Implicit Representations of Meaning in Neural Language Models

Belinda Z. Li **Maxwell Nye** **Jacob Andreas**
Massachusetts Institute of Technology
{bzl,mnye,jda}@mit.edu

Abstract

Does the effectiveness of neural language models derive entirely from accurate modeling of surface word co-occurrence statistics, or do these models represent and reason about the world they describe? In BART and T5 transformer language models, we identify contextual word representations that function as *models of entities and situations* as they evolve throughout a discourse. These neural representations have functional similarities to linguistic models of dynamic semantics: they support a linear readout of each entity's current properties and relations, and can be manipulated with predictable effects on language generation. Our results indicate that prediction in pretrained neural language models is supported, at least in part, by dynamic representations of meaning and implicit simulation of entity state, and that this behavior can be learned with only text as training data.[1]

1 Introduction

Neural language models (NLMs), which place probability distributions over sequences of words, produce contextual word and sentence embeddings that are useful for a variety of language processing tasks (Peters et al., 2018; Lewis et al., 2020). This usefulness is partially explained by the fact that NLM representations encode lexical relations (Mikolov et al., 2013) and syntactic structure (Tenney et al., 2019). But the extent to which NLM training also induces representations of *meaning* remains a topic of ongoing debate (Bender and Koller, 2020; Wu et al., 2021). In this paper, we show that NLMs represent meaning in a specific sense: in simple semantic domains, they build representations of situations and entities that encode logical descriptions of each entity's dynamic state.

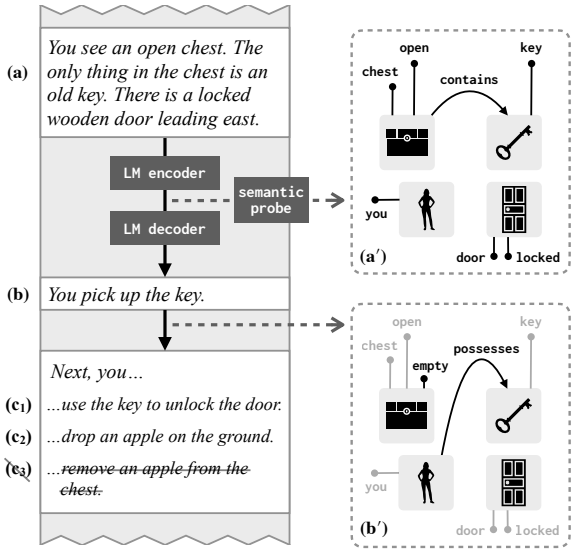

Figure 1: Neural language models trained on text alone (a–c) produce semantic representations that encode properties and relations of entities mentioned in a discourse (a′). Representations are updated when the discourse describes changes to entities' state (b′).

Consider the text in the left column of Fig. 1. Sentences (a) describe the contents of a room; this situation can be formally characterized by the graph of entities, properties, and relations depicted in (a′). Sentence (b), *You pick up the key*, causes the situation to change: a chest becomes `empty`, and a key becomes `possessed` by *you* rather than `contained` by the chest (b′). None of these changes are explicitly described by sentence (b). Nevertheless, the set of sentences that can follow (a)–(b) to form a semantically coherent discourse is determined by this new situation. An acceptable next sentence might feature the person using the key (c_1) or performing an unrelated action (c_2). But a sentence in which the person takes an apple out of the chest (c_3) cannot follow (a)–(b), as the chest is now empty.

Formal models of situations (built, like (a′)–(b′), from logical representations of entities and their

[1]Code and data are available at https://github.com/belindal/state-probes.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 1813–1827
August 1–6, 2021.

attributes) are central to linguistic theories of meaning. NLMs face the problem of learning to generate coherent text like (a–c) without access to any explicit supervision for the underlying world state (a′)–(b′). Indeed, recent work in NLP points to the lack of exposure of explicit representations of the world external to language as *prima facie* evidence that LMs cannot represent meaning at all, and thus cannot in general output coherent discourses like (a)–(c) (Bender and Koller, 2020).

The present paper can be viewed as an empirical response to these arguments. It is true that current NLMs do not reliably output coherent descriptions when trained on data like (a)–(c). But from text alone, even these imperfect NLMs appear to learn *implicit* models of meaning that are translatable into formal state representations like (a′)–(b′). These state representations capture information like the `emptiness` of the `chest` in (b′), which is not explicitly mentioned and cannot be derived from any purely syntactic representation of (a)–(b), but follows as a semantically necessary consequence. These implicit semantic models are roughly analogous to the simplest components of discourse representation theory and related formalisms: they represent sets of entities, and update the facts that are known about these entities as sentences are added to a discourse. Like the NLMs that produce them, these implicit models are approximate and error-prone. Nonetheless, they do most of the things we expect of world models in formal semantics: they are structured, queryable and manipulable. In this narrow sense, NLM training appears to induce not just models of linguistic form, but models of meaning.

This paper begins with a review of existing approaches to NLM probing and discourse representation that serve as a foundation for our approach. We then formalize a procedure for determining whether NLM representations encode representations of situations like Fig. 1 (a′)–(b′). Finally, we apply this approach to BART and T5 NLMs trained on text from the English-language Alchemy and TextWorld datasets. In all cases, we find evidence of implicit meaning representations that:

1. Can be linearly decoded from NLM encodings of entity mentions.
2. Are primarily attributable to open-domain pretraining rather than in-domain fine-tuning.
3. Influence downstream language generation.

We conclude with a discussion of the implications of these results for evaluating and improving factuality and coherence in NLMs.

2 Background

What do LM representations encode? This paper's investigation of state representations builds on a large body of past work aimed at understanding how other linguistic phenomena are represented in large-scale language models. NLM representations have been found to encode syntactic categories, dependency relations, and coreference information (Tenney et al., 2019; Hewitt and Manning, 2019; Clark et al., 2019). Within the realm of semantics, existing work has identified representations of word meaning (e.g., fine-grained word senses; Wiedemann et al. 2019) and predicate–argument structures like frames and semantic roles (Kovaleva et al., 2019).

In all these studies, the main experimental paradigm is *probing* (Shi et al., 2016; Belinkov and Glass, 2019): given a fixed source of representations (e.g. the BERT language model; Devlin et al. 2019) and a linguistic label of interest (e.g. semantic role), a low-capacity "probe" (e.g a linear classifier) is trained to predict the label from the representations (e.g. to predict semantic roles from BERT embeddings). A phenomenon is judged to be encoded by a model if the probe's accuracy cannot be explained by its accuracy when trained on control tasks (Hewitt and Liang, 2019) or baseline models (Pimentel et al., 2020).

Our work extends this experimental paradigm to a new class of semantic phenomena. As in past work, we train probes to recover semantic annotations, and interpret these probes by comparison to null hypotheses that test the role of the model and the difficulty of the task. The key distinction is that we aim to recover a representation of the *situation described by a discourse* rather than representations of the *sentences that make up the discourse*. For example, in Fig. 1, we aim to understand not only whether NLMs encode the (sentence-level) semantic information that there was a *picking up* event whose patient was *you* and whose agent was *the key*—we also wish to understand whether LMs encode the *consequences* of this action for all entities under discussion, including the chest from which the key was (implicitly) taken.

