Is "*My Favorite New Movie*" My Favorite Movie? Probing the Understanding of Recursive Noun Phrases

Qing Lyu[1] **Hua Zheng**[2] **Daoxin Li**[3] **Li Zhang**[1]
Marianna Apidianaki[1] **Chris Callison-Burch**[1]
[1]Department of Computer and Information Science, University of Pennsylvania
[2]Key Lab of Computational Linguistics (MOE), Peking University
[3]Department of Linguistics, University of Pennsylvania
{lyuqing,zharry,marapi,ccb}@seas.upenn.edu
zhenghua@pku.edu.cn
daoxinli@sas.upenn.edu

Abstract

Recursive noun phrases (NPs) have interesting semantic properties. For example, *my favorite new movie* is not necessarily my favorite movie, whereas *my new favorite movie* is. This is common sense to humans, yet it is unknown whether language models have such knowledge. We introduce the Recursive Noun Phrase Challenge (**RNPC**), a dataset of three textual inference tasks involving textual entailment and event plausibility comparison, precisely targeting the understanding of recursive NPs. When evaluated on RNPC, state-of-the-art Transformer models only perform around chance. Still, we show that such knowledge is learnable with appropriate data. We further probe the models for relevant linguistic features that can be learned from our tasks, including modifier semantic category and modifier scope. Finally, models trained on RNPC achieve strong zero-shot performance on an extrinsic Harm Detection evaluation task, showing the usefulness of the understanding of recursive NPs in downstream applications.[1]

1 Introduction

Recursion, the self-embedding of a linguistic structure, constitutes a fundamental property of human language. Due to its hierarchical structure, it poses many challenges to human language acquisition. One such challenge occurs in the context of recursive Noun Phrases (NPs), i.e., NPs with multiple prenominal modifiers. For instance, in Figure 1, when asked to point to *the second green ball* in a series of balls, children sometimes erroneously point to the second **and** green ball (intersective interpretation), instead of the second **among** green balls (recursive interpretation) (Matthei, 1982; Hamburger and Crain, 1984; Marcilese et al., 2013).

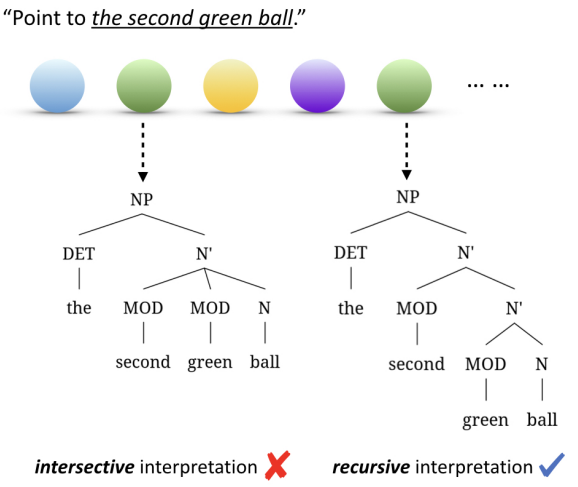

Figure 1: The intersective (incorrect) and the recursive (correct) interpretation of *the second green ball*.

We investigate whether language models (LMs) make similar errors, since the understanding of recursive NPs is also fundamental in real-world AI applications. For example, a summarization system should know that *the former US president* cannot be shortened as *the president*, since they are no longer in power. Also, a self-driving car asked to take *the first left-hand exit* should not assume that it is always the first exit.

Previous work has studied the syntactic parsing of recursive NPs (Nakov and Hearst, 2005; Pitler et al., 2010), as well as the semantic categorization of modifiers in NPs with only one prenominal modifier (Kamp and Partee, 1995; McCrae et al., 2014). However, neither parsing nor modifier categorization alone can sufficiently capture the meaning of recursive NPs (§2).

In this paper, using recursive NPs with two modifiers as our test-bed, we address the following questions about LMs' understanding of recursion:

(a) **Is the knowledge of how to interpret recursive NPs present in LMs (§5)?** We propose

[1]Our code and data are available at https://github.com/veronica320/Recursive-NPs.

Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5286 - 5302

Task	ID	Input	Label
Single-Premise Textual Entailment (SPTE)	(1a)	Premise: This is my **new favorite** movie. Hypothesis: This is my favorite movie.	Entailment
	(1b)	Premise: This is my **favorite new** movie. Hypothesis: This is my favorite movie.	Non-Entailment
Multi-Premise Textual Entailment (MPTE)	(2a)	Premise 1: He is a skillful American violinist. Premise 2: He is a father. Hypothesis: He is an **American** father.	Entailment
	(2b)	Premise 1: He is a skillful American violinist. Premise 2: He is a father. Hypothesis: He is a **skillful** father.	Non-Entailment
Event Plausibility Comparison (EPC)	(3a)	Event 1: The actress is known by everyone. Event 2: The famous former actress is known by everyone.	(Event 2 is) More Plausible
	(3b)	Event 1: The actress lives in France. Event 2: The famous former actress lives in France.	(Event 2 is) Equally Plausible
	(3c)	Event 1: The actress stars in many latest movies. Event 2: The famous former actress stars in many latest movies.	(Event 2 is) Less Plausible

Table 1: Examples for each task in our dataset. The NPs of interest are underlined. Differences between examples are in bold. See Section 3 for details.

the Recursive Noun Phrase Challenge (RNPC), a challenge set containing three classification tasks: Single-Premise Textual Entailment, Multi-Premise Textual Entailment, and Event Plausibility Comparison (§3). Table 1 provides examples for each task. Results show that state-of-the-art (SOTA) LMs finetuned on standard benchmarks of the same format (e.g., MNLI (Williams et al., 2018)) all struggle on our dataset, suggesting that the target knowledge is not readily available.

(b) **Is such knowledge learnable with appropriate data (§6)?** We adopt the challenge set analysis technique proposed by Liu et al. (2019a), which exposes models to a small amount of data and assesses how well they can adapt. All models achieve a noticeable performance improvement with as few as 200 examples, indicating that the target knowledge is potentially learnable.

