to accurately reflect tasks in spite of this requires rethinking our task definitions as well. One particular approach that seems promising to me is to develop **narrowly-scoped tasks which facilitate exhaustive annotation** over text. This can minimize the artifacts that result from annotators producing a convenience sample of language. Examples of the approach include Text-Based NP Enrichment (Elazar et al., 2021), where pairs of pre-specified noun phrases are related to each other using prepositions, or Question-Answer driven Semantic Role Labeling (He et al., 2015b, QA-SRL), which labels semantics with natural language question-answer pairs that conform to a highly restrictive template, facilitating relatively exhaustive annotation in practice (FitzGerald et al., 2018; Roit et al., 2020). While the scope of such tasks is small, we can expect the evaluation results to be more meaningful because annotation is less biased, and modules performing these tasks can be composed into larger systems (Section 2.6.2). These approaches also define tasks to explicate linguistic structure while minimizing arbitrary or arguable decisions about how to map natural language into formal ontologies.

This brings me to another pressing problem for designing task specifications: making sure they are coherent, well-formed, and mutually understood by all stakeholders. This is also difficult: some tasks have many multiple overlapping and competing definitions, as with hate speech and toxicity (Poletto et al., 2021), whereas in natural language inference annotators often systematically disagree with each other (Pavlick and Kwiatkowski, 2019; Nie et al., 2020). Filtering out data points with low agreement or patching these problems with caveats in the annotation guidelines risk hiding the symptoms of the underlying cause: an ill-specified task. Improving this situation requires careful examination of our tasks in the context of their potential interpretations by annotators and users. For example, Zeinert et al. (2021) conduct a deep investigation of online misogyny and present a taxonomy which can form a basis for the development of models sensitive to a variety of distinctions in abusive language. As for ambiguity, we can examine its sources and then **directly represent ambiguity in the task**: for example, Min et al. (2020) find that half of the questions in the NATURAL QUESTIONS dataset (Kwiatkowski et al., 2019) are ambiguous, and formulate a task and dataset where multiple possible interpretations of each question are accounted for and disambiguation is done in the output. The result is both more reliably measurable and more reflective of real-world question answering than the original task; similar work has subsequently been done for visual question answering (Elias Stengel-Eskin et al., 2022) and natural language inference (Liu et al., 2023). Another notable example is gender-specific machine translation: When translating into a language with gendered personal pronouns, Google Translate has a facility to detect gender ambiguity in the source sentence and provide multiple, gender-specific translations (Johnson, 2018, 2020). This method enriches the system's output to reflect more of the true complexity of the machine translation task. With this extra context, the user can choose the translation which best fits their needs (i.e., has the desired grammatical gender) — an outcome which would be impossible to achieve in general if the system had to make a forced choice and output a single translation (as the user may not even know that the output text is gendered, or what the relevant gendered pronouns are).

2.6.2 Build Decomposable Models

It is sometimes said that that no individual understands every part of a computer: there are too many subsystems, too much complexity, too many layers of abstraction for any one person to learn them all — both in the hardware and software. (In this way, traditional computers and deep learning systems have something in common.) Yet, many can use computers productively, and indeed, software is eating the world (Andreessen, 2011). The key to this is the power of abstraction: once a module is defined and its API specified, it can be freely used and re-used on the basis of its specification without the user needing to know exactly how it works. As a result, successively more complex systems can be built, at ever-larger scales, and these systems can be made reliable as a whole on the condition that they are reliable at each of their parts.

It is possible, though by no means trivial, to build powerful AI systems in the same spirit. While pipeline-based NLP systems have fallen out of favor with the ease and effectiveness of end-to-end learning, there is evidence that even when modern models encode a lot of information about language that we think they should use for a task, they don't use it — as with the success of syntactic probes over pretrained representations (Hewitt and Manning, 2019) but the failure of fine-tuned models to depend on syntax in the right ways for natural language inference (McCoy et al., 2019). Now there is new potential for pipelines, as the power of pretrained language models allows for the development of unprecedently robust submodules for simple tasks. For example, Ernst et al. (2022) introduce a pipeline model for multi-document summarization which extracts individual propositions from each document, clusters them using a proposition-level alignment submodule (Ernst et al., 2021), and then uses sentence fusion to produce full summary sentences. Pretrained models like the Cross-Document Language Model (Caciularu et al., 2021) and BART (Lewis et al., 2020) enable high-accuracy pipeline components without the need for extremely large supervised training sets, and the use of explicit clusters produces faithful-by-construction explanations for the summarization system's output. As another example, Chen et al. (2021) use submodules which convert question-answer pairs to declarative form (Demszky et al., 2018) together with sentence decontextualization (Choi et al., 2021) to leverage natural language inference models as verifiers for question answering. In such cases, The creation of more simple, semantically-aware text manipulation modules can potentially enable new, creative model pipelines for reliably performing complex tasks without the need for combinatorially complex and comprehensive evaluation sets.

These models come *pre-packaged* with a theory of their task, which provides new ways of verifying properties of their outputs: For example, each sentence in a summary produced by the system of Ernst et al. (2022) can be traced directly to a set of propositions in the source documents, so hallucination by the summarization system can be largely reduced to hallucination in the (much simpler and easier to audit) sentence fusion module, or mistakes in the clustering step. Integrating retrieval and copying directly into language models, as in GopherCite (Menick et al., 2022), is another way of directly connecting the model's output to external source material, grounding its behavior in information that informs the user on how to act in light of its behavior.

2.7 Conclusion

I have presented a framework for thinking about meaningful progress in the construction of AI systems that perform natural language understanding or, more generally, interface with humans and the world. In particular, I lay out a set of principles for developing theories of machine behavior, which can allow for successful communication between humans and these systems. This is not only fundamental to achieving language understanding between humans and machines, but it is critical to ensuring that the development and deployment of AI systems produces the intended outcomes.

Chapter 3

TOWARDS SCALABLE THEORIES

Formal theories of language structure, world knowledge, and reasoning have long served a vital role in the development of reliable NLP systems, whether as components in traditional NLP pipelines, or as organizing elements for robustness and interpretability tests of end-to-end neural networks. However, these theories are often challenging to theoretically specify with broad coverage, and formal representations are difficult to reliably annotate at scale. As deep learning and large scale continue to advance the capabilities of machine learning models, some argue that intelligence and the world is simply too complex for us to assume we can capture it with domain theories (Sutton, 2019).

In this chapter, I present a framework for the development of *scalable theories*: systematic accounts of aspects of intelligent behavior (*e.g.*, ontologies and knowledge bases) which can automatically scale up alongside large datasets and machine learning models. The key idea in my proposal is an epistemological shift: in contrast with the theory-first "Rationalist" paradigms of old-fashioned AI and the benchmarking-focused "Empiricist" paradigm of modern deep learning, I propose embracing *Pragmatism*: an epistemology which employs theories as tools for articulating and systematizing knowledge while prioritizing holding these theories accountable to useful data external to the theory. The upshot is an approach to data collection and modeling which emphasizes decoupling the phenomena represented in the data from the theories we use to reason about them. Relegating the latter to modeling then allows us to use machine learning and automated methods to develop of new explanatory theories of that data at large scale.

3.1 Introduction

Formal representations of linguistic structure and meaning have long guided our understanding of how to build natural language processing systems, for example, with syntactic and semantic processing in the traditional NLP pipeline (Jurafsky and Martin, 2008). However, this approach has always had limitations:

1. Fully specifying formal representations requires resolving challenging theoretical questions which have been long contentious among linguists;
2. Reliably producing these representations with broad coverage using machine learning systems has proven difficult, requiring new, expensive and difficult-to-annotate data for new domains of text; and,
3. Even ostensibly correct linguistic representations are often difficult to effectively apply in downstream tasks.

These challenges, combined with the unprecedented effectiveness of neural network function approximators, have led to the proliferation of end-to-end neural network models which directly perform end tasks without relying on intermediate representations of linguistic structure (He et al., 2017a; Lee et al., 2017a; Seo et al., 2017, *inter alia*). Especially with additional improvements from unsupervised pre-training (Peters et al., 2018; Devlin et al., 2019), these systems tend to outperform earlier approaches across the board (Wang et al., 2019b,a). Still, they exhibit serious gaps in generalization and robustness (Jia and Liang, 2017; Niven and Kao, 2019) and remain highly uninterpretable.

To address these problems, formal theory, *e.g.*, of linguistic structure, common sense, reasoning, and world knowledge, can provide guiding frameworks for evaluation. For example, they inform the design and construction of challenge sets (McCoy et al., 2019; Naik et al., 2018; Wang et al., 2019b), measures of systematicity (Yanaka et al., 2020; Kim and Linzen, 2020), behavioral tests (Linzen et al., 2016), and probing experiments (Liu et al., 2019a; Tenney et al., 2019b). As these theories allow us to characterize the generalization behaviors we desire, they are likely to play a pivotal role in the design and training of highly general systems. However, the aforementioned shortcomings also pose challenges in this setting: defining systematic generalization patterns can be difficult, limiting scope and coverage; furthermore, performance on targeted tests (e.g., probing) is not always reflective of a model's sensitivity to relevant linguistic phenomena when deployed on downstream tasks (McCoy et al., 2019).

In light of these issues, core improvements in formal theories of aspects of intelligent behavior may yield boons for both the construction and evaluation of NLP systems. But the question remains of how to achieve this: decades of work on resources like semantic ontologies (Baker et al., 1998a; Palmer et al., 2005), commonsense knowledge bases (Lenat, 1995; Speer et al., 2017), and formal reasoning systems (Lifschitz, 2008) have to a large extent been superseded in NLP based on deep learning, displaced from model pipelines by end-to-end neural networks (Sutskever et al., 2014; Lee et al., 2017a; Dozat and Manning, 2017; He et al., 2017a).

To understand this difficulty, I examine the recent history of NLP from the lens of epistemology. Drawing from Church (2007)'s description of the field on a *pendulum* between Rationalism and Empiricism (Section 3.2), I present an alternative in Pragmatism: a century-old epistemology which we can use to characterize what it means for us to *know* something about language (Section 3.3). This will serve as the basis for guiding principles for the development of *scalable theories*, moving us towards a better understanding of the behaviors that can be built into AI systems.

3.2 The Rationalist–Empiricist Divide

Church (2007) describes the history of computational linguistics and AI on a *pendulum*, swinging between Rationalist (roughly, theory-driven) and Empiricist (roughly, data-driven) paradigms every 20 years. Church lists the "swings" as follows (with my comments):

- 1950s: Empiricism (Shannon, Skinner, Firth, Harris) — information theory, psychological behaviorism, early corpus linguistics

- 1970s: Rationalism (Chomsky, Minsky) — generative linguistics, logic-based AI

- 1990s: Empiricism (IBM Speech Group, AT&T Bell Labs) — statistical NLP, machine learning, modern distributional semantics

- 2010s: A Return to Rationalism?

As the reader may know, the predicted "Return to Rationalism" did not happen. NLP, for its part, is more Empiricist than ever. Ironically, the Empiricism of 2007 is that of statistical modeling with loglinear models in the NLP pipeline — by today's standards, downright Rationalist in its adherence to theoretical conceptions of the language system. In this section, I will discuss advantages and disadvantages of each paradigm, recent developments in Empiricist NLP, and how we ended up where we are today.

3.2.1 Rationalist Roots: What's in a Theory?

The theory-first approach to creating knowledge embodied in these approaches is sometimes termed *Rationalist*, a name that hearkens to the epistemological school of Rationalism — the most prominent members including René Descartes (1596–1650) and Immanuel Kant (1724–1804) — centered (roughly) around the ideas that reason is the primary source of knowledge and that indubitable truths can be accessed through reason alone.[1] The benefits of a theoretical approach are manifold; here I discuss two.

Understanding Fundamental Limitations In computational linguistics and AI, Church (2007) identifies Noam Chomsky and Marvin Minsky among prominent Rationalists. Among other contributions, they are known for criticisms of the theoretical limits of statistical models: Chomsky (1957) introduced a widely-known critique of n-gram language models, highlighting their inability to correctly capture unbounded-length syntactic dependencies, and Minsky and Papert (1969) showed that certain kinds of neural networks (which, functionally, act like linear separators) cannot model the XOR function. More recently, critics such as Bender and Koller (2020), Marcus and

[1] The most famous of such purported truths is probably Descartes's *cogito ergo sum*.

Davis (2020), and Hofstadter (2018) take a more modern Rationalist angle, arguing that there are theoretical limits to the capabilities of language models which are trained on large swathes of text but lack real-world situational experience or formal reasoning capabilities. In all of these cases, a theoretical understanding of language (*i.e.*, as containing unbounded dependencies, and as expressing intents grounded outside of language) or machine learning classifiers (*i.e.*, as limited to linearly separable classes for certain models, or lacking the systematic ability to manipulate symbols) has been instrumental in identifying fundamental limitations of language technologies.

Capturing Deep Generalities Chomsky is known for regarding probabilistic models as giving "no particular insight into some of the basic problems of syntactic structure" (Chomsky, 1957). The problem that he identified was that some deep generalities of language — *i.e.*, the unbounded length of certain grammatical dependencies (such as subject agreement) — could not be systematically captured in n-grams alone. The flip side of this observation is that formal theories can serve as powerful tools for capturing and modeling such generalities.

Indeed, the development of linguistic theory is often centered around capturing remarkable generalities. Chomsky's early arguments captured systematic generalities in English syntax such as the behaviors of unbounded-length agreement dependencies and extractions (Chomsky, 1957, 1965); theories of semantic roles are often designed to capture and predict a wide range of behaviors in verbs' syntactic realizations on the basis of their meanings (Dowty, 1991; Pustejovsky, 2011); Davidsonian semantic representations use logical theory to capture regularities in entailment (Davidson, 1967); and, there is a rich tradition in linguistics of making predictions about cross-linguistic universals (*e.g.*, ordering constraints) on the basis of theories of language structure (Cinque, 1999; Haspelmath, 2010; Stanojević and Steedman, 2021). When building NLP systems, we also wish for them to capture important behavioral regularities about language. Leveraging a theory, either through direct implementation or synthesizing data to provide inductive bias to a system, may seem like a natural way to work towards such general capabilities.

The key to both aforementioned advantages of theory-driven thinking — understanding limitations and unlocking generalities — is a mechanistic description (on some level) of the system being implemented, or more generally, a mentally-manipulable model of an aspect of (putatively) intelligent behavior. While such theories clearly bring advantages, not all is rosy with Rationalism, which has largely fell out of favor in NLP. So next, I will discuss what we have to learn from the Empiricist developments in modern AI.

3.2.2 The Era of Empiricism

Empiricism is the school of thought that sensory data and experience are the primary sources (and objects) of knowledge. Prominent early figures in the Empiricist tradition include Francis Bacon (1561–1626) and David Hume (1711–1776). When applied to computational linguistics and NLP, the Empiricist perspective is generally understood to mean that the knowledge or capabilities possessed by system (such as a language processing model) should be assessed primarily by its behavior on data rather than the form of the model itself, and to a large extent the acquisition of knowledge by an artificial intelligence system should be done on the basis of data and experience. This perspective dates back to the origins of AI and information theory, with Turing (1950)'s *Imitation Game* and Shannon (1948)'s noisy channel model of communication. While the traditional NLP pipeline may be counted in the Empiricist tradition, it contains Rationalist elements in its reliance on linguistic theory to form its components. More recently, the field has taken a turn further into Empiricism: On one hand, modeling is dominated by end-to-end neural networks where the model's form is only considered relevant insofar as it affects measurable behaviors on data (Collobert and Weston, 2008; Wang et al., 2019b). On the other, the injection of knowledge and capabilities into these models is increasingly done via exposure to a wide range of behavioral and observational data, rather than theoretically-informed inductive bias (Mikolov et al., 2013; Peters et al., 2018; Devlin et al., 2019; Brown et al., 2020).

The recent rise of Empiricism can to some extent be attributed to its agnosticism with respect to underlying theories of the phenomena being modeled. Here I will review two arguments along these lines.

The Map is Not the Territory One useful perspective is provided by Norvig (2011), written in reply to public remarks by Chomsky. Norvig highlights crucial flaws in Chomsky's criticisms of statistical modeling. In particular, Chomsky's criticism about unbounded-length dependencies applies to n-gram models, but not necessarily probabilistic models in general (for example, a probabilistic context-free grammar can capture such dependencies). Furthermore, he assumes that any probabilistic model would assign zero probability to unseen sentences — but this ignores the existince of smoothing methods. Overall, Chomsky's criticisms do not apply to the Empiricist *approach* so much as particular, naïve modeling methods; in extending them to all of probabilistic modeling, Chomsky argues only on the basis of a lack of imagination. Minsky and Papert (1969)'s point about linear separators has suffered a similar fate: it no longer applies to the nonlinear neural network models that are ubiquitous today. The more general problem is that theoretical arguments about the limitations of a system only apply when the theory correctly describes that system; if the methods of system construction change so that the theory no longer applies, the limitations are not so fundamental.

In a similar vein, recent arguments on principle that language models cannot 'understand' language in any meaningful sense (Bender and Koller, 2020; Marcus and Davis, 2020) are received with skepticism: it is not clear *a priori* how theoretical considerations regarding the nature of meaning should imply constraints on the possible behaviors of a system. After all, any behavioral test which could meaningfully expose the difference between 'understanding' and not could also potentially be present in the training routine of such a model, and any difference which could not be observed through behavioral testing seems like metaphysics (Michael, 2020). Overall, the victories that Empiricism has had over these criticisms stem to a large extent from the flexibility that comes from its lack of commitment to any particular theory or form by which to describe the world.

The World is Extremely Complex The flexibility of not committing to a theory, on that note, is not just a practical advantage: Norvig (2011) describes it as an epistemic virtue, a more careful and humble way of approaching the world than what he calls the "Platonism" of Chomsky. Along these lines, Sutton (2019) remarks on a 'Bitter Lesson' from the history of AI:

> We have to learn the bitter lesson that building in how we think we think does not work in the long run... the actual contents of minds are tremendously, irredeemably complex; we should stop trying to find simple ways to think about the contents of minds, such as simple ways to think about space, objects, multiple agents, or symmetries. All these are part of the arbitrary, intrinsically-complex, outside world. They are not what should be built in, as their complexity is endless; instead we should build in only the meta-methods that can find and capture this arbitrary complexity.

In NLP, the hard-learned lesson that theoretical considerations from linguistics often do not contribute to performance in the real world has been cast in stone as a famous quote by Fred Jelinek from the 1980s.[2] Training a general-purpose algorithm on data that directly represents the problem one is trying to solve has, for the complex problems in AI and language, consistently been a more reliable route to building a decent system than relying on theories which inevitably oversimplify the problem or construe it in unhelpful ways. The upshot of these observations is general skepticism on the part of some ML and NLP practitioners that formal theories, such as grammars, linguistic ontologies, and explicit representations of knowledge and common sense, provide any meaningful or useful explanation of language use.

3.2.3 The Perils of Empiricism

Yet, not all is well in the land of Empiricists. End-to-end neural network models near-uniformly perform best on every test suite we have, especially when subjected to large-scale pretraining on language text (Devlin et al., 2019; Lewis et al., 2020). But this success has exposed another set of problems: the shortcomings of our evaluations and benchmarks. Before modern neural network modeling, it was rare for a machine learning system to achieve performance metrics on par with humans in complex language understanding (or linguistic structure prediction) tasks. Now, however, it is quite common (Wang et al., 2019a). The problem is that these metrics are not robust: even a model that achieves very high performance within its training distribution may be easily fooled

[2] No, I will not repeat it here. It has been said enough.

or led astray with perturbations to its input that would not affect human judgment (Jia and Liang, 2017), or will have unacceptably degraded performance out of domain. The Empiricist answer to this has been to incorporate more data in the pretraining process (Radford et al., 2019; Brown et al., 2020), but there are at least two fundamental limitations to what this approach can achieve, which we will discuss now.

Underspecification The first problem is *underspecification*: when building a system using machine learning, the only definition of the task that is accessible to the model is its training set; the goal, of course, is to create a system that generalizes in desired ways on new data. Kharitonov and Chaabouni (2021) found that training neural language models on deeply underspecified data leads to strikingly different generalization behaviors, based on both model architecture and random chance. But more concerningly, D'Amour et al. (2020) illustrate how this kind of underspecification is rampant in machine learning pipelines: models trained using standard methods and datasets which perform just as well in-distribution as each other may diverge wildly in the way that they generalize. The problem is that fundamentally, a training set *itself* cannot suffice to specify *all* of the desired behaviors of a model (*i.e.*, on inputs outside that set), in the same way that a universal quantification can not be proven with a set of examples. Said another way, it is impossible to distinguish annotation artifacts in the training set (Gururangan et al., 2018) — which only apply to a subset of examples — from generalizable reasoning rules, without some understanding of what generalizable reasoning *should* look like. Building this understanding requires comprehensible ways of exhaustively carving up the space of inputs into subsets, and characterizing a model's desired behavior on each subset.

As is well known in the field of formal verification (Pierce et al., 2017), this is precisely the advantage that a formal theory provides over example-based testing. In in the context of NLP, when model failures are identified on out-of-distribution data, in order to understand them as systematic problems rather than an ad-hoc or cherry-picked sets of mistakes, they must be characterized in terms of a theory — whether this means a theory of the relationship between syntactic structure and entailment (McCoy et al., 2019), compositionality of sentence meaning (Kim and Linzen, 2020), monotonicity (Yanaka et al., 2020), or other subsystems of language (Lake and Baroni, 2018; Yanaka et al., 2021). In this context, what theory offers us is a way of making sense of our measurements and assigning them meaning.

Intelligibility The second problem, broadly speaking, is *intelligibility* (Weld and Bansal, 2019). In order to trust AI systems, be able to anticipate and correct their failures, and integrate them with other software, we need to be able to reason effectively about their behavior. This, as Weizenbaum (1976) points out, requires some kind of theory. In this vein, Olah et al. (2020) argue for reverse-engineering trained neural networks back into mechanistic theories which can help us understand their behavior. However, what is needed in order to correctly specify the behavior of an intelligible model is a theory of its *desired* behavior — not just its *existing* behavior.

Overall, this leaves us in a very difficult situation. It seems that Empiricism alone — while it scales incredibly well with complexity — is not sufficient for the development of robust, intelligible systems, which requires a comprehensible theory of their desired behaviors. On the other hand, Rationalism doesn't scale to handle the sheer complexity of the domains we wish to model. Is there a scalable approach to the development of comprehensible, robust theories? Perhaps there is, in an alternative epistemology which we will explore next.

3.3 Prospects of Pragmatism: Building Theories that Work

An odd feature of the Rationalism/Empiricism dichotomy is that neither epistemology accurately describes the pursuit of science in most fields either. In fields like physics, chemistry, and biology, theoretical and experimental approaches are not in conflict; rather, they synergize and inform each other, as theories are continually updated to align with new experimental data. To make sense of this, we can turn to an epistemology inspired by how people actually operate in the world: Pragmatism.

Pragmatism is an epistemological framework which (roughly speaking) conceptualizes *knowing* in terms of the *actions* that the knowledge licenses, *i.e.*, by the predictions that follow from that knowledge. Prominent Pragmatists include Charles Sanders Peirce (1839–1914) and William James (1842–1910). Like Empiricism, Pragmatism embraces experience as the primary source from which we can derive knowledge. Unlike Empiricists, however, Pragmatists such as James embrace the use of formal and linguistic categories as comprising the content of knowledge, on the basis of their *usefulness* in making predictions and licensing actions (James, 1907). The Pragmatist conception of truth differs from the Rationalist one in that it makes no claim to fundamentally describe the form of the world: the Pragmatist search for truth is not a search for one true theory, but for an ever-expanding set of theoretical tools and concepts that can be picked up and put down according to the needs of the knower. In pithy terms, a Pragmatist might agree with the statistical aphorism that that "All models are wrong; some are useful" (Box, 1976). Pragmatists such as James (1907) claim that this perspective more accurately describes human behavior with respect to knowledge (and indeed, the pursuit of science) than prior epistemologies.

Combining the core ideology of Pragmatism with observations from computational linguistics, we can derive two guiding principles for the development of theories that may have prospective use in NLP: decouple data from theory (Section 3.3.1), and make data reflect use (Section 3.3.2).

3.3.1 Decouple Data from Theory

One feature that distinguishes much NLP work (particularly involving linguistic structure) from traditional sciences is the status of theory with respect to data. In most empirical sciences, data takes the form of concrete measurements of the world, and the task of a theory is to explain those measurements. In NLP, many benchmarks and datasets are constructed under the *assumption* of a theory, whether it be one of syntactic structure (Marcus et al., 1993a; de Marneffe et al., 2021), semantic structure (Palmer et al., 2005; Banarescu et al., 2013a), or some other task-specific labeling scheme.

Having a theory, *e.g.*, of syntactic or semantic structure, is useful in that it provides a straightforward way of annotating phenomena involving disambiguation of text — and syntactic and semantic disambiguation is important for understanding language. However, errors in the theory and inconsistencies in its annotation resulting from theoretical complexity of vagueness of its definitions limit what can be learned by models, as human performance can be surprisingly low (Nangia and Bowman, 2019) and high inter-annotator agreement is very challenging to achieve. For example, the OntoNotes compendium of semantic annotations (Hovy et al., 2006) was presented as "The

90% solution" because of 90% agreement rates — which imply that the dataset cannot validate performance numbers any higher than 90%.