How might LMs encode meaning? Like in other probing work, an attempt to identify neural

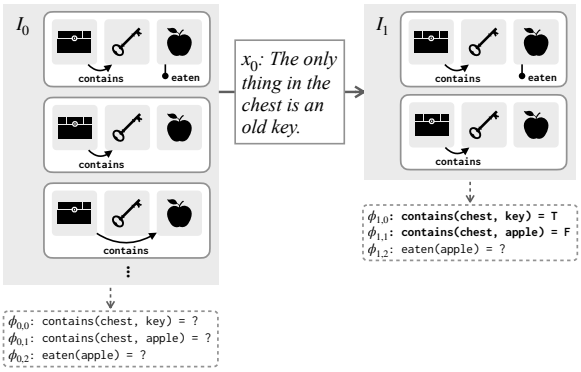

Figure 2: A collection of possible situations is represented as an information state (I_0). Information states assign values to propositions $\phi_{i,j}$ according to whether they are true, false, or undetermined in all the situations that make up an information state. Appending a new sentence discourse causes the information state to be updated (I_1). In this case, the sentence *The only thing in the chest is an old key* causes `contains(chest, key)` to become true, `contains(chest, apple)` to become false, and leaves `eaten(apple)` undetermined.

encodings of entities and situations must begin with a formal framework for representing them. This is the subject of **dynamic semantics** in linguistics (Heim, 2008; Kamp et al., 2010; Groenendijk and Stokhof, 1991). The central tool for representing meaning in these approaches is the *information state*: the set of possible states of the world consistent with a discourse (I_0 and I_1 in Fig. 2). Before anything is said, all logically consistent situations are part of the information state (I_0). Each new sentence in a discourse provides an update (that constrains or otherwise manipulates the set of possible situations). As shown in the figure, these updates can affect even unmentioned entities: the sentence *the only thing in the chest is a key* ensures that the proposition `contains(chest, `x`)` is false for all entities x other than the key. This is formalized in §3 below.[2]

The main hypothesis explored in this paper is that *LMs represent (a particular class of) information states*. Given an LM trained on text alone, and a discourse annotated post-hoc with information states, our probes will try to recover these information states from LM representations. The semantics literature includes a variety of proposals for how information states should be represented; here, we will represent information states logically, and decode information states via the truth values that they assign to logical propositions ($\phi_{i,j}$ in Fig. 2).[3]

[2]See also Yalcin (2014) for an introductory survey.

[3]In existing work, one of the main applications of dynamic semantics is a precise treatment of quantification and scope at the discourse level. The tasks investigated in this paper do not involve any interesting quantification, and rely on the simplest parts of the formalism. More detailed exploration of quantification in NLMs is an important topic for future study.

LMs and other semantic phenomena In addition to work on interpretability, a great deal of past research uses language modeling as a pretraining scheme for more conventional (supervised) semantics tasks in NLP. LM pretraining is useful for semantic parsing (Einolghozati et al., 2019), instruction following (Hill et al., 2020), and even image retrieval (Ilharco et al., 2021). Here, our primary objective is not good performance on downstream tasks, but rather understanding of representations themselves. LM pretraining has also been found to be useful for tasks like factoid question answering (Petroni et al., 2019; Roberts et al., 2020). Our experiments do not explore the extent to which LMs encode static background knowledge, but instead the extent to which they can build representations of novel situations described by novel text.

3 Approach

Overview We train probing models to test whether NLMs represent the information states specified by the input text. We specifically probe for the truth values of logical propositions about entities mentioned in the text. For example, in Figure 1, we test whether a representation of sentences (a)–(b) encodes the fact that `empty(chest)` is true and `contains(chest, key)` is false.

Meanings as information states To formalize this: given a universe consisting of a set of entities, properties, and relations, we define a **situation** as a *complete* specification of the properties and relations of each entity. For example, the box labeled I_0 in Fig. 2 shows three situations involving a `chest`, a `key`, an `apple`, an `eaten` property and a `contains` relation. In one situation, the chest contains the key and the apple is eaten. In another, the chest contains the apple, and the apple is not eaten. In general, a situation assigns a value of true or false to *every* logical proposition of the form $P(x)$ or $R(x, y)$ (e.g. `locked(door)` or `contains(chest, key)`).

Now, given a natural language discourse, we can view that discourse as specifying a *set* of possible situations. In Fig. 2, the sentence x_0 picks out the subset of situations in which the chest contains the key. A collection of situations is called an **information state**, because it encodes a listener's

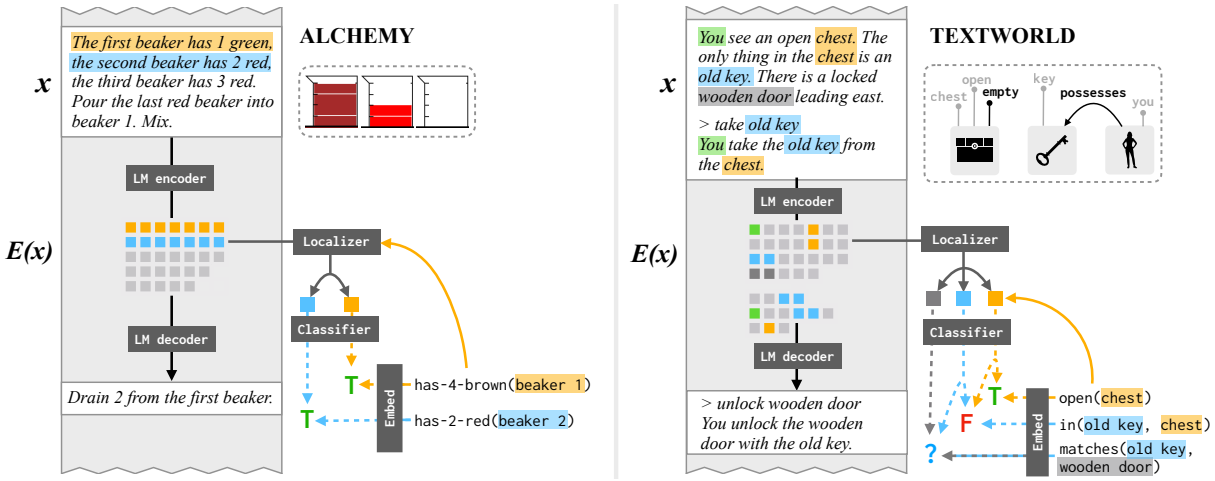

Figure 3: Overview of the probe model. **Left**: Alchemy. **Right**: Textworld. The LM is first trained to generate the next instruction from prior context (left side, both figures). Next, the LM encoder is frozen and a probe is trained to recover (the truthfulness of) propositions about the current state from specific tokens of encoder outputs.

knowledge of (and uncertainty about) the state of the world resulting from the events described in a discourse.[4] In a given information state, the value of a proposition might be *true* in all situations, *false* in all situations, or *unknown*: true in some but false in others. An information state (or an NLM representation) can thus be characterized by the label it assigns to every proposition.

Probing for propositions We assume we are given:

- A sequence of sentences $x_{1:n} = [x_1, \ldots, x_n]$.
- For each i, the information state I_i that results from the sentences $x_{1:i}$. We write $I(\phi) \in \{T, F, ?\}$ for the value of the proposition ϕ in the information state I.
- A language model encoder E that maps sentences to sequences of d-dimensional word representations.

To characterize the encoding of semantic information in $E(x)$, we design a semantic **probe** that tries to recover the contents of I_i from $E(x_{1:i})$ proposition-by-proposition. Intuitively, this probe aims to answer three questions: (1) **How** is the truth value of a given proposition ϕ encoded? (Linearly? Nonlinearly? In what feature basis?) (2) **Where** is information about ϕ encoded? (Distributed across all token embeddings? Local to particular tokens?) (3) **How well** is semantic information encoded? (Can it be recovered better than chance? Perfectly?)