(c) **What can models learn from recursive NPs (§7)?** We probe the finetuned models for two well-studied linguistic features in previous work, modifier semantic category and modifier scope. We show that both features can be learned from RNPC, with techniques including edge probing (Tenney et al., 2019) and attention visualization (Vig, 2019).

(d) **Is such knowledge useful for downstream tasks (§8)?** When evaluated on an extrinsic Harm Detection task, models finetuned on RNPC achieve strong zero-shot performance. This shows that the understanding of recursive NPs can benefit downstream language understanding tasks.

In summary, our work identifies an interesting linguistic phenomenon that is common sense to humans but challenging for models. It contributes to the characterization of LMs' limitations and capabilities in language understanding.

2 Related Work

Noun Phrases (NPs) have been extensively studied in both linguistics and NLP, primarily from the following perspectives.

Syntactic structure. A line of work focuses on the syntactic structure of NPs, which essentially explains the **modifier scope** (Campbell, 2002) in NPs. One classic task is NP bracketing, i.e., deciding whether an NP is right-branching (e.g., [*world* [*oil prices*]]) or left-branching (e.g., [[*crude oil*] *prices*]) (Lauer, 1995; Nakov and Hearst, 2005). A harder task is full parsing (Vadas and Curran, 2007; Pitler et al., 2010), i.e., reconstructing the complete dependency tree.

Modifier semantics. Another line of research revolves around the semantics of simple modifier-noun composition, starting with ways to **categorize modifiers** based on their inference patterns (Kamp and Partee, 1995; Bouillon and Viegas, 1999; Chierchia and McConnell-Ginet, 2000). With M as the modifier and N as the noun, a representative taxonomy summarized by McCrae et al. (2014) is:
(1) **intersective**: X is a M N $\implies$ X is $M \wedge X$ is a N, e.g., "an *American* surgeon" describes someone who is both American and a surgeon;
(2) **subsective**: X is a M N $\implies$ X is a N, but X is a M N $\not\Rightarrow$ X is M, e.g., someone who is "a *skillful* surgeon" is not necessarily skillful in all disciplines;

(3) **privative**: X is a $M\ N \not\Rightarrow X$ is a N, e.g., "a *former* surgeon" describes someone who is no longer a surgeon.

Despite the variations[2] and debates[3] on the taxonomy, we follow these conventional terms in subsequent sections.

With the advances in NLP, more recent works starts modeling the semantics of simple modifier-noun constructions with first-order logic (McCrae et al., 2014), linear mapping (Baroni and Zamparelli, 2010), and other explicit compositional operations (Boleda et al., 2012, 2013). In particular, Pavlick and Callison-Burch (2016a,b) propose a novel contextualized inference-based approach. They define the Add-One Entailment task with natural contexts from textual corpora, where the hypothesis differs from the premise by the insertion of one modifier. For example, *The crowd roared* entails *The enthusiastic crowd roared*, though *enthusiastic crowd* denotes a subset of *crowd* without context. However, natural contexts also introduce complications from monotonicity (Van Benthem, 1983). For instance, *red apple* entails *apple*, but *He didn't eat any red apple* does not entail *He didn't eat any apple* due to the downward entailment context. In our proposed approach, we handle this issue by controlling for context monotonicity.

Other related work explores which attributes of the head noun are affected by the presence of modifiers. Mullenbach et al. (2019) look at how modifiers project from a noun to its parts (e.g., does a *red jeep* have *red tires*?). Emami et al. (2021) test the likelihood change of an event when a modifier is added (e.g., *a false key* is less likely to *open a door* than *a key*). Apidianaki and Garí Soler (2021) study the prototypical properties of nouns (e.g., *a strawberry* entails *a red strawberry*). Researchers also examine the interpretation of noun compounds (Shwartz and Waterson, 2018; Hendrickx et al., 2013) (e.g., olive oil is made *of* olives, while baby oil is made *for* babies).

Summary. Neither syntactic parsing nor modifier semantics alone can fully capture the meaning of recursive NPs. In terms of syntax, modifier scope cannot always explain NPs due to the influence from modifier semantics. For instance, *a [big [fake gun]]* and *a [big [black gun]]* have the same structure but different inference patterns, i.e. only the latter is a gun. Meanwhile, modifier category itself does not suffice without taking into account modifier scope. For example, *a so-called healthy food* and *a so-called homeopathy expert* start with the same privative modifier (*so-called*). However, *so-called* questions truthfulness of the second modifier (*healthy*) in the former case while that of the noun (*expert*) in the latter. Therefore, we introduce a dataset containing three novel and challenging textual inference tasks, which rely on the interplay of syntax and semantics in determining the meaning of recursive NPs.

3 Task Formulation

Our dataset contains three tasks. Let us denote a canonical two-modifier recursive NP by $\mathbf{Det\ M_1\ M_2\ N}$ (Determiner, Modifier 1, Modifier 2, Noun). With this notation, the tasks are outlined below. See Table 1 for concrete examples.

Single-Premise Textual Entailment (SPTE) follows the conventional TE task format. Given a premise and a hypothesis, the model decides whether the premise semantically entails the hypothesis. The labels include `entailment` and `non-entailment`.[4] An SPTE example can be represented in regular expression as:

$$\textbf{Premise} : P\ Det\ M_1\ M_2\ N$$
$$\textbf{Hypothesis} : P\ Det\ (M_1|M_2)?\ N$$
$$\textbf{Label} : \texttt{entailment}|\texttt{non-entailment}$$

where $\mathbf{P}$ is a sentence prefix, which can be instantiated as *This is/He is/She is*, etc., depending on the NP. Intuitively, this task tests **whether an NP entails its various components**. This holds for most simple NPs (e.g., *the second ball* entails *ball*), but recursive NPs offer interesting counterexamples (e.g., (1b) in Table 1).

Multi-Premise Textual Entailment (MPTE) is adapted from the attributive propagation test described in Lalisse (2015). The format differs from SPTE only in that it has two premises instead of one. Given that both are true, the task is to determine whether the hypothesis is also true. The first premise is of the same form as in SPTE. The second premise contains a noun other than $\mathbf{N}$, denoted

[2] For example, other studies call category (3) "non-subsective" instead, and further decompose it into "privative" (X is a $M\ N$ contradicts X is a N, e.g., *fake*) and "non-privative" (X is a $M\ N$ is neutral to X is a N, e.g., *alleged*).