As another example, Palmer et al. (2006) find that fine-grained sense distinctions produce considerable disagreement among humans annotating English text. But fixing the problem can't just be a matter of improving the sense inventory: they find that coarser-grained sense groups designed to improve agreement lack the distinctions from the fine-grained senses that are necessary for predicting how words should translate into typologically distant languages like Chinese and Korean. When different tasks might require different theoretical distinctions, locking them in stone at annotation time is a problem. This is especially true considering there will almost certainly be missing categories: word sense annotations disambiguating only between the *river* and *money* senses of the word *bank* will have no way of accounting for new senses or homonyms that come along, *e.g.*, if the *bet* (verbal) sense is encountered in new data. More generally, refining annotation guidelines to increase agreement between annotators does not necessarily solve the problem, as the extra assumptions built into the annotation process do not necessarily encode any more scientifically meaningful information in the data — a problem known in the philosophy of science as the "problem of theoretical terms" (Riezler, 2014).

Building a robust theory that can scale to unexpected phenomena and new data, and can be adjusted for new tasks, requires theoretical agility which is precluded by the commitment that comes with a theory-based annotation standard. An alternative is to directly annotate the phenomena that the theory is meant to explain, and derive the theory on the basis of this data. This, for example, is how work on *grammar engineering* is done in the DELPH-IN consortium (Bender and Emerson, 2021). For each language, a broad-coverage Head-driven Phrase Structure Grammar (HPSG) is maintained separately from its associated treebank, which is annotated not with full syntactic analyses but with *discriminants* (Carter, 1997) such as prepositional phrase attachment sites which constrain the set of possible parses in a way that is independent of the grammar. This way, when the grammar is updated, the discriminants can be used to automatically update the treebank accordingly while also providing considerable data to sanity-check the updated theory (Oepen et al., 2004; Flickinger et al., 2017). Among examples that push the envelope further are the Decompositional Semantics Initiative (White et al., 2016) and MegaAttitude project (White and Rawlins, 2016).[3] In these projects, annotating large-scale corpora with the phenomena that are posited to underly linguistic theories in question — such as Dowty (1991)'s proto-role properties, or entailments corresponding to neg-raising (An and White, 2020) and projection (White and Rawlins, 2018) — has facilitated data-driven insights regarding argument selection (Reisinger et al., 2015a) and lexically-specified syntactic subcategorization rules (White, 2021), as well as automatically inducing lexicon-level ontologies of semantic roles (White et al., 2017a) and event structure (Gantt et al., 2021) that are derived directly from the phenomena they are designed to explain.

The lesson of Empiricism is that for a model to work, it must be learned from data; while Rationalism tells us that for a model to be intelligible and general, it must be grounded in theory. A wealth of innovative prior work shows us that Pragmatism is possible: we can have both.

3.3.2 Make Data Reflect Use

Crafting a satisfying, data-driven theory of a few linguistic phenomena is not sufficient to serve as a backbone for general language understanding systems. The second relevant lesson of Pragmatism is that the model must be fit to its use. The approaches reviewed in Section 3.3.1 are, by and large, targeted at theoretical questions in language syntax and semantics, *e.g.*, regarding the nature of syntactic structure across many languages (Bender et al., 2002) or the syntactic realization of a verb's arguments (Reisinger et al., 2015a). On the other hand, general-purpose language processing relies on a huge amount of lexical and world knowledge and inferential ability which is outside the scope of traditional linguistic theories. While general-purpose syntactic and semantic representations have some direct uses in NLP end-tasks, such as for search and retrieval (Schäfer et al., 2011; Shlain et al., 2020), their application in downstream tasks requiring higher-level reasoning or inference, like reading comprehension, translation, and information extraction has been less fruitful. This is at least in part because these theories are far insufficient to serve as mechanistic accounts of the

[3] `https://decomp.io`, `https://megaattitude.io`

inferential phenomena which are required to perform those tasks.[4]

Constructing theories which *can* account for such phenomena is a monumental challenge. But it is a challenge which, I argue, we must address if we want to pursue the goal of accurate, reliable, and intelligible systems. Pragmatism tells us that the first step is to catalog the phenomena we wish to explain in such a way that we can produce and maintain theories of these phenomena in an agile way. This will require carefully carving up the space of phenomena in such a way that useful abstractions can be designed to facilitate future progress (Dijkstra, 1974).

3.4 Scaling Theories with Data Simulation

Section 3.3 provided a general framework for building useful theories: annotate data in a theoretically-minimal way, scope it carefully to reflect specific phenomena that we want to explain, and then and automatically induce theories to explain those phenomenia using computational methods like machine learning. But how does this method scale in practice? Even if the resulting theories are high-quality, they still require annotated data, which limits their scope to orders of magnitude less than what is leveraged by the pretrained generative models which are now the standard in NLP (Liu et al., 2019b; Brown et al., 2020).

Black-Box Data Simulators This is where black-box models may actually be able to help with the development of theory. Even if they are uninterpretable on their own, their high accuracy and data efficiency under fine-tuning means they can be used as *data simulators*, generating phenomenological data — potentially at a level of granularity or exhaustivity unobtainable from humans — which can be fed into another, more interpretable algorithm to distill a theory from it. Indeed, this is exactly the approach we will take in Chapter 6: we first train a black-box model to generate questions in Question-Answer driven Semantic Role Labels, where each role is labeled

[4] There has been some recent success in inducing syntactic structures from free text data (Kim et al., 2019) using the open-ended objective of optimizing likelihood; as many tasks can be framed in terms of the likelihood of text continuations (Paperno et al., 2016; Radford et al., 2019; Brown et al., 2020), investigating the use of these structures as well as those in other syntactically structured language models (Kuncoro et al., 2017, 2019) as a mechanism for model intelligibility may be an interesting line of future work; however, their capacity to explain model behavior is still bounded by the limits of syntax in explaining inferential behaviors.

with a single question in the training data. But then we decode full question distributions from this model, and induce an ontology of semantic roles by clustering arguments based on the overlap of their question distributions.

It may seem like the use of a black-box model as a data simulator begs the question: if our concern is that the black-box model isn't learning the underlying function we hope it is, then doesn't using it to simulate data risk leading us to a theory of the wrong function? Actually, yes — but if the theory is wrong, *we can see it, and we can do something about it.* If the results of theory induction are not as expected (for example, the set of induced semantic roles looks wrong), it provides an opportunity to do error analysis on the level of *theory* rather than individual examples. In particular, examining the "wrong" parts of the resulting theory (*e.g.*, induced semantic roles that don't match what we intuitively expect based on existing theory), and their connection to the training data, will lead to one of the following outcomes:

- Systematic gaps in the data or mistakes in the model used for data simulation — which can then be corrected. See Chapter 6 for an in-depth analysis of this kind.

- Mistakes in the modeling assumptions used in the theory induction algorithm — giving us interesting information that we can use to improve our theories

- Mistakes in our intuition about what the theory should have looked like in the first place — which means we've learned something.

All of these are positive outcomes for long-term progress.

Scaling in Complexity I have discussed the problem of scale in the sense of the *size* of the theory — an issue with large knowledge bases or ontologies of linguistic structure. But this does not handle the case of more *complex* tasks, with more nuanced relations between input and output (such as open-ended question answering or common sense inference tasks). Since theoretical modeling requires narrowly-scoped data (discussed more in Part II), we cannot necessarily expect to be able to construct theories of such broad capabilities in the short term. However, if we carve up the space of these tasks, start with theories of simple sub-phenomena of reading, inference, and language understanding, then we may be able to bootstrap from these theories to then annotate and make sense of more complex data. Constructing a “complete” or “true” theory of these tasks may indeed be impossible even in principle, but — in the spirit of Pragmatism — that doesn’t mean we can’t construct theories that are *useful* for understanding and controlling the behavior of such systems. How the proposed framework would scale with task complexity is as-of-yet unclear. However, scalable theories of narrow phenomena at least provide a step in the right direction.

3.5 Conclusion

In this chapter, I presented a program for the development of scalable theories, grounded in reflections on recent developments in NLP from an epistemological lens. By applying the principles of Pragmatism — scoping our data carefully to reflect specific phenomena of interest, and applying computational methods to automatically induce theories of those phenomena — I hope that we can make firm, if small, steps to create scientific progress in NLP.

Part II

DATA: ANNOTATING NATURAL LANGUAGE WITH NATURAL LANGUAGE

In Part I, I outlined the idea of *scalable, data-driven theories* of language behavior, and argued for their central role in scientific progress in NLP. As discussed in Chapter 2, building AI systems we can understand requires having theories of our models and data, as well as theories of the tasks we wish to accomplish. The first step to developing these theories in an empirical, data-driven way is to carefully choose the data (Chapter 3). That is what we will discuss in this part of the thesis. To guide the *data* aspect of data-driven theory, I propose **Four Principles of Scientific Data for NLP**:

1. **Theoretical minimalism.** The data itself should encode as few theoretical assumptions as possible. For example, if you wish to capture natural language syntax, you should directly annotate the *phenomena* that syntactic theories are intended to explain rather than directly annotating theoretical constructs like syntactic trees. This avoids the "problem of theoretical terms" (Riezler, 2014) discussed in Chapter 3 and creates the space for an underlying theory to explain this data.

2. **Broad comprehensibility.** In order to run data collection at large scale or on-demand in new domains, it should be possible to recruit non-expert annotators and affordable to pay them to label large amounts of data (*e.g.*, through crowdsourcing), or it should be feasible to automatically generate the data (*e.g.*, with language models).

3. **Annotation constraints.** The output space of the task should not be too open-ended. Requiring annotators to, for example, write whole sentences or stories without tight constraints allows them to produce data from a convenience sample of the space, resulting in biased data that doesn't capture the full complexity of the phenomena of interest (Cai et al., 2017;

Gururangan et al., 2018). Constraining their annotations to a specific space can help ensure exhaustive coverage of the phenomena of interest, as Chapters 4 and 5 will show.

4. **Narrow scope.** The task should not capture too much complexity in the relationship between input and output. Not only can this make it difficult for annotators to reliably produce high-quality data, but it makes it more difficult to model the phenomena expressed in the data with a comprehensible theory. We will see in Chapter 4 that allowing for too much complexity makes modeling difficult, while in Part III I will show how more constrained data allows for theory-driven modeling for use in end tasks and the automated induction of linguistic structure.

The call for highly constrained and tightly scoped data collection may seem to contravene some recent trends in deep learning for language. In particular, there have been huge documented successes from large-scale training on heterogeneous corpora to perform highly open-ended tasks like language modeling (Peters et al., 2018; Radford et al., 2019; Brown et al., 2020) and masked language modeling (Devlin et al., 2019). I am not saying it has been a mistake to pursue such directions. Models trained on large, open-ended corpora (like language models) can be very powerful tools for a wide range of purposes, and indeed can synergize with my proposed approach through their use as *data simulators* (see Section 3.4). What I am proposing is to *also* work towards the development of data-driven theories of language tasks, which requires the kind of data work I describe in this part of the thesis. My view is that this is essential to making scientific and long-term progress in NLP, and will aid in the development of engineerable systems, particularly for tasks where it is difficult or impossible to curate training data (see the introduction of Part III).

In Chapters 4 and 5, I describe two data annotation projects which aided in the development of the Four Principles. This work is focused on annotations of shallow semantic structure: syntax, semantic roles, and other predicate-argument structure relations expressed in text. The key strategy in our methodology is to stay theoretically minimal and crowd-comprehensible by *annotating natural language with natural language*. This line of work was pioneered by He et al. (2015b), who propose *Question-Answer driven Semantic Role Labeling* (QA-SRL), a framework for annotating

English verbal predicate-argument relations using simple, highly constrained question-answer pairs (described in more detail in Chapter 5).

In subsequent years, we explored variations of this approach, which illustrate some of the basic tensions between the Four Principles:

- He et al. (2016)[5] explored a more constrained approach, targeting natural language syntax. We introduced *human-in-the-loop parsing*, wherein we extract syntactic attachment ambiguities from an existing parser's n-best list of parse trees, convert these ambiguities into multiple-choice questions, get crowdsourced workers to answer the questions, and then re-parse the original sentence with constraints derived from the results. Testing this approach on the English CCGbank (Hockenmaier and Steedman, 2007), our results are positive, but weak — there was a very small improvement in parser performance. A core challenge was the *syntax–semantics mismatch*, where workers would provide semantically correct answers to questions which would correspond to the wrong syntactic attachment — for example, because of coreference between noun phrases or head-finding choices in partitives. So even though our annotation task was tightly scoped, our interpretation of the results required theoretical assumptions about syntax which did not match up with the intuitions of non-expert workers. See He et al. (2016) for more details.

- Michael et al. (2018), reproduced in Chapter 4, took the opposite tack. In human-in-the-loop parsing, the annotation constraints and narrow task scope interfered with the first two Principles of theoretical minimalism and broad comprehensibility. So in this project, we broadened the scope of the task and sought to more directly capture annotator intuitions. We solicited open-ended questions from annotators with the goal of having them capture as many interesting semantic relationships as possible in the source sentence. Doing this successfully required adding many careful constraints and incentives to the crowdsourcing procedure, but we were careful to still allow for open-ended questions that expressed annotator

[5] I participated heavily in this project, experimenting with approaches for generating questions from sets of parse trees in Combinatory Categorial Grammar (Steedman, 2000).

creativity. The result was a dataset of “Question-Answer Meaning Representation” (QAMR) annotations over English encyclopedic and news text covering many interesting phenomena. However, achieving high recall of predicate-arugment relations was not economical (requiring high annotation redundancy) and the resulting unstructured question-answer pairs were hard to use downstream. The most successful use of QAMRs in follow-up work were probably QuASE (He et al., 2020), which used QAMRs for pretraining, and Supervised Open Information Extraction (Stanovsky et al., 2018), which converted the QAMR dataset to Open Information Extraction tuples — but had to parse the questions with an existing syntactic parser to do so. While an early version of this work (Michael et al., 2017) showed how to use token statistics from the question–answer pairs to convert QAMR annotations into graph-based meaning representations, this required highly redundant annotations (5 annotators per sentence) and the results only matched the reliability of the automated semantic graph parsers of the time. The lesson from these results is that leaving the annotation space too open and unconstrained leads to difficulties with recall and challenges with downstream modeling and theory.

- FitzGerald et al. (2018), covered in Chapter 5, returned to QA-SRL. In the original QA-SRL work (He et al., 2015b), annotators were individually trained and used an excel spreadsheet with drop-down menus to specify the questions. In this work, we streamlined the annotation process greatly and collected data using crowdsourcing. We automated quality control by using a two-stage generate/validate crowdsourcing pipeline, and increased annotation speed, reliability, and coverage using an autocomplete and auto-suggest system which keeps track of the syntactic structure of QA-SRL questions as the annotator types, and uses it to suggest completions as well as whole questions. We were able to gather data for over 64,000 sentences with high coverage, as well as further increase coverage by bringing a model into the loop. In terms of both semantic richness (*i.e.*, inherent complexity) and annotation constraints, these annotations are somewhere between our work on human-in-the-loop parsing and question-answer meaning representations, showing the importance of striking an appropriate balance between the Four Principles.

Our results over the course of these projects suggests that we should search for tasks in a “goldilocks zone”: Their scope should not be so constrained or beholden to prior theory as to be unintuitive, but not so unconstrained that it is hard to get exhaustive and reliable annotation of interesting phenomena. As annotation constraints depend on *some* prior theory of the phenomena to be captured, these constraints need to be carefully chosen so as to minimize arbitrary assumptions in the task setup and make sure the task is natural for annotators.

In the case of QA-SRL, the prior theory we incorporated is a small grammar fragment of English encompassing QA-SRL questions. Our findings support that QA-SRL, with the annotation aids developed in Chapter 5 (FitzGerald et al., 2018), strikes a good balance of the Four Principles. The question remains, then: what can be done with this data? Can we use it to build data-driven theories, and if so, how can such theories be used? We will return to these issues in Part III.

Chapter 4
QUESTION-ANSWER MEANING REPRESENTATIONS

In this chapter, we introduce Question-Answer Meaning Representations (QAMRs), which represent the predicate-argument structure of a sentence as a set of question-answer pairs. We develop a crowdsourcing scheme to show that QAMRs can be labeled with very little training, and gather a dataset of English QAMRs with over 5,000 sentences and 100,000 questions. A qualitative analysis demonstrates that the crowd-generated question-answer pairs cover the vast majority of predicate-argument relationships in existing datasets (including PropBank, NomBank, and QA-SRL) along with many previously under-resourced ones, including implicit arguments and relations. We also report baseline models for question generation and answering, and summarize a recent approach for using QAMR labels to improve an Open IE system. These results suggest the freely available[1] QAMR data and annotation scheme should support significant future work.[2]

4.1 Introduction

Predicate-argument relationships form a key part of sentential meaning representations, and support answering basic questions such as *who did what to whom*. Resources for predicate-argument structure are well-developed for verbs (e.g. PropBank (Palmer et al., 2005)) and there have been efforts to study other parts of speech (e.g. NomBank (Meyers et al., 2004) and FrameNet (Baker et al., 1998b)) and introduce whole-sentence structures (e.g. AMR (Banarescu et al., 2013b)). However, highly skilled and trained annotators are required to label data within these formulations for each new domain, and it takes significant effort to model each new type of relationship (e.g., noun arguments in NomBank). We propose a new method to annotate relatively complete representations

[1] `github.com/uwnlp/qamr`

[2] This chapter is drawn from Michael et al. (2018).

Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29.

Who will **join** as **nonexecutive director**? - Pierre Vinken
What is **Pierre**'s last name? - Vinken
Who is **61 years old**? - Pierre Vinken
How **old** is **Pierre Vinken**? - 61 years old
What will he **join**? - the board
What will he **join the board** as? - nonexecutive director
What type of **director** will **Vinken** be? - nonexecutive
What day will **Vinken join the board**? - Nov. 29

Figure 4.1: Example QAMR.

of the predicate-argument structure of a sentence, which can be done easily by non-experts.

We introduce Question-Answer Meaning Representations (QAMRs), which represent the predicate-argument structure of a sentence as a set of question-answer pairs (see Figure 4.1). Following the QA-SRL formalism (He et al., 2015a), each question-answer pair corresponds to a predicate-argument relationship. There is no need for a carefully curated ontology and the labels are highly interpretable. However, we differ from QA-SRL in focusing on *all words* in the sentence rather than just verbs, and allowing *free form* questions instead of using templates.

The QAMR formulation provides a new way of thinking about predicate-argument structure. Any form of sentence meaning—from a vector of real numbers to a logical form—should support the challenge of determining which questions are answerable by the sentence, and what the answers are. A QAMR sidesteps intermediate formal representations by surfacing those questions and answers *as* the representation. As with any other representation, this can then be reprocessed for downstream tasks. Indeed, the question-answer format facilitates reprocessing for tasks that are similar in form, for example Open IE (see Section 4.4).

A key advantage of QAMRs is that they can be annotated with crowdsourcing. The main challenge is coverage, as it can be difficult for a single annotator to write all possible QA pairs for a sentence. Instead, we distribute the work between multiple annotators in a novel crowdsourcing scheme, which we use to gather a dataset of over 100,000 QA pairs for 5,000 sentences in Newswire and Wikipedia domains.

Although QAMR questions' free-form nature is crucial for our approach, it means that predicates are not explicitly marked. However, with a simple predicate-finding heuristic, we can align QAMR to PropBank, NomBank, and QA-SRL and show high coverage of predicate-argument structure, including more than 90% of non-discourse relationships. Further analysis reveals that QAMRs also capture many phenomena that are not modeled in traditional representations of predicate-argument structure, including coreference, implicit and inferred arguments, and implicit relations (for example, with noun adjuncts).

Finally, we report simple neural baselines for QAMR question generation and answering. We also highlight a recent result (Stanovsky et al., 2018) showing that QAMR data can be used to improve performance on a challenging task: Open Information Extraction. Together, these results show that there is significant potential for follow up work on developing innovative uses of QAMR and modeling their relatively comprehensive and complex predicate-argument relationships.

4.2 Crowdsourcing

We gather QAMRs with a two-stage crowdsourcing pipeline[3] using monetary incentives and crowd-driven quality control to ensure high coverage of predicate-argument structure. *Generation* workers write QA pairs and *validation* workers answer or reject the generated questions. Full details of our setup are given in Appendix A.

Generation Workers receive an English sentence with up to four *target words*. They are asked to write as many QA pairs as possible containing each target word in the question or answer, subject to light constraints (for example, the question must contain a word from the sentence and be answered in the sentence, and they must highlight the answer in the sentence). Workers must write at least one QA pair for each target word to receive the base pay of 20c. An increasing bonus of $3(k+1)$ cents is paid for each k-th additional QA pair they write that passes the validation stage.

[3] Built using Amazon Mechanical Turk: www.mturk.com

Validation Workers receive a sentence and a batch of questions written by an annotator in the first stage (with no marked target words or answers). The worker must mark each question as *invalid* or *redundant* with another question, or highlight its answer in the sentence. Two workers validate and answer each set of questions. They are paid a base rate of 10c for each batch, with an extra 2c for each question past four.

Quality control Question writers are disqualified if the percentage of valid judgments on their questions falls below 75%. Validators need to pass a qualification test and maintain above 70% agreement with others, where overlapping answer spans are considered to agree.

Treatment of workers The minimum pay of 20c per sentence was chosen to yield a rough reward of $12/hr for working at moderate pace (60s per sentence, subjectively judged by the authors). Pay per-hour depended on worker pace and how exhaustively they annotated questions, with some making more and some making less than this amount, and we take a few other measures to ensure fair treatment of crowd workers. First, for the question generating task, we allow workers to start on the task and immediately get paid by default, instead of withholding payment or asking them to work for free in a trial phase. Then, if a worker is disqualified from further work, it is done using Mechanical Turk's qualification system rather than account blocks, which many workers report can affect the status of their account and whether they can get work in the future.[4] Finally, workers are always paid for all of the work they provide.

4.2.1 Data Preparation and Annotation

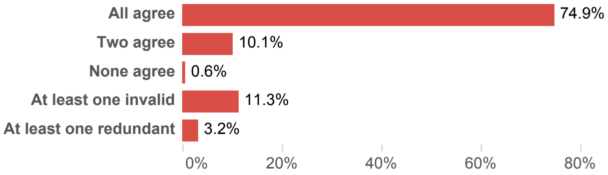
We drew our data from 1,000 Wikinews articles from 2012–2015 and 1,000 articles from Wikipedia's 1,000 core topics,[5] partitioned by document into train, dev, and test, and preprocessed using the Stanford CoreNLP tools (Manning et al., 2014). We also annotated 253 sentences from the Penn

[4] Blocks were used at the beginning of annotation, but after workers raised concerns, all blocks were lifted, we replaced blocking with the qualification system, and we allowed all workers to try the task again.

[5] https://en.wikipedia.org/wiki/Wikipedia:1,000_core_topics

	PTB	Train	Dev	Test
Sentences	253	3,938	499	480
Annotators	5	1	3	3
QA Pairs	27,082	73,561	27,535	26,994
Filtered	18,789	51,063	19,069	18,959
Cost	$2,862	$7,879	$2,919	$2,919
Cost/token	$0.44	$0.08	$0.25	$0.25

(a) Summary of the data gathered.



(b) Agreement and validation statistics. Answers were considered to agree if their spans overlapped. High agreement on answers indicates that questions were generally interpretable and answers were consistent.

Figure 4.2: Statistics of the QAMR dataset.

Treebank (Marcus et al., 1993b) chosen to overlap with existing resources for comparison (see Section 4.3).

For each sentence, we group its non-stopwords sequentially into groups of 3 or 4 target words, removing sentences with no content words. By presenting workers with nearly-contiguous lists of target words, enforcing non-redundancy, and providing bonuses, we encourage exhaustiveness over all possible QA pairs. By allowing the target word to appear in the question *or* the answer, we make the requirements flexible enough that there is almost always some QA pair that can be written.

Figure 4.2b shows agreement statistics for question validation. We removed questions either validator counted invalid or redundant, as well as questions not beginning with a wh-word,[6] which we found to be of low quality. We also annotated the partitions at different levels of redundancy to allow for more exhaustive dev, test, and comparison sets. See Figure 4.2a for statistics.

4.3 Data Analysis

In this section, we show that QAMR has high coverage of predicate-argument structure and uses a rich vocabulary to label fine-grained and implicit semantic relations.

[6] *who, what, when, where, why, how, which,* and *whose*

Coverage To show that QAMR captures the same kinds of predicate-argument relations as existing formalisms, we compare our data to PropBank, NomBank, and QA-SRL. Since predicates in the questions are not explicitly marked, we use a simple predicate-finding heuristic to help align to other formalisms: for each minimal span that appears in the QAMR questions and answers (i.e., none of its subspans appear independently of it elsewhere in the QAMR), we compute its *predicate score* as the proportion of its appearances that are in a question rather than in an answer.[7] We then choose the span with the highest predicate score in each question as its predicate.

We measure recall on the shared Penn Treebank sentences for each resource by randomly sampling n annotators out of 5 for each group of target words, which simulates the situation for the training set (1 annotator) and the dev/test sets (3 annotators). For each n we took the mean of 10 runs. Full details of our comparison are in Appendix B.

Results are shown in Figure 4.3a. Single annotators cover over 60% of relationships, and coverage quickly increases with the number of annotators, reaching over 90% with all five. This shows that QAMR's representational capacity covers the vast majority of relevant predicate-argument relations in existing resources. However, coverage in our training set is low due to low annotation density.