[4] An individual sentence is associated with a *context change potential*: a map from information states to information states.

The probe is built from three components, each of which corresponds to one of the questions above:

1. A **proposition embedder** $\texttt{embed} : L \to \mathbb{R}^d$ (where L is the set of logical propositions).
2. A **localizer** $\texttt{loc} : L \times \mathbb{R}^d \to \mathbb{R}^d$ which extracts and aggregates LM representations as candidates for encoding ϕ. The localizer extracts tokens of $E(x)$ at positions corresponding to particular tokens in the underlying text x. We express this in notation as $E(x)[*]$, where $*$ is a subsequence of x. (For example, if $x =$ "*the third beaker is empty*". $E(x) = [v_1, v_2, v_3, v_4, v_5]$ has one vector per token. $E(x)[\text{"}\textit{third beaker}\text{"}] = [v_2, v_3]$.)
3. A **classifier** $\texttt{cls}_\theta : \mathbb{R}^d \times \mathbb{R}^d \to \{T, F, ?\}$, which takes an embedded proposition and a localized embedding, and predicts the truth value of the proposition.

We say that a proposition ϕ is encoded by $E(x)$ if:

$$\texttt{cls}_\theta(\texttt{embed}(\phi), \texttt{loc}(\phi, E(x))) = I(\phi)\ . \quad (1)$$

Given a dataset of discourses $\mathcal{D}$, we attempt to find a classifier parameters θ from which all propositions can be recovered for all sentences in Eq. (1). To do so, we label each with the truth/falsehood of every relevant proposition. We then train the parameters of a $\texttt{cls}_\theta$ on a subset of these propositions and test whether it generalizes to held-out discourses.

4 Experiments

Our experiments aim to discover to what extent (and in what manner) information states are encoded in NLM representations. We first present a specific instantiation of the probe that allows us to determine how well information states are encoded in two NLMs and two datasets (§4.2); then provide a more detailed look at *where* specific propositions are encoded by varying `loc` (§4.3). Finally, we describe an experiment investigating the causal role of semantic representations by directly manipulating $E(x)$ (§4.4).[5]

4.1 Preliminaries

Model In all experiments, the encoder E comes from a BART (Lewis et al., 2020) or T5 (Raffel et al., 2020) model. Except where noted, BART is pretrained on OpenWebText, BookCorpus, CC-News, and Stories (Lewis et al., 2020), T5 is pretrained on C4 (Raffel et al., 2020), and both are fine-tuned on the TextWorld or Alchemy datasets described below. Weights of E are frozen during probe training.

Data: Alchemy Alchemy, the first dataset used in our experiments, is derived from the SCONE (Long et al., 2016) semantic parsing tasks. We preserve the train / development split from the original dataset (3657 train / 245 development). Every example in the dataset consists of a human-generated sequence of instructions to drain, pour, or mix a beaker full of colored liquid. Each instruction is annotated with the ground-truth state that results from following that instruction (Figure 3).

We turn Alchemy into a language modeling dataset by prepending a declaration of the initial state (the initial contents of each beaker) to the actions. The initial state declaration always follows a fixed form ("*the first beaker has* [amount] [color]*, the second beaker has* [amount] [color]*, ...*"). Including it in the context provides enough information that it is (in principle) possible to deterministically compute the contents of each beaker after each instruction. The NLM is trained to predict the next instruction based on a textual description of the initial state and previous instructions.

The state representations we probe for in Alchemy describe the contents of each beaker. Because execution is deterministic and the initial state is fully specified, the information state associated with each instruction prefix consists of only a single possible situation, defined by a set of propositions:

$$\begin{aligned}\Phi = \{&\texttt{has-}v\texttt{-}c(b) : \\ &b \in \{\texttt{beaker 1}, \texttt{beaker 2}, \ldots\}, \\ &v \in 1..4, \\ &c \in \{\texttt{red}, \texttt{orange}, \texttt{yellow}, \ldots\}\} \,.\end{aligned} \tag{2}$$

In the experiments below, it will be useful to have access to a natural language representation of each proposition. We denote this:

$$\texttt{NL}(\texttt{has-}v\texttt{-}c(b)) = \text{“}\textit{the } b \textit{ beaker has } v \textit{ } c\text{”} \,. \tag{3}$$

Truth values for each proposition in each instruction sequence are straightforwardly derived from ground-truth state annotations in the dataset.

Data: TextWorld TextWorld (Côté et al., 2018) is a platform for generating synthetic worlds for text-based games, used to test RL agents. The game generator produces rooms containing objects, surfaces, and containers, which the agent can interact with in various predefined ways.

We turn TextWorld into a language modeling dataset by generating random game rollouts following the "simple game" challenge, which samples world states with a fixed room layout but changing object configurations. For training, we sample 4000 rollouts across 79 worlds, and for development, we sample 500 rollouts across 9 worlds. Contexts begin with a description of the room that the player currently stands in, and all visible objects in that room. This is followed by a series of actions (preceded by >) and game responses (Fig. 3).

The NLM is trained to generate both an action and a game response from a history of interactions.

We probe for both the properties of and relations between entities at the end of a sequence of actions. Unlike Alchemy, these may be undetermined, as the agent may not have explored the entire environment by the end of an action sequence. (For example, in Fig. 3, the truth value of `matches(old key, door)` is *unknown*). The set of propositions available in the TextWorld domain has form

$$\begin{aligned}\Phi = &\{p(o) : o \in O, p \in P\} \\ &\cup \{r(o_1, o_2) : o_1, o_2 \in O, r \in R\}\end{aligned} \tag{4}$$

for objects $O = \{\texttt{player}, \texttt{chest}, \ldots\}$, properties $P = \{\texttt{open}, \texttt{edible}, \ldots\}$ and relations $R =$

[5] Sections here are also discussed in more detail in Appendix A.1 (for §4.1), A.2 (for §4.2), and A.3 (for §4.3).

		Alchemy				TextWorld			
		State EM		Entity EM		State EM		Entity EM	
		BART	T5	BART	T5	BART	T5	BART	T5
main probe (§4.2)		7.6	14.3	75.0	75.5	48.7	53.8	95.2	96.9
baselines & model ablations (§4.2)	+pretrain, -fine-tune	1.1	4.3	69.3	74.1	23.2	38.9	91.1	94.3
	-pretrain, +fine-tune	1.5		62.8		14.4		81.2	
	random init.	0.4		64.9		11.3		74.5	
	no change	0.0		62.7		9.73		74.1	
	no LM	0.0		32.4		1.77		81.8	
locality (§4.3)	first	-		-		49.6	51.5	93.6	95.9
	last	-		-		55.1	58.6	96.5	97.6

Table 1: Probing results. For each dataset, we report *Entity EM*, the % of entities for which all propositions were correct, and *State EM*, the % of states for which all proposition were correct. For non-pretrained baselines (*-pretrain, +fine-tune* and *random init.*), we report the single best result from all model configurations examined. Semantic state information can be recovered at the entity level from both language models on both datasets, and successful state modeling appears to be primarily attributable to pretraining rather than fine-tuning.

$\{\texttt{on}, \texttt{in}, \ldots\}$. We convert propositions to natural language descriptions as:

$$\begin{aligned} \texttt{NL}(p(o)) &= \text{“}\textit{the } p \textit{ is } o\text{”} \\ \texttt{NL}(r(o_1, o_2)) &= \text{“}\textit{the } o_1 \textit{ is } r \textit{ } o_2\text{”} \ . \end{aligned} \tag{5}$$

The set of propositions and their natural language descriptions are pre-defined by TextWorld's simulation engine. The simulation engine also gives us the set of true propositions, from which we can compute the set of false and unknown propositions.