[3] Some linguists (for example, Partee (2010)) argue that (3) should be subsumed by (2), since privative modifiers can coerce the noun they modify into a looser interpretation.

[4] We do not distinguish between `neutral` and `contradiction` in order to minimize label ambiguity.

Category	Count	**Examples**: modifier (ATTRIBUTE)
Intersective	296	red (COLOR), female (GENDER), German (NATIONALITY)
Subsective	269	short (HEIGHT), small (SIZE), far (DISTANCE)
Privative	124	former (TIME), vice (AUTHORITY), fake (AUTHENTICITY)

Table 2: Statistics and examples for each semantic category in our modifier lexicon.

by $\mathbf{N_2}$.[5] A regular expression representation is:

Premise 1 : $P\ Det\ M_1\ M_2\ N$

Premise 2 : $P\ Det\ N_2$

Hypothesis : $P\ Det\ (M_1|M_2)\ N_2$

Label : `entailment|non-entailment`

This test targets the **compositionality of modifiers and nouns**. While most of the time a modifier can be freely "detached" and "attached" (e.g., (2a)), sometimes it cannot (e.g., (2b)).

Event Plausibility Comparison (EPC) follows the task formalization by Emami et al. (2021) for single-modifier NPs. Given two events, **Event1** and **Event2**, a model needs to assess the plausibility of **Event2** compared to that of **Event1**. The two events have the same event predicate **E**, and differ only in the **NP**. A regular expression representation is:

Event 1 : $Det\ (M_1|M_2)?\ N\ E$

Event 2 : $Det\ M_1\ M_2\ N\ E$

Label : `more|equally|less plausible`

This task tests the **influence of adding modifier(s) on the plausibility of different events about the noun**. Not all events are affected in the same way: in (3), *stars in many latest movies* becomes less plausible, while *is known by everyone* is more so.

We choose the three tasks defined above because they allow us to study different interesting properties of recursive NPs that conventional parsing tasks do not. For example, SPTE is convenient for comparing the impact of modifier order on the meaning of the NP (e.g., (1a) and (1b)); MPTE precisely reflects the property of subsective modifiers (e.g., *skillful*); whereas EPC is suitable for NPs with privative modifiers, since the other formats often cause ambiguity in this case.[6]

[5]For both premises to hold at the same time, we need an $\mathbf{N_2}$ that can refer to the same entity as **N**.

[6]For example, *fake fur* might or might not be considered a kind of fur (Partee, 2010). Annotators would thus probably disagree on the label if it were an SPTE example.

Task	**Total**	Entail	Non-entail
SPTE	1,163	582	581
Task	**Total**	Entail	Non-entail
MPTE	1,063	541	522
Task	**Total**	More	Equal	Less
EPC	1,479	508	392	579

Table 3: Number of examples in each RNPC task. Entail/Non-entail stand for Entailment/Non-entailment, and More/Equal/Less stand for More Plausible/Equally Plausible/Less Plausible.

4 Dataset Construction

Our dataset is constructed in four stages: (a) modifier lexicon construction, (b) NP extraction and selection, (c) instance creation and review, and (d) label verification. Among them, (c) and (d) involve crowdsourcing.[7]

Modifier lexicon construction. We first construct a lexicon of modifiers following the taxonomy in Section 2 (McCrae et al., 2014). We include modifiers studied in relevant linguistics literature (Nayak et al., 2014; Lalisse, 2015) and complement the list with modifiers that are missing or have not been addressed before under this lens (for example, modifiers that describe material, such as *wooden*, can also be viewed as privative). Each entry in the lexicon contains the modifier itself, its category (intersective, subsective, or privative), and its attribute (e.g., *green* is a COLOR). In total, the lexicon contains 689 modifiers, the largest resource of this kind. See Table 2 for category distribution and examples.

NP extraction and selection. Next, we collect recursive NPs from a variety of resources: linguistics literature (Matthei, 1982; Abdullah and Frost, 2005; Teodorescu, 2006; Morzycki, 2016), text corpora (Penn Treebank (Marcus et al., 1993) and the Annotated Gigaword corpus (Napoles et al., 2012)), and our creation. From text corpora, we extract all NPs with more than two consecutive modifiers in our lexicon, and manually select NPs considering a set of factors: lexical diversity, class balance, whether there is an interaction between the modifiers, etc. Finally, we complement the set with deliberately designed challenging cases of our invention, resulting in 1,299 NPs in total.

[7]See more statistics, crowdsourcing setup, and agreement details in Appendix A; see annotation guidelines and HIT design in the Supplementary Materials.