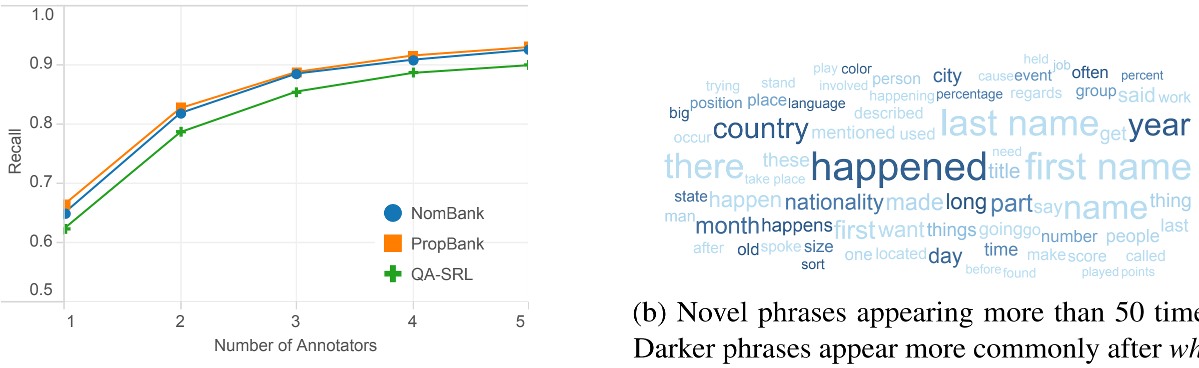
For a qualitative analysis, we sample 150 QA pairs (see Table 4.1 for examples).[8] Of our sample, over 90% of question-answer pairs correspond to a predicate-argument relation expressed in the sentence,[9] including arguments and modifiers of nouns and verbs as well as relationships like those within proper names (Table 4.1, ex. 2c, 3a) and coreference (ex. 3c, 4c). Questions that do not align to predicate-argument structure often target shallow inferences (ex. 3b, 7c).

Rich vocabulary Annotators use the open question format to introduce a large vocabulary of *external phrase*s which do not appear in the sentence. Overall, 5,687 different external phrases

[7] This follows the intuition that predicates are more likely to appear in the question; for example, see **join** in Figure 4.1.

[8] This sample, and statistics for the remainder of this section, were taken from the filtered train and development sets, with a total of about 70k QA pairs.

[9] We assume a QA pair targets the relation corresponding to the semantic role of the wh-word in the question.



(a) Recall of predicate-argument relations for sentences shared with each of our reference datasets, with increasing number of annotators.

(b) Novel phrases appearing more than 50 times. Darker phrases appear more commonly after *who, which*, or *how*. The vast majority of external phrases are used to denote entity/event types or semantic relations.

Figure 4.3: Presence of predicate-argument relations and novel phrases in QAMR.

are introduced (excluding stopwords), appearing 25,952 times in 38.7% of the questions (see Figure 4.3b). These include typing words like *state* and *country* (Table 4.1, ex. 5), most often directly after the *wh*-word, and relation-denoting phrases like *work for* (ex. 2b), *last name* (ex. 3a), and *victim* (ex. 7a). Despite the open format, synonyms are not a major issue, obscuring the semantic relation in only 2% of our sample (ex. 2a).

We also find verbal paraphrases of noun compounds, as proposed by Nakov (2008). For example, where *Gallup poll* appears in the text, one annotator has written *Who conducted the poll?*, which explicates the relationship between *Gallup* and *poll*. Similarly, *Who received the bailouts?* is written for the phrase *bank bailouts*.

Semantics, not just syntax Only 63% of QA pairs characterize their predicate-argument relation using the same syntactic relationship as in the sentence. 5% have answers coreferent with the syntactic argument (Table 4.1, ex. 3c, 4c); 17% exhibit syntactic variation, using different prepositions (ex. 4c, 6a), alternating between active and passive (ex. 1b), or changing between the noun and verb form of the predicate (ex. 8a); 6% ask about implicit arguments (ex. 4b, 5c, 8b); and 6% ask about inferred relations (ex. 3b).

4.4 Models

To establish initial baselines, we apply existing neural models for QAMR question generation and answering. We also briefly summarize a recent end task result, where QAMR annotations were used to improve an Open IE system.

Question generation In question generation (QG), we learn a mapping from a sentence $\mathbf{w}$ to a set of questions $q_1, \dots, q_m$. We enumerate pairs of words $(w_{\text{q}}, w_{\text{a}})$ from the sentence to seed the generator. During training, outputs are questions q and inputs are tuples $(\mathbf{w}, w_{\text{q}}, w_{\text{a}})$, where $w_{\text{q}} \in q$ and w_{a} is in q's answer. We also add negative samples where the output is a special token and the input has w_{q}, w_{a} that never appear together.

We use an encoder-decoder model with a copying mechanism (Zhou et al., 2017) to generate a question from an input sentence with tagging features for part of speech, w_{q}, and w_{a}. At test time, we run all pairs of content words (w_i, w_j) where $|i - j| \leq 5$ through the model to yield a set of questions. On the QAMR test set, this achieves 28% precision and 24% recall with fuzzy matching (multi-BLEU[10] > 0.8).

Question answering The format of QAMRs allows us to apply an existing question-answering model (Seo et al., 2016) designed for the SQuAD (Rajpurkar et al., 2016) reading comprehension task to answer QAMR questions. Training and testing with the SQuAD metrics on QAMR, the model achieves 70.8% exact match and 79.7% F1 score. We further improve performance to 75.7% exact match and 83.9% F1 by pooling our training set with the SQuAD training data. The relative ease of QA in comparison to QG suggests that in QAMR, most of the information is contained in the questions.

Open IE Finally, we also expect that the predicate-argument relationships represented in QAMRs will be useful for many end tasks. Such a result was recently shown for Open IE (Stanovsky et al., 2018), using our QAMR corpus. Open IE involves extracting tuples of natural language phrases

[10] An average of the BLEU1–BLEU4 scores.

that express the propositions asserted by a sentence. They show that, using a syntactic dependency parser, a QAMR can be converted to a list of Open IE extractions. Augmenting their training data with a conversion of our QAMR dataset yields state-of-the-art performance on several Open IE benchmarks (Stanovsky and Dagan, 2016b; Xu et al., 2013; de Sá Mesquita et al., 2013; Schneider et al., 2017). The gains come largely from the extra extractions (e.g., with nominal predicates) that QAMRs support over traditional resources focusing on verbal predications.

4.5 Related Work

In addition to the semantic formalisms (Palmer et al., 2005; Meyers et al., 2004; Banarescu et al., 2013b; He et al., 2015a) we have already discussed, FrameNet (Baker et al., 1998b) also focuses predicate-argument structure, but has more fine-grained argument types. Gerber and Chai (2010) target implicit nominal arguments. Stanovsky and Dagan (2016a) annotate non-restrictive noun phrase modifiers on top of QA-SRL. Other linguistically motivated annotation schemes include UCCA (Abend and Rappoport, 2013a), HSPG treebanks (Flickinger et al., 2017), and the Groningen meaning bank (Basile et al., 2012).

Crowdsourcing has also been applied to gather annotations of structure in the setup of multiple choice questions, for example, for Dowty's semantic proto-roles (Reisinger et al., 2015b; White et al., 2016) and human-in-the-loop parsing and classification (He et al., 2016; Duan et al., 2016; Werling et al., 2015), while Wang et al. (2017) use crowdsourcing with question-answer pairs to annotate some PropBank roles directly. Our approach recovers paraphrases of noun compounds similar to those crowdsourced by Nakov (2008).

More broadly, non-expert annotation has been used extensively to gather question-answer pairs over natural language texts, for example in reading comprehension (Rajpurkar et al., 2016; Richardson et al., 2013; Nguyen et al., 2016) and visual question answering (Antol et al., 2015). However, while these treat question answering as an end task, we regard it as a representation of predicate-argument structure, and focus annotators on a smaller selection of text (a few target words in a single sentence, rather than a paragraph) aiming to achieve high coverage.

4.6 Conclusion

QAMR provides a new way of thinking about meaning representation: using open-ended natural language annotation to represent rich semantic structure. This paradigm allows for representing a broad range of semantic phenomena with data easily gathered from native speakers. Our dataset has already been used to improve the performance of an Open IE system, and how best to leverage the data and model its complex phenomena is an open challenge which our annotation scheme could support studying at a relatively large scale.

4.7 Notes

4.7.1 Crowdsourcing Details

In this section we provide full details of the data collection methodology described in Section 2. For the exact text of the instructions shown to workers, and code to reproduce the annotation or demo the interface, see `github.com/uwnlp/qamr`.

Stages Data collection proceeded in two stages: *generation* and *validation*. These were run as two types of HITs (Human Intelligence Tasks) on the Amazon Mechanical Turk platform. Workers wrote questions and answers for the generation task, and those questions would be immediately uploaded as new HITs for the validation task, which ran concurrently. Two workers would validate each question. The worker writing the question would be assessed based on the validators' judgments, and the validators would be assessed based on their agreement. In this way, the quality of workers in either stage could be quickly assessed so spammers or low-quality workers could be disqualified before causing much damage.

Question constraints In both stages, we define a *valid* question to

(1) contain at least one word from the sentence,

(2) be about the sentence's meaning,

(3) be answered obviously and explicitly in the sentence,

(4) not be a yes/no question, and

(5) not be *redundant*,

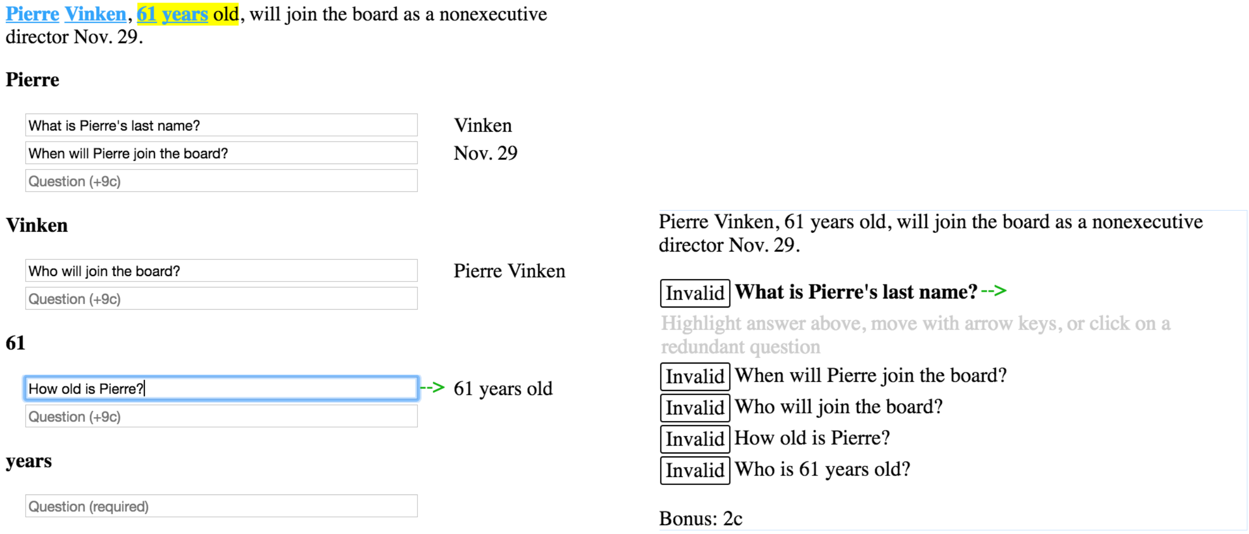
where we define two questions as being redundant by the informal criterion of “having the same meaning” and the same answer. These requirements are illustrated with examples.

Workers in the generation phase are instructed only to write valid questions, while workers in the validation phase are instructed only to answer valid questions (marking the rest invalid or redundant).

When we ask that the question contains a word from the sentence, we allow for changing forms of the word through inflectional or derivational morphology (with examples of both). The only constraint on questions that is strictly enforced by the interface is a length limit of 50 characters.

Target words In the generation task, each sentence is presented to the worker with several underlined *target words*. They are required to write at least one QA pair for each target word, where the target word must appear either in the question or the answer. We choose sets of target words by chunking consecutive words (ignoring stopwords) into groups of 3 or 4 (or fewer for very short sentences). Because the target words shown to a single worker are close to each other—and often share a constituent—it restricts the set of QA pairs they write to relate to a certain part of the sentence. However, asking that the target word appears either in the question *or* the answer makes it flexible enough so that the worker is almost never stuck with no reasonable question to write. We identified this approach after some experimentation, finding that together with the monetary incentives described below, it struck the appropriate balance of scope that was small enough to get exhaustive annotation, but not so small that it cornered workers into writing awkward questions or getting frustrated.

Interface In the generation stage, below the sentence, each target word is listed with a text field below it where they write a question for that target word. While the text field is focused, they



Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29.

Pierre

What is Pierre's last name? — Vinken

When will Pierre join the board? — Nov. 29

Question (+9c)

Vinken

Who will join the board? — Pierre Vinken

Question (+9c)

61

How old is Pierre? --> 61 years old

Question (+9c)

years

Question (required)

(a) Annotation interface for generation.

Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29.

Invalid **What is Pierre's last name?** -->

Highlight answer above, move with arrow keys, or click on a redundant question

Invalid When will Pierre join the board?

Invalid Who will join the board?

Invalid How old is Pierre?

Invalid Who is 61 years old?

Bonus: 2c

(b) Annotation interface for validation.

Figure 4.4: Screenshots of the annotation interfaces.

highlight the answer tokens in the sentence using custom implemented highlighting functionality. The highlighted tokens then appear next to the focused question. The answer tokens need not be a contiguous span in our interface (though they almost always are in practice). Once a question is written and answered, a new text field appears directly below it for another question, allowing the annotator to write as many questions as they can. See Figure 4.4a for a screenshot of the generation task interface.

In the validation stage, no target words are indicated to the user; they only see a list of questions written in a single HIT by a worker in the generation stage. They use the arrow keys to switch between the questions, and use the mouse to assess them: either highlighting the answer in the sentence, clicking another question to mark the selected one redundant, or clicking the *invalid* button to mark a question invalid. See Figure 4.4b.

Incentives and payment Base pay for the generation stage was 20c, with a bonus of $3(k+1)$ cents for each question beyond the number required (so, the first extra question would reward them 6c, the next 9c, and so on). However, their bonuses were only calculated based on the number of questions considered valid by annotators. So if a worker in the generation task wrote 2 extra questions, but any 2 (or 3, or more) of their questions were judged invalid, then they would receive no bonus.

In the validation stage, workers were paid 10c plus a bonus of 2c per question beyond four.

Quality control We used Mechanical Turk's quality control mechanisms in several ways. First, we used the built-in Locale qualification to limit the tasks to workers based in the United States as a proxy for English proficiency. Second, we wrote a multiple-choice qualification test for the validation task, which tested workers' understanding of the definitions of question validity and redundancy. Workers were required to get a score of 75% on this test before working on the validation task.

Finally, we used Mechanical Turk's built-in qualification mechanism to keep track of worker accuracy and agreement ratings. Before working on either task, a worker would have to request a qualification which stored their accuracy or agreement value. Then as they worked, it would be updated over time and they could check its value in their Mechanical Turk account to see how they were doing. In the generation task, accuracy was calculated as the proportion of all judgments (aggregating those from both validators) that were not *invalid* or *redundant*, and accuracy had to remain above 75% to avoid disqualification. In the validation task, agreement was calculated by treating answer spans as agreeing if they had any overlap, and *redundant* judgments agreeing if their targets had agreeing answer spans. A worker's agreement had to stay above 70% for them to remain qualified.

If a worker's accuracy or agreement rating dropped within 5% of the threshold, the worker was automatically sent an email with a warning and a list of common mistakes and tips they might use to improve.

Implementation All of our code was written in Scala, using the Java AWS SDK on the backend to interface with Mechanical Turk, Akka Actors and Akka HTTP to implement the web server and quality control logic, and Scala.js with React to implement the user interface.

Dataset Our dataset was gathered over the course of 1 month from 330 unique workers. See Section 2.1 for details.

4.7.2 SRL Comparison

In this section we provide the full details of the comparison of QAMR to PropBank, NomBank, and QA-SRL given in Section 3.

Preprocessing For each of these resources, there were certain predicate-argument relationships that we filtered out of the comparison for being out of scope.

For PropBank, we filter out predicates and arguments that are auxiliary verbs, as well as reference (`R-`) roles since aligning these properly is difficult and their function is primarily syntactic. We also remove discourse (-DIS) arguments such as *but* and *instead*: these may be regarded as involved in *discourse* structure separately from the predicate-argument structure we are investigating. 78% of the original dependencies remain.

For NomBank, we also remove auxiliaries, and we remove arguments that include the predicate—which are present for words like *salesman* and *teacher*—leaving 83% of the original dependencies.

For QA-SRL, we use all dependencies, and where multiple answers were provided to a question, we take the union of the answer spans to be the argument span.

Alignment Because QAMR does not mark predicates explicitly, we use a simple predicate-finding heuristic to align the QA pairs in a QAMR to the predicate-argument relations in each resource independently.

For each QAMR, we identify every *minimal span* appearing in its questions and answers, i.e., a span from the sentence where none of its subspans appear independently of it in the QAMR. We then calculate a *predicate score* for each span, as the proportion of times it appeared in a question versus an answer. Then for each QA pair, we identify the span in the question with highest predicate score as its *predicate span*, and the answer as its argument span. This is then aligned to the predicate-argument arc in the chosen resource with the greatest non-zero argument overlap such that the predicate is contained within the question's predicate span. If there is no such alignment, we check for an opposite-direction alignment where the predicate is in the answer of a QA pair and the argument completely contains the question's predicate span.

Results See Section 3 for a description of the results. With 1 annotator, we get around 60% recall, but it begins to level off over 85% with 3 annotators.

We manually examined 25 sentences to study sources of coverage loss in the 5-annotator case. In comparison to PropBank and NomBank, the missing dependencies are due to missing QA pairs (44%), mistakes in our alignment heuristic (28%), and subtle modifiers/idiomatic uses (28%). For example, annotators sometimes overlook phrases such as *so far* (marked as a temporal modifier in PropBank) or *let's* (where *'s* is marked as a core verbal argument). Comparing to QA-SRL, 60% of the missed relations are inferred/ambiguous relations that are common in that dataset. Missed QA pairs in QA-SRL account for another 20%.

In aggregate, these analyses show that the QAMR labels capture the same kinds of predicate-argument structures as existing resources. However, while our development and test sets can be expected to have reasonable coverage, where we have labels from only one annotator for each target word (as in our training set), the recall low compared to expert-annotated structures, which may pose challenges to learning.

Sentence	Ann.	Question	Answers
(1) Climate change affects distribution of weeds, pests, and diseases.		(a) What affects distribution of diseases?	Climate change
	VAR	(b) What is affected?	distribution of... / distribution
(2) Baruch ben Neriah, Jeremiah's scribe, used this alphabet to create the later scripts of the Old Testament.	SYN	(a) Who wrote the scripts?	Baruch ben Neriah
	ROLE	(b) Who did Baruch work for?	Jeremiah
		(c) What is old?	Testament / the Old Testament
(3) Mahlunga has said he did nothing wrong and Judge Horn said he "failed to express genuine remorse".	ROLE	(a) What is the Judge's last name?	Horn
	INF	(b) Who doubted his remorse was genuine?	Judge Horn
	CO	(c) Who didn't express genuine remorse?	Mahlunga
(4) In Byron's later memoirs, "Mary Chaworth is portrayed as the first object of his adult sexual feelings."		(a) Who is portrayed in the work?	Mary Chaworth
	IMP	(b) Who was the object of his sexual feelings?	Mary Chaworth
	VAR	(c) Who was Mary the object of sexual feelings for?	Byron
(5) Volunteers are presently renovating the former post office in the town of Edwards, Mississippi, United States for the doctor to have an office.		(a) What town is the post office in?	Edwards
		(b) What state is the post office in?	Mississippi
	IMP	(c) What country are the volunteers renovating in?	United States
	VAR	(d) What country is the city of Edwards in?	United States
(6) The ossicles are the malleus (hammer), incus (anvil), and the stapes (stirrup).	VAR	(a) What is the malleus one of?	The ossicles / ossicles
(7) Liam "had his whole life in front of him", said Detective Inspector Andy Logan, who was the senior investigator of his murder.	ROLE	(a) Who is the murder victim Logan is investigating?	Liam
	ROLE	(b) What rank of investigator is Andy Logan?	Detective Inspector / senior
	INF	(c) Who was Detective Logan speaking about?	Liam
(8) This cemetery dates from the time of Menkaure (Junker) or earlier (Reisner), and contains several stone-built mastabas dating from as late as the 6th dynasty.	INF	(a) How old are the stone-built mastabas?	dating from as late as the 6th dynasty / from as late as the 6th dynasty
	IMP	(b) What period was earlier than Menkaure?	Reisner
		(c) What dates from the 6th dynasty?	mastabas / several stone-built mastabas

Table 4.1: Examples of question-answer pairs capturing various semantic relations, annotated with interesting phenomena they exhibit: syntactic variation (VAR), synonym use (SYN), explicit role names for implicit relations (ROLE), coreference (CO), implicit arguments (IMP), and inferred relations (INF).

Chapter 5

CROWDSOURCING QA-SRL

In this chapter, we present a new large-scale corpus of Question-Answer driven Semantic Role Labeling (QA-SRL) annotations, and the first high-quality QA-SRL parser. Our corpus, QA-SRL Bank 2.0, consists of over 250,000 question-answer pairs for over 64,000 English sentences across 3 domains and was gathered with a new crowd-sourcing scheme that we show has high precision and good recall at modest cost. We also present neural models for two QA-SRL subtasks: detecting argument spans for a predicate and generating questions to label the semantic relationship. The best models achieve question accuracy of 82.6% and span-level accuracy of 77.6% (under human evaluation) on the full pipelined QA-SRL prediction task. They can also, as we show, be used gather additional annotations at low cost.[1]

5.1 Introduction

Learning semantic parsers to predict the predicate-argument structures of a sentence is a long standing, open challenge (Palmer et al., 2005; Baker et al., 1998b). Such systems are typically trained from datasets that are difficult to gather,[2] but recent research has explored training non-experts to provide this style of semantic supervision (Abend and Rappoport, 2013a; Basile et al., 2012; Reisinger et al., 2015b; He et al., 2015a). In this paper, we show for the first time that it is possible to go even further by crowdsourcing a large scale dataset that can be used to train high quality parsers at modest cost.

We adopt the Question-Answer-driven Semantic Role Labeling (QA-SRL) (He et al., 2015a)

[1] This chapter is based on FitzGerald et al. (2018). In this project, I led the crowdsourcing task design, data collection, quality control, and data analysis.

[2] The PropBank (Bonial et al., 2010) and FrameNet (Ruppenhofer et al., 2016) annotation guides are 89 and 119 pages, respectively.

In 1950 Alan M. Turing ***published*** "Computing machinery and intelligence" in Mind, in which he ***proposed*** that machines could be ***tested*** for intelligence ***using*** questions and answers.

Predicate		**Question**	**Answer**
published	1	Who published something?	Alan M. Turing
	2	What was published?	“Computing Machinery and Intelligence”
	3	When was something published?	In 1950
proposed	4	Who proposed something?	Alan M. Turing
	5	What did someone propose?	that machines could be tested for intelligent using questions and answers
	6	When did someone propose something?	In 1950
tested	7	What can be tested?	machines
	8	What can something be tested for?	intelligence
	9	How can something be tested?	using questions and answers
using	10	What was being used?	questions and answers
	11	Why was something being used?	tested for intelligence

Figure 5.1: An annotated sentence from our dataset. Question 6 was not produced by crowd workers in the initial collection, but was produced by our parser as part of Data Expansion (see Section 5.5.)

annotation scheme. QA-SRL is appealing because it is intuitive to non-experts, has been shown to closely match the structure of traditional predicate-argument structure annotation schemes (He et al., 2015a), and has been used for end tasks such as Open IE (Stanovsky and Dagan, 2016b). In QA-SRL, each predicate-argument relationship is labeled with a question-answer pair (see Figure 5.1). He et al. (2015a) showed that high precision QA-SRL annotations can be gathered with limited training but that high recall is challenging to achieve; it is relatively easy to gather answerable questions, but difficult to ensure that every possible question is labeled for every verb. For this reason, they hired and trained hourly annotators and only labeled a relatively small dataset (3000 sentences).

Our first contribution is a new, scalable approach for crowdsourcing QA-SRL. We introduce a streamlined web interface (including an auto-suggest mechanism and automatic quality control to boost recall) and use a validation stage to ensure high precision (i.e. all the questions must be answerable). With this approach, we produce QA-SRL Bank 2.0, a dataset with 133,479 verbs from 64,018 sentences across 3 domains, totaling 265,140 question-answer pairs, in just 9 days. Our analysis shows that the data has high precision with good recall, although it does not cover every possible question. Figure 5.1 shows example annotations.

Using this data, our second contribution is a comparison of several new models for learning a QA-SRL parser. We follow a pipeline approach where the parser does (1) unlabeled *span detection* to determine the arguments of a given verb, and (2) *question generation* to label the relationship between the predicate and each detected span. Our best model uses a span-based representation similar to that introduced by Lee et al. (2016) and a custom LSTM to decode questions from a learned span encoding. Our model does not require syntactic information and can be trained directly from the crowdsourced span labels.

Experiments demonstrate that the model does well on our new data, achieving up to 82.2% span-detection F1 and 47.2% exact-match question accuracy relative to the human annotations. We also demonstrate the utility of learning to predict easily interpretable QA-SRL structures, using a simple data bootstrapping approach to expand our dataset further. By tuning our model to favor recall, we over-generate questions which can be validated using our annotation pipeline, allowing for greater recall without requiring costly redundant annotations in the question writing step. Performing this procedure on the training and development sets grows them by 20% and leads to improvements when retraining our models. Our final parser is highly accurate, achieving 82.6% question accuracy and 77.6% span-level precision in an human evaluation. Our data, code, and trained models will be made publicly available.[3]

5.2 Data Annotation

A QA-SRL annotation consists of a set of question-answer pairs for each verbal predicate in a sentence, where each answer is a set of contiguous spans from the sentence. QA-SRL questions are defined by a 7-slot template shown in Table 5.1. We introduce a crowdsourcing pipeline to collect annotations rapidly, cheaply, and at large scale.