Evaluation We evaluate probes according to two metrics. **Entity Exact-Match (EM)** first aggregates the propositions by *entity* or *entity pair*, then counts the percentage of entities for which *all* propositions were correctly labeled. **State EM** aggregates propositions by *information state* (i.e. context), then counts the percentage of states for which all facts were correctly labeled.

4.2 Representations encode entities' final properties and relations

With this setup in place, we are ready to ask our first question: is semantic state information encoded at all by pretrained LMs fine-tuned on Alchemy and TextWorld? We instantiate the probing experiment defined in §3 as follows:

The **proposition embedder** converts each proposition $\phi \in \Phi$ to its natural language description, embeds it using the same LM encoder that is being probed, then averages the tokens:

$$\texttt{embed}(\phi) = \texttt{mean}(E(\texttt{NL}(\phi))) \tag{6}$$

The **localizer** associates each proposition ϕ with specific tokens corresponding to the entity or entities that ϕ describes, then averages these tokens. In Alchemy, we average over tokens in the *initial description* of the beaker in question. For example, let x be the discourse in Figure 3 (left) and ϕ be a proposition about the first beaker. Then, e.g.,

$$\begin{aligned} &\texttt{loc}(\texttt{has-1-red(beaker 1)}, E(x)) = \\ &\quad \texttt{mean}(E(x)[\textit{The first beaker has 2 green,}]). \end{aligned} \tag{7}$$

In TextWorld, we average over tokens in *all mentions* of each entity. Letting x be the discourse in Figure 3 (right), we have:

$$\begin{aligned} &\texttt{loc}(\texttt{locked(wooden door)}, E(x)) = \\ &\quad \texttt{mean}(E(x)[\textit{wooden door}]) \ . \end{aligned} \tag{8}$$

Relations, with two arguments, are localized by taking the mean of the two mentions.

Finally, the **classifier** is a *linear model* which maps each NLM representation and proposition to a truth value. In Alchemy, a linear transformation is applied to the NLM representation, and then the proposition with the maximum dot product with that vector is labelled T (the rest are labelled F). In TextWorld, a bilinear transformation maps each (proposition embedding, NLM representation) pair to a distribution over $\{T, F, ?\}$.

As noted by Liang and Potts (2015), it is easy to construct examples of semantic judgments that cannot be expressed as linear functions of purely syntactic sentence representations. We expect (and verify with ablation experiments) that this probe is not expressive enough to compute information states directly from surface forms, and *only* expressive enough to read out state information already computed by the underlying LM.

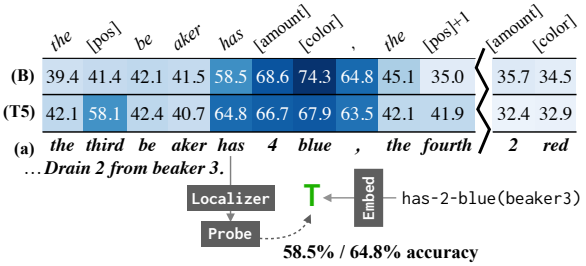

Figure 4: Locality of information state in Alchemy. We focus on the initial state declaration. Linear probes are trained to decode the final state of a beaker conditioned on the individual contextualized representations for each word. Separate probes are trained for each position. Tokens in the same relative position (with respect to the target beaker) are superimposed and the averaged entity EM is reported in (B) for BART and (T5) for T5. (a) shows the paradigm on a concrete example.

Results Results are shown in Table 1. A probe on T5 can exactly recover 14.3% of information states in Alchemy, and 53.8% in TextWorld. For context, we compare to two **baselines**: a *no LM* baseline, which simply predicts the most frequent final state for each entity, and a *no change* baseline, which predicts that the entity's final state in the discourse will be the same as its initial state. The *no LM* baseline is correct 0% / 1.8% of the time and the *no change* baseline is correct 0% / 9.7% of the time—substantially lower than the main probe.

To verify that this predictability is a property of the NLM representations rather than the text itself, we apply our probe to a series of **model ablations**. First, we evaluate a *randomly initialized* transformer rather than the pretrained and fine-tuned model, which has much lower probe accuracy. To determine whether the advantage is conferred by LM pretraining or fine-tuning, we ablate either open-domain pretraining, in a *-pretrain,+fine-tune* ablation, or in-domain fine-tuning, in a *+pretrain,-fine-tune* ablation. (For all experiments *not* using a pretrained model checkpoint, we experimented with both a BART-like and T5-like choice of depth and hidden size, and found that the BART-like model performed better.) While both fine-tuning and pretraining contribute to the final probe accuracy, pretraining appears to play a much larger role: semantic state can be recovered well from models with no in-domain fine-tuning.

Finally, we note that there may be lexical overlap between the discourse and natural language descriptions of propositions. How much of the probe's performance can be attributed to this overlap? In Alchemy, the *no change* baseline (which performs much worse than our probe) also acts as a lexical overlap baseline—there will be lexical overlap between true propositions and the initial state declaration only if the beaker state is unchanged. In TextWorld, each action induces multiple updates, but can at most overlap with one of its affected propositions (e.g. *You close the chest* causes `closed(chest)` and ¬`open(chest)`, but only overlaps with the former). Moreover, only ∼50% of actions have lexical overlap with any propositions at all. Thus, lexical overlap cannot fully explain probe performance in either domain.

	State EM		Entity EM	
	BART	T5	BART	T5
remap	50.2	50.4	88.9	93.2
main probe	50.2	53.8	91.3	94.6

Table 2: Locality of information state in TextWorld (T5). Entity state information tends to be slightly more present in mentions of the target entity (main probe) rather than of other entities (remap), but not by much.

In summary, *pretrained NLM representations model state changes and encode semantic information about entities' final states*.

4.3 Representations of entities are local to entity mentions

The experiment in §4.2 assumed that entity state could be recovered from a fixed set of input tokens. Next, we conduct a more detailed investigation into *where* state information is localized. To this end, we ask two questions: first, can we assume state information is localized in the corresponding entity mentions, and second, if so, *which* mention encodes the most information, and what kind of information does it encode?

4.3.1 Mentions or other tokens?

We first contrast tokens *within* mentions of the target entity to tokens *elsewhere* in the input discourse. In Alchemy, each beaker b's initial state declaration is tokenized as: $\texttt{toks}_b = \{the_b,$ [position]$_b$, be_b, $aker_b$, has_b, [volume]$_b$, [color]$_b$, $,_b\}$, where b signifies the beaker position. Rather than pooling these tokens together (as in §4.2), we construct a **localizer ablation** that associates beaker b's state with *single tokens* t in either the initial mention of beaker b, or the initial mention of *other beakers* at an integer offset Δ. For each (t, Δ) pair, we construct a localizer that matches propositions about beaker b with $t_{b+\Delta}$. For example, the $(has, +1)$ localizer associates the *third* beaker's final state

with the vector in $E(x)$ at the position of the "has" token in *the fourth beaker has 2 red*.