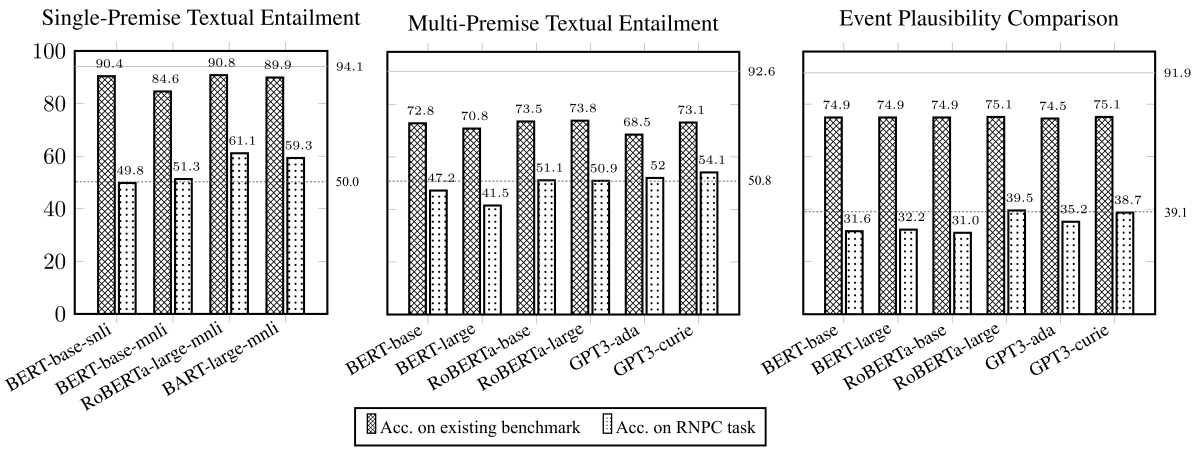

Figure 2: Given SOTA models finetuned on existing benchmark(s) of the same format as each RNPC task, we compare their accuracy on these benchmark(s) and on the RNPC task. The dotted line represents the majority baseline, and the solid line stands for human performance. Models for SPTE are finetuned on MNLI and SNLI, while models for the other two tasks are finetuned on MPE and ADEPT, respectively.

Instance creation and review. We hire college students[8] to write examples for the three tasks based on our collection of NPs. Each student is given a screening test containing five NPs. If $\geq 75\%$ of their created examples across all tasks are valid, they are qualified to continue. Each instance is then reviewed and/or revised by one of the authors, resulting in 8,260 valid instances.

Label verification. We again hire college students to verify instance labels via Amazon Mechanical Turk. Each task has a screening test of 10 easy instances with an unambiguous answer, and only students with an accuracy of $\geq 90\%$ can proceed. During the official annotation, a HIT contains 10 questions of a task, including one control question. Each HIT is completed by three people, excluding its creator. Annotations are then filtered based on the accuracy on control questions and the time used. Only examples with ≥ 2 people agreeing with the gold label are retained, yielding 4,567 examples. We then down-sample the examples in each task for a relatively balanced ratio among classes, resulting in 3,705 examples. See Table 3 for details.

5 Do LMs understand recursive NPs?

To answer question (a), whether the knowledge of how to interpret recursive NPs is present in pretrained LMs, we use the "behavioral test" probing method (Belinkov et al., 2020). Namely, we evaluate SOTA models finetuned on existing benchmark(s) of the same format as each RNPC task. The rationale is that LMs should acquire the ability of textual inference in the required format during finetuning, which allows us to elicit their potential knowledge about recursive NPs.[9]

Experimental setup. We consider the following datasets that address similar phenomena as our tasks: (1) MNLI (Williams et al., 2018) and SNLI (Bowman et al., 2015) for our SPTE; (2) MPE (Lai et al., 2017) for our MPTE; and (3) ADEPT (Emami et al., 2021) for our EPC. We choose SOTA and close-to-SOTA models on these benchmarks as probing candidates, including BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019b), BART (Lewis et al., 2020), and GPT3 (Brown et al., 2020).[10]

Results and analysis. We evaluate the finetuned models on each RNPC task. When the finetuning dataset has more classes than our task does, we map the model prediction to one of our classes by summing probability scores.[11] Figure 2 compares the performance of the models on the relevant

[8] Specifically, undergraduate and graduate students in an Artificial Intelligence class.

[9] LMs can also overfit the finetuning dataset and thus "forget" the target knowledge acquired during pretraining. Thus, we also directly probe the pretrained LMs in a complementary "likelihood scoring" experiment, described in Appendix C.

[10] Due to the size of MNLI and SNLI, we only evaluate available checkpoints from the Huggingface Transformers model hub. For the other two benchmarks, all models are trained by us. Also, the largest GPT3-davinci is unavailable for finetuning and thus excluded. See Appendices B and E.1 for dataset, model and hyperparameter details.

[11] For example, for a model trained on MNLI (with three labels), we compare the score of `entailment` and the summed score of `neutral` and `contradiction`. If the former is higher, we predict `entailment` on SPTE; otherwise `non-entailment`. Empirically, this strategy results in higher performance than directly mapping the highest-score MNLI label to its corresponding SPTE label.

Task	ID	Input	Gold Label	Predicted Label
Single-Premise Textual Entailment	(1a)	Premise: This is my new favorite movie. Hypothesis: This is my favorite movie.	Entailment	Entailment ✓
	(1b)	Premise: This is my favorite new movie. Hypothesis: This is my favorite movie.	Non-Entailment	Entailment ✗
Multi-Premise Textual Entailment	(2a)	Premise 1: He is a short American basketball player. Premise 2: He is a man. Hypothesis: He is an American man.	Entailment	Entailment ✓
	(2b)	Premise 1: He is a short American basketball player. Premise 2: He is a man. Hypothesis: He is a short man.	Non-Entailment	Entailment ✗
Event Plausibility Comparison	(3a)	Event 1: An animal can be harmful to people. Event 2: A dead dangerous animal can be harmful to people.	Less Plausible	Less Plausible ✓
	(3b)	Event 1: An animal can be harmful to people. Event 2: A dangerous dead animal can be harmful to people.	More Plausible	Less Plausible ✗

Table 4: Minimal-pair examples where the best-performing models make errors for each RNPC task. Differences between each pair are underlined.

benchmarks and our tasks. We also include human performance, calculated by averaging the accuracy of three college student annotators on a random sample of 300 examples for each task.

All models struggle on RNPC with performance around chance, while human accuracy is constantly above 90. On SPTE and MPTE, almost all models have a high false-positive rate. As long as all tokens in the hypothesis (e.g., *This is the second ball*) appear in the premise (e.g., *This is the second green ball*), they tend to predict `entailment`, indicating that they are making the same intersective interpretation errors as children do. On EPC, most models over-predict `equally plausible`, arguably due to the class imbalance during finetuning. This also shows that our task is not trivially solvable by models that understand non-recursive NPs, which the finetuning dataset comprises.