[3] http://qasrl.org

Wh	**Aux**	**Subj**	**Verb**	**Obj**	**Prep**	**Misc**
Who			blamed	someone		
What	did	someone	blame	something	on	
Who			refused		to	do something
When	did	someone	refuse		to	do something
Who	might		put	something		somewhere
Where	might	someone	put	something		

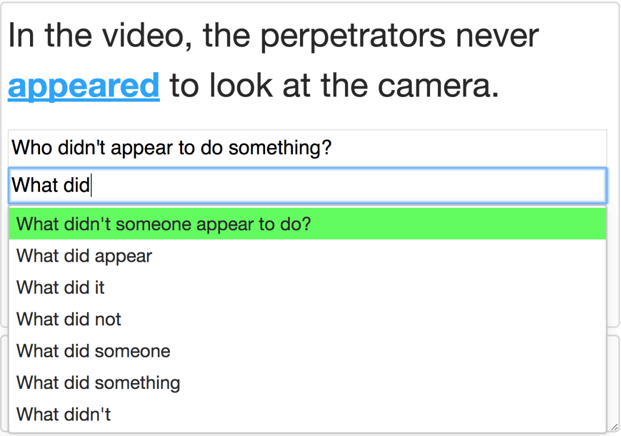
Table 5.1: Example QA-SRL questions, decomposed into their slot-based representation. See He et al. (2015a) for the full details. All slots draw from a small, deterministic set of options, including verb tense ($present$, $pastparticiple$, etc.) Here we have replaced the verb-tense slot with its conjugated form.

Pipeline Our crowdsourcing pipeline consists of a *generation* and *validation* step. In the generation step, a sentence with one of its verbs marked is shown to a single worker, who must write QA-SRL questions for the verb and highlight their answers in the sentence. The questions are passed to the validation step, where n workers answer each question or mark it as *invalid*. In each step, no two answers to distinct questions may overlap with each other, to prevent redundancy.

Instructions Workers are instructed that a *valid* question-answer pair must satisfy three criteria: 1) the question is grammatical, 2) the question-answer pair is asking about the time, place, participants, etc., of the target verb, and 3) all correct answers to each question are given.

Autocomplete We provide an autocomplete drop-down to streamline question writing. Autocomplete is implemented as a Non-deterministic Finite Automaton (NFA) whose states correspond to the 7 QA-SRL slots paired with a partial representation of the question's syntax. We use the NFA to make the menu more compact by disallowing obviously ungrammatical combinations (e.g., *What did been appeared?*), and the syntactic representation to auto-suggest complete questions about arguments that have not yet been covered (see Figure 5.2). The auto-suggest feature significantly reduces the number of keystrokes required to enter new questions after the first one, speeding up the annotation process and making it easier for annotators to provide higher recall.

Payment, quality control, and worker treatment Generation pays 5c for the first QA pair (required), plus 5c, 6c, etc. for each successive QA pair (optional), to boost recall. The validation step pays 8c per verb, plus a 2c bonus per question beyond four. Generation workers must write at least 2 questions per verb and have 85% of their questions counted valid, and validators must maintain 85% answer span agreement with others, or they are disqualified from further work. A validator's answer is considered to agree with others if their answer span overlaps with answer spans provided by a majority of workers.



In the video, the perpetrators never appeared to look at the camera.

Who didn't appear to do something?

What did

What didn't someone appear to do?

Figure 5.2: Interface for the generation step. Autocomplete shows completions of the current QA-SRL slot, and auto-suggest shows fully-formed questions (highlighted green) based on the previous questions.

The minimum pay of 8–10c per sentence was chosen to yield a rough reward of $12/hr for working at moderate pace (30s per sentence for question generation, subjectively judged by the authors). Pay per-hour depended on worker pace, with some making more and some making less than this amount, and we take a few other measures to ensure fair treatment of crowd workers. First, for the question generating task, we allow workers to start on the task and immediately get paid by default, instead of withholding payment or asking them to work for free in a trial phase. Then, if a worker is disqualified from further work, it is done using Mechanical Turk's qualification system rather than account blocks, which many workers report can affect the status of their account and whether they can get work in the future. Finally, workers are always paid for all of the work they provide. Unlike the generation phase, we used a qualification test for the validation phase because the validation step determined the bonuses sent to generation workers. This meant that low-quality validators or spammers could affect the pay of generation workers by answering "Invalid" to every question for a short while before getting disqualified.

	Wikipedia	Wikinews	Science
Sentences	15,000	14,682	46,715
Verbs	32,758	34,026	66,653
Questions	75,867	80,081	143,388
Valid Qs	67,146	70,555	127,455

(a) Statistics for the dataset with questions written by workers across three domains.

	P	**R**	**F**
He et al. (2015a)	97.5	86.6	91.7
This work	95.7	72.4	82.4
This work (unfiltered)	94.9	85.4	89.9

(b) Precision and recall of our annotation pipeline on a merged and validated subset of 100 verbs. The unfiltered number includes questions that some validators marked as invalid.

Table 5.2: Dataset statistics and coverage of questions.

Preprocessing We use the Stanford CoreNLP tools (Manning et al., 2014) for sentence segmentation, tokenizing, and POS-tagging. We identify verbs by POS tag, with heuristics to filter out auxiliary verbs while retaining non-auxiliary uses of "have" and "do." We identify conjugated forms of each verb for the QA-SRL templates by finding them in Wiktionary.[4]

Dataset We gathered annotations for 133,479 verb mentions in 64,018 sentences (1.27M tokens) across 3 domains: Wikipedia, Wikinews, and science textbook text from the Textbook Question Answering (TQA) dataset (Kembhavi et al., 2017). We partitioned the source documents into train, dev, and test, sampled paragraph-wise from each document with an 80/10/10 split by sentence.

Annotation in our pipeline with $n = 2$ validators took 9 days on Amazon Mechanical Turk.[5] 1,165 unique workers participated, annotating a total of 299,308 questions. Of these, 265,140 (or 89%) were considered valid by both validators, for an average of 1.99 valid questions per verb and 4.14 valid questions per sentence. See Table 5.2a for a breakdown of dataset statistics by domain. The total cost was $43,647.33, for an average of 32.7c per verb mention, 14.6c per question, or 16.5c per valid question. For comparison, He et al. (2015a) interviewed and hired contractors to annotate data at much smaller scale for a cost of about 50c per verb. Our annotation scheme is cheaper, far more scalable, and provides more (though noisier) supervision for answer spans.

[4] www.wiktionary.org

[5] www.mturk.com

To allow for more careful evaluation, we validated 5,205 sentences at a higher density (up to 1,000 for each domain in dev and test), re-running the generated questions through validation with $n = 3$ for a total of 6 answer annotations for each question.

Quality Judgments of question validity had moderate agreement. About 89.5% of validator judgments rated a question as valid, and the agreement rate between judgments of the same question on whether the question is invalid is 90.9%. This gives a Fleiss's Kappa of 0.51. In the higher-density re-run, validators were primed to be more critical: 76.5% of judgments considered a question valid, and agreement was at 83.7%, giving a Fleiss's Kappa of 0.55.

Despite being more critical in the denser annotation round, questions marked valid in the original dataset were marked valid by the new annotators in 86% of cases, showing our data's relatively high precision. The high precision of our annotation pipeline is also backed up by our small-scale manual evaluation (see Coverage below).

Answer spans for each question also exhibit good agreement. On the original dataset, each answer span has a 74.8% chance to exactly match one provided by another annotator (up to two), and on the densely annotated subset, each answer span has an 83.1% chance to exactly match one provided by another annotator (up to five).

Coverage Accurately measuring recall for QA-SRL annotations is an open challenge. For example, question 6 in Figure 5.1 reveals an inferred temporal relation that would not be annotated as part of traditional SRL. Exhaustively enumerating the full set of such questions is difficult, even for experts.

However, we can compare to the original QA-SRL dataset (He et al., 2015a), where Wikipedia sentences were annotated with 2.43 questions per verb. Our data has lower—but loosely comparable—recall, with 2.05 questions per verb in Wikipedia.

In order to further analyze the quality of our annotations relative to (He et al., 2015a), we reannotate a 100-verb subset of their data both manually (aiming for exhaustivity) and with our crowdsourcing pipeline. We merge the three sets of annotations, manually remove bad questions (and their answers), and calculate the precision and recall of the crowdsourced annotations and those of He et al. (2015a) against this pooled, filtered dataset (using the span detection metrics described in Section 5.4). Results, shown in Table 5.2b, show that our pipeline produces comparable precision with only a modest decrease in recall. Interestingly, re-adding the questions rejected in the validation step greatly increases recall with only a small decrease in precision, showing that validators sometimes rejected questions considered valid by the authors. However, we use the filtered dataset for our experiments, and in Section 5.5, we show how another crowdsourcing step can further improve recall.

5.3 Models

Given a sentence $\boldsymbol{X} = x_0, \ldots, x_n$, the goal of a QA-SRL parser is to produce a set of tuples $(v_i, \boldsymbol{Q_i}, \mathcal{S}_i)$, where $v \in \{0, \ldots, n\}$ is the index of a verbal predicate, $\boldsymbol{Q}_i$ is a question, and $\mathcal{S}_i \in \{(i,j) \mid i,j \in [0,n], j \geq i\}$ is a set of spans which are valid answers. Our proposed parsers construct these tuples in a three-step pipeline:

1. *Verbal predicates* are identified using the same POS-tags and heuristics as in data collection (see Section 5.2).
2. *Unlabeled span detection* selects a set $\mathcal{S}_v$ of spans as arguments for a given verb v.
3. *Question generation* predicts a question for each span in $\mathcal{S}_v$. Spans are then grouped by question, giving each question a set of answers.

We describe two models for unlabeled span detection in section 5.3.1, followed by question generation in section 5.3.2. All models are built on an LSTM encoding of the sentence. Like He et al. (2017b), we start with an input $\boldsymbol{X}_v = \{\boldsymbol{x}_0 \ldots \boldsymbol{x}_n\}$, where the representation $\boldsymbol{x}_i$ at each time step is a concatenation of the token w_i's embedding and an embedded binary feature $(i = v)$ which indicates whether w_i is the predicate under consideration. We then compute the output representation $\boldsymbol{H}_v = \textsc{BiLSTM}(\boldsymbol{X}_v)$ using a stacked alternating LSTM (Zhou and Xu, 2015a) with highway connections (Srivastava et al., 2015) and recurrent dropout (Gal and Ghahramani, 2016). Since the span detection and question generation models both use an LSTM encoding, this component could in principle be shared between them. However, in preliminary experiments we found that sharing hurt performance, so for the remainder of this work each model is trained independently.

5.3.1 Span Detection

Given an encoded sentence $\boldsymbol{H}_v$, the goal of span detection is to select the spans $\mathcal{S}_v$ that correspond to arguments of the given predicate. We explore two models: a sequence-tagging model with BIO encoding, and a span-based model which assigns a probability to every possible span.

BIO Sequence Model

Our BIO model predicts a set of spans via a sequence $\boldsymbol{y}$ where each $y_i \in \{\boldsymbol{B}, \boldsymbol{I}, \boldsymbol{O}\}$, representing a token at the beginning, interior, or outside of any span, respectively. Similar to He et al. (2017b), we make independent predictions for each token at training time, and use Viterbi decoding to enforce hard BIO-constraints[6] at test time. The resulting sequences are in one-to-one correspondence with sets $\mathcal{S}_v$ of spans which are pairwise non-overlapping. The locally-normalized BIO-tag distributions are computed from the BiLSTM outputs $\boldsymbol{H}_v = \{\boldsymbol{h}_{v0}, \ldots, \boldsymbol{h}_{vn}\}$:

$$p(y_t \mid \boldsymbol{x}) \propto exp(\boldsymbol{w}_{\text{tag}}^{\intercal}\text{MLP}(\boldsymbol{h}_{vt}) + \boldsymbol{b}_{\text{tag}}) \tag{5.1}$$

Span-based Model

Our span-based model makes independent binary decisions for all $O(n^2)$ spans in the sentence. Following Lee et al. (2016), the representation of a span (i, j) is the concatenation of the BiLSTM output at each endpoint:

$$\boldsymbol{s}_{vij} = [\boldsymbol{h}_{vi}, \boldsymbol{h}_{vj}]. \tag{5.2}$$

The probability that the span is an argument of predicate v is computed by the sigmoid function:

$$p(y_{ij} \mid \boldsymbol{X}_v) = \sigma(\boldsymbol{w}_{\text{span}}^{\intercal}\text{MLP}(\boldsymbol{s}_{vij}) + \boldsymbol{b}_{\text{span}}) \tag{5.3}$$

At training time, we minimize the binary cross entropy summed over all n^2 possible spans, counting a span as a positive example if it appears as an answer to any question.

[6] E.g., an I-tag should only follow a B-tag.

At test time, we choose a threshold τ and select every span that the model assigns probability greater than τ, allowing us to trade off precision and recall.

5.3.2 Question Generation

We introduce two question generation models. Given a span representation $\boldsymbol{s}_{vij}$ defined in Section 5.3.1, our models generate questions by picking a word for each question slot (see Section 5.2). Each model calculates a joint distribution $p(\boldsymbol{y} \mid \boldsymbol{X}_v, \boldsymbol{s}_{vij})$ over values $\boldsymbol{y} = (y_1, \ldots, y_7)$ for the question slots given a span $\boldsymbol{s}_{vij}$, and is trained to minimize the negative log-likelihood of gold slot values.

Local Model

The local model predicts the words for each slot independently:

$$p(y_k \mid \boldsymbol{X}_v, \boldsymbol{s}_{vij}) \propto \exp(\boldsymbol{w}_k^\intercal \text{MLP}(\boldsymbol{s}_{vij}) + \boldsymbol{b}_k). \tag{5.4}$$

Sequence Model

The sequence model uses the machinery of an RNN to share information between slots. At each slot k, we apply a multiple layers of LSTM cells:

$$\boldsymbol{h}_{l,k}, \boldsymbol{c}_{l,k} = \text{LSTMC}\textsc{ell}_{l,k}(\boldsymbol{h}_{l-1,k}, \boldsymbol{h}_{l,k-1}, \boldsymbol{c}_{l,k-1}) \tag{5.5}$$

where the initial input at each slot is a concatenation of the span representation and the embedding of the previous word of the question: $\boldsymbol{h}_{0,k} = [\boldsymbol{s}_{vij}; \boldsymbol{y}_{k-1}]$. Since each question slot predicts from a different set of words, we found it beneficial to use separate weights for the LSTM cells at each slot k. During training, we feed in the gold token at the previous slot, while at test time, we use the predicted token. The output distribution at slot k is computed via the final layers' output vector $\boldsymbol{h}_{Lk}$:

$$p(y_k \mid \boldsymbol{X}_v, \boldsymbol{s}_{vij}) \propto \exp(\boldsymbol{w}_k^\intercal \text{MLP}(\boldsymbol{h}_{Lk}) + \boldsymbol{b}_k) \tag{5.6}$$

5.4 *Initial Results*

Automatic evaluation for QA-SRL parsing presents multiple challenges. In this section, we introduce automatic metrics that can help us compare models. In Section 5.6, we will report human evaluation results for our final system.

5.4.1 Span Detection

Metrics We evaluate span detection using a modified notion of precision and recall. We count predicted spans as correct if they match any of the labeled spans in the dataset. Since each predicted span could potentially be a match to multiple questions (due to overlapping annotations) we map each predicted span to one matching question in the way that maximizes measured recall using maximum bipartite matching. We use both exact match and intersection-over-union (IOU) greater than 0.5 as matching criteria.

Results Table 5.3a shows span detection results on the development set. We report results for the span-based models at two threshold values τ: $\tau = 0.5$, and $\tau = \tau^*$ maximizing F1. The span-based model significantly improves over the BIO model in both precision and recall, although the difference is less pronounced under IOU matching.

5.4.2 Question Generation

Metrics Like all generation tasks, evaluation metrics for question generation must contend with the fact that there are in general multiple possible valid questions for a given predicate-argument pair. For instance, the question "Who did someone blame something on?" may be rephrased as "Who was blamed for something?" However, due to the constrained space of possible questions defined by QA-SRL's slot format, accuracy-based metrics can still be informative. In particular, we report the rate at which the predicted question exactly matches the gold question, as well as a relaxed match where we only count the question word (WH), subject (SBJ), object (OBJ) and Miscellaneous (Misc) slots (see Table 5.1). Finally, we report average slot-level accuracy.

Exact Match	**P**	**R**	**F**
BIO	69.0	75.9	72.2
Span ($\tau = 0.5$)	**81.7**	80.9	81.3
Span ($\tau = \tau*$)	80.0	**84.7**	**82.2**
IOU $\geq$ 0.5	**P**	**R**	**F**
BIO	80.4	86.0	83.1
Span ($\tau = 0.5$)	**87.5**	84.2	85.8
Span ($\tau = \tau*$)	83.8	**93.0**	**88.1**

(a) Results for Span Detection on the dense development dataset. Span detection results are given with the cutoff threshold τ at 0.5, and at the value which maximizes F-score. We report precision, recall and F-score both with exact span matching and matches where the intersection over union (IOU) is ≥ 0.5.

	EM	**PM**	**SA**
Local	44.2	62.0	**83.2**
Seq.	**47.2**	**62.3**	82.9

(b) Question Generation results on the dense development set. **EM** - Exact Match accuracy, **PM** - Partial Match Accuracy, **SA** - Slot-level accuracy

	P	**R**	**F**
Span + Local	37.8	43.7	40.6
Span + Seq. ($\tau = 0.5$)	**39.6**	**45.8**	**42.4**

(c) Joint span detection and question generation results on the dense development set, using exact-match for both spans and questions.

Table 5.3: Experimental results on the dense development set for our initial trained models.

Results Table 5.3b shows the results for question generation on the development set. The sequential model's exact match accuracy is significantly higher, while word-level accuracy is roughly comparable, reflecting the fact that the local model learns the slot-level posteriors.

5.4.3 Joint results

Table 5.3c shows precision and recall for joint span detection and question generation, using exact match for both. This metric is exceedingly hard, but it shows that almost 40% of predictions are exactly correct in both span and question. In Section 5.6, we use human evaluation to get a more accurate assessment of our model's accuracy.

5.5 Data Expansion

Since our trained parser can produce full QA-SRL annotations, its predictions can be validated by the same process as in our original annotation pipeline, allowing us to focus annotation efforts towards filling potential data gaps.

By detecting spans at a low probability cutoff, we over-generate QA pairs for already-annotated sentences. Then, we filter out QA pairs whose answers overlap with answer spans in the existing annotations, or whose questions match existing questions. What remains are candidate QA pairs which fill gaps in the original annotation. We pass these questions to the validation step of our crowdsourcing pipeline with $n = 3$ validators, resulting in new labels.

We run this process on the training and development partitions of our dataset. For the development set, we use the trained model described in the previous section. For the training set, we use a relaxed version of jackknifing, training 5 models over 5 different folds. We generate 92,080 questions at a threshold of $\tau = 0.2$. Since in this case many sentences have only one question, we restructure the pay to a 2c base rate with a 2c bonus per question after the first (still paying no less than 2c per question).

Data statistics 46,017 (50%) of questions run through the expansion step were considered valid by all three annotators. In total, after filtering, the expansion step increased the number of valid questions in the train and dev partitions by 20%. However, for evaluation, since our recall metric identifies a single question for each answer span (via bipartite matching), we filter out likely question paraphrases by removing questions in the expanded development set whose answer spans have two overlaps with the answer spans of one question in the original annotations. After this filtering, the expanded development set we use for evaluation has 11.5% more questions than the original development set.

The total cost including MTurk fees was $8,210.66, for a cost of 8.9c per question, or 17.8c per valid question. While the cost per valid question was comparable to the initial annotation, we gathered many more negative examples (which may serve useful in future work), and this method allowed us to focus on questions that were missed in the first round and improve the exhaustiveness of the annotation (whereas it is not obvious how to make fully crowdsourced annotation more exhaustive at a comparable cost per question).

Exact Match	**P**	**R**	**F**	**AUC**
Original	80.8	**86.8**	83.7	.906
Expanded	**82.9**	86.4	**84.6**	**.910**
IOU $\geq$ 0.5	**P**	**R**	**F**	**AUC**
Original	87.1	**93.2**	90.1	.946
Expanded	**87.9**	93.1	**90.5**	**.949**

(a) Span detection results with $\tau*$.

	EM	**PM**	**SA**
Original	50.5	64.4	**84.1**
Expanded	**50.8**	**64.9**	**84.1**

(b) Question generation.

	P	**R**	**F**
Original	**47.5**	46.9	47.2
Expanded	44.3	**55.0**	**49.1**

(c) Joint span detection and question generation with $\tau = 0.5$.

Table 5.4: Results on the expanded development set comparing the full model trained on the original data, and with the expanded data.

Retrained model We retrained our final model on the training set extended with the new valid questions, yielding modest improvements on both span detection and question generation in the development set (see Table 5.4). The span detection numbers are higher than on the original dataset, because the expanded development data captures true positives produced by the original model (and the resulting increase in precision can be traded off for recall as well).

5.6 Final Evaluation

We use the crowdsourced validation step to do a final human evaluation of our models. We test 3 parsers: the span-based span detection model paired with each of the local and sequential question generation models trained on the initial dataset, and our final model (span-based span detection and sequential question generation) trained with the expanded data.

Methodology On the 5,205 sentence densely annotated subset of dev and test, we generate QA-SRL labels with all of the models using a span detection threshold of $\tau = 0.2$ and combine the questions with the existing data. We filter out questions that fail the autocomplete grammaticality check (counting them invalid) and pass the data into the validation step, annotating each question to a total of 6 validator judgments. We then compute question and span accuracy as follows: A question is considered correct if 5 out of 6 annotators consider it valid, and a span is considered correct if its generated question is correct and the span is among those selected for the question by validators. We rank all questions and spans by the threshold at which they are generated, which allows us to compute accuracy at different levels of recall.

Results Figure 5.3 shows the results. As expected, the sequence-based question generation models are much more accurate than the local model; this is largely because the local model generated many questions that failed the grammaticality check. Furthermore, training with our expanded data results in more questions and spans generated at the same threshold. If we choose a threshold value which gives a similar number of questions per sentence as were labeled in the original data annotation (2 questions / verb), question and span accuracy are 82.64% and 77.61%, respectively.

Table 5.5 shows the output of our best system on 3 randomly selected sentences from our development set (one from each domain). The model was overall highly accurate—only one question and 3 spans are considered incorrect, and each mistake is nearly correct,[7] even when the sentence contains a negation.

5.7 Related Work

Resources and formalisms for semantics often require expert annotation and underlying syntax (Palmer et al., 2005; Baker et al., 1998b; Banarescu et al., 2013c). Some more recent semantic resources require less annotator training, or can be crowdsourced (Abend and Rappoport, 2013a; Reisinger et al., 2015b; Basile et al., 2012; Michael et al., 2018). In particular, the original QA-SRL (He et al., 2015a) dataset is annotated by freelancers, while we developed streamlined crowdsourcing approaches for more scalable annotation.

Crowdsourcing has also been used for indirectly annotating syntax (He et al., 2016; Duan et al., 2016), and to complement expert annotation of SRL (Wang et al., 2017). Our crowdsourcing

[7] The incorrect question "When did someone appear?" would be correct if the Prep and Misc slots were corrected to read "When did someone appear to do something?"

approach draws heavily on that of Michael et al. (2018), with automatic two-stage validation for the collected question-answer pairs.

More recently, models have been developed for these newer semantic resources, such as UCCA (Teichert et al., 2017) and Semantic Proto-Roles (White et al., 2017b). Our work is the first high-quality parser for QA-SRL, which has several unique modeling challenges, such as its highly structured nature and the noise in crowdsourcing.

Several recent works have explored neural models for SRL tasks (Collobert and Weston, 2007; FitzGerald et al., 2015; Swayamdipta et al., 2017; Yang and Mitchell, 2017), many of which employ a BIO encoding (Zhou and Xu, 2015a; He et al., 2017b). Recently, span-based models have proven to be useful for question answering (Lee et al., 2016) and coreference resolution (Lee et al., 2017b), and PropBank SRL (He et al., 2018).

5.8 Conclusion

In this paper, we demonstrated that QA-SRL can be scaled to large datasets, enabling a new methodology for labeling and producing predicate-argument structures at a large scale. We presented a new, scalable approach for crowdsourcing QA-SRL, which allowed us to collect QA-SRL Bank 2.0, a new dataset covering over 250,000 question-answer pairs from over 64,000 sentences, in just 9 days. We demonstrated the utility of this data by training the first parser which is able to produce high-quality QA-SRL structures. Finally, we demonstrated that the validation stage of our crowdsourcing pipeline, in combination with our parser tuned for recall, can be used to add new annotations to the dataset, increasing recall.

