My research aims to develop generative models of text that are powerful, reliable, and intelligent enough to enable non-experts to use them to automate complex tasks. Think of a machine that expands an outline into a gripping screenplay or composes a nuanced, long form answer to a complex question, both of which can then be further edited through suggestions made in natural language. To build such models requires not just better algorithms, but also better methods for understanding how current large language models work and what extra information about the world is needed to complete an ever wider range of text-based tasks.

Controlling Generative Models (§1) Unlike traditional machine learning models, data-hungry generative models are trained on raw data scraped from the web, which makes eliciting specific behaviors a significant technical challenge. I develop algorithms that allow for finer grained control, e.g. to make sure that a model generating a story does not accidentally slip into writing a book review instead. [4, 8, 12, 2]

Understanding Models via Generative Behavior (§2) Current language models (LMs), with their enormous number of parameters ($> 10^9$) are challenging to explain mechanistically. I design methods to backwards-engineer models through observed behavior, so we can answer questions such as: Why did it say the most populous country in North America is Mexico? [6, 9, 1]

Connecting Language Models to the World (§3) When humans speak, they draw on *extralinguistic* information observed in the physical and social world. I engineer systems that connect information about the world to its corresponding meaning in language, e.g., by embedding generative models in simulated physical worlds to understand the properties and relationships between objects. [5, 7, 13, 3]

1 Controlling Generative Models

Our understanding of how generative models work (i.e. what functions they are internally computing) is still limited, but **the ultimate test of understanding a generative model is the ability to manipulate its outputs precisely**. If we can reliably predict how changes in the input will effect the output, this establishes **causal proof** that we understand how a model functions. The primary obstacle is that neural generative models are **opaque by construction**—purposefully over-parameterized to soak up information, with architectural structure that is so general it is difficult to interpret.

Generative models often overfit easy-to-learn patterns, e.g., language models (LMs) overfit to the natural patterns of word reuse by repeating words and phrases far too much. I have shown that this results in a positive feedback loop (illustrated in Figure 1), causing LMs to generate a single phrase over and

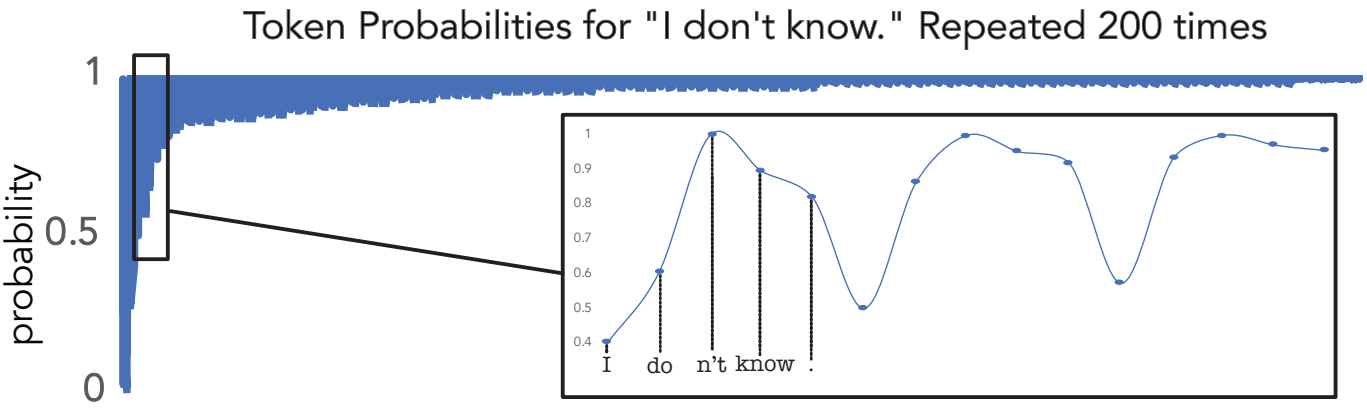

Figure 1: The probability of a sequence increases with each repetition, creating a positive feedback loop.