Next, we closely examine the best-performing models on each task, including RoBERTa-large finetuned on MNLI, GPT3-curie finetuned on MPE, and RoBERTA-large finetuned on ADEPT. On MPTE and EPC, even the best model barely surpasses chance performance. On SPTE, the best accuracy (61.2) is still unimpressive for a binary classification task. To understand where exactly the models fail, we further present a qualitative minimal-pair analysis in Table 4. On SPTE, the two examples differ only in the order of modifiers (*new* and *favorite*) in the premise, leading to opposite labels. However, the model predicts `entailment` for both, suggesting its insensitivity to subtle meaning differences incurred by modifier order changes. On MPTE, the difference between the two examples lies in the modifier in the hypothesis, *an American man* vs. *a short man*. As basketball players are generally tall, the second hypothesis should not be entailed. Again, the model predicts `entailment` for both cases, which shows its lack of relevant world knowledge. Finally, on EPC, *a dead dangerous animal* and *a dangerous dead animal* have subtly different meanings – the former refers to a *dangerous animal* that is dead (e.g., a dead lion, which is no longer harmful to people), while the latter refers to a *dead animal* that has become dangerous (e.g., a dead squirrel carrying viruses, which is indeed harmful). The model fails to distinguish between them, predicting `less plausible` for both. All the above observations show that the knowledge for interpreting recursive NPs is not present in LM representations.

6 Can LMs Learn the Meaning of Recursive NPs?

We investigate the reasons behind the models' low performance on RNPC, specifically whether their failure is due to the lack of in-domain training data or an intrinsic deficiency in their architecture. Namely, we attempt to answer question (b): Is the target knowledge learnable with appropriate data?

We adopt the challenge set analysis technique from Liu et al. (2019a), which exposes a model to a small amount of challenge data and assesses how well it can adapt. Specifically, we split each RNPC task dataset into a training set of 200 examples and a new test set containing the rest, ensuring that they have different modifiers in the same position. For example, if a modifier appears as the M_1 of an NP in the training set, it cannot appear in the same position of any NP in the test set. Then, we finetune

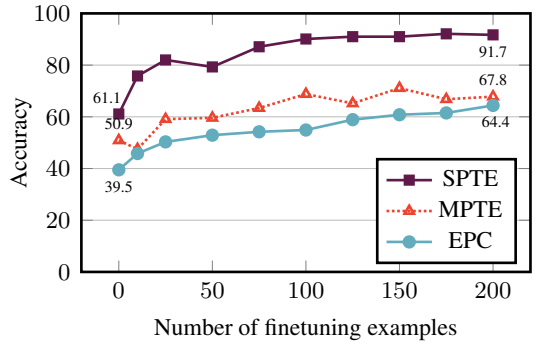

Figure 3: Learning curves of the best models on each RNPC task with an increasing number of finetuning examples.

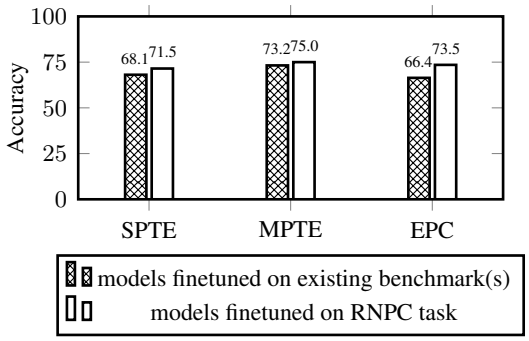

Figure 4: Probing accuracy for the "modifier semantic category" feature, before (left) and after (right) finetuning on each RNPC task.

each model from Figure 2 on an increasing number of examples (10 to 200). The learning curves of the best-performing models (RoBERTa-large (MNLI), RoBERTa-base (MPE), and RoBERTA-large (ADEPT)) are plotted in Figure 3.[12]

On SPTE, the accuracy rapidly climbs from 61.1 to 75.8 with only 10 examples, and saturates around 92 with 100 examples, approaching human performance (94.1). The learning curve on MPTE has more fluctuations, with a peak at 71.1 (150 examples) and a final score of 67.8. On EPC, starting around chance (39.5), the accuracy progressively increases up to 64.4 with 200 examples. These results indicate that the target knowledge is learnable with appropriate training data. Furthermore, SPTE may be the easiest task, since it only requires local knowledge about the meaning of the modifiers and the noun. By contrast, MPTE and EPC involve world knowledge (e.g., basketball players are generally tall among the population), as well as global reasoning between components in a sentence (e.g., the relationship between the event and the modifiers), which may explain the remaining large gap between model and human performance (> 90).

7 What can LMs learn from RNPC?

Given that the target knowledge is learnable, we now address question (c): What linguistic features have the models learned from RNPC? We probe for two features extensively studied in the relevant literature (cf. §2), using different techniques.

Modifier semantic category. We first investigate if models have learned the semantic category of modifiers using the "edge probing technique" (Tenney et al., 2019). Namely, each modifier is categorized as intersective, subsective, or privative (McCrae et al., 2014). The entailment pattern of individual modifiers is an important factor in determining the meaning of the entire NP.

Given a finetuned model, we take the contextualized representation of each modifier in the last hidden layer. Then, we attach a linear head on top of the token representation as an "auxiliary classifier". We choose linear classifiers because more expressive ones like Multi-Layer Perceptron are more likely to capture the target feature themselves (Hewitt and Liang, 2019). The token representations are then frozen, while the linear head is trained to predict the semantic category of the modifiers.[13]

We probe the models finetuned on RNPC from Section 6, as well as the models finetuned on existing benchmarks for comparison. The results are shown in Figure 4. For all tasks, the probing accuracy is higher for models finetuned on RNPC than on existing benchmarks. The increase is small for SPTE (3.4) and MPTE (2.8), but more obvious for EPC (7.1). This is somewhat counter-intuitive since modifier category is defined in terms of entailment patterns, but models learn it better from EPC than from TE tasks. Nonetheless, the overall trend shows that models can learn the semantic category of modifiers to some extent after being finetuned on our datasets. Since the absolute increase is limited, we plan to explore ways to quantify the actual amount of learned knowledge in future work.