5.9 Notes

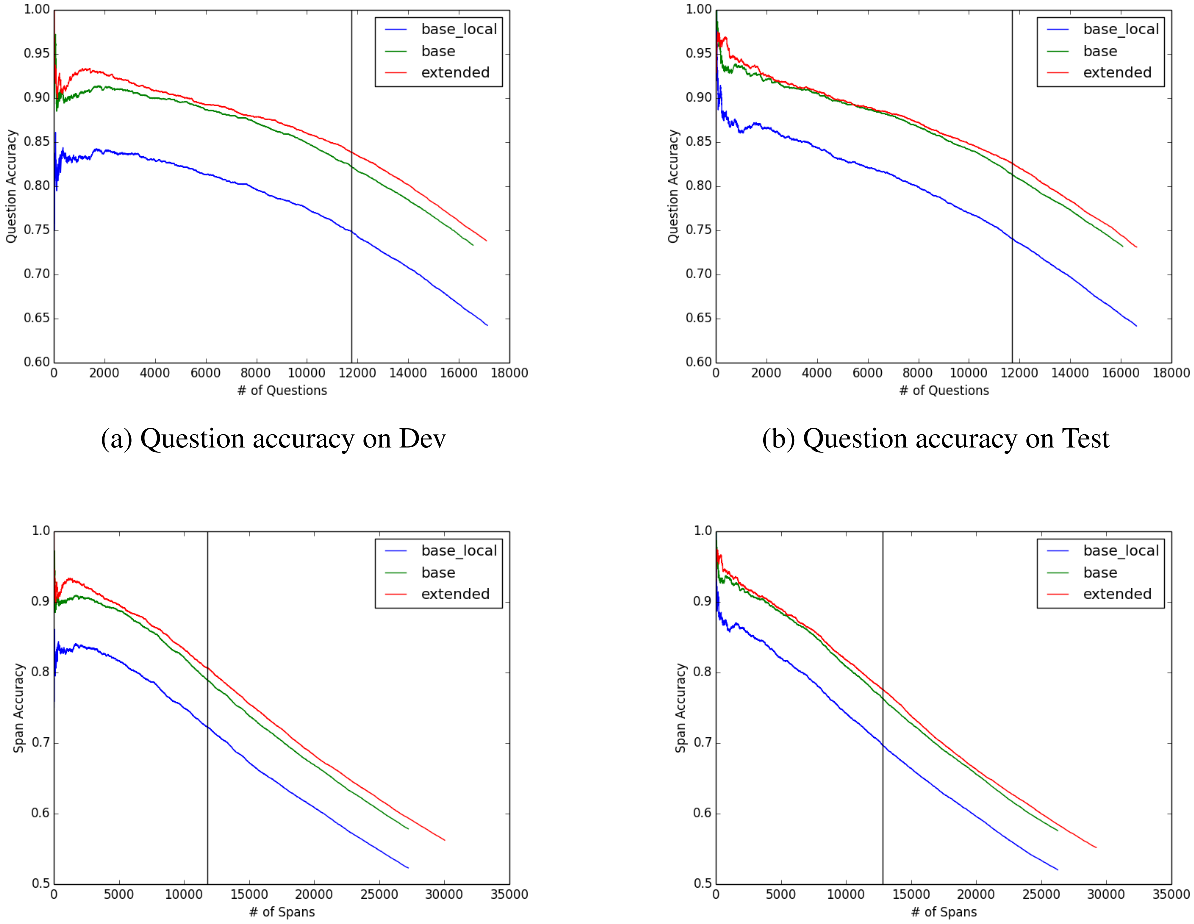
5.9.1 Experimental Setup

Hyperparameters The parameters of our LSTMs are initialized with random orthonormal matrices as described by Saxe et al. (2014). Input tokens are lower-cased, and the word vectors are pre-initialized with the 100-dimensional Glove embeddings trained on 6B tokens (Pennington et al., 2014a) and fine-tuned during training. Tokens which are not covered by the Glove embeddings are assigned to the UNK vector. The embedding of the binary predicate indicator feature is also 100 dimensions. The text-encoder BiLSTM consists of 4 layers, uses a hidden size of 300 and . The output prediction feed-forward neural network for each model consists of a single 100 dimensional hidden layer with the non-rectified linear unit nonlinearity. For the sequential question generation model, each timestep consists of 4 layers of LSTMCells with a hidden size of 200.

Training All models are trained using Adadelta (Zeiler, 2012) with $\epsilon = 1e^{-6}$ and $\rho = 0.95$ and a mini-batch size of 80. The span encoding BiLSTM uses a recurrent dropout rate of 0.1, and we clip gradients with a norm greater than 1. All models were trained until performance on the development set did not improve for 10 epochs[8]. Our models were implemented in PyTorch[9] using the AllenNLP toolkit (Gardner et al., 2018).

[8] All models completed training within 40 epochs, which took less than 4 hours on a single Titan X Pascal GPU.

[9] `http://pytorch.org/`



(a) Question accuracy on Dev

(b) Question accuracy on Test

(c) Span accuracy on Dev

(d) Span accuracy on Test

Figure 5.3: Human evaluation accuracy for questions and spans, as each model's span detection threshold is varied. Questions are considered correct if 5 out of 6 annotators consider it valid. Spans are considered correct if their question was valid, and the span was among those labeled by human annotators for that question. The vertical line indicates a threshold value where the number of questions per sentence matches that of the original labeled data (2 questions / verb).

A much larger super eruption in Colorado **produced** over 5,000 cubic kilometers of material.	Produced	What produced something?	A much larger super eruption
		Where did something produce something?	in Colorado
		What did something produce?	over 5,000 cubic kilometers of material
In the video, the perpetrators never **appeared** to **look** at the camera.	appeared	Where didn't someone appear to do something?	In the video
		Who didn't appear to do something?	the perpetrators
		When did someone appear?	never
		What didn't someone appear to do?	look at the camera
			to look at the camera
	look	Where didn't someone look at something?	In the video
		Who didn't look?	the perpetrators
		What didn't someone look at?	the camera
Some of the vegetarians he **met** were members of the Theosophical Society, which had been **founded** in 1875 to further universal brotherhood, and which was **devoted** to the study of Buddhist and Hindu literature.	met	Who met someone?	Some of the vegetarians
			vegetarians
		Who met?	he
		What did someone meet?	members of the Theosophical Society
	founded	What had been founded?	members of the Theosophical Society
			the Theosophical Society
		When was something founded?	in 1875
			1875
		Why has something been founded?	to further universal brotherhood
	devoted	What was devoted to something?	members of the Theosophical Society
		What was something devoted to?	the study of Buddhist and Hindu literature

Table 5.5: System output on 3 randomly sampled sentences from the development set (1 from each of the 3 domains). Spans were selected with $\tau = 0.5$. Questions and spans with a red background were marked incorrect during human evaluation.

Part III

THEORY: FROM LANGUAGE, STRUCTURE

In Part I, we introduced the idea of *data-driven theories*. In Part II, we discussed what the *data* part may look like. Now we turn to theory.

In the physical sciences, developing and testing a theory requires constructing a mathematical model, designing an experiment, predicting the experiment's outcome using the model, and then performing the experiment and comparing the result to the predicted one. If the predicted and observed measurements don't match, the experimental setup and theory must be scrutinized and revised until harmony is reached.

In linguistics and AI, rarely does it seem so simple. When it comes to the kind of behaviors we want to build into NLP systems, there are few widely accepted theories of language behavior that have the kind of concrete, testable predictive power that we can get in the physical sciences. I think there are a couple main reasons for this:

- **Irreducible complexity.** The phenomena we are trying to explain are fundamentally extremely complex: theories of language behavior must account for the lexicon, compositional structure, world knowledge, common sense, reasoning, and more. It is impossible to fully specify any explanatory theory of such things by hand, particularly because the details of such a theory will also vary by individual speakers and change over time. So those who work in the realm of theory (*e.g.*, linguists) generally work with very small subtheories (*e.g.*, grammar fragments) targeted towards their phenomena of interest, and assume the rest — which is out of scope for them — happens to work out.[10] In the context of NLP and AI, we are often forced to deal with a great deal of the complexity all at once: for example, while a syntactician may decide

[10] An exception to this is the practice of grammar engineering, which closely approximates the idea of data-driven theory advocated for in this thesis. The main difference is my greater explicit commitment to theoretical minimalism and the scope of phenomena I am interested in explaining being much broader.

not to concern themselves with the mechanisms that drive syntactic disambiguation, building a syntactic parser requires facing that problem head-on.

- **Measurement challenges.** Another (related) problem is that people may not always agree on what it means for a theory to be "falsified," because of questions about what exactly the theory is *supposed* to predict. The clearest example of this from linguistics is probably the *competence–performance distinction*, whereby language behavior which is not accounted for by a theory of so-called linguistic "competence" can be dismissed as a "performance" issue (or due to "genre effects"). I am not saying no such distinction exists, but (again) in the context of AI we are not so easily afforded an *à la carte* approach to the phenomena we wish to model.

In data-driven theory, I propose a resolution to both of these problems. Carefully specifying and scoping the data, then gathering it at large scale, eliminates any uncertainty over what phenomena a theory is supposed to explain. Then, using computational modeling to automatically induce the theory from that data gives us a way of managing and making sense of the complexity of the data in a way that is globally consistent.

To make this idea concrete, I will describe how through two concurrent works, coauthors and I have shown that QA-SRL forms a powerful backbone for a data-driven theory of semantic roles:

- In Michael and Zettlemoyer (2021), covered in Chapter 6, we show how to use QA-SRL to automatically induce an ontology of semantic roles. The approach is based on a key insight: the *set* of QA-SRL questions that are correctly answered by a given answer span identify an underlying semantic role through its set of syntactic alternations. We leverage this insight by using a trained QA-SRL question generator as a data simulator (see Section 3.4), generating a full distribution over (simplified) QA-SRL questions for each argument of a verb appearing through an entire corpus. Clustering these distributions of questions according to a simple maximum-likelihood objective yields a set of discrete semantic roles that exhibits high agreement with existing resources. This presents an approach which could potentially be used to develop semantic role ontologies in new domains where they are not currently available.

- Pyatkin et al. (2021)[11] show how to use QA-SRL to structure a language generation system. The task is to generate fluent questions that ask about the arguments corresponding to specific semantic roles in context, where the roles are drawn from a pre-existing ontology. The primary challenge here is a lack of training data: only QA-SRL provides such exhaustive annotations of questions for each verb, but it does not produce fully fluent questions (maintaining the simple QA-SRL format) and we have no annotations for semantic roles which aren't expressed in a given sentence. We overcome these challenges with two key insights: First, we find that QA-SRL questions corresponding to a specific role generally maintain their semantics (*i.e.*, correspond to the same role) across many contexts. This allows us to prime our question generation system with a template QA-SRL question corresponding to the correct role, allowing it to generate semantically correct questions even when the answer isn't provided in context (a situation never encountered during training). Second, we find that we can leverage the syntactic structure within a QA-SRL question to align the placeholders (*who*, *what*, *someone*, *something*, etc.) in each question with the answers of other questions, translating the QA-SRL Bank 2.0 (FitzGerald et al., 2018, Chapter 5) into the *Frame-Aligned QA-SRL Bank* which contains exhaustive annotation of more fluent questions, closer to what is provided in a QAMR (Michael et al., 2018, Chapter 4).

Together this work illustrates not only the promise for the development of large-scale ontologies in a data-driven way (Michael and Zettlemoyer, 2021), but it also illustrates how having these ontologies computationally grounded in the phenomena they are designed to explain, *i.e.*, question-answer pairs (Pyatkin et al., 2021), facilitates the downstream use of such an ontology. It's not hard to imagine next steps combining these works: In future systems, using semantic roles automatically induced from QA-SRL can obviate the need for pre-specified role ontologies altogether. This would be great news for NLP systems working with semantic roles, but my hope is that these insights may translate to other task settings as well — an issue I will come back to in Part IV.

[11] In this project, I served in an advisory role as well as developing the placeholder-alignment algorithm and Frame-Aligned QA-SRL Bank.

Chapter 6

INDUCING SEMANTIC ROLES WITHOUT SYNTAX

Semantic roles are a key component of linguistic predicate-argument structure, but developing ontologies of these roles requires significant expertise and manual effort. Methods exist for automatically inducing semantic roles using syntactic representations, but syntax can also be difficult to define, annotate, and predict. We show that in English, it is possible to automatically induce semantic roles from QA-SRL, a scalable and ontology-free semantic annotation scheme that uses question-answer pairs to represent predicate-argument structure. By associating arguments with distributions over QA-SRL questions and clustering them in a mixture model, our method outperforms all previous models as well as a new state-of-the-art baseline over gold syntax. We show that our method works because QA-SRL acts as *surrogate syntax*, capturing non-overt arguments and syntactic alternations, which are central motivators for the use of semantic role labeling systems.[1]

6.1 Introduction

Semantic role labeling (SRL) requires extracting propositional predicate-argument structure from language, *i.e.*, *who* is doing *what* to *whom*. Applications of SRL include information extraction (Christensen et al., 2011), machine reading (Wang et al., 2015), and model analysis (Tenney et al., 2019a; Kuznetsov and Gurevych, 2020), and semantic roles form the backbone of many more general meaning representations (Banarescu et al., 2013a; Abend and Rappoport, 2013b).

The primary challenge, and promise, for SRL systems is to distill syntactically variable surface structures into semantic predicate-argument structures from an ontology (Palmer et al., 2005; Baker et al., 1998a). However, ontologies and their associated training data require time and expertise to

[1] This chapter is based on Michael and Zettlemoyer (2021). Code, models, and a web interface to explore the results are available at `https://github.com/julianmichael/qasrl-roles`.

*The plane was **diverting** around weather formations over the Java Sea when contact with air traffic control (ATC) in Jakarta was **lost**.*

wh	aux	subj	verb	obj	prep	obj2	?	Answer
What	was		being diverted		around		?	*weather formations*
What	was		diverting				?	*The plane*
What	was		being diverted				?	*The plane*
What	was		lost				?	*contact with air traffic control*
Where	was	something	lost				?	*over the Java Sea*

Table 6.1: Example QA-SRL question-answer pairs from the development set of the QA-SRL Bank 2.0 (FitzGerald et al., 2018). Questions may be represented in a verb-agnostic way by recording the form of the verb in the **verb** slot (e.g., *stem*, *past participle*). Note that the syntax used in questions may differ from the syntax in the source sentence, for example in the above questions using *diverted* in its passive form.

Labels	Questions	
A1 (98%)	What is given?	.30
	What does something give something?	.21
	What does something give?	.20
	What is something given?	.11
A0 (98%)	What gives something?	.44
	What gives something something?	.27
	What gives something to something?	.08
A2 (94%)	What is given something?	.28
	What does something give something to?	.18
	What does something give something?	.14
	What is given?	.09
	What is something given to?	.07
TMP (46%),	When does something give something?	.20
ADV (22%),	How does something give something?	.09
MNR (12%)	When is something given?	.09
	When is something given something?	.09
PNC (30%),	Why does something give something?	.18
ADV (22%),	Why does something give up something?	.07
TMP (14%)	Why is something given something?	.07

Table 6.2: Roles for *give* produced by our final model. Core arguments are captured almost perfectly, exhibiting both passive and dative alternations.

annotate and do not readily generalize to new domains, limiting their broad-coverage applicability. Prior work towards mitigating this problem includes unsupervised induction of semantic roles from syntactic representations (Lang and Lapata, 2010). However, the need for formal syntactic supervision retains some of the annotation and generalization difficulties of supervised SRL, and it has proven difficult to do much better than a simple syntactic baseline (Lang and Lapata, 2011). An alternative is to use an ontology-free annotation scheme like QA-SRL (He et al., 2015b), which represents roles with natural language questions. While QA-SRL can be annotated at large scale (FitzGerald et al., 2018), many different QA-SRL questions may correspond to the same role, making it more difficult to use in downstream tasks.

We show how to overcome this difficulty, by automatically inducing an ontology of semantic roles corresponding to clusters of QA-SRL questions (see Table 6.2 for an example clustering). We use a model to predict a distribution over QA-SRL questions associated with each argument in a corpus, and cluster them to maximize likelihood under a simple model we call a *Hard Unigram Mixture*. Our model can be effectively optimized both by EM and greedy methods, which affords the benefits of tunable hierarchical clustering without sacrificing scalability (Section 6.3).

Experiments in semantic role induction (Section 6.4) show that our method outperforms all previous methods in the literature, as well as a new state-of-the-art baseline over gold syntax. This is despite requiring no formal syntactic supervision or theory, where the formalism used by previous work is highly informative of gold standard semantic roles (Section 6.5). We also present a detailed analysis (Section 6.6) showing why our method works: QA-SRL acts as *surrogate syntax*, removing (role-irrelevant) syntactic variation in the source text such as that from *non-overt* arguments (*e.g.*, phrases extracted from relative clauses), while itself exhibiting (role-relevant) syntactic alternations which capture the behavior of verbal predicates (Table 6.2). While our model is only defined for English, as that is how QA-SRL is defined, the principle of surrogate syntax could conceivably be applied to other languages as well. Taken together, these results paint a path towards on-the-fly, data-driven construction of useful, interpretable ontologies of semantic structure.

6.2 Task Setting

The input to our task is a set of natural language sentences, where a subset of the tokens are marked as *predicates*. Each predicate has a set of *arguments*, and each argument x corresponds to a set of spans $x = \{s_1, \dots, s_m\}$ in the predicate's sentence.[2]

An ontology of semantic roles is a set of *frames* (corresponding to semantic predicates), and each frame has a set of associated *roles* (corresponding to participants in the event or state denoted by its frame). There may also be a set of *modifier roles* (*e.g.*, location or time) which can appear with any frame. In supervised semantic role labeling, each predicate in the input data must be assigned to one of the frames in a given ontology, and each of a predicate's arguments must be assigned roles from its frame (or modifier roles). In semantic role induction, our task is to produce both the ontology and these assignments.

We follow prior work (Lang and Lapata, 2010) in treating semantic role induction as a clustering problem and assuming a single frame per predicate lemma.[3] Given input data marked with predicates and their arguments, we cluster the arguments for each predicate into sets corresponding to semantic roles. We may then compare these clusters to gold labels using clustering metrics (Section 6.4.3).

6.3 Modeling

Our model treats each argument x as a set of counts of QA-SRL questions, denoted $\phi(x)$. We produce these counts from a trained QA-SRL question generator (Section 6.3.1) and cluster them by maximizing their likelihood under a mixture model (Section 6.3.2) using a hybrid of flat and hierachical clustering (Section 6.3.3).

[2] Previous work (Lang and Lapata, 2010) assumes a syntactic dependency tree and marks each argument by its syntactic head, which allows for features based on argument lemmas and dependency paths. We instead assume sets of argument spans, but no syntax tree; this allows for features based on spans (such as QA-SRL questions). Both approaches are ways of featurizing the same gold arguments.

[3] Some ontologies, like FrameNet (Baker et al., 1998a), define frames that span multiple lemmas (e.g., *buy* and *sell* share a *Commercial Transaction* frame), whereas others like PropBank (Palmer et al., 2005) use frames which are specific to each lemma, denoting something closer to word sense. In our case, assuming a single frame per lemma simplifies modeling and allows us to compare to previous work. However, modeling predicate sense is an important problem for future work, as we will suggest in Section 6.6.3.

6.3.1 Generating QA-SRL Features

For each argument x of a predicate, we leverage a trained QA-SRL parser to generate pseudocounts $\phi(x)$ of simplified QA-SRL questions, which will form the input features for the clustering step.

Simplified QA-SRL Example QA-SRL questions are shown in Table 6.1. These questions contain information which is not directly relevant to semantic roles, such as tense, aspect, modality, and negation. Since this creates sparsity for our model, we remove it as a preprocessing step. In particular, we replace the **aux** and **verb** slot values with either *is* and *past participle* (for passive voice), _ and *present* (for active voice when **subj** is blank), or *does* and *stem* (for active voice when **subj** is present). We also replace all occurrences of *who* and *someone* with *what* or *something*.

Generating Question Counts Let p denote a predicate, s denote a span, and q denote a simplified QA-SRL question. To generate our question count vectors ϕ, we reproduce the QA-SRL question generator of FitzGerald et al. (2018), which generates a distribution $\mathrm{P}(q \mid p, s)$ over QA-SRL questions conditioned on a predicate p and answer span s in a sentence. This model uses a BiLSTM encoder, concatenating the output representations of span endpoints and feeding them into a custom LSTM decoder which models the QA-SRL slot values in sequence. We modify the model to use BERT (Devlin et al., 2019) features as input embeddings for the BiLSTM (details in Section 6.8.1).

Recall from Section 6.2 that an argument x consists of a set of spans from its sentence. We generate question counts $\phi(x) \in \mathbb{R}_{\geq 0}^{|q|}$ by taking the mean

$$\phi(x) = \frac{1}{|x|} \sum_{s \in x} \mathrm{P}(q \mid p, s),$$

where $\mathbb{R}_{\geq 0}$ denotes the nonnegative real numbers and $|q|$ is the number of possible simplified QA-SRL questions. Since $|q|$ is large, to make this tractable we approximate $\mathrm{P}(q \mid p, s)$ with beam search, using a sparse representation and assigning counts of 0 to questions outside the beam.

6.3.2 Objective

Let $\mathbf{X} = \{x_1, \ldots, x_n\}$ be the set of input arguments for clustering. Our goal is a clustering $\mathbf{C} = \{C_1, \ldots, C_k\}$ which is a partition of $\mathbf{X}$. We model each argument's questions $\phi(x)$ as being drawn from a mixture model over latent roles, each corresponding to a cluster $C \in \mathbf{C}$. We maximize likelihood under this model, which we call a Hard Unigram Mixture, with the addition of a connectivity penalty which encourages roles not to appear twice for the same predicate instance.

The Hard Unigram Mixture (HUM) Recall that $\phi : \mathbf{X} \to \mathbb{R}^d_{\geq 0}$ assigns question pseudocounts to each $x \in \mathbf{X}$. Let π denote a probability distribution over $\{1, \ldots, k\}$ and θ a distribution over $\{1, \ldots, d\}$. We propose the *Hard Unigram Mixture* loss

$$\mathcal{L}^{\text{HUM}}_{\lambda}(\mathbf{C}) = -\log \mathrm{P}(\mathbf{X} \mid \mathbf{C}) - \lambda \log \mathrm{P}(\mathbf{C}),$$

where

$$\mathrm{P}(\mathbf{X} \mid \mathbf{C}) = \prod_{i}^{k} \max_{\theta} \prod_{x \in C_i} \mathrm{P}(\phi(x) \mid \theta)$$

is the *data likelihood* and

$$\mathrm{P}(\mathbf{C}) = \max_{\pi} \prod_{i}^{k} \pi_i^{||C_i||}$$

is the *clustering likelihood*, writing $||C||$ for the sum of the ϕ counts in a cluster C. The data likelihood prefers more, smaller clusters, the clustering likelihood prefers fewer clusters, and λ is a hyperparameter that trades off between them.[4]

Connectivity Penalty Let $p(x)$ denote the predicate instance corresponding to an argument x. We propose a *connectivity penalty*

$$\mathcal{L}^{\text{cp}}(\mathbf{C}) = \frac{1}{2} \sum_{i}^{k} \sum_{x_1, x_2 \in C_i} \delta(p(x_1) = p(x_2)),$$

[4] Here, $\mathcal{L}^{\text{HUM}}_{1}$ is equivalent to the negative log likelihood under the maximum likelihood estimate of a mixture of unigrams model (Nigam et al., 2000) constrained to hard assignments $\mathbf{C}$; hence the name *Hard Unigram Mixture*. Further theoretical and empirical comparison to prior work is provided in Section 6.8.7.

where δ is the indicator function, which discourages clusterings where multiple arguments of the same predicate instance are assigned the same role. This assumption has also been leveraged by prior models (Lang and Lapata, 2011; Titov and Klementiev, 2012).

Loss Function Our full loss is then

$$\mathcal{L}_\lambda(\mathbf{C}) = \mathcal{L}^{\text{HUM}}_\lambda(\mathbf{C}) + \mathcal{L}^{\text{cp}}(\mathbf{C})$$

with the single hyperparameter λ.

6.3.3 Hybrid Clustering

We optimize $\mathcal{L}_\lambda$ in three steps: flat pre-clustering, greedy merging, and tuned splitting. This approach provides us with both the efficiency benefits of flat clustering and the relative determinism, interpretability and tunability of hierarchical clustering.

Flat Pre-Clustering For pre-clustering, we minimize $\mathcal{L}_0$ via hard EM. To avoid likelihoods of 0 in $\mathcal{L}^{\text{HUM}}_0$, we smooth our estimates of θ using a Dirichlet prior. To optimize $\mathcal{L}^{\text{cp}}$ via EM, we draw x_1 from the previous iteration's clustering in order to compute the contribution of each x_2 to the loss. With sufficiently large k, this can produce a high-precision clustering in $O(nk)$ time to serve as input to the merging step.

Greedy Merging After pre-clustering, we produce a binary cluster tree by iteratively merging pairs of clusters which greedily minimize $\mathcal{L}_0$. Since $\lambda = 0$, the loss grows monotonically when merging clusters. The loss at each merge can be efficiently updated by maintaining maximum likelihood estimates θ for each cluster.

Tuned Splitting Finally, we iteratively split the cluster tree produced by the merging stage. At each step, we split the cluster C_i with the lowest log data likelihood per item $\frac{\log \mathrm{P}(C_i|\mathbf{C})}{|C_i|}$. We then choose the clustering which minimizes $\mathcal{L}_\lambda$, with $\lambda > 0$ tuned during model development.[5]

[5] A comparison of this method against a constant-k baseline and oracle upper bound is given in Section 6.8.5.

6.4 Experimental Setup

Data We run experiments on the distribution of PropBank (Palmer et al., 2005) provided for the CoNLL 2008 Shared Task (Surdeanu et al., 2008). We use the same setup as previous work, removing arguments annotated with reference (R-) and continuation (C-) roles, keeping only verbal predicates,[6] and using the development set for model development and the training set for testing.