In TextWorld, which does not have such easily categorizable tokens, we investigate whether information about the state of an entity is encoded in mentions of *different entities*. We sample a random mapping `remap` between entities, and construct a localizer ablation in which we decode propositions about w from mentions of $\texttt{remap}(w)$. For example, we probe the value of `open(chest)` from mentions of *old key*. These experiments use a different evaluation set—we restrict evaluation to the subset of entities for which *both* w and $\texttt{remap}(w)$ appear in the discourse. For comparability, we re-run the main probe on this restricted set.[6]

Results Fig. 4 shows the locality of BART and T5 in the Alchemy domain. Entity EM is highest for words corresponding to the correct beaker, and specifically for color words. Decoding from any token of an incorrect beaker barely outperforms the *no LM* baseline (32.4% entity EM). In TextWorld, Table 2 shows that decoding from a remapped entity is only 1-3% worse than decoding from the right one. Thus, *the state of an entity e is (roughly) localized to tokens in mentions of e*, though the degree of locality is data- and model-dependent.

4.3.2 Which mention?

To investigate facts encoded in different mentions of the entity in question, we experiment with decoding from the first and last mentions of the entities in x. The form of the localizer is the same as 4.2, except instead of averaging across all mentions of entities, we use the first mention or the last mention. We also ask whether *relational* propositions can be decoded from just one argument (e.g., `in(old key, chest)` from just mentions of *old key*, rather than the averaged encodings of *old key* and *chest*).

Results As shown in Table 1, in TextWorld, probing the *last* mention gives the highest accuracy. Furthermore, as Table 3 shows, *relational facts can be decoded from either side of the relation*.

	Relations		Properties	
	BART	T5	BART	T5
1-arg	41.4	55.5	83.2	90.9
2-arg	49.6	54.4	94.5	98.5
random init.	21.9		35.4	

Table 3: State EM for decoding each type of fact (relations vs. properties), with each type of probe (1- vs. 2- argument decoding). Though decoding from two entities is broadly better, one entity still contains a nontrivial amount of information, even regarding relations.

4.4 Changes to entity representations cause changes in language model predictions

The localization experiments in Section 4.3 indicate that state information is localized within contextual representations in predictable ways. This suggests that modifying the representations themselves could induce systematic and predictable changes in model behavior. We conduct a series of causal intervention experiments in the Alchemy domain which measure effect of *manipulating encoder representations* on NLM output. We *replace* a small subset of token representations with those from a different information state, and show that this causes the model to behave as if it were in the new information state.[7]

A diagram of the procedure is shown in Fig. 5. We create two discourses, x_1 and x_2, in which *one* beaker's final volume is zero. Both discourses describe the same initial state, but for each x_i, we append the sentence *drain v_i from beaker b_i*, where v_i is the initial volume of beaker b_i's contents. Though the underlying initial state tokens are the same, we expect the contextualized representation $C_1 = E(x_1)[\textit{the ith beaker}\ldots]$ to differ from $C_2 = E(x_2)[\textit{the ith beaker}\ldots]$ due to the different final states of the beakers. Let $\textsc{cont}(x)$ denote the set of sentences constituting *semantically acceptable continuations* of a discourse prefix x. (In Fig. 1, $\textsc{cont}$(a, b) contains c_1 and c_2 but not c_3.)[8] In Alchemy, $\textsc{cont}(x_1)$ should not contain mixing, draining, or pouring actions involving b_1 (similarly for $\textsc{cont}(x_2)$ and b_2). Decoder samples given C_i should fall into $\textsc{cont}(x_i)$.

Finally, we *replace the encoded description of beaker 2 in C_1 with its encoding from C_2*, creating a new representation C_{mix}. C_{mix} was not derived from any real input text, but implicitly represents a situation in which *both* b_1 and b_2 are empty. A decoder generating from C_{mix} should generate instructions in $\textsc{cont}(x_1) \cap \textsc{cont}(x_2)$ to be consistent with this situation.

[6] The `remap` and $\Delta \neq 0$ probes described here are analogous to *control tasks* (Hewitt and Liang, 2019). They measure probes' abilities to predict labels that are structurally similar but semantically unrelated to the phenomenon of interest.

[7] This experiment is inspired by Geiger et al. (2020).

[8] In order to automate evaluation of consistency, we use a version of Alchemy with synthetically generated text. The underlying LM has also been fine-tuned on synthetic data.

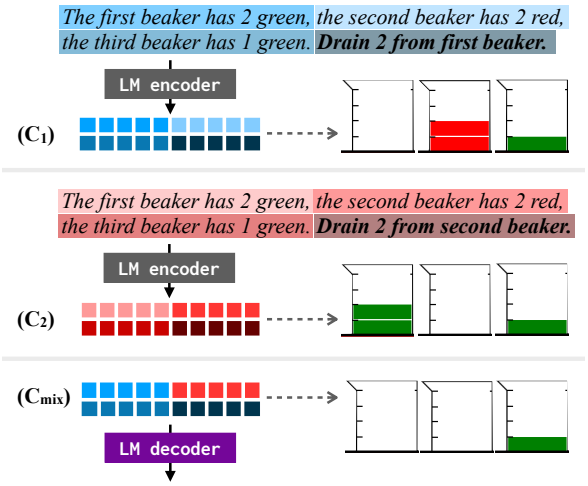

Figure 5: Intervention experiments. Construct C_1, C_2 by appending text to empty one of the beakers (e.g. the first and second beakers) and encoding the result. Then, create C_{mix} by taking encoded tokens from C_1 and replacing the encodings corresponding to the second beaker's initial state declaration with those from C_2. This induces the LM to model both the first and second beakers as empty, and the LM decoder should generate actions consistent with this state.

	% of generations within...					
	$\text{CONT}(x_1) \cap \text{CONT}(x_2)$		$\text{CONT}(x_1)$		$\text{CONT}(x_2)$	
	BART	T5	BART	T5	BART	T5
C_1	20.4	37.9	**96.2**	**93.0**	21.6	40.8
C_2	16.1	29.1	24.1	37.9	**87.7**	**87.2**
C_{mix}	**57.7**	**75.4**	86.7	86.8	64.8	84.5

Table 4: Results of intervention experiments. Though imperfect, generations from C_{mix} are more often consistent with both contexts compared to those from C_1 or C_2, indicating that its underlying information state (approximately) models both beakers as empty.

Results We generate instructions conditioned on C_{mix} and check whether they are in the expected sets. Results, shown in Table 4, align with this prediction. For both BART and T5, substantially more generations from C_{mix} fall within $\text{CONT}(x_1) \cap \text{CONT}(x_2)$ than from C_1 or C_2. Though imperfect (compared to C_1 generations within $\text{CONT}(x_1)$ and C_2 generations within $\text{CONT}(x_2)$), this suggests that the information state associated with the synthetic encoding C_{mix} is (approximately) one in which *both* beakers are empty.

5 Limitations

...of large NLMs: It is important to emphasize that both LM output and implicit state representations are imperfect: even in the best case, complete information states can only be recovered 53.8% of the time in tasks that most humans would find very simple. (Additional experiments described in Appendix A.5 offer more detail about these errors.) The success of our probing experiments should not be taken to indicate that the discovered semantic representations have anything near the expressiveness needed to support human-like generation.

...of our experimental paradigm: While our probing experiments in §4.2 provide a detailed picture of structured state representations in NLMs, the intervention experiments in §4.4 explain the relationship between these state representations and model behavior in only a very general sense. They leave open the key question of whether *errors* in language model prediction are attributable to errors in the underlying state representation. Finally, the situations we model here are extremely simple, featuring just a handful of objects. Thought experiments on the theoretical capabilities of NLMs (e.g. Bender and Koller's "coconut catapult") involve far richer worlds and more complex interactions. Again, we leave for future work the question of whether current models can learn to represent them.