Modifier scope. We also probe for the scope of the first modifier ($\mathbf{M_1}$) in recursive NPs ($\mathbf{Det\ M_1\ M_2\ N}$). Specifically, we focus on privative $\mathbf{M_1}$'s, since they can have different scopes when interacting with different $\mathbf{M_2}$'s and $\mathbf{N}$'s. For instance, in the NP *a former American diplomat*, *former* negates *diplomat* ($\mathbf{N}$), but the person is still American; while in *a former beginner drummer*, it negates *beginner* ($\mathbf{M_2}$), but the person may still be

[12] See Appendix E.2 for model and hyperparameter details.

[13] See Appendix E.3 for an illustration of the method.

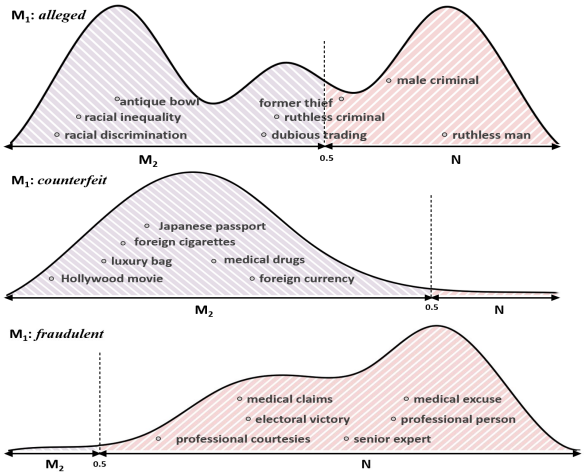

Figure 5: A case study of modifier scope. Each sub-figure shows the frequency distribution of the attention ratio r ($0 < r < 1$) for an $\mathbf{M_1}$, divided into two sides at 0.5. The $\mathbf{M_2}$ side contains NPs where $\mathbf{M_1}$ attends more to $\mathbf{M_2}$ than to $\mathbf{N}$; vice versa for the $\mathbf{N}$ side.

a drummer.[14] This difference cannot be captured by the semantic category of *former*.

As a proxy for the scope of $\mathbf{M_1}$, we use attention visualization, a widely adopted technique to study token correlations (Vig, 2019).[15] We choose BERT-base finetuned on 200 MPTE examples from Section 6 as the model to be probed for a case study.

Let us denote any token in a given NP as $\mathbf{x}$. We define $\mathbf{A_x}$, the average of the weights of all attention heads from $\mathbf{M_1}$ to $\mathbf{x}$ in the final layer, representing how much $\mathbf{M_1}$ attends to token $\mathbf{x}$. We then calculate the ratio $r = \mathbf{A_N}/(\mathbf{A_N} + \mathbf{A_{M_2}})$ ($0 < r < 1$). If $r < 0.5$, then $\mathbf{M_1}$ attends more to $\mathbf{M_2}$; else, $\mathbf{M_1}$ attends more to $\mathbf{N}$. For each privative modifier, we take all NPs containing it in the M_1 position in our dataset and plot the distribution of r. Figure 5 shows three examples (*alleged*, *counterfeit*, or *fraudulent*) representing different patterns.

As shown in the first sub-figure, *alleged* attends more to either $\mathbf{M_2}$ and $\mathbf{N}$ depending on the NP. For example, it attends more to $\mathbf{M_2}$ in *an alleged antique bowl* (0.454), since the NP describes a bowl that may not be antique. Inversely, *an alleged male criminal* is on the $\mathbf{N}$ side (0.517), since they are most likely male but may not be a criminal.

The second sub-figure indicates that *counterfeit* mainly attends to $\mathbf{M_2}$. For instance, *a counterfeit Hollywood movie* (0.382) is still a movie, but is probably not made in Hollywood. This is similar to the cases of *luxury bag*, *medical drugs*, *foreign cigarettes*, etc. On the contrary, *fraudulent* mainly attends to $\mathbf{N}$, as shown in the third sub-figure. The *fraudulent medical claims* (0.559) are not valid claims but still on medical grounds. The same holds for *electoral victory*, *medical excuse*, etc.

Additionally, we notice that there are some boundary cases close to the $\mathbf{r = 0.5}$ division line, like *ruthless criminal* and *former thief* in the *alleged* sub-figure. A plausible explanation is that $\mathbf{M_1}$ is questioning both $\mathbf{M_2}$ and $\mathbf{N}$ in these cases (e.g., *an alleged ruthless criminal* is not necessarily ruthless or a criminal). Overall, the above results indicate that models finetuned on our tasks can capture modifier scope in recursive NPs.

8 Is RNPC useful for downstream tasks?

We finally address question (d): How can such knowledge benefit downstream tasks? We choose the task of Harm Detection (Banko et al., 2020) for extrinsic evaluation. Concretely, we consider the scenario where a user interacts with a task-oriented agent like Siri or Alexa, and the agent needs to determine whether the involved activity in the user query is potentially harmful. The definition of "harm" can be user-dependent. Here, we consider an activity to be harmful if it may cause pain, physical injury, or be illegal for **minors**. We choose this task because many false positives come from recursive NPs. For example, *how to make a homemade bomb* is obviously `harmful` while *how to make a homemade bath bomb* is `harmless`.

We collect a small test set from wikiHow, a website of how-to articles. Each article title is considered a query (e.g., *how to make a cake*). Then, we compile a list of 74 keywords about harmful entities (e.g., *bomb*, *fire*, *drugs*), only 12 of which occur in RNPC. We then select wikiHow queries containing at least an NP with one of the 74 keywords as the head noun, and sample a small subset for manual annotation. Each query is labeled as `harmful` or `harmless`, depending on whether it involves a harmful activity as defined above. After data cleaning and re-balancing, we obtain 170 queries, with a 1:1 positive/negative ratio.

[14] Admittedly, there can be alternative interpretations: say, one can also imagine that *a former beginner drummer* describes a person who is no longer a drummer at all. However, in that case, it is enough to say *a former drummer* instead, considering the Gricean maxim of quantity. Therefore, here we still focus on the first interpretation, which is more straightforward.

[15] There have been recent debates on the faithfulness of this method (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019). Therefore, we do not use attention weights to make claims about **how** our models work, but only **what** they capture, with attention weights.