Our one preprocessing difference from previous work is that instead of using the dependency-based SRL annotations provided in the CoNLL 2008 dataset, we use full answer spans, which we reconstruct by aligning the CoNLL 2008 data back to the original annotations in the Penn Treebank (Marcus et al., 1993a) and PropBank.[7]

6.4.1 Models

***HUM of QA-SRL Questions* (HUM-QQ)** We train a QA-SRL parser on the expanded set of the QA-SRL Bank 2.0 (FitzGerald et al., 2018) using the architecture described in Section 6.3.1. In the pre-clustering step, we estimate $k = 100$ clusters. For tuned splitting, we choose λ to maximize performance on the development set. Hyperparameters are detailed in Section 6.8.2.

SYNTF This model assigns each argument to a cluster corresponding to the label of its syntactic dependency to its parent, using the syntactic formalism provided in CoNLL 2008 Shared Task data. Past work has found SYNTF to be a strong baseline (Lang and Lapata, 2011).

Prior Work We compare to Bayesian generative modeling (Titov and Klementiev, 2012, BAYES), which is state-of-the-art on gold syntax, and an embedding-based method (Luan et al., 2016, SYMDEP/ASYMDEP) which is state-of-the-art using automatic syntax. These as well as all other

[6] While we ignore nominal predicates, our method naturally generalizes to nominalizations, which are provided with QA-SRL annotations in QANom (Klein et al., 2020).

[7] Using gold spans is necessary in order to compare to previous work and use the CoNLL 2008 dataset for evaluation of role induction. In a more realistic setting where gold argument spans are not available, we could use the span detector of FitzGerald et al. (2018) to construct argument spans.

prior approaches (*e.g.*, Lang and Lapata, 2011; Titov and Khoddam, 2015; Woodsend and Lapata, 2015) crucially rely on syntactic features.

6.4.2 Auxiliary Clustering Rules

For SYNTF and HUM-QQ, we experiment with several auxiliary clustering rules.

Lexical Rules We employ three lexical rules, each producing a separate cluster for all arguments whose spans exactly match a phrase contained in the rule's lexicon. Our rules are for negation (5 phrases), modals (23 phrases), and discourse modifiers (55 phrases). These lexica were written to correspond to the AM-NEG, AM-MOD, and AM-DIS roles on the basis of the PropBank annotation guidelines (Babko-Malaya, 2005) and development set.[8]

Passive to Active Conversion We also propose a syntactic rule that applies only to SYNTF, where we the transform the dependencies as follows:

- The LGS label, meaning "logical subject," is a dependency label given for *by*-phrases modifying a passive verb whose object denotes what is normally the subject of the verb's active form (Surdeanu et al., 2008). We change this to SBJ.

- Passive voice can be detected when the predicate verb is in past participle form (part-of-speech tag VBN) and its syntactic parent is a *be*-verb (part of speech VC, lemma "be"). In these cases, we change the syntactic label of any SBJ dependents into OBJ.

6.4.3 Metrics

Purity/Collocation To compare with previous work, we follow Lang and Lapata (2010) in using purity and collocation based F1 score for our main evaluation. Purity measures cluster homogeneity: it assigns to each cluster the gold label for which it has the most points, and then measures

[8] Full lexica for these rules are provided in Section 6.8.3.

the proportion of points which have their cluster's assigned label. Collocation measures cluster concentration: it assigns each gold label to the cluster which contains the most of its points, and then measures the proportion of points which are in their gold label's assigned cluster. These are calculated independently for each verb and averaged, weighing each verb by its number of argument instances. The harmonic mean of the final results is reported as an F1 score.

$\mathbf{B^3}$ For deeper analysis, we use the B^3 (*B-cubed*) family of clustering metrics (Bagga and Baldwin, 1998). B^3 precision and recall are the precision and recall of each point's predicted cluster against its gold cluster, averaging over points. In comparison to purity and collocation, these metrics are tougher and more discriminative between clusterings, respecting important constraints like the cluster completeness constraint of Rosenberg and Hirschberg (2007), among others (Amigó et al., 2009). B^3 also allows us to reliably report scores along slices of the data for analysis purposes, as well as account for each slice's contribution to the total error. We report full B^3 results for our models in Section 6.8.6 and encourage future work to use these as the primary metrics.

6.5 Results

Main results are shown in Table 6.3. Our auxiliary rules put SYNTF significantly above the state of the art for gold syntax (with 85.2 F1 versus 83.0). HUM-QQ surpasses it with 87.1 F1 in the best case, despite not using gold syntax at all.

6.5.1 A Stronger Syntactic Baseline

For SYNTF, the addition of either lexical (negation, modal, and discourse) rules or the passive-to-active conversion produce competitive models, covering over 75% of the gap from baseline to BAYES. Used together, our rules bring the score to 85.2 F1, surpassing BAYES by 2.2 points. Table 6.5 breaks down these improvements by measuring B^3 performance on relevant roles.

For the lexical rules, we find that the negation and modal rules nearly completely capture their roles, with the discourse rule providing significant improvements as well. In contrast, previous models have struggled with these roles, as reported by Lang and Lapata (2011, Table 4, `NEG` and `DIS`

Model	**PU**	**CO**	**F1**	**ΔF1**
Gold Syntax				
SyntF	81.6	77.8	79.6	0.0
+ lex	85.2	79.8	82.4	+2.8
+ pass→act	83.6	80.8	82.2	+2.6
+ all rules	87.3	**83.1**	**85.2**	**+5.6**
Bayes (SotA)	**88.7**	78.1	83.0	+3.4
AsymDep	85.6	78.3	81.8	+2.2
Automatic Syntax				
Bayes	86.2	72.7	78.8	-0.8
SymDep (SotA)	81.9	**76.6**	**79.2**	-0.4
Automatic QA-SRL				
HUM-QQ	80.9	83.4	82.1	+2.5
- conn. penalty	79.0	82.7	80.8	+1.2
+ lex	**85.4**	**88.8**	**87.1**	**+7.5**

Table 6.3: Main results. The addition of a few simple rules to the SyntF baseline puts it significantly above existing approaches, and incorporating these rules into our QA-SRL-based model pushes performance even further, despite not using gold syntax at all. Evaluation numbers for baselines besides SyntF are drawn directly from prior work.

B^3 F1	A0	A1	A2	A3	A4	Args	TMP	ADV	MNR	LOC	PNC	CAU	Mods	All
SyntF + lex	78	71	63	**55**	**67**	73	**87**	**51**	**60**	**81**	**65**	**67**	**74**	74
HUM-QQ + lex	**90**	**87**	**69**	54	65	**85**	78	39	50	55	56	36	61	**82**
% Err \ Freq	.26	.37	.09	.01	.01	.74	.07	.04	.03	.03	.01	.01	.18	1.0
SyntF + lex	.23	.41	.13	.03	.02	.79	.04	.06	.04	.02	.01	.01	.18	1.0
HUM-QQ + lex	.15	.26	.14	.04	.02	.61	.08	.12	.07	.06	.02	.02	.38	1.0

Table 6.4: B^3 F1 scores on the training set for the most common labels, excluding `NEG`, `MOD`, and `DIS`.

roles). However, this is better seen as a shortcoming of the evaluation than the models: these roles are relatively uninteresting from the perspective of semantic role induction, as they are closed-class, not specific to particular predicates, and don't correspond to a semantic argument or modifier of the event denoted by the predicate. It might have been reasonable to exclude these arguments from the task at the outset, but instead, using our rules can mostly account for them while maintaining some comparability to prior work.

The passive-to-active conversion also produces a sizable gain, particularly on the core roles A0 and A1 (Table 6.5). Titov and Klementiev (2012) informally note that the BAYES model learns some syntactic alternations; of these, the passive alternation is perhaps the most impactful as it can apply to any transitive verb. What we've found is that a simple rule accounting for the passive construction in the syntax provided to the BAYES model can account for a large majority of its gains.

These results provide extra context in which to interpret the existing literature on semantic role induction. The fact that our simple auxiliary rules bring the syntactic baseline beyond the existing state of the art raises questions about whether the performance differences between previously published models are due to their relative abilities in capturing their intended phenomena — such as selectional restrictions and distributions over argument heads (Lang and Lapata, 2014) — or capturing these rules. It is not clear how much of the 5.2 F1 gain over SYNTF from our auxiliary rules is redundant with previous models. It seems likely that applying our rules to them would produce a result competitive with HUM-QQ, but it would still rely on gold syntax. Our focus is the utility of QA-SRL as features; indeed, it is also conceivable that applying a hierarchical model like BAYES to QA-SRL features would bring further improvements as well.

6.5.2 Superiority Without Syntax

HUM-QQ benefits disproportionately from the lexical rules, with a 5 F1 gain as opposed to the 2.8 F1 gain for SYNTF. This is because PropBank's NEG, MOD, and DIS arguments almost never occur in QA-SRL, so they get nonsense questions from the model (see Section 6.8.10, Table 6.12).[9] However,

[9] In practice, when using arguments predicted by a QA-SRL span detector (FitzGerald et al., 2018), we can remove the lexical rules entirely since the corresponding arguments will not be present.

	B^3 F1 Score				
Model	NEG	MOD	DIS	A0	A1
SyntF	41	45	50	78	71
+ all rules	**98**	**98**	**80**	**83**	**78**
Frequency	.01	.04	.03	.26	.37

Table 6.5: Breakdown of B^3 F1 scores on the training set for the labels most relevant to our auxiliary rules. The lexical rules capture AM-NEG, AM-MOD, and AM-DIS very well, and the active/passive rule significantly improves performance on A0 and A1, which are by far the most frequent role labels in the data. A rule-by-rule performance breakdown is provided in Section 6.8.4.

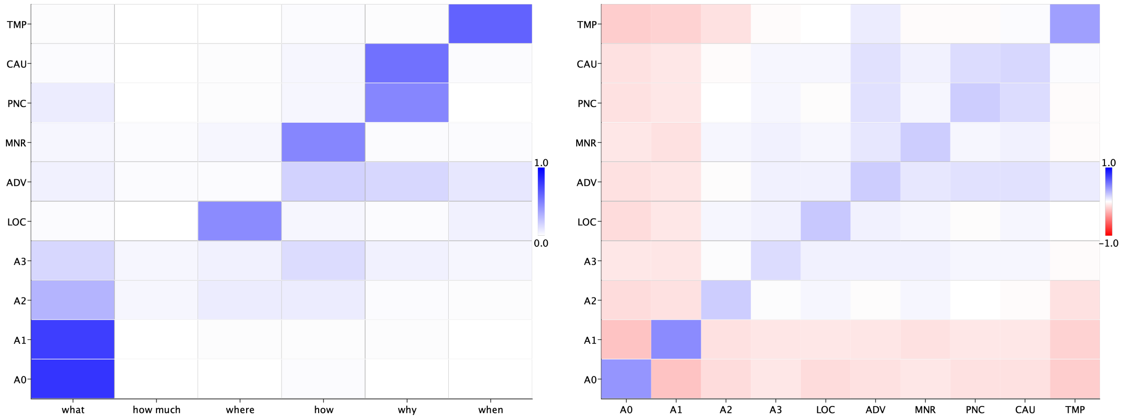
even the baseline model with no lexical rules or connectivity penalty surpasses the performance of the baselines using automatic syntax, all of which fall short of SyntF on gold.[10] With these additions, HUM-QQ sets a new state of the art beyond our enhanced SyntF baseline, with 87.1 F1.

Table 6.4 compares our model to SyntF + lex on the most common roles using B^3. HUM-QQ greatly improves over SyntF on core arguments (73→85 F1), but performs worse on modifiers (74→61). Since core arguments make up 74% of arguments in the corpus, HUM-QQ brings a large improvement overall (74→82) and core arguments still account for a majority of its error (at 61%).

SyntF's high performance on modifiers can be traced back to representational choices in the CoNLL 2008 Shared Task syntax (Surdeanu et al., 2008), which uses several dependency types that are semantic in nature, such as TMP, LOC, MNR, and DIR, among others. These often correlate well with gold modifier role labels, especially TMP (87 F1) and LOC (81 F1).[11] This fact has led some prior work, *e.g.*, Titov and Klementiev (2012), to use these dependency labels as clusters directly, so as to avoid the need to model modifier roles and instead focus on core arguments. Since we eschew syntactic features, we are forced to recover PropBank modifier roles from the ground up, making the task more difficult (explored more in Section 6.6.2).

[10] To be fair, these models use the automatic parses provided with the CoNLL 2008 data, which were produced by MaltParser (Nivre et al., 2006) at the time. Using state-of-the-art methods to predict the parses today would almost certainly improve the semantic role induction results, but probably not past gold parses.

[11] See Lang and Lapata (2014, Table 2) for a detailed contingency table.



(a) Distribution of *wh*-words for each role.

(b) Normalized PMIs between gold roles.

Figure 6.1: Cooccurrence between gold role labels and *wh*-words in QA-SRL (left) or each other in HUM-QQ's predicted clusters (right). The distributions of *wh*-words are normalized per role, and NPMI between gold labels is chance-corrected, where negative values (red) are clustered apart more often than by chance, and positive values (blue) are preferentially grouped together.

6.6 *What does QA-SRL Encode About Semantic Roles?*

Semantic roles are traditionally characterized as abstractions over syntactic arguments and modifiers (Gruber, 1965; Fillmore, 1968). Despite their deep entanglement with syntax, we have found that significant improvements in semantic role induction are possible without explicit syntactic analysis of the sentence, instead leveraging distributions of QA-SRL questions for each argument. In this section, we show that this is because QA-SRL questions provide *surrogate syntax*, recapitulating the aspects of syntax that are important for semantic roles (Section 6.6.1). Where QA-SRL questions fail to capture aspects of PropBank semantic roles, this arises in part from ontological differences with PropBank on modifiers (Section 6.6.2) and limitations of our experimental setup ignoring predicate sense (Section 6.6.3).

6.6.1 Surrogate Syntax

HUM-QQ brings the largest improvement over SYNTF on core arguments `A0` and `A1`. To investigate this, we identify the verbs which saw the greatest increase in B^3 F1 score on each role individually.

What we find is that QA-SRL works by acting as *surrogate syntax*: it removes much of the (role-irrelevant) syntactic variation in the source text, while still exhibiting (role-relevant) syntactic alternations which capture the syntactic behavior of the predicate verb.

Reducing Syntactic Variation For A0, the three verbs with the greatest improvement from SYNTF to HUM-QQ are *compete*, *conduct*, and *connect*, all with gaps of over 40 F1.[12] For each of these, their A0 arguments have a wide range of syntactic functions assigned by SYNTF, with SBJ less than 50% of the time — despite the fact that where the A0 role is present, it is designed to correspond to the grammatical subject (Babko-Malaya, 2005). We found that this is because these verbs frequently have *non-overt* subjects, which are not direct syntactic dependents of the predicate in CoNLL 2008 syntax (74% of a random sample of 30 sentences with A0 arguments of these three verbs, 10 from each; see Section 6.8.8). They appear in phrases like 'two competing *objectives*' (with adjectival clauses), 'urging *directors* to conduct a fair auction' (with control verbs), or '*a maze of halls* that connects film rooms' (with relative clauses). In these cases, the SYNTF baseline does poorly, as the correspondence between the SBJ dependency and A0 role only holds consistently for overt subjects.

In contrast, HUM-QQ assigns the vast majority of A0 arguments in these cases with questions that put the *wh*-word in subject position, *e.g.*, *What competes with something?* or *What conducts?* Here, QA-SRL removes much of the syntactic variation from the source text and recovers something close to the underlying grammatical relation between the argument and the verb, while also providing information about the verb's subcategorization frames (*e.g.*, the presence of an object in *What connects something?*), aiding in recovery of the semantic role.

Capturing Syntactic Alternations For A1, The verbs with the greatest improvement are *propose*, *prefer*, *price*, and *relate*, with a gap of >50 F1 between models. Of the top 50 such verbs, 48 are transitive with A1 as the transitive object (see Section 6.8.8). In these cases, the passive alternation allows the argument to be asked about in either the subject (*What is proposed?*) or object (*What*

[12] To reduce variance from low-frequency verbs, we measure this gap after smoothing their precision and recall with 10 counts of the weighted aggregate for the model.

does something propose?) position. We find that QA-SRL does this, frequently combining questions about passive subject and active object into one role: for 62% of the top 50 verbs, the cluster corresponding to A1 gives greater than 20% probability *each* to passive subject and active object questions. This happens because the Hard Unigram Mixture objective clusters together distributions whose uncertainty is spread over the same set of elements, which here correspond to syntactic alternations. As an example, Table 6.2 shows the induced clusters for *give*, which exhibit both passive and dative alternations; *give* gained 31 F1 on A1 in HUM-QQ.

6.6.2 Mismatched Modifiers

HUM-QQ struggles to identify PropBank modifier roles, and it has room for improvement on trailing arguments like A2 and A3. In QA-SRL, the semantics of these roles are primarily expressed by the initial *wh*-word, such as *when*, *where*, *why*, *how*, etc. Figure 6.1a shows the distribution of *wh*-words appearing for each role in the training set. To a large extent, each role is concentrated on a corresponding *wh*-word, but there are exceptions. A2, A3, and AM-ADV are widely spread between *wh*-words, and *how* and *why* account for a significant portion of questions for several roles each. See Section 6.8.10, Table 6.11 for full questions.

To visualize how this affects clustering results, Figure 6.1b shows the normalized pointwise mutual information (NPMI; Bouma, 2009) between gold labels in HUM-QQ's predicted clusters (see Section 6.8.9 for how this is calculated). While A0 and A1 are distinguished well from all other roles, the trailing arguments A2 and A3 are not well distinguished from modifiers, reflecting the difficulty of the argument–adjunct distinction for these arguments, which often have similar meanings to modifiers and form a significant error case for supervised labelers (He et al., 2017a). AM-ADV tends to be confused with other modifier roles, which reflects its definition in the PropBank guidelines as a sort of "catch-all" role for meanings not captured in the other modifiers (Babko-Malaya, 2005). Finally, AM-CAU (*cause*) and AM-PNC (*purpose, not cause*) tend to be confused with each other, since they both elicit *why* questions.

B^3 F1	A0	A1	A2	A3	A4	Args	TMP	ADV	MNR	LOC	PNC	CAU	Mods	All
SyntF + lex	78	71	63	55	67	73	**87**	**51**	60	**81**	65	**67**	**74**	74
HUM-QQ + lex	90	87	69	54	65	85	78	39	50	55	56	36	61	82
+ gold arg/adj	91	89	78	65	78	88	77	47	57	61	58	35	64	85
+ gold sense	91	90	74	58	75	87	80	43	55	64	63	47	65	84
+ both	**92**	**92**	**83**	**70**	**81**	**90**	81	**51**	**64**	69	**66**	46	69	**87**

Table 6.6: Breakdown of B^3 F1 scores on the training set for the most common labels in our ablation studies. The first two rows are repeated from Table 6.4.

Argument–Adjunct Distinction Scores are significantly lower for trailing core arguments A2–4 than for A0 and A1. Since part of the problem seems to be confusion with modifier roles (Figure 6.1b), we conduct an oracle experiment to enforce the argument–adjunct distinction by doubling the size of the feature space to $\phi(x) \in \mathbb{R}^{2|q|}_{\geq 0}$ and projecting gold core arguments and modifiers into orthogonal subspaces.

Results are shown in Table 6.6 (+ gold arg/adj). The oracle boosts performance by 3 points, with particular focus on trailing arguments A2 (69→78) and A4 (65→78), as well as modifiers AM-ADV (39→47), AM-MNR (50→57), and AM-LOC (55→61). However, overall performance on modifiers is still far below the syntactic baseline. Given the coarse semantics of English *wh*-words in comparison to PropBank modifier roles (Figure 6.1a), it may be that finer-grained features are necessary to significantly increase performance on modifiers.

6.6.3 Scrambled Senses

Despite core arguments significantly improving under HUM-QQ, they remain the largest source of error. To investigate this, we examine the verbs with the worst F1 on core arguments. The top verbs are *go*, *settle*, *confuse*, *turn*, and *follow*, with <60 F1. Half of the top 20 have 4 or more predicate senses annotated in PropBank, where different senses often manifest their roles differently: for example, the subject is A0 when *settling with the IRS* (sense 2), but A1 when *settling into a new job* (sense 3). To quantify this, we run an oracle experiment where we induce roles for each verb sense separately instead of each verb lemma. Results are shown in Table 6.6 (+ gold sense). Performance improves particularly on trailing arguments A2, A3 and A4, which tend to differ greatly in meaning and realization for different predicate senses. A combined oracle (+ both) shows that the gains are mostly complementary with those from the argument/adjunct distinction oracle. These results suggest that future work on semantic role induction should prioritize modeling predicate senses.

6.7 Conclusion

We have shown that QA-SRL provides a way to do state-of-the-art semantic role induction without the need for formal syntax. It works by providing *surrogate* syntax: it captures long-distance dependencies to non-overt arguments and exhibits syntactic alternations which allow us to detect varied ways of expressing the same role. These results suggest that QA-SRL can provide some of the practical benefits of sophisticated syntactic formalisms that have separate layers of functional structure, like Combinatory Categorial Grammar (Steedman, 1996, 2000), Head-Driven Phrase Structure Grammar (Pollard and Sag, 1994), or Lexical Functional Grammar (Bresnan et al., 2015) — but without grammar engineering or expert data annotation.

One challenge is that QA-SRL is currently only defined for English. Future work may benefit from our lessons about the utility of surrogate syntax when designing similar annotation methodologies for other languages; combining this with insights from existing work on grammar development for diverse languages (Bender et al., 2002) may be key.

While formal ontologies of semantic roles and syntax are difficult to formulate and scale, our results show how it may be comparatively feasible to formulate, scale, and build robust models for the *phenomena* that such ontologies are meant to explain. QA-SRL exhibits enough of these phenomena that a relatively simple model over it (the Hard Unigram Mixture in Section 6.3) yields state-of-the-art induced semantic roles which are interpretable and linguistically meaningful. This suggests that identifying and gathering supervision for more phenomena (e.g., those related to word sense or modifier semantics) in a relatively theory-agnostic way, then building models grounded in linguistic theory, may be a promising avenue for future work. This general approach has recently been applied to syntax as well, for example leveraging constituency tests (Cao et al., 2020) and naturally-occurring bracketings (Shi et al., 2021).

The fact that discrete structures can be reliably derived from ontology-free annotation schemes like QA-SRL can potentially inform future efforts to construct large-scale ontologies of semantic structure. QA-SRL has the further benefit over traditional SRL of including a broader scope of *implicit* arguments than those addressed by supervised systems, as shown by Roit et al. (2020). Taken together, our results suggest that with the right kind of annotation scheme, it should be possible to construct rich semantic ontologies in new domains, without expert curation and in a data-driven, linguistically motivated way.

6.8 Notes

6.8.1 QA-SRL Question Generator

We reproduce FitzGerald et al. (2018)'s architecture, encoding sentences with a stacked alternating LSTM (Zhou and Xu, 2015b) with highway connections (Srivastava et al., 2015) and recurrent dropout (Gal and Ghahramani, 2016), and representing spans by concatenating the output embeddings of their endpoints (Lee et al., 2016). The question generator is a specialized LSTM decoder which only outputs the tokens allowed in each QA-SRL slot. The current predicate is indicated by an embedded binary feature input to BiLSTM encoder, and answer span representations are input at each step of the LSTM decoder. We make two changes from FitzGerald et al. (2018): 1) As opposed to GloVe (Pennington et al., 2014b) or ELMo (Peters et al., 2018), We embed the inputs with BERT-base (Devlin et al., 2019) in the 'feature' style with a learned scalar mix over layers, and 2) we additionally concatenate the output embedding of the predicate to the input of the LSTM decoder.

6.8.2 Hyperparameters

QA-SRL Question Generator The BiLSTM encoder uses a hidden size of 300, 4 layers, 0.1 recurrent dropout probability, and a 100-dimensional predicate indicator embedding. The LSTM decoder has a 100-dimensional hidden state and predicts QA-SRL slots with 200-dimensional embeddings via an MLP with a 100-dimensional hidden layer. We train on all QA pairs in the

Model	PU	CO	F1	ΔF1
SyntF	81.6	77.8	79.6	0.0
+ negation	82.8	77.8	80.2	+0.6
+ modals	83.0	79.8	81.3	+1.7
+ discourse	82.6	77.8	80.1	+0.5
+ pass→act	83.6	80.8	82.2	+2.6
+ all rules	87.3	83.1	85.2	+5.6

Table 6.7: Detailed results for auxiliary rules on SyntF.

QA-SRL Bank 2.0 expanded training set using BERT's variant of Adam (Kingma and Ba, 2015) with a learning rate of $5\text{e}{-5}$ and batch size of 32, selecting the model with minimal perplexity on the expanded development set. To produce our feature vectors ϕ, we decode questions with a beam size of 20 and a minimum probability cutoff of 0.01.

Flat Pre-Clustering We perform flat clustering with 100 clusters, skipping this step for verbs with 100 arguments or less. We use a concentration parameter of $\alpha = 0.01$ (i.e., uniform base measure with a sum of 0.01) and do 5 random restarts, each running until the loss decreases by less than $1\text{e}{-5}$ per iteration, and choose the run that yields the lowest loss.

Tuned Splitting Our final model (HUM-QQ + lex) uses $\lambda = 0.35$.

6.8.3 Rule Lexica

Here we list the full lexica for the auxiliary clustering rules described in Section 6.4.2.

Negation 5 items: *n't*, *never*, *no*, *no longer*, *not*.

These are drawn directly from the PropBank guidelines (Babko-Malaya, 2005, p. 32).