6 Conclusion

Even when trained only on language data, NLMs encode simple representations of meaning. In experiments on two domains, internal representations of text produced by two pretrained language models can be mapped, using a linear probe, to representations of the state of the world described by the text. These internal representations are structured, interpretably localized, and editable. This finding has important implications for research aimed at improving factuality and and coherence in NLMs: future work might probe LMs for the the states and properties ascribed to entities the first time they are mentioned (which may reveal biases learned from training data; Bender et al. 2021), or correct errors in generation by directly editing representations.

Acknowledgments

Thanks to Ekin Akyürek, Evan Hernandez, Joe O'Connor, and the anonymous reviewers for feedback on early versions of this paper. MN is supported by a NSF Graduate Research Fellowship. This work was supported by a hardware donation from NVIDIA under the NVAIL program.

Impact Statement

This paper investigates the extent to which neural language models build meaning representations of the world, and introduces a method to probe and modify the underlying information state. We expect this can be applied to improve factuality, coherence, and reduce bias and toxicity in language model generations. Moreover, deeper insight into how neural language models work and what exactly they encode can be important when deploying these models in real-world settings.

However, interpretability research is by nature dual-use and improve the effectiveness of models for generating false, misleading, or abusive language. Even when not deliberately tailored to generation of harmful language, learned semantic representations might not accurately represent the world because of errors both in prediction (as discussed in §5) and in training data.

References

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623.

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.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Ruo Yu Tao, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. 2018. Textworld: A learning environment for text-based games. *CoRR*, abs/1806.11532.

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.

Arash Einolghozati, Panupong Pasupat, Sonal Gupta, Rushin Shah, Mrinal Mohit, Mike Lewis, and Luke Zettlemoyer. 2019. Improving semantic parsing for task oriented dialog.

Atticus Geiger, Kyle Richardson, and Christopher Potts. 2020. Neural natural language inference models partially embed theories of lexical entailment and negation. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 163–173, Online. Association for Computational Linguistics.

Jeroen Groenendijk and Martin Stokhof. 1991. Dynamic predicate logic. *Linguistics and Philosophy*, 14:39–100.

Irene Heim. 2008. *File Change Semantics and the Familiarity Theory of Definiteness*, pages 223 – 248.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. 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 2733–2743, Hong Kong, China. Association for Computational Linguistics.

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.

Felix Hill, Sona Mokra, Nathaniel Wong, and Tim Harley. 2020. Human instruction-following with deep reinforcement learning via transfer-learning from text.

Gabriel Ilharco, Rowan Zellers, Ali Farhadi, and Hannaneh Hajishirzi. 2021. Probing contextual language models for common ground with visual representations. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5367–5377, Online. Association for Computational Linguistics.

Hans Kamp, Josef Genabith, and Uwe Reyle. 2010. *Discourse Representation Theory*, pages 125–394.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. 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 4365–4374, Hong Kong, China. Association for Computational Linguistics.

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.

Percy Liang and Christopher Potts. 2015. Bringing machine learning and compositional semantics together. *Annu. Rev. Linguist.*, 1(1):355–376.

Reginald Long, Panupong Pasupat, and Percy Liang. 2016. Simpler context-dependent logical forms via model projections. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1456–1465, Berlin, Germany. Association for Computational Linguistics.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia. 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, New Orleans, Louisiana. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? 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 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Tiago Pimentel, Naomi Saphra, Adina Williams, and Ryan Cotterell. 2020. Pareto probing: Trading off accuracy for complexity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3138–3153, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural MT learn source syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1526–1534, Austin, Texas. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. 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.

Gregor Wiedemann, Steffen Remus, Avi Chawla, and Chris Biemann. 2019. Does BERT make any sense? Interpretable word sense disambiguation with contextualized embeddings.

Zhaofeng Wu, Hao Peng, and Noah A. Smith. 2021. Infusing Finetuning with Semantic Dependencies. *Transactions of the Association for Computational Linguistics*, 9:226–242.

Seth Yalcin. 2014. Introductory notes on dynamic semantics.

A Appendix

A.1 Datasets Details (§4.1)

Alchemy Alchemy is downloadable at https://nlp.stanford.edu/projects/scone/. Alchemy propositions are straightforwardly derived from existing labels in the dataset. We preserve the train/dev split from the original dataset (3657 train/245 dev), which we use for training the underlying LM *and* the probe. In subsequent sections, we include additional results from a synthetic dataset that we generated (3600 train/500 dev), where actions are created following a fixed template, making it easy to evaluate consistency.

Textworld We generate a set of worlds for training, and a separate set of worlds for testing. We obtain transcripts from three agents playing on each game: a perfect agent and two (semi-)random agents, which intersperse perfect actions with several steps of random actions. For training, we sample 4000 sequences from the 3 agents across 79 worlds. For development, we sample 500 sequences from the 3 agents across 9 worlds.

During game generation, we are given the set of all propositions that are *True* in the world, and how the set updates after each player action. However, the player cannot infer the *full* state before interacting with and seeing all objects, and neither (we suspect) can a language model trained on partial transcripts. For example, a player that starts in the bedroom cannot infer `is-in(refrigerator, kitchen)` without first entering the kitchen. One solution would be to hard-code rules–a player can only know about the states of entities it has directly affected or seen. However, since synthetically-generated worlds might share some commonalities, a player that has played many games before (or an LM trained on many transcripts) might be able to draw particular conclusions about entities in unseen worlds, even before interacting with them.

To deal with these factors, we train a **labeller** model `label` to help us classify propositions as *known true*, *known false*, and *unknown*. We generate a training set (separate from the training set for the probe and probed LM) to train the labeller. We again use BART, but we give it the text transcripts and train it to directly decode the full set of True propositions and False proposition by the end of the transcript (recall we have the ground-truth full True set, and we label all propositions that aren't in the True set as False). This allows the labeller model to pick up both patterns *between the discourse and its information state*, as well as infer *general patterns among various discourses*. Thus, on unknown worlds, given text T, if proposition A is True most or all of the time given T, the model should be confident in predicting A. We label A as True in these cases. However, if proposition A is True only half of the time given T, the model is unconfident. We label A as Unknown in these cases. Thus, we create our *unknown* set using a confidence threshold τ on `label`'s output probability.

A.2 Probe Details + Additional Results (§4.2)

Below, we give a more detailed account of our probing paradigm in each domain, including equations.

Alchemy Probe The proposition embedder converts propositions ϕ to natural language descriptions $\widetilde{\phi}$ ("`the` b`th beaker has` v c") and encodes them with the BART or T5 LM encoder.