over again [4]. It turns out this is largely a problem of a **misaligned inference method**. The standard algorithm for generating text with an LM was to greedily choose the highest probability word from left to right, which encourages more repetition once the first repetition is generated. Inventing a solution requires us to **identify and exploit how models naturally structure their predictions**. Most generative LMs are neural models trained to approximate the probability distribution $P(w_i|w_1 \cdots w_{i-1})$ of a word drawn from a fixed vocabulary $w_i \in \mathcal{V}$, given the previous words in a document $w_1 \cdots w_{i-1}$. In [4] I showed that, for any individual prediction, LMs assign most probability mass to a small portion of contextually relevant vocabulary: the "nucleus" of the distribution. I proposed sampling directly from this subset, which eliminates repetition loops and increases coherence. Since publication, this method has become the standard decoding method for open-ended text generation and is used in thousands of academic and industrial systems, including user-facing systems such as the GPT-3 API [11].

Of course, there are many other ways in which we wish to control models. Model behavior changes with model size and we can **exploit model differences to factor out errors**, since large LMs make most of the same mistakes as smaller LMs, but simply do so with lower frequency. In [8] I proposed a decoding algorithm that uses the over-confidence of smaller LMs as an indicator of likely errors, using this to guide large LMs to avoid their already reduced errors. Text generated with this method matches the human distribution of text more closely, e.g., mentioning the same entities as humans do given the same prompt. Because of the extremely general nature of language models, **we can use generative models to probe for information in *their own* outputs**. For instance, I designed a system to guide an LM to summarize input text, by using that same LM to verify whether model-generated summaries contained the same information as the original text [12]. **Training neural models with user manipulable structure** can give us even finer grained control. My colleagues and I proposed a new neural architecture with parallel subnetworks, each specialized to a different information source. This allows us to remove certain sources at inference time, without further training [2]. We can thus train one subnetwork on undesirable content, e.g. hate speech, then remove it at inference time, removing the corresponding undesirable behavior.

2 Understanding Models via Generative Behavior

Generative models are not like traditional machine-learning models designed to complete a specific task. Instead, most generative models are controlled through *prompting*, by providing them with a string prefix for which they must generate a completion. If we want a model to answer a question we can *prompt* it by formatting the input correctly, e.g., "Q: What is the chemical formula for salt? A:" and let the model generate "NaCl". This raises the question: **what kind of input causes what kind of model behavior?**

Through prompting, LMs are commonly used to solve multiple-choice questions (similar to the GRE format) by choosing the highest-probability option. This works surprisingly well, but I show in [6] that ranking by probability is suboptimal, because different strings compete for probability mass (Figure 2), e.g. the model reserves finite probability mass for "USA" and "United States" when asked "What is the largest country in North America?" This causes any concept with multiple different representative strings to be assigned less probability mass under the model, even if it is conceptually correct. The issue is that **generative models attempt to predict all behavior that may result from a given input simultaneously**, with probability assigned according to predicted frequency in human-generated text. To directly measure how *plausible* a model considers an answer we introduce an alternative scoring function that reweighs multiple choice probabilities according to their a priori probability as predicted by the model. This demonstrates how **models can be used to decompose their own predictions into human understandable parts**.

One common method for improving task performance is to condition generative models on example input-output pairs of the desired task—under the assumption that the model can learn the gist of the task from these examples. This tends to improve performance, but **high scores on a task do not tell us what information the model uses to complete the task**. I show that in most cases models are not directly

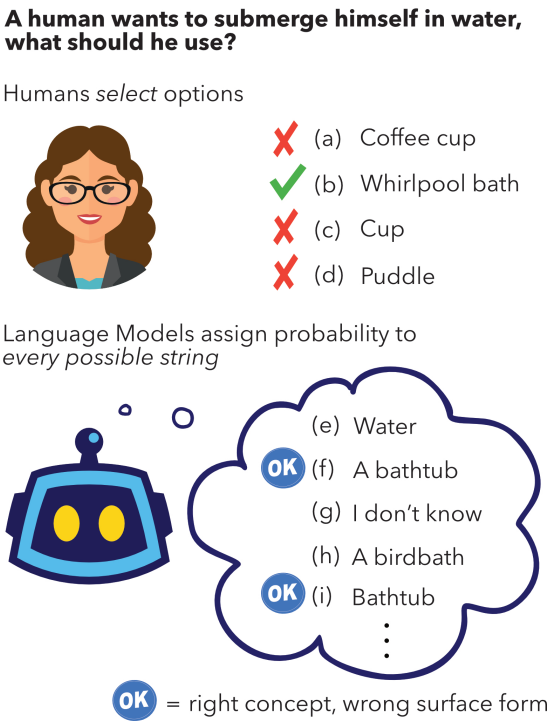

Figure 2: Humans can select from given options; LMs assign probability to all possible strings, even conceptually equivalent ones.