Model	Acc.	P	R	F_1
Always `harmful`	50.0	50.0	**100.0**	66.7
GPT3-ada	49.4	49.7	98.8	66.1
GPT3-curie	59.4	60.5	54.1	57.1
GPT3-davinci	51.3	50.6	**100.0**	67.2
RoBERTa-large (SPTE) (**ours**)	58.2	54.5	**100.0**	70.5
RoBERTa-large (EPC) (**ours**)	**72.9**	**66.4**	92.9	**77.5**

Table 5: Zero-shot performance of models trained on RNPC on the Harm Detection task. Baselines include a model that always predicts `harmful` and GPT3.

We design two zero-shot harm classifiers using models finetuned on our entire SPTE and EPC dataset. They share a few pre-processing steps: first, all NPs are extracted from the input query; then, NPs containing a keyword from our list in the head noun position are retained. For each retained NP (e.g., *a water gun*), we check if it is indeed a harmful entity using either the SPTE or the EPC model. The input to the SPTE model is a premise of the form "This is {NP}" (e.g., *This is a water gun*) and a hypothesis of the form "This is (a/an) {N}" (e.g., *This is a gun*). If the output label is `entailment`, we classify the query as `harmful`, otherwise `harmless`. Likewise, using the EPC model, we form two events given the retained NP: "(A/An) {N} is harmful" and "{NP} is harmful". If the second event is predicted as more or equally plausible compared to the first, the query is considered `harmful`.

We compare our two classifiers to a simple baseline that always predicts `harmful` as well as to three GPT3 models.[16] Both classifiers meaningfully exceed the simple baseline, and the EPC-based classifier outperforms all the other methods by 10+ in terms of accuracy and F_1. This shows that the understanding of recursive NPs is beneficial for downstream tasks without any training data. To understand why EPC is more suitable than SPTE for this task, we further examine the errors they make. One major error type concerns polysemous keywords such as *shot*. For instance, the SPTE model mistakenly predicts *how to have a good basketball shot* to be `harmful` because *a good basketball shot* is still a *shot* (*shot* can mean both "shooting a gun" and "shooting a ball"). There are also some queries out of the scope of the EPC model, e.g., *how to make a sake bomb*. Since *sake bomb* is a cocktail, the gold label is `harmful` as our target users are minors. The EPC model correctly predicts that *a sake bomb* is less harmful than *a bomb*, but fails to capture that it may still be harmful (for minors).

[16] Used in a zero-shot setting; see Appendix E.4 for details.

9 Conclusion

We introduce RNPC, a challenge set targeting the understanding of recursive NPs, a fundamental aspect of human common sense. Pretrained LMs with SOTA performance on Natural Language Understanding benchmarks have poor mastery of this knowledge, but can still learn it when exposed to small amounts of data from RNPC. Using different probing techniques, we show that models can learn relevant linguistic features, including modifier category and scope, from RNPC. They also achieve strong zero-shot performance on an extrinsic Harm Detection task, indicating the transferability of this knowledge. For future work, we hope to investigate other linguistic phenomena as a step towards comprehensively characterizing LMs' limitations and capabilities in language understanding.

Acknowledgments

This research is based upon work supported in part by the DARPA KAIROS Program (contract FA8750-19-2-1004), the DARPA LwLL Program (contract FA8750-19-2-0201), and the IARPA BETTER Program (contract 2019-19051600004). Approved for Public Release, Distribution Unlimited. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, IARPA, or the U.S. Government.

Special thanks go to our annotators, students in CIS 421/521 and MCIT 521 at the University of Pennsylvania. We also thank Artemis Panagopoulou for providing the extrinsic evaluation data. Meanwhile, we appreciate the support from OpenAI on finetuning GPT-3. Finally, we thank Haochen Zhang, Pengyuan Lu, Daniel Deutsch, Daphne Ippolito, Lara Martin, Young-Min Cho, Yi Zhang, Helen Jin, Siyi Liu, Eleni Miltsakaki, Jordan Kodner, Mingming Liu, Peng Zhou, Christopher Cieri, James J. Fiumara, Ellie Pavlick, Charles Yang, Yejin Choi, Alexander Koller, Chris Potts, and Mitch Marcus for their valuable feedback.

References

Nabil Abdullah and Richard A. Frost. 2005. Adjectives: A uniform semantic approach. In *Advances in*

Artificial Intelligence, pages 330–341, Berlin, Heidelberg. Springer Berlin Heidelberg.

Marianna Apidianaki and Aina Garí Soler. 2021. ALL dolphins are intelligent and SOME are friendly: Probing BERT for nouns' semantic properties and their prototypicality. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 79–94, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michele Banko, Brendon MacKeen, and Laurie Ray. 2020. A unified taxonomy of harmful content. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 125–137, Online. Association for Computational Linguistics.

Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1183–1193, Cambridge, MA. Association for Computational Linguistics.

Yonatan Belinkov, Sebastian Gehrmann, and Ellie Pavlick. 2020. Interpretability and analysis in neural NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 1–5, Online. Association for Computational Linguistics.

Gemma Boleda, Marco Baroni, The Nghia Pham, and Louise McNally. 2013. Intensionality was only alleged: On adjective-noun composition in distributional semantics. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013) – Long Papers*, pages 35–46, Potsdam, Germany. Association for Computational Linguistics.

Gemma Boleda, Eva Maria Vecchi, Miquel Cornudella, and Louise McNally. 2012. First order vs. higher order modification in distributional semantics. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1223–1233, Jeju Island, Korea. Association for Computational Linguistics.

Pierrette Bouillon and Evelyne Viegas. 1999. The description of adjectives for natural language processing: Theoretical and applied perspectives. In *Proceedings of Description des Adjectifs pour les Traitements Informatiques. Traitement Automatique des Langues Naturelles*, pages 20–30.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher 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 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Richard Campbell. 2002. Computation of modifier scope in NP by a language-neutral method. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Gennaro Chierchia and Sally McConnell-Ginet. 2000. *Meaning and grammar: An introduction to semantics*. MIT press.

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.

Ali Emami, Ian Porada, Alexandra Olteanu, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2021. ADEPT: An adjective-dependent plausibility task. 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 7117–7128, Online. Association for Computational Linguistics.