Given a proposition `has-`v`-`$c(b)$, the localizer `loc` maps `has-`v`-`$c(b)$ to tokens in $E(x)$ that corresponding to the initial state of beaker b. Since x always begins with an initial state declaration of the form "*the first beaker has [amount] [color], the second beaker has [amount] [color], ...*", tokens at position $8b-8 \cdots 8b-1$ of x correspond to the initial state of beaker b. (Each state has 8 tokens: 'the', 'bth', 'be', 'aker', 'has', '[amount]', '[color]', ','). Thus,

$$\begin{aligned}&\texttt{loc}(\texttt{has-}v\texttt{-}c(b), E(x)) = \\ &\quad \texttt{mean}(E(x)[8b-8], \cdots, E(x)[8b-1])\end{aligned}$$

We train a linear probe $\texttt{cls}_\theta$ to predict the final beaker state given the encoded representation $E(x)$ of text x. For our probe, we learn linear projection weights $W^{(d\times d)}$ and bias $b^{(d)}$ to maximize the dot product between the LM representation and the embedded proposition. Formally, it computes $v_b^{(max)}, c_b^{(max)}$ as

$$\begin{aligned}v_b^{(max)}, c_b^{(max)} = \arg\max_{v',c'} \big(&\texttt{embed}(\texttt{has-}v'\texttt{-}c'(b))\cdot \\ &(W \cdot \texttt{loc}(\texttt{has-}v'\texttt{-}c'(b), E(x)) + b)\big)\end{aligned} \tag{9}$$

In other words, $v_b^{(max)}, c_b^{(max)}$ are the values of v and c that maximize this dot product. The probe then returns

$$\begin{aligned}&\texttt{cls}_\theta(\texttt{embed}(\texttt{has-}v\texttt{-}c(b)), \texttt{loc}(\texttt{has-}v\texttt{-}c(b), E(x))) \\ &\qquad = \begin{cases} T \text{ if } v, c = v_b^{(max)}, c_b^{(max)} \\ F \text{ if } v, c \neq v_b^{(max)}, c_b^{(max)} \end{cases}\end{aligned} \tag{10}$$

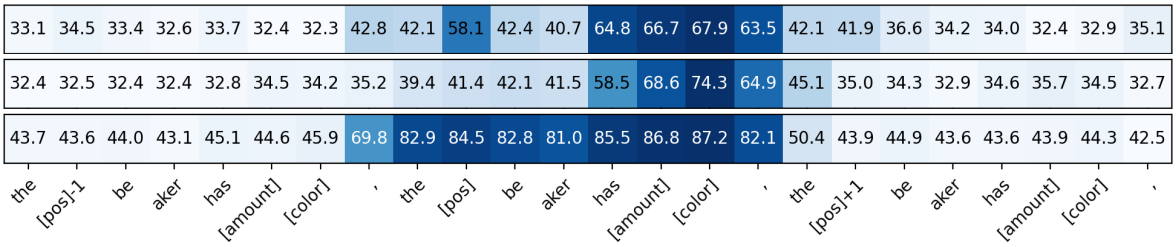

Figure 6: Alchemy locality - full results. **Top**: T5, Finetuned+probed on real data. **Middle**: BART, Finetuned+probed on real data. **Bottom**: BART, Finetuned+probed on synthetic data. We note that for the synthetic data, accurate decoding is possible from a much wider set of tokens, but all still correspond to the relevant beaker.

Note that $\texttt{cls}_\theta$ selects the optimal final state *per beaker*, from the set of all possible states of beaker b, taking advantage of the fact that only one proposition can be true per beaker.

Textworld Probe For Textworld, the proposition embedder converts propositions ϕ to natural language descriptions $\tilde{\phi}$ ("`the` o `is` p" for properties and "`the` o_1 `is` r o_2" for relations) and encodes them with the BART or T5 LM encoder.

Given a proposition $p(o)$ pertaining to an entity or $r(o_1, o_2)$ pertaining to an entity pair, we define localizer `loc` to map the proposition to tokens of $E(x)$ corresponding to *all mentions* of its arguments, and averages across those tokens:

$$\begin{aligned}\texttt{all\_idx}(e) &= \text{set of indices of } x \text{ correspond} \\ &\quad \text{-ing to all instances of } e \\ \texttt{loc}(r(o_1,o_2),E(x)) &= \texttt{mean}\big(E(x)[\texttt{all\_idx}(o_1) \cup \\ &\quad \texttt{all\_idx}(o_2)]\big) \\ \texttt{loc}(p(o),E(x)) &= \texttt{mean}\left(E(x)[\texttt{all\_idx}(o)]\right)\end{aligned} \tag{11}$$

We train a bilinear probe $\texttt{cls}_\theta$ that classifies each (proposition embedding, LM representation) pair to $\{T, F, ?\}$. The probe has parameters $W^{(3\times d\times d)}, b^{(3)}$ and performs the following bilinear operation:

$$\texttt{scr}(\phi, E(x)) = \texttt{embed}(\phi)^T \cdot W \cdot \texttt{loc}(\phi, E(x)) + b \tag{12}$$

where `scr` is a vector of size 3, with one score per $T, F, ?$ label. The probe then takes the highest-scoring label

$$\texttt{cls}_\theta(\texttt{embed}(\phi), \texttt{loc}(\phi, E(x))) = \begin{cases} T \text{ if } \texttt{scr}(\phi,E(x))[0] > \texttt{scr}(\phi,E(x))[1], \texttt{scr}(\phi,E(x))[2] \\ F \text{ if } \texttt{scr}(\phi,E(x))[1] > \texttt{scr}(\phi,E(x))[2], \texttt{scr}(\phi,E(x))[0] \\ ? \text{ if } \texttt{scr}(\phi,E(x))[2] > \texttt{scr}(\phi,E(x))[0], \texttt{scr}(\phi,E(x))[1] \end{cases} \tag{13}$$

A.3 Localization Experiment Details + Additional Results (§4.3)

Below we provide a specific, formulaic account of each of our localizer experiments.

Mentions vs. Other Tokens (§4.3.1) – Alchemy Recall that we train a probe for each (t, Δ) pair to extract propositions about b from token $t_{b+\Delta} \in \texttt{toks}_{b+\Delta}$, where Δ is the beaker offset. Specifically, the localizer for this probe takes form

$$\begin{aligned}&\texttt{off}: \\ &\quad \{\textit{'the'} \to 0, \textit{[position]} \to 1, \textit{'be'} \to 2, \textit{'aker'} \to 3, \\ &\quad \textit{'has'} \to 4, \textit{[amount]} \to 5, \textit{[color]} \to 6, \textit{','} \to 7\} \\ &\texttt{loc}_{(t,\Delta)}(\texttt{has-}v\texttt{-}c(b, E(x))) = \\ &\quad \texttt{mean}(E(x)[8(b+\Delta) - 8 + \texttt{off}(t)])\end{aligned}$$

The full token-wise results for beaker states in a 3-beaker (24-token) window around the target beaker is shown in Figure 6 (Top/Middle).

Additional localizer ablations results for a BART probe trained and evaluated on synthetic Alchemy data are shown in Figures 6 (Bottom). Similar to the non-synthetic experiments, we point the localizer to just a single token of the initial state. Interestingly, BART's distribution looks very different in the synthetic setting. Though state information is still local to the initial state description of the target beaker, it is far more distributed within the description–concentrated in not just the amount and color tokens, but also the mention tokens.

Mentions vs. Other Tokens (§4.3.1) – Textworld The specific localizer for this experiment has form

$$\begin{aligned}&\texttt{loc}(r(o_1,o_2),E(x)) = \\ &\quad \texttt{mean}(E(x)[\texttt{all\_idx}(\texttt{remap}(o_1)) \cup \\ &\qquad \texttt{all\_idx}(\texttt{remap}(o_2))]) \\ &\texttt{loc}(p(o),E(x)) = \texttt{mean}(E(x)[\texttt{all\_idx}(\texttt{remap}(o))])\end{aligned}$$

Note the evaluation set for this experiment is slightly different as we exclude contexts which do not mention $\texttt{remap}(w)$.

	Alchemy	
	Entity EM	State EM
main probe (§4.2)	75.0	7.55
human-grounded-features (§A.4)	45.7	0.71
synthetic data	88.2	35.9

Table 5: Additional Alchemy results. We compare our full encoded-NL embedder with a featurized embedder (§A.4). We also report results on synthetic data.

.