learning the semantics of a task from examples because their performance remains the same when conditioning on incorrect examples that do not actually represent the task [9], instead learning the vague distribution of the inputs, outputs, and formatting. To find out what models *do* know, we can **use generated text to examine model behavior directly**.

For example, in [1] I use generated sentences to investigate how much LMs know about the physical world, finding that models can infer physical attributes observed directly in language (e.g. "Elephants weigh more than humans.") but struggle with attributes that are only indirectly implicated. For instance the fact that a door can be *opened* is revealed when one says "I opened the door." but "Doors are openable." is almost never explicitly stated. To bridge this gap, the third branch of my work explores connecting models to the real world.

3 Connecting Language Models to the World

While the internet provides textual descriptions of almost every aspect of human life, **understanding language requires representing and reasoning about the world**. Since LMs do not have direct exposure to the physical and social world, they often fail to make inferences in these domains, and generate nonsense such as "I put the elephant in the office fridge."

An aspect of the world that has been especially tricky to encode are the **principles of cooperative communication** that regulate what is worth saying and how. Humans do not say things that are obvious or irrelevant ("By the way, this hotel exchanges money for shelter."). Without an understanding of such common knowledge, models misunderstand sarcasm as literal and rare events as common ("The rooms in this hotel are free!"). One path forward is **formalizing principles of communication directly in model behavior**, e.g., training models that are meant to directly encode these assumptions and can guide generative models towards more realistic text. In [5], I trained discriminative models to encode a theory of cooperative communication, and used them to guide a base generative model. I proposed an algorithm in which a base generative model self-critiques its own generated text using these discriminative models, then adjusts the influence the discriminative models have on text generation. The result was text that stated the obvious less and had **more coherent narrative structure**.

Many other principles of communication still need to be formalized to fix unrealistic model behavior. For instance, **taking orders in natural language requires understanding when you might have done something incorrectly.** In [7] I studied the problem of an agent attempting to follow directions in a photorealistic simulated environment, proposing an algorithm that allows the model to introspect on whether it has fallen off of the correct path and retrace its steps, achieving much more efficient navigation. To understand the physical dynamics of objects in the world, I showed that giving generative models exposure to full 3D environments where they can interact with objects to complete goals improves their textual representation of the objects they encounter [13]. I am especially excited by using multimodal models, with access to more than just text, to create **model-based metrics that more closely approximate human judgement**. I created an evaluation for automatic image captioning text that relies on co-trained image and text encoders [3], that is meant to explicitly model how text fits an image, rather than overlap with human reference captions. This metric correlates better with human judgements than all previous metrics, *without using a human reference text* that other metrics depend on, paving the way for model-based evaluations.

4 Future Directions

The most exciting frontier in generative modeling is *long-form generation*: text generated by current models barely remains coherent for a paragraph, images only remain realistic to a limited size, and generated videos only last a few seconds. Long-form generation has remained a thorny problem, because the number of dependencies within a generation grows exponentially with size. To make the problem tractable we need to *factor model behavior into its elementary components* the way the periodic table of elements factors the behavior of chemicals, explaining basic principles that can be built into more complex structures. Even with our currently limited understanding of generative models, a key weakness has become clear: *pragmatics*, the ability to infer unspoken information by reasoning about the current context and generate text that is useful rather than obvious. I expect my future work to focus on all three of these areas.

Long-form Generation Currently, it is difficult for models to coherently generate more than a paragraph, but my goal is to create systems that automatically plans and generates entire essays, technical documents, and even books by **training the model to plan for intended outcomes**. The current obstacle is that models are specifically trained to predict one token at a time, without any plan in mind. Rather than throwing away the current progress, the solution is to use current generative models as a powerful *prior* about natural language, while training models that specifically try to predict **higher-level narrative structure** in text. Initial attempts at this have achieved some success, but *evaluating generated text* is a significant bottleneck. A promising direction here is **indirect evaluation through downstream tasks**: rather than attempting to directly verify whether a model wrote an email correctly, test whether a person or another model given that email can correctly complete a desired task.