Modals 23 items: *'d*, *'ll*, *'ve*, *able*, *ca*, *can*, *can't*, *could*, *going*, *gon*, *gonna*, *have*, *may*, *might*, *must*, *ought*, *shall*, *should*, *used*, *will*, *wo*, *won't*, *would*.

Note the inclusion of *have*, *used*, *able*, and *going*, which are parts of phrasal modals (e.g., *have to*), which are included in AM-MOD according to the PropBank guidelines (Babko-Malaya, 2005, p. 32).

Discourse 55 items: *after all*, *ah*, *also*, *and*, *and so*, *as a result*, *as we've seen before*, *as well*, *but*, *certainly*, *damn*, *either*, *for example*, *for instance*, *for one*, *for one thing*, *frankly*, *furthermore*, *gosh*, *hence*, *however*, *in addition*, *in any case*, *in any event*, *in contrast*, *in fact*, *in other words*, *in particular*, *in that case*, *in this case*, *in turn*, *indeed*, *instead*, *ironically*, *moreover*, *nonetheless*, *of course*, *oh gosh*, *oh my god*, *oh my gosh*, *on the other hand*, *or*, *particularly*, *rather*, *regardless*, *similarly*, *so*, *specifically*, *thereby*, *therefore*, *though*, *thus*, *too*, *uh*, *um*.

Note the inclusion of some interjections, (*ah*, *oh my gosh*, etc.), which are included in AM-DIS according to the PropBank guidelines (Babko-Malaya, 2005, p. 31).

6.8.4 Auxiliary Rule Performance Breakdown

In Table 6.7, we provide a more detailed accounting of the improvements that arise from our auxiliary rules described in Section 6.4.2 and Table 6.5. The negation and discourse rules bring precision improvements, likely because they mostly have ADV dependencies outgoing. The modal rule improves both precision and recall because modals have many different kinds of outgoing dependencies, due to their status as heads of clauses (which can serve in many syntactic capacities). Finally, the passive alternation rule aids precision by splitting SBJ between active and passive uses, and aids recall by grouping LGS with the active SBJ and passive SBJ with active OBJ. This mainly affects the core argument labels A0 and A1, as shown in Table 6.5 — especially A1, as we also find for QA-SRL questions in Section 6.6.1.

6.8.5 Tuned Splitting Evaluation

Our model has a single parameter λ which determines the number of clusters for each verb via the tradeoff between the data likelihood and clustering likelihood. We compare this to a constant baseline (the same number of clusters for all verbs) and an oracle upper bound which chooses

Tuning Method	**PU**	**CO**	**F1**
Constant $k = 6$	83.9	86.7	85.3
$\lambda = 0.35$	85.4	88.8	87.1
F1 Oracle	**87.6**	**89.6**	**88.6**

Table 6.8: Comparison of methods to set the number of clusters for each verb, reported for HUM-QQ+ lex.

Setting	**Objective**
$\lambda = 1$	Mixture of Unigrams Likelihood
$\lambda = 0$	Jensen-Shannon Divergence
$\lambda = -1$	Mutual Information

Table 6.9: Objectives reproduced by the HUM loss for different settings of λ, described in Section 6.8.7.

Model	**B^{3}P**	**B^{3}R**	**F1**	**ΔF1**
Gold Syntax				
SYNTF	74.7	68.3	71.3	0.0
+ lex	79.1	70.4	74.5	+3.2
+ pass→act	77.4	72.1	74.7	+3.4
+ all rules	**82.2**	**74.7**	**78.3**	**+7.0**
Automatic QA-SRL				
HUM-QQ	71.1	79.0	74.8	+3.5
- conn. pen.	71.6	75.7	73.6	+2.3
+ lex	**79.8**	**83.4**	**81.6**	**+10.3**
+ lex + MI	77.7	82.1	79.9	+8.6

Table 6.10: B^3 Results on models we tested. The gap between HUM-QQ and SYNTF is larger than for purity and collocation, as B^3 is a tougher metric which is more discriminative between clusterings. The last model variant (+MI) is described in Section 6.8.7.

the split that maximizes the purity/collocation F1 score for each verb independently. As shown in Table 6.8, we improve on the constant baseline by 1.8 points (85.3→87.1), but fall short of the oracle by 1.5 points (87.1→88.6). There is room for improvement, but errors in the tuning step may not be the most significant factor to concern future work.

6.8.6 B^3 *Results*

Results using B^3 metrics on models we tested are shown in Table 6.10.

6.8.7 Related Clustering Algorithms

Recall the Hard Unigram Mixture loss

$$\mathcal{L}_{\lambda}^{\mathrm{HUM}}(\mathbf{C}) = -\log \mathrm{P}(\mathbf{X} \mid \mathbf{C}) - \lambda \log \mathrm{P}(\mathbf{C}).$$

Different settings of λ reproduce several objectives present in the literature, summarized in Table 6.9. As written in Section 6.3, when $\lambda = 1$, minimizing $\mathcal{L}_{1}^{\mathrm{HUM}}$ maximizes likelihood of the data $\mathbf{X}$ under a mixture of unigrams model (Nigam et al., 2000).

When the number of clusters k is fixed, setting $\lambda = 0$ as in our greedy merging step (Section 6.3.3) is equivalent to enforcing a uniform prior π over mixture components. In this case, the gain in loss on each merge is the Jensen-Shannon Divergence (JSD) between the merged clusters, scaled by their total size and using each cluster's size to determine its mixing weights in the divergence, as in the mixture-based definition of JSD by Lin (1991). JSD is used in the same way by Chrupała (2012), without the scaling and weighting, as a similarity measure for agglomerative clustering.

Finally, setting $\lambda = -1$ reduces the HUM loss to the mutual information between the QA-SRL questions under ϕ and the cluster assignment C, which has been used in prior work to encourage informative clusterings (Michael et al., 2020). This is related to the *distributional clustering* paradigm of Pereira et al. (1993), which aims to identify common factors that explain distributional data, and which Slonim and Tishby (1999) frame in terms of an information bottleneck that maximizes mutual information between the data and a jointly distributed 'relevance' variable (though in our case, the reference variable is the cluster assignment itself). Setting $\lambda = -1$ in the greedy merging step, we find (in Table 6.10) that using a mutual information criterion in this way hurts performance. We guess this is because the objective incentivizes clusters of uniform size, which does not match the highly skewed distributions of gold semantic roles.

6.8.8 Manual Analysis Results

Improved Verbs on `A0` The top 50 verbs by F1 gain on `A0` from SYNTF to HUM-QQ are: *compete*, *conduct*, *connect*, *combine*, *dominate*, *restore*, *require*, *yield*, *limit*, *ban*, *direct*, *tie*, *oversee*, *contain*, *identify*, *increase*, *evaluate*, *specialize*, *allow*, *assist*, *restrict*, *found*, *grant*, *feature*, *propose*, *detail*, *force*, *convert*, *veto*, *rate*, *bolster*, *appoint*, *enact*, *design*, *list*, *lead*, *resolve*, *retire*, *schedule*, *reach*, *analyze*, *remove*, *speed*, *manage*, *deliver*, *underlie*, *revise*, *emerge*, *enable*, *block*.

We examined 30 sentences containing the top 3 verbs (*compete*, *conduct*, and *connect*). There were 31 A0 arguments of these verbs in these sentences. Of these, 8 (26%) were overt, 11 (35%) were extracted subjects of relative clauses, 5 (16%) were modified by the predicate appearing in an adjectival clause, 5 (16%) were subjects of open complements of control verbs, and 2 (6%) were otherwise implicit (subject of an adverbial clause or open complement not under a control verb).

Improved Verbs on A1 We examined the top 50 verbs by their difference in B^3 performance on A1 between SYNTF and HUM-QQ. 48 of them are transitive; the other two are bolded. In decreasing order of F1 gain, they are: *propose*, *prefer*, *price*, *relate*, *involve*, *help*, *choose*, *consider*, *design*, *mention*, *identify*, *release*, *include*, ***exist***, ***range***, *value*, *revise*, *lead*, *associate*, *need*, *increase*, *import*, *prove*, *feel*, *place*, *determine*, *limit*, *found*, *enact*, *control*, *cancel*, *dilute*, *disclose*, *select*, *exclude*, *force*, *insure*, *accrue*, *damage*, *calculate*, *hurt*, *secure*, *delay*, *regard*, *record*, *open*, *use*, *concern*, *weaken*, *adjust*.

6.8.9 Calculating Normalized PMI

Here we describe some special concerns for our use of normalized PMI in Section 6.6.2.

Pointwise mutual information (PMI) is a measure of how likely two items (such as tokens in a corpus) are to occur together relative to chance (Church and Hanks, 1989). One feature of PMI is that it tends to be larger for rare events: if two items x and y always occur together, then their PMI is $-\log \mathrm{P}(x, y)$. This can make it difficult to assess association patterns among items with greatly varying probabilities (*e.g.*, the AM-CAU role appears for 1% of arguments, while A1 appears for 27%). So we use *normalized* PMI (NPMI; Bouma, 2009), which factors out the effect of item frequency on PMI. Formally, the NPMI of x and y is

$$\left(\log \frac{\mathrm{P}(x,y)}{\mathrm{P}(x)\,\mathrm{P}(y)}\right) \Big/ -\log(\mathrm{P}(x,y)) \,, \tag{6.1}$$

taking the limit value of -1 when they never occur together, 1 when they only occur together, and 0 when they occur independently. We use NPMI to analyze the co-occurrence of *gold labels* in *predicted clusters*: A pair of gold labels with high NPMI are preferentially grouped together by the induced roleset, whereas two labels with low NPMI are preferentially distinguished. The joint distribution between gold labels is generated by drawing one point (x) uniformly at random from the data, drawing another (y) uniformly at random from x's predicted cluster, and reading the gold labels of both. NPMI has been used to analyze clusters in this way by Michael et al. (2020).

Calculating NPMI naïvely on our full clustering has a caveat. The denominator of the PMI term in Equation 6.1, $\mathrm{P}(x)\,\mathrm{P}(y)$, uses marginal probabilities of x and y over the corpus to calculate chance co-occurrence. But our clusters are constrained not to overlap between verbs, so this does not correctly estimate chance cooccurrence in our setting. Instead, we use the expectation over verbs of within-verb chance cooccurrence:

$$\sum_{v} \mathrm{P}(x \mid v)\,\mathrm{P}(y \mid v)\,\mathrm{P}(v),$$

where $\mathrm{P}(v)$ is proportional to the number of arguments for the verb v.

6.8.10 Question Distributions by Role

We list the top questions and their probabilities for modifier roles in Table 6.11. Questions for core roles and the ones covered by our lexical rules are in Table 6.12. We use *verb* (or *verbs*, or *verbed*) as a placeholder for the verb, which in practice is replaced with the predicate for a given instance.

Role	Top Questions	Prob
TMP	When does something verb something?	0.34
	When does something verb?	0.21
	When is something verbed?	0.18
	When does something verb somewhere?	0.03
	When does sth. verb to do something?	0.02
	How does something verb?	0.01
	How is something verbed?	0.01
ADV	Why does something verb something?	0.13
	How does something verb something?	0.12
	When does something verb something?	0.09
	How is something verbed?	0.08
	How does something verb?	0.08
	Why does something verb?	0.05
	Why is something verbed?	0.04
	When does something verb?	0.04
	What does something verb?	0.03
	When is something verbed?	0.03
MNR	How is something verbed?	0.25
	How does something verb?	0.22
	How does something verb something?	0.19
	What does something verb?	0.02
	Where does something verb?	0.02
	Why does something verb something?	0.02
	How does something verb somewhere?	0.02
LOC	Where does something verb something?	0.24
	Where is something verbed?	0.22
	Where does something verb?	0.21
	When does something verb something?	0.04
	How does something verb something?	0.03
	How does something verb?	0.02
	How is something verbed?	0.02
PNC	Why does something verb something?	0.29
	Why is something verbed?	0.21
	Why does something verb?	0.08
	Why does something verb somewhere?	0.05
	What is something verbed to do?	0.03
	How is something verbed?	0.03
	What is something verbed for?	0.02
CAU	Why does something verb something?	0.32
	Why does something verb?	0.16
	Why is something verbed?	0.16
	Why does something verb somewhere?	0.04
	How does something verb?	0.04
	Why does sth. verb to do something?	0.03
DIR	Where does something verb?	0.40
	How does something verb?	0.17
	Where is something verbed?	0.10
	Where does something verb something?	0.07
	How is something verbed?	0.06
	How does something verb something?	0.03

Table 6.11: Top questions for modifier roles. Most roles align well with a *wh*-word, especially *when*, *where*, or *why*. AM-ADV takes many *wh*-words, and *how* appears often for nearly all modifier roles.

Role	Top Questions	Prob
A0	What verbs something?	.65
	What verbs?	.14
	How is something verbed?	.02
A1	What does something verb?	.42
	What is verbed?	.25
	What verbs?	.09
	What verbs something?	.03
	What does something verb to do?	.02
A2	What does something verb?	.12
	How is something verbed?	.07
	What verbs something?	.07
	Where is something verbed?	.06
	What is verbed?	.06
	How does something verb?	.06
	How much does something verb?	.04
A3	How does something verb?	.15
	What does something verb?	.09
	How is something verbed?	.07
	Why does something verb something?	.06
	How does something verb something?	.05
	When does something verb?	.05
	Where does something verb?	.04
A4	What does something verb to?	.17
	Where does something verb?	.17
	How does something verb?	.16
	How much does something verb?	.14
	What does something verb something to?	.04
	How is something verbed?	.03
NEG	What verbs something?	.40
	What verbs?	.15
	What is verbed?	.12
	How is something verbed?	.05
	How does something verb?	.03
	How does something verb something?	.03
MOD	What verbs something?	.22
	How does something verb something?	.11
	What verbs?	.09
	Why does something verb something?	.07
	What is verbed?	.06
	How does something verb?	.06
	How is something verbed?	.06
DIS	When does something verb something?	.15
	How does something verb something?	.15
	What verbs something?	.08
	How is something verbed?	.07
	How does something verb?	.07
	Why does something verb something?	.06
	When does something verb?	.05

Table 6.12: Top questions for core roles and lexical rules. The questions for AM-NEG, AM-MOD, and AM-DIS don't make sense, while core roles are as expected: A0 is mostly subjects; A1 mixes subjects and objects, with some complements; A2 and on are more varied.

Part IV

CONCLUDING THOUGHTS

In this document I proposed the development of *data-driven theory* as a paradigm for scientific progress in NLP. Beginning from the perspective of the philosophy of language, I propose *theories* as essential to creating common understanding between humans and machines, allowing for the construction of language understanding systems and the employment of AI systems in service of humans' goals (Chapter 2). To make it feasible to construct theories of complex language behaviors, I propose a framework for *scalable theories* based in Pragmatist epistemology (Chapter 3). In this framework, developing a scalable theory involves:

- Collecting carefully-scoped data in a way that directly represents a phenomenon of interest while imposing minimal prior theoretical assumptions,
- Potentially increasing the scale and coverage of that data using a learned black-box model as a data simulator,
- Inducing simple, comprehensible models of this high-coverage data using machine learning, and
- Examining the results to debug the theory and data, iterating to improve the quality and coverage of both while also potentially improving our scientific understanding of the phenomenon of interest.

As a case study and proving ground for scalable theory, I present a series of projects involving the annotation of question-answer pairs as representations of linguistic structure and meaning (Chapters 4 and 5) — most notably QA-SRL, which I find to strike a balance of four proposed **Principles**

of Data for Scientific NLP (Part II): 1) theoretical minimalism, 2) broad comprehensibility, 3) annotation constraints, and 4) narrow scope. Using QA-SRL, I show how to leverage black-box data simulation together with simple probabilistic modeling to automatically induce an ontology of semantic roles which is directly and comprehensibly grounded in phenomena that the theory of semantic roles is meant to explain. This not only lays the groundwork for new scalable theoretical developments in semantic representation, but can serve as an example to guide future work on scalable theories in other domains.

Why now?

The justification for building scalable, data-driven theories can be summarized as follows (given in longer form in Chapters 2 and 3):

1. To build systems which generalize in controllable, predictable ways and can usefully inform us of how to act in the world, we need comprehensible theories of their desired behavior.

2. However, the behaviors we wish to produce in AI and NLP are too complex for us to easily write down theories of how they should work.

3. So instead, we must use machines (i.e., statistical models) to construct our theories on the basis of data in a scalable way. The role for the scientist here is twofold:

    - to carefully determine the scope of the phenomena to be explained and curate the data accordingly, and
    - to define the meta-theory which relates the learned theory to the data.

This argument could have been made at any point in the history of NLP, so it is worth discussing why it has not been articulated in this way before, and why it is particularly worth articulating now.

First, as a caveat: the essence of the argument *has* been made before, even if not as explicitly or using the same language. Approaches similar to my proposal are used in grammar engineering (Oepen et al., 2004; Flickinger et al., 2017) and the Decompositional Semantics Initiative (White et al., 2016). In linguistic typology, Haspelmath (2010) argues for *framework-free grammatical theory*, making similar points about the relationship between data and theory. There are some differences, however, between my approach and that of these authors, since I am focused on applications in NLP where the vastness and complexity of the domain becomes more of a challenge (Sutton, 2019). The relationship between my approach and theirs is discussed in more depth in Chapter 3.

Returning to the question: Why is the argument for scalable, data-driven theory particularly relevant now? My view is that it is because until the deep learning revolution, the field was in an *era of underfitting*. Pre-neural statistical models such as Conditional Random Fields (Lafferty et al., 2001) struggle even in-distribution on tasks like syntactic and semantic parsing, let alone more complex end-user tasks like question answering or language generation tasks. It may seem premature to argue about the generalization properties of NLP systems out-of-distribution when they are not even expressive enough to perform well in-distribution with lots of training data. In addition, the performance of these systems was weak enough that many were convinced that they would benefit from the continued development of hand-curated linguistic resources like PropBank (Palmer et al., 2005).

With deep learning, all of these factors changed: the limits of hand-curated resources like PropBank have been surpassed, and highly-expressive models can fit all kinds of data distributions, leaving us face-to-face with the problem of generalization and the need for data-driven theory. On top of this, we also have essential new tools available for data simulation: the semantic role induction algorithm in Chapter 6 would not have been workable without a neural model to simulate dense annotation of QA-SRL questions. So we are finally in a position to make such theories scalable.

Looking forward

Extending the basic paradigm of scalable theory to more facilities of language (*e.g.*, syntax or word sense) and more complex phenomena (*e.g.*, representations of world knowledge, common sense, reasoning and more) remains a major challenge. As the scope of the phenomena to be represented increases, greater annotation constraints will be necessary in order to ensure that the desired phenomena are adequately covered. However, doing so is challenging while maintaining theoretical minimalism. My hope is that scalable theories of simple, narrowly-scoped subproblems (*e.g.*, semantic roles) will provide annotation constraints that can make more complex tasks tractable to exhaustively annotate, without introducing the same problems as in the Rationalist paradigm where inconsistencies, underspecification, and arbitrary choices in the theory limit the usefulness of the data. In this way, it may be possible to bootstrap from narrowly-scoped theories into progressively broad accounts of language structure and meaning, and intelligent behavior more generally.

At this point, such talk is speculation. It is unclear how the paradigm of data-driven theory will generalize to more complex tasks. However, in this work I hope to have provided an argument this kind of work is at least worth attempting, and perhaps laid some groundwork and principles which can be used as a starting point for it to be done in the future.

BIBLIOGRAPHY

Omri Abend and Ari Rappoport. 2013a. Universal conceptual cognitive annotation (ucca). In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*.

Omri Abend and Ari Rappoport. 2013b. Universal Conceptual Cognitive Annotation (UCCA). In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 228–238, Sofia, Bulgaria. Association for Computational Linguistics.

Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. 2009. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information retrieval*, 12(4):461–486.

Hannah Youngeun An and Aaron Steven White. 2020. The lexical and grammatical sources of neg-raising inferences. In *Proceedings of the Society for Computation in Linguistics 2020*, pages 386–399, New York, New York. Association for Computational Linguistics.

Marc Andreessen. 2011. Why software is eating the world.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. VQA: Visual Question Answering. In *International Conference on Computer Vision (ICCV)*.

Olga Babko-Malaya. 2005. Propbank annotation guidelines. Retrieved 2022.

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 79–85, Montreal, Quebec, Canada. Association for Computational Linguistics.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998a. The Berkeley FrameNet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada. Association for Computational Linguistics.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998b. The Berkeley FrameNet project. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*, pages 86–90. Association for Computational Linguistics.

Dare A. Baldwin. 1995. Understanding the link between joint attention and language. In *Joint Attention: Its Origins and Role in Development*, pages 131–158. Lawrence Erlbaum Associates, Inc.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013a. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013b. Abstract meaning representation for sembanking. In *Proceedings of the Linguistic Annotation Workshop*.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013c. Abstract meaning representation for sembanking. In *7th Linguistic Annotation Workshop and Interoperability with Discourse*.

Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. 2012. Developing a large semantically annotated corpus. In *Proceedings of the 2012 International Conference on Language Resources and Evaluation*.

Emily M. Bender and Guy Emerson. 2021. Computational linguistics and grammar engineering. In Stefan Müller, Anne Abeillé, Robert D. Borsley, and Jean-Pierre Koenig, editors, *Head-Driven Phrase Structure Grammar: The handbook*, Empirically Oriented Theoretical Morphology and Syntax, pages 1101–1148. Language Science Press., Berlin.

Emily M. Bender, Dan Flickinger, and Stephan Oepen. 2002. The grammar matrix: An open-source starter-kit for the rapid development of cross-linguistically consistent broad-coverage precision grammars. In *COLING-02: Grammar Engineering and Evaluation*.

Emily M. Bender and Alexander Koller. 2020. 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. Association for Computational Linguistics.

Eric Berne. 1964. *Games People Play: The Basic Handbook of Transactional Analysis*. Ballantine Books, The Random House Publishing Group, New York.

Claire Bonial, Olga Babko-Malaya, Jinho D Choi, Jena Hwang, and Martha Palmer. 2010. Propbank annotation guidelines. *Center for Computational Language and Education Research, CU-Boulder*.

Gerlof Bouma. 2009. Normalized (pointwise) mutual information in collocation extraction. In *GSCL*.

Samuel R. Bowman and George Dahl. 2021. What will it take to fix benchmarking in natural language understanding? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4843–4855, Online. Association for Computational Linguistics.

George E. P. Box. 1976. Science and statistics. *Journal of the American Statistical Association*, 71(356):791–799.

Leo Breiman. 2001. Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical Science*, 16:199–231.

Joan Bresnan, Ash Asudeh, Ida Toivonen, and Stephen Wechsler. 2015. *Lexical-functional syntax*. John Wiley & Sons.

Rechele Brooks and Andrew N. Meltzoff. 2005. The development of gaze following and its relation to language. *Developmental science*, 8 6:535–43.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Avi Caciularu, Arman Cohan, Iz Beltagy, Matthew Peters, Arie Cattan, and Ido Dagan. 2021. CDLM: Cross-document language modeling. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2648–2662, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zheng Cai, Lifu Tu, and Kevin Gimpel. 2017. Pay attention to the ending:strong neural baselines for the ROC story cloze task. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 616–622, Vancouver, Canada. Association for Computational Linguistics.

Steven Cao, Nikita Kitaev, and Dan Klein. 2020. Unsupervised parsing via constituency tests. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4798–4808, Online. Association for Computational Linguistics.

David Carter. 1997. The TreeBanker: a tool for supervised training of parsed corpora. In *Computational Environments for Grammar Development and Linguistic Engineering*.

Jifan Chen, Eunsol Choi, and Greg Durrett. 2021. Can NLI models verify QA systems' predictions? In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3841–3854, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Eunsol Choi, Jennimaria Palomaki, Matthew Lamm, Tom Kwiatkowski, Dipanjan Das, and Michael Collins. 2021. Decontextualization: Making sentences stand-alone. *Transactions of the Association for Computational Linguistics*, 9:447–461.

Noam Chomsky. 1957. *Syntactic Structures*. Mouton & Co., The Hague.

Noam Chomsky. 1965. *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA.

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. 2022. Palm: Scaling language modeling with pathways. *CoRR*.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2011. An analysis of open information extraction based on semantic role labeling. In *K-CAP*.

Grzegorz Chrupała. 2012. Hierarchical clustering of word class distributions. In *Proceedings of* *the NAACL-HLT Workshop on the Induction of Linguistic Structure*, pages 100–104, Montréal, Canada. Association for Computational Linguistics.

Kenneth Church. 2007. A pendulum swung too far. *Linguistic Issues in Language Technology*, 2.

Kenneth Ward Church and Patrick Hanks. 1989. Word association norms, mutual information, and lexicography. In *27th Annual Meeting of the Association for Computational Linguistics*, volume 16, pages 76–83. Association for Computational Linguistics.

Guglielmo Cinque. 1999. *Adverbs and Functional Heads: A Crosslinguistic Perspective*. Oxford University Press, Oxford, UK.

Robin Clark. 2012. *Meaningful Games: Exploring Language with Game Theory*. MIT Press, Cambridge, MA.

Ronan Collobert and Jason Weston. 2007. Fast semantic extraction using a novel neural network architecture. In *ACL 2007*.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 160–167, New York, NY, USA. Association for Computing Machinery.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D. Hoffman, Farhad Hormozdiari, Neil Houlsby, Shaobo Hou, Ghassen Jerfel, Alan Karthikesalingam, Mario Lucic, Yian Ma, Cory McLean, Diana Mincu, Akinori Mitani, Andrea Montanari, Zachary Nado, Vivek Natarajan, Christopher Nielson, Thomas F. Osborne, Rajiv Raman, Kim Ramasamy, Rory Sayres, Jessica Schrouff, Martin Seneviratne, Shannon Sequeira, Harini Suresh, Victor Veitch, Max Vladymyrov, Xuezhi Wang, Kellie Webster, Steve Yadlowsky, Taedong Yun, Xiaohua Zhai, and D. Sculley. 2020. Underspecification presents challenges for credibility in modern machine learning.