Which Mention? (§4.3.2) – first/last The localizer for this experiment is constructed by replacing all instances of `all_idx` in Eq. 10 with either `first_idx` or `last_idx`, defined as:

$$\texttt{first\_idx}(e) = \text{set of indices of } x \text{ corresponding to first instance of } e$$
$$\texttt{last\_idx}(e) = \text{set of indices of } x \text{ corresponding to last instance of } e$$

Which Mention? (§4.3.2) – single- vs. both-entity probe. The specific localizer for the single-entity probe has form

$$\texttt{loc}(r(o_1, o_2), E(x)) = \left\{ \begin{matrix} \texttt{mean}(E(x)[\texttt{all\_idx}(o_1)]), \\ \texttt{mean}(E(x)[\texttt{all\_idx}(o_2)]) \end{matrix} \right\}$$
$$\texttt{loc}(p(o), E(x)) = \texttt{mean}(E(x)[\texttt{all\_idx}(o)])$$

Note the localizer returns a 2-element *set* of encodings from each relation. We train the probe to decode $r(o_1, o_2)$ from *both* elements of this set.

The full results are in Table 3. As shown, the both-mentions probe is slightly better at both decoding relations and properties. However, this may simply be due to having less candidate propositions per entity pair, than per entity (which includes relations from *every other* entity paired with this entity). For example, entity pair *(apple, chest)* has only three possibilities: `in(apple, chest)` is True/Unknown/False, while singular entity *(chest)* has much more: `in(apple, chest)`, `in(key, chest)`, `open(chest)`, etc. can each be True/Unknown/False. A full set of results broken down by property/relation can be found in Table 6. Overall, the single-entity probe outperforms all baselines, suggesting that each entity encoding contains information about its relation with other entities.

A.4 Proposition Embedder Ablations

We experiment with a featurized `embed` function in the Alchemy domain. Recall from Section 4.2 and A.2 that our main probe uses encoded natural-language assertions of the state of each beaker (Eq. 6). We experiment with featurized vector where each beaker proposition is the concatenation of a 1-hot vector for beaker position and a sparse vector encoding the amount of each color in the beaker (with 1 position per color). For example, if there are 2 beakers and 3 colors `[green,red,brown]`, `has-3-red(2)` is represented as $[0, 1, 0, 3, 0]$. A multi-layer perceptron is used as the `embed` function to map this featurized representation into a dense vector, which is used in the probe as described by Eq. 10. In this setting, the `embed` MLP is optimized jointly with the probe.

Results are shown in Table 5. Using a featurized representation (45.7) is significantly worse than using an encoded natural-language representation (75.0), suggesting that the form of the fact embedding function is important. In particular, *the encoding is linear in sentence-embedding space, but nonlinear in human-grounded-feature space.*

A.5 Error Analysis

We run error analysis on the BART model. For the analysis below, it is important to note that we make no distinction between *probe errors* and *representation errors*—we do not know whether the errors are attributable to the linear probe's lack of expressive power, or whether the underlying LM indeed does fail to capture certain phenomena. We note that a *BART decoder* trained to decode the final information state from $E(x)$ is able to achieve 53.5% state EM on Alchemy (compared to 0% on random initialization baseline) whereas the linear decoder was only able to achieve 7.55% state EM—suggesting that certain state information may be non-linearly encoded in NLMs.

Alchemy The average number of incorrect beakers per sample is 25.0% (2.7 beakers out of 7).

We note that the distribution is skewed towards longer sequences of actions, where the % of wrong beakers increases from 11.3% (at 1 action) to 24.7% (2 actions), 30.4% (3 actions), 33.4% (at 4 actions). For beakers not acted upon (final state unchanged), the error rate is 13.3%. For beakers acted upon (final state changed), the error rate is 44.6%. Thus, errors are largely attributed to failures in reasoning about the effects of actions, rather than failures in decoding the initial state. (This is unsurprising, as in Alchemy, the initial state is explicitly written in the text—and we're decoding from those tokens).

For beakers that were predicted wrong, 36.8% were predicted to be its unchanged initial state and

	Overall	Relations	Properties	True Facts			False Facts		
	EM	EM	EM	Pre	Rec	F1	Pre	Rec	F1
	48.7	49.6	94.5	95.1	98.1	96.4	99.6	**98.9**	99.2
+pretrain, -finetune	23.2	32.7	75.4	93.2	94.9	93.8	97.0	96.1	96.4
-pretrain, +finetune	14.4	26.8	44.0	87.3	86.6	86.3	93.3	93.0	92.9
randomly initialized	11.3	21.9	35.4	91.5	83.8	87.2	91.8	84.1	86.5
no LM	1.77	24.8	33.4	88.3	80.9	84.4	88.8	86.9	87.6
no change	9.73	30.1	9.73	77.9	73.0	75.3	79.1	61.8	68.9
locality (first)	49.6	50.9	88.5	95.4	97.2	96.1	99.1	98.7	98.9
locality (last)	**55.1**	**56.6**	**96.5**	**96.1**	**98.7**	**97.3**	**99.7**	98.9	**99.3**

Table 6: TextWorld Probing. Metrics are reported on *whole state*. Precision is computed against *all* gold, ground-truth True facts in the state. Recall is computed against the `label`-model-generated True facts in the state. All numbers reported are averaged across all samples. Relations are overall much harder to probe than properties.

the remaining 63.2% were predicted to be empty — thus, probe mistakes are largely attributable to a tendency to over-predict empty beakers. This suggests that the downstream decoder may tend to generate actions too conservatively (as empty beakers cannot be acted upon). Correcting this could encourage LM generation diversity.

Finally, we examine what *type* of action tends to throw off the probe. When there is a *pouring* or *mixing*-type action present in the sequence, the model tends to do worse (25.3% error rate for drain-type vs. 31.4 and 33.3% for pour- and mix-type), though this is partially due to the higher concentration of drain actions in short action sequences.

Textworld Textworld results, broken down by properties/relations, are reported in Table 6. The probe seems to be especially bad at classifying relations, which make sense as relations are often expressed indirectly. A breakdown of error rate for each proposition type is shown in Table 7, where we report what % of that type of proposition was labelled incorrectly, each time it appeared. This table suggests that the probe consistently fails at decoding locational relations, i.e. failing to classify `east-of(kitchen,bedroom)` and `west-of(kitchen,bedroom)` as True, despite the layout of the simple domain being fixed. One hypothesis is that location information is made much less explicit in the text, and usually require reasoning across longer contexts and action sequences. For example, classifying `in(key, drawer)` as True simply requires looking at a single action: $>$ *put key in drawer*. However, classifying `east-of(kitchen,bedroom)` as True requires reasoning across the following context:

You are in the bedroom [. . .]

> go east
You enter the kitchen.

where the ellipses possibly encompass a long sequence of other actions.

Proposition Type	Error Rate
`{north|south|east|west}-of(A,B)`	11.8%
`is-on(A,B)`	6.17%
`is-in(A,B)`	1.20%
`locked(A)`	0.47%
`closed(A)`	0.35%
`eaten(A)`	0.049%
`open(A)`	0.039%

Table 7: Error rate per proposition type in Textworld.

A.6 Infrastructure and Reproducibility

We run all experiments on a single 32GB NVIDIA Tesla V100 GPU. On both Alchemy and Textworld, we train the language models to convergence, then train the probe for 20 epochs. In Alchemy and Textworld, both training the language model and the probe takes approximately a few (less than 5) hours. We probe BART-base, a 12-layer encoder-decoder Transformer model with 139M parameters, and T5-base, a 24-layer encoder-decoder Transformer model which has 220M parameters. Our probe itself is a linear model, with only two parameters (weights and bias).