Taxonomizing Model Behavior We understand the low-level components of generative models perfectly by construction, but their high-level behavior is *emergent* as increased training data and model size results in unexpected (but often desirable) behavior. This means that **current generative models fit the definition of complex systems** [10]. To study complex systems, we need to **formalize definitions of behaviors that do not yet have low-level explanations**, e.g., LMs can often answer factual questions correctly, but can they distinguish factual questions from very common fictional scenarios? The final product of such an effort would be methodology for constructing a **predictive model of when and how behaviors interact** given black-box access to a model. This would help us control models, even models with new architectures and training objectives, more reliably, as well as predict failure cases, e.g., predicting that certain inputs will bias a given model so much towards copying previous text that it will inevitably regurgitate the input.

Making Generative Models Pragmatic Reasoners Current generative models are risky to deploy directly to users, because they are often offensive, obtuse, and chaotic—they commit social faux pas at an alarming rate. To overcome this we need to train models that **condition on contextual information**, e.g., correctly inferring the relationships between people in a conversation from **connotation and implication**. This means studying models that actually interact with simulated human worlds, as I began to in [13] and [7]. However, it also means **formally defining the inferences we want models to draw** from a given input. It is challenging for untrained humans to annotate these relationships, because we are largely unconscious of the social inferences we draw in daily life. Instead, I believe the path forward is **designing autodidact models that hypothesize what they are missing** and interact with humans and the internet in order to select among competing hypotheses. As children many of us had to ask a parent or mentor, "Was it rude when I said 'hate' instead of 'dislike'?" a rich source of information that generative models have thus far rarely been given access to.

References

[1] M. Forbes, A. Holtzman, and Y. Choi. Do neural language representations learn physical commonsense? In *CogSci*, 2019.

[2] S. Gururangan, M. Lewis, A. Holtzman, N. A. Smith, and L. Zettlemoyer. Demix layers: Disentangling domains for modular language modeling. *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2022.

[3] J. Hessel, A. Holtzman, M. Forbes, R. Le Bras, and Y. Choi. Clipscore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7514–7528, 2021.

[4] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi. The curious case of neural text degeneration. *International Conference on Learning Representations (ICLR)*, 2020.

[5] A. Holtzman, J. Buys, M. Forbes, A. Bosselut, D. Golub, and Y. Choi. Learning to write with cooperative discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1638–1649, 2018.

[6] A. Holtzman, P. West, V. Shwartz, Y. Choi, and L. Zettlemoyer. Surface form competition: Why the highest probability answer isn't always right. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7038–7051, 2021.

[7] L. Ke, X. Li, Y. Bisk, A. Holtzman, Z. Gan, J. Liu, J. Gao, Y. Choi, and S. Srinivasa. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6741–6749, 2019.

[8] X. L. Li, A. Holtzman, D. Fried, P. Liang, J. Eisner, T. Hashimoto, L. Zettlemoyer, and M. Lewis. Contrastive decoding: Open-ended text generation as optimization. *arXiv preprint arXiv:2210.15097*, 2022.

[9] S. Min, X. Lyu, A. Holtzman, M. Artetxe, M. Lewis, H. Hajishirzi, and L. Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

[10] S. Mittal, S. Diallo, and A. Tolk. *Emergent behavior in complex systems engineering: A modeling and simulation approach*. John Wiley & Sons, 2018.

[11] OpenAI. The openai api. *OpenAI*, 2020.

[12] P. West, A. Holtzman, J. Buys, and Y. Choi. Bottlesum: Unsupervised and self-supervised sentence summarization using the information bottleneck principle. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3752–3761, 2019.

[13] R. Zellers, A. Holtzman, M. E. Peters, R. Mottaghi, A. Kembhavi, A. Farhadi, and Y. Choi. Piglet: Language grounding through neuro-symbolic interaction in a 3d world. 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 2040–2050, 2021.