Donald Davidson. 1967. The logical form of action sentences.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

Filipe de Sá Mesquita, Jordan Schmidek, and Denilson Barbosa. 2013. Effectiveness and efficiency of open relation extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 447–457.

Ferdinand de Saussure. 1916. Cours de linguistique générale [course in general linguistics]. Edited by Charles Bally and Albert Sechehaye.

Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming question answering datasets into natural language inference datasets. *CoRR*, abs/1809.02922.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Edsger W. Dijkstra. 1974. Ewd 447: On the role of scientific thought. In *Selected Writings on Computing: A Personal Perspective*, pages 60–66. Springer-Verlag. Book published in 1982.

DKB. 2022. Google search is dying.

David Dowty. 1991. Thematic proto-roles and argument selection. *Language*, 67(3):547–619.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Manjuan Duan, Ethan Hill, and Michael White. 2016. Generating disambiguating paraphrases for structurally ambiguous sentences. In *Proceedings of the 10th Linguistic Annotation Workshop*.

Yanai Elazar, Victoria Basmov, Yoav Goldberg, and Reut Tsarfaty. 2021. Text-based np enrichment.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

Elias Stengel-Eskin, Jimena Guallar-Blasco, Yi Zhou, and Benjamin Van Durme. 2022. Why did the chicken cross the road? rephrasing and analyzing ambiguous questions in vqa.

Ori Ernst, Avi Caciularu, Ori Shapira, Ramakanth Pasunuru, Mohit Bansal, Jacob Goldberger, and Ido Dagan. 2022. Proposition-level clustering for multi-document summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1765–1779, Seattle, United States. Association for Computational Linguistics.

Ori Ernst, Ori Shapira, Ramakanth Pasunuru, Michael Lepioshkin, Jacob Goldberger, Mohit Bansal, and Ido Dagan. 2021. Summary-source proposition-level alignment: Task, datasets and supervised baseline. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 310–322, Online. Association for Computational Linguistics.

Charles J. Fillmore. 1968. The case for case. In Emmon Bach and Robert T. Harms, editors, *Universals in Linguistic Theory*. Holt, Rinehart & Winston.

Nicholas FitzGerald, Julian Michael, Luheng He, and Luke Zettlemoyer. 2018. Large-scale QA-SRL parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2051–2060. Association for Computational Linguistics.

Nicholas FitzGerald, Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Semantic role labeling with neural network factors. In *EMNLP 2015*.

Dan Flickinger, Stephan Oepen, and Emily M. Bender. 2017. Sustainable development and refinement of complex linguistic annotations at scale. In *Handbook of Linguistic Annotation*, pages 353–377, Dordrecht. Springer Netherlands.

Michael C. Frank and Noah D. Goodman. 2012. Predicting pragmatic reasoning in language games. *Science*, 336(6084):998–998.

Morris F. Friedell. 1969. On the structure of shared awareness. *Systems Research and Behavioral Science*, 14:28–39.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *NIPS*.

William Gantt, Lelia Glass, and Aaron Steven White. 2021. Decomposing and recomposing event structure. *CoRR*, abs/2103.10387.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2021. The pile: An 800gb dataset of diverse text for language modeling.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Matt Gardner, William Merrill, Jesse Dodge, Matthew Peters, Alexis Ross, Sameer Singh, and Noah A. Smith. 2021. Competency problems: On finding and removing artifacts in language data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1801–1813, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michael Gelfond. 2021. Some thoughts on logic, declarative programming, and knowledge representation. Talk given for World Logic Day, January 14.

Matthew Gerber and Joyce Y Chai. 2010. Beyond nombank: A study of implicit arguments for nominal predicates. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1583–1592. Association for Computational Linguistics.

H. P. Grice. 1975. Logic and conversation. In Peter Cole and Jerry L. Morgan, editors, *Syntax and Semantics: Vol. 3: Speech Acts*, pages 41–58. Academic Press, New York.

Paul Grice. 1989. *Studies in the Way of Words.* Harvard University Press, Cambridge, MA.

Jeffrey S. Gruber. 1965. *Studies in Lexical Relations.* Ph.D. thesis, Massachusetts Institute of Technology.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Martin Haspelmath. 2010. Framework-free grammatical theory. In *The Oxford Handbook of Linguistic Analysis*, Oxford, UK. Oxford University Press.

Hangfeng He, Qiang Ning, and Dan Roth. 2020. QuASE: Question-answer driven sentence encoding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8743–8758, Online. Association for Computational Linguistics.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 364–369. Association for Computational Linguistics.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017a. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483, Vancouver, Canada. Association for Computational Linguistics.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017b. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483, Vancouver, Canada. Association for Computational Linguistics.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015a. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015b. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 643–653, Lisbon, Portugal. Association for Computational Linguistics.

Luheng He, Julian Michael, Mike Lewis, and Luke Zettlemoyer. 2016. Human-in-the-loop parsing. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2337–2342, Austin, Texas. Association for Computational Linguistics.

Irene Heim. 1983. On the projection problem for presuppositions. In P. Portner and B. H. Partee, editors, *Formal Semantics - the Essential Readings*, pages 249–260. Blackwell.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Julia Hockenmaier and Mark Steedman. 2007. CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.

Douglas Hofstadter. 2018. The shallowness of google translate. *The Atlantic*.

Douglas R Hofstadter. 1979. *Gödel, Escher, Bach: an eternal golden braid*. Penguin books. Basic Books, New York, NY.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60, New York City, USA. Association for Computational Linguistics.

IEEE. 2021. IEEE standard for information technology–telecommunications and information exchange between systems local and metropolitan area networks–specific requirements part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications amendment 1: Enhancements for high-efficiency WLAN. *IEEE Std 802.11ax-2021 (Amendment to IEEE Std 802.11-2020)*, pages 1–767.

William James. 1907. *Pragmatism: a New Name for some Old Ways of Thinking*. Project Gutenberg.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Melvin Johnson. 2018. Providing gender-specific translations in google translate.

Melvin Johnson. 2020. A scalable approach to reducing gender bias in google translate.

Dan Jurafsky and James Martin. 2008. *Speech and Language Processing*, 2nd edition. Prentice Hall, Upper Saddle River, NJ.

Hans Kamp. 1981. A theory of truth and semantic representation. In P. Portner and B. H. Partee, editors, *Formal Semantics - the Essential Readings*, pages 189–222. Blackwell.

Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *CVPR 2017*.

Eugene Kharitonov and Rahma Chaabouni. 2021. What they do when in doubt: a study of inductive biases in seq2seq learners. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Najoung Kim and Tal Linzen. 2020. COGS: A compositional generalization challenge based on semantic interpretation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9087–9105, Online. Association for Computational Linguistics.

Yoon Kim, Chris Dyer, and Alexander Rush. 2019. Compound probabilistic context-free grammars for grammar induction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2369–2385, Florence, Italy. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Ayal Klein, Jonathan Mamou, Valentina Pyatkin, Daniela Stepanov, Hangfeng He, Dan Roth, Luke Zettlemoyer, and Ido Dagan. 2020. QANom: Question-answer driven SRL for nominalizations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3069–3083, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Kalpesh Krishna, Aurko Roy, and Mohit Iyyer. 2021. Hurdles to progress in long-form question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the*

Association for Computational Linguistics: Human Language Technologies, pages 4940–4957, Online. Association for Computational Linguistics.

Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, Graham Neubig, and Noah A. Smith. 2017. What do recurrent neural network grammars learn about syntax? In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1249–1258, Valencia, Spain. Association for Computational Linguistics.

Adhiguna Kuncoro, Chris Dyer, Laura Rimell, Stephen Clark, and Phil Blunsom. 2019. Scalable syntax-aware language models using knowledge distillation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3472–3484, Florence, Italy. Association for Computational Linguistics.

Ilia Kuznetsov and Iryna Gurevych. 2020. A matter of framing: The impact of linguistic formalism on probing results. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 171–182, Online. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Brenden M. Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2879–2888. PMLR.

Joel Lang and Mirella Lapata. 2010. Unsupervised induction of semantic roles. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 939–947, Los Angeles, California. Association for Computational Linguistics.

Joel Lang and Mirella Lapata. 2011. Unsupervised semantic role induction via split-merge clustering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1117–1126, Portland, Oregon, USA. Association for Computational Linguistics.

Joel Lang and Mirella Lapata. 2014. Similarity-driven semantic role induction via graph partitioning. *Computational Linguistics*, 40(3):633–669.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017a. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017b. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197. Association for Computational Linguistics.

Kenton Lee, T. Kwiatkowski, Ankur P. Parikh, and Dipanjan Das. 2016. Learning recurrent span representations for extractive question answering. *ArXiv*, abs/1611.01436.

Douglas B. Lenat. 1995. CYC: A large-scale investment in knowledge infrastructure. *Commun. ACM*, 38(11):32–38.

David Lewis. 1969. *Convention: A Philosophical Study*. Harvard University Press, Cambridge, MA.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Thomas Liao, Rohan Taori, Inioluwa Deborah Raji, and Ludwig Schmidt. 2021. Are we learning yet? a meta review of evaluation failures across machine learning. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Vladimir Lifschitz. 2008. What is answer set programming? In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, AAAI'08, page 1594–1597. AAAI Press.

Jianhua Lin. 1991. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37:145–151.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Alisa Liu, Zhaofeng Wu, Julian Michael, Alane Suhr, Peter West, Alexander Koller, Swabha Swayamdipta, Noah A. Smith, and Yejin Choi. 2023. We're afraid language models aren't modeling ambiguity.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike

Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yi Luan, Yangfeng Ji, Hannaneh Hajishirzi, and Boyang Li. 2016. Multiplicative representations for unsupervised semantic role induction. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 118–123, Berlin, Germany. Association for Computational Linguistics.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60. Association for Computational Linguistics.

Gary Marcus and Ernest Davis. 2020. Gpt-3, bloviator: Openai's language generator has no idea what it's talking about. *MIT Technology Review*.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993a. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993b. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Jacob Menick, Maja Trebacz, Vladimir Mikulik, John Aslanides, Francis Song, Martin Chadwick, Mia Glaese, Susannah Young, Lucy Campbell-Gillingham, Geoffrey Irving, and Nat McAleese. 2022. Teaching language models to support answers with verified quotes.

Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The nombank project: An interim report. In *HLT-NAACL 2004 workshop: Frontiers in corpus annotation*.

Julian Michael. 2020. To dissect an octopus: Making sense of the form/meaning debate.

Julian Michael, Jan A. Botha, and Ian Tenney. 2020. Asking without telling: Exploring latent ontologies in contextual representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6792–6812, Online. Association for Computational Linguistics.

Julian Michael, Gabriel Stanovsky, Luheng He, Ido Dagan, and Luke Zettlemoyer. 2017. Crowdsourcing question-answer meaning representations. *CoRR*, abs/1711.05885.

Julian Michael, Gabriel Stanovsky, Luheng He, Ido Dagan, and Luke Zettlemoyer. 2018. Crowdsourcing question-answer meaning representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 560–568, New Orleans, Louisiana. Association for Computational Linguistics.

Julian Michael and Luke Zettlemoyer. 2021. Inducing semantic roles without syntax. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4427–4442, Online. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint 1301.3781*.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint*.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. In *Proceedings of the 2020 Conference on Empirical*

Methods in Natural Language Processing (EMNLP), pages 5783–5797, Online. Association for Computational Linguistics.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. Compositional questions do not necessitate multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4249–4257, Florence, Italy. Association for Computational Linguistics.

Marvin Minsky. 1990. Thoughts about artificial intelligence. In Ray Kurzweil, editor, *The Age of Intelligent Machines*. MIT Press, Cambridge, MA.

Marvin Minsky and Seymour Papert. 1969. *Perceptrons: an Introduction to Computational Geometry*. MIT Press, Cambridge, MA.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Preslav Nakov. 2008. Noun compound interpretation using paraphrasing verbs: Feasibility study. In *Artificial Intelligence: Methodology, Systems, and Applications: 13th International Conference, AIMSA 2008, Varna, Bulgaria, September 4-6, 2008. Proceedings*, pages 103–117, Berlin, Heidelberg. Springer Berlin Heidelberg.

Nikita Nangia and Samuel R. Bowman. 2019. Human vs. muppet: A conservative estimate of human performance on the GLUE benchmark. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4566–4575, Florence, Italy. Association for Computational Linguistics.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Advances in Neural Information Processing Systems*.

Yixin Nie, Xiang Zhou, and Mohit Bansal. 2020. What can we learn from collective human opinions on natural language inference data? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9131–9143, Online. Association for Computational Linguistics.

K. Nigam, A. McCallum, S. Thrun, and Tom Michael Mitchell. 2000. Text classification from labeled and unlabeled documents using em. *Machine Learning*, 39:103–134.

Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4658–4664, Florence, Italy. Association for Computational Linguistics.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2006. MaltParser: A data-driven parser-generator for dependency parsing. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

Peter Norvig. 2011. On chomsky and the two cultures of statistical learning.

Stephan Oepen, Dan Flickinger, Kristina Toutanova, and Christopher D. Manning. 2004. LinGO Redwoods: A rich and dynamic treebank for HPSG. *Research on Language and Computation*, 2:575–596.

Christopher Olah, Nick Cammarata, L. Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. 2020. Zoom in: An introduction to circuits. *Distill*, 5(3).

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2022. In-context learning and induction heads. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

Martha Palmer, H. Dang, and C. Fellbaum. 2006. Making fine-grained and coarse-grained sense distinctions, both manually and automatically. *Natural Language Engineering*, 13:137–163.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany. Association for Computational Linguistics.

Ellie Pavlick and Tom Kwiatkowski. 2019. Inherent disagreements in human textual inferences. *Transactions of the Association for Computational Linguistics*, 7:677–694.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014a. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014b. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Fernando Pereira, Naftali Tishby, and Lillian Lee. 1993. Distributional clustering of English words. In *31st Annual Meeting of the Association for Computational Linguistics*, pages 183–190, Columbus, Ohio, USA. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics:*

Human Language Technologies, Volume 1 (Long Papers), pages 2227–2237. Association for Computational Linguistics.

Benjamin C. Pierce, Arthur Azevedo de Amorim, Chris Casinghino, Marco Gaboardi, Michael Greenberg, Cătălin Hriţcu, Vilhelm Sjöberg, and Brent Yorgey. 2017. *Software Foundations*. Electronic textbook. Version 5.0. http://www.cis.upenn.edu/ bcpierce/sf.

Fabio Poletto, Valerio Basile, Manuela Sanguinetti, Cristina Bosco, and Viviana Patti. 2021. Resources and benchmark corpora for hate speech detection: a systematic review. *Lang. Resour. Evaluation*, 55:477–523.

Carl Pollard and Ivan A Sag. 1994. *Head-driven phrase structure grammar*. University of Chicago Press.

Paul Portner. 2004. The semantics of imperatives within a theory of clause types. *Semantics and Linguistic Theory*, 14:235–252.

James Pustejovsky. 2011. Coercion in a general theory of argument selection. *Linguistics*, 49(6):1401–1431.

Valentina Pyatkin, Paul Roit, Julian Michael, Yoav Goldberg, Reut Tsarfaty, and Ido Dagan. 2021. Asking it all: Generating contextualized questions for any semantic role. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1429–1441, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. Retrieved 2022.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *EMNLP 2016*.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with clip latents.

Yasaman Razeghi, Robert L. Logan, Matt Gardner, and Sameer Singh. 2022. Impact of pretraining term frequencies on few-shot reasoning.

Drew Reisinger, Rachel Rudinger, Francis Ferraro, Craig Harman, Kyle Rawlins, and Benjamin Van Durme. 2015a. Semantic proto-roles. *Transactions of the Association for Computational Linguistics*, 3:475–488.

Drew Reisinger, Rachel Rudinger, Francis Ferraro, Craig Harman, Kyle Rawlins, and Benjamin Van Durme. 2015b. Semantic proto-roles. *Transactions of the Association for Computational Linguistics*, pages 475–488.

Matthew Richardson, Christopher JC Burges, and Erin Renshaw. 2013. MCTest: A challenge dataset for the open-domain machine comprehension of text. In *EMNLP*, pages 193–203.

Stefan Riezler. 2014. Last words: On the problem of theoretical terms in empirical computational linguistics. *Computational Linguistics*, 40(1):235–245.

Paul Roit, Ayal Klein, Daniela Stepanov, Jonathan Mamou, Julian Michael, Gabriel Stanovsky, Luke Zettlemoyer, and Ido Dagan. 2020. Controlled crowdsourcing for high-quality QA-SRL annotation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7008–7013, Online. Association for Computational Linguistics.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, Prague, Czech Republic. Association for Computational Linguistics.

Josef Ruppenhofer, Michael Ellsworth, Miriam RL Petruck, Christopher R Johnson, and Jan Scheffczyk. 2016. *FrameNet II: Extended theory and practice.* Institut für Deutsche Sprache, Bibliothek.

Andrew M Saxe, James L McClelland, and Surya Ganguli. 2014. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *ICLR 2014.*

Ulrich Schäfer, Bernd Kiefer, Christian Spurk, Jörg Steffen, and Rui Wang. 2011. The ACL Anthology searchbench. In *Proceedings of the ACL-HLT 2011 System Demonstrations*, pages 7–13, Portland, Oregon. Association for Computational Linguistics.

Thomas Schelling. 1960. *The Strategy of Conflict.* Harvard University Press, Cambridge, MA.

Rudolf Schneider, Tom Oberhauser, Tobias Klatt, Felix A. Gers, and Alexander Loser. 2017. Analysing errors of open information extraction systems. *CoRR*, abs/1707.07499.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603.*

Claude E. Shannon. 1948. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423.

Tianze Shi, Ozan İrsoy, Igor Malioutov, and Lillian Lee. 2021. Learning syntax from naturally-occurring bracketings. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2941–2949, Online. Association for Computational Linguistics.

Micah Shlain, Hillel Taub-Tabib, Shoval Sadde, and Yoav Goldberg. 2020. Syntactic search by example. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 17–23, Online. Association for Computational Linguistics.

Noam Slonim and Naftali Tishby. 1999. Agglomerative information bottleneck. In *Proceedings of the 12th International Conference on Neural Information Processing Systems*, NIPS'99, pages 617–623, Cambridge, MA, USA. MIT Press.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4444–4451. AAAI Press.

R. Srivastava, Klaus Greff, and J. Schmidhuber. 2015. Training very deep networks. In *NIPS*.

Robert Stalnaker. 1978. Assertion. *Syntax and Semantics (New York Academic Press)*, 9:315–332.

Robert C. Stalnaker. 2002. Common ground. *Linguistics and Philosophy*, 25(5-6):701–721.

Miloš Stanojević and Mark Steedman. 2021. Formal Basis of a Language Universal. *Computational Linguistics*, 47(1):9–42.

Gabriel Stanovsky and Ido Dagan. 2016a. Annotating and predicting non-restrictive noun phrase modifications. In *Proceedings of the 54rd Annual Meeting of the Association for Computational Linguistics (ACL 2016)*.

Gabriel Stanovsky and Ido Dagan. 2016b. Creating a large benchmark for open information extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Austin, Texas. Association for Computational Linguistics.

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised open information extraction. In *Proceedings of the 2018 Conference of the North American Chapter of* *the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 885–895, New Orleans, Louisiana. Association for Computational Linguistics.

Mark Steedman. 1996. *Surface Structure and Interpretation.* MIT Press.

Mark Steedman. 2000. *The Syntactic Process.* MIT Press.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL 2008 shared task on joint parsing of syntactic and semantic dependencies. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 159–177, Manchester, England. Coling 2008 Organizing Committee.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3104–3112, Cambridge, MA, USA. MIT Press.

Rich Sutton. 2019. The bitter lesson.

Swabha Swayamdipta, Sam Thomson, Chris Dyer, and Noah A Smith. 2017. Frame-semantic parsing with softmax-margin segmental rnns and a syntactic scaffold. *arXiv preprint arXiv:1706.09528*.

Adam Teichert, Adam Poliak, Benjamin Van Durme, and Matthew Gormley. 2017. Semantic proto-role labeling. In *Proceedings of AAAI*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? probing for sentence structure in contextualized word representations. In

7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net.

Ivan Titov and Ehsan Khoddam. 2015. Unsupervised induction of semantic roles within a reconstruction-error minimization framework. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1–10, Denver, Colorado. Association for Computational Linguistics.

Ivan Titov and Alexandre Klementiev. 2012. A Bayesian approach to unsupervised semantic role induction. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 12–22, Avignon, France. Association for Computational Linguistics.

Michael Tomasello, Malinda Carpenter, and Ulf Liszkowski. 2007. A new look at infant pointing. *Child development*, 78(3):705–722.

Michael Tomasello and Michael Jeffrey Farrar. 1986. Joint attention and early language. *Child development*, 57 6:1454–63.

Alan M. Turing. 1950. Computing machinery and intelligence. *Mind*, LIX(236):433–460.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. SuperGLUE: A multi-task benchmark and analysis platform for natural language understanding. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 3261–3275. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Chenguang Wang, Alan Akbik, laura chiticariu, Yunyao Li, Fei Xia, and Anbang Xu. 2017. Crowd-in-the-loop: A hybrid approach for annotating semantic roles. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1914–1923. Association for Computational Linguistics.

Hai Wang, Mohit Bansal, Kevin Gimpel, and David McAllester. 2015. Machine comprehension with syntax, frames, and semantics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 700–706, Beijing, China. Association for Computational Linguistics.

Joseph Weizenbaum. 1976. *Computer Power and Human Reason: From Judgment to Calculation*. W. H. Freeman & Co., USA.

Daniel S. Weld and Gagan Bansal. 2019. The challenge of crafting intelligible intelligence. *Commun. ACM*, 62(6):70–79.

Keenon Werling, Arun Tejasvi Chaganty, Percy S Liang, and Christopher D Manning. 2015. On-the-job learning with bayesian decision theory. In *Advances in Neural Information Processing Systems*.

Aaron Steven White. 2021. On believing and hoping whether. *Semantics and Pragmatics*, 14(6):1–21.

Aaron Steven White and Kyle Rawlins. 2016. A computational model of s-selection. *Semantics and Linguistic Theory*, 26:641–663.

Aaron Steven White and Kyle Rawlins. 2018. The role of veridicality and factivity in clause selection. In *Proceedings of the 48th Annual Meeting of the North East Linguistic Society*, pages 221–234, Amherst, MA. GLSA Publications.

Aaron Steven White, Kyle Rawlins, and Benjamin Van Durme. 2017a. The semantic proto-role linking model. In *Proceedings of the 15th Conference of the European Chapter of the Association* *for Computational Linguistics: Volume 2, Short Papers*, pages 92–98, Valencia, Spain. Association for Computational Linguistics.

Aaron Steven White, Kyle Rawlins, and Benjamin Van Durme. 2017b. The semantic proto-role linking model. In *ACL 2017*.

Aaron Steven White, Drew Reisinger, Keisuke Sakaguchi, Tim Vieira, Sheng Zhang, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme. 2016. Universal decompositional semantics on Universal Dependencies. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1713–1723, Austin, Texas. Association for Computational Linguistics.

Kristian Woodsend and Mirella Lapata. 2015. Distributed representations for unsupervised semantic role labeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2482–2491, Lisbon, Portugal. Association for Computational Linguistics.

Ying Xu, Mi-Young Kim, Kevin Quinn, Randy Goebel, and Denilson Barbosa. 2013. Open information extraction with tree kernels. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 868–877, Atlanta, Georgia. Association for Computational Linguistics.

Hitomi Yanaka, Koji Mineshima, Daisuke Bekki, and Kentaro Inui. 2020. Do neural models learn systematicity of monotonicity inference in natural language? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6105–6117, Online. Association for Computational Linguistics.

Hitomi Yanaka, Koji Mineshima, and Kentaro Inui. 2021. SyGNS: A systematic generalization testbed based on natural language semantics. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 103–119, Online. Association for Computational Linguistics.

Bishan Yang and Tom Mitchell. 2017. A joint sequential and relational model for frame-semantic parsing. In *EMNLP 2017*, pages 1247–1256.

F. Yergeau. 2003. UTF-8, a transformation format of ISO 10646. RFC 3629, RFC Editor.

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

Philine Zeinert, Nanna Inie, and Leon Derczynski. 2021. Annotating online misogyny. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3181–3197, Online. Association for Computational Linguistics.

Jie Zhou and Wei Xu. 2015a. End-to-end learning of semantic role labeling using recurrent neural networks. In *ACL 2015*.

Jie Zhou and Wei Xu. 2015b. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1127–1137, Beijing, China. Association for Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017. Neural question generation from text: A preliminary study. *arXiv preprint arXiv:1704.01792*.

VITA

Julian Michael grew up in Plano, Texas and received his B.S. in Computer Science from the University of Texas at Austin in 2015, where he first conducted research with Vladimir Lifschitz. He did his PhD in Computer Science and Engineering at the University of Washington, where he was advised by Luke Zettlemoyer. Julian moved to New York University in 2022 to work with Sam Bowman as a research scientist, and received his PhD in June of 2023 when he finally got around to turning in his dissertation. Outside of research, Julian enjoys body movement practices like parkour, rock climbing, and yoga, as well as learning about psychology, physiology, and medicine, and playing ball with philosophers. Julian also enjoys learning how to do new things and then never doing them, trying to figure out and resolve the cruxes of people's disagreements, and occasionally pretending to meditate.