Henry Hamburger and Stephen Crain. 1984. Acquisition of cognitive compiling. *Cognition*, 17(2):85–136.

Iris Hendrickx, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Stan Szpakowicz, and Tony Veale. 2013. SemEval-2013 task 4: Free paraphrases of noun compounds. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 138–143, Atlanta, Georgia, USA. Association for Computational Linguistics.

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.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. 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 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Hans Kamp and Barbara Partee. 1995. Prototype theory and compositionality. *Cognition*, 57(2):129–191.

Ronald Kemker, Marc McClure, Angelina Abitino, Tyler L. Hayes, and Christopher Kanan. 2018. Measuring catastrophic forgetting in neural networks. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 3390–3398. AAAI Press.

Alice Lai, Yonatan Bisk, and Julia Hockenmaier. 2017. Natural language inference from multiple premises. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 100–109, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Matthias Lalisse. 2015. Distinguishing Intersective and Non-Intersective Adjectives in Compositional Distributional Semantics. Master's thesis, University of Oxford.

Mark Lauer. 1995. Corpus statistics meet the noun compound: Some empirical results. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 47–54, Cambridge, Massachusetts, USA. 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.

Nelson F. Liu, Roy Schwartz, and Noah A. Smith. 2019a. Inoculation by fine-tuning: A method for analyzing challenge datasets. 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 2171–2179, 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*.

Mercedes Marcilese, LMS Corrêa, and Marina RA Augusto. 2013. Recursive pre-nominal modifiers interpretation in language acquisition. *Advances in Language Acquisition*, pages 138–146.

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

Edward H Matthei. 1982. The acquisition of prenominal modifier sequences. *Cognition*, 11(3):301–332.

John P. McCrae, Francesca Quattri, Christina Unger, and Philipp Cimiano. 2014. Modelling the semantics of adjectives in the ontology-lexicon interface. In *Proceedings of the 4th Workshop on Cognitive Aspects of the Lexicon (CogALex)*, pages 198–209, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.

Marcin Morzycki. 2016. *Modification*. Cambridge University Press.

James Mullenbach, Jonathan Gordon, Nanyun Peng, and Jonathan May. 2019. Do nuclear submarines have nuclear captains? a challenge dataset for commonsense reasoning over adjectives and objects. 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 6052–6058, Hong Kong, China. Association for Computational Linguistics.

Preslav Nakov and Marti Hearst. 2005. Search engine statistics beyond the n-gram: Application to noun compound bracketing. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 17–24, Ann Arbor, Michigan. Association for Computational Linguistics.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated Gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, pages 95–100, Montréal, Canada. Association for Computational Linguistics.

Neha Nayak, Mark Kowarsky, Gabor Angeli, and Christopher D Manning. 2014. A dictionary of non-subsective adjectives. Technical report.

Barbara H Partee. 2010. Privative adjectives: subsective plus coercion. *Presuppositions and discourse: Essays offered to Hans Kamp*, 21:273–285.

Ellie Pavlick and Chris Callison-Burch. 2016a. Most "babies" are "little" and most "problems" are "huge": Compositional entailment in adjective-nouns. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2164–2173, Berlin, Germany. Association for Computational Linguistics.

Ellie Pavlick and Chris Callison-Burch. 2016b. So-called non-subsective adjectives. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 114–119, Berlin, Germany. 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.

Emily Pitler, Shane Bergsma, Dekang Lin, and Kenneth Church. 2010. Using web-scale n-grams to improve base NP parsing performance. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 886–894, Beijing, China. Coling 2010 Organizing Committee.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

Vered Shwartz and Chris Waterson. 2018. Olive oil is made *of* olives, baby oil is made *for* babies: Interpreting noun compounds using paraphrases in a neural model. 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 218–224, New Orleans, Louisiana. 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. 2019. 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.

Alexandra Teodorescu. 2006. Adjective ordering restrictions revisited. In *25th West Coast Conference on Formal Linguistics*, pages 399–407. Cascadilla Proceedings Project.

David Vadas and James Curran. 2007. Adding noun phrase structure to the Penn Treebank. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 240–247, Prague, Czech Republic. Association for Computational Linguistics.

Johan Van Benthem. 1983. Determiners and logic. *Linguistics and Philosophy*, 6(4):447–478.

Jesse Vig. 2019. A multiscale visualization of attention in the transformer model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 37–42, Florence, Italy. Association for Computational Linguistics.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. 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 11–20, Hong Kong, China. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

A Dataset Construction Details

A.1 RNPC Statistics

NPs. RNPC has 1,299 NPs. For an NP in the form of $\mathbf{Det\ M_1\ M_2\ N}$, the two modifiers M_1 and M_2 can each belong to one of three possible semantic categories (intersective, subsective, or privative), resulting in nine possible combinations. We plot the distribution of NPs with different combinations in RNPC in Table 6. Note that the distribution is not balanced because certain categories (e.g., NPs containing privative modifiers) yield many more minority class examples for our three tasks (e.g., `non-entailment` in SPTE). Thus, considering the final class balance in RNPC tasks, we include more NPs of certain categories.

Training and test sets for finetuning. In the experiment where we finetune models on RNPC, described in Section 6, we split again the dataset for each task into a training set and a new test set, ensuring no overlap of modifiers occurring in the same position. The training set contains 200 examples, which are gradually provided to the model. The test set contains the remaining examples. Table 7 shows the number of examples for each task.

A.2 Crowdsourcing Details

In the construction of RNPC, we hire college students as crowdworkers for instance creation and label verification. Specifically, they are undergraduate and graduate students in an Artificial Intelligence class (CIS 421/521 and MCIT 521 at the University of Pennsylvania), with good English proficiency. Both tasks are given as optional extra credit assignments in the class. Participation is solely voluntary. Before participation, students can preview the tasks, and are given a clear description of how the data will be used at the beginning of the instructions.

During instance creation, we provide detailed instructions on how to write high-quality examples for each task, which can be found in the Supplementary Materials. Annotations are collected via Google Forms. With 100 valid instances (equivalent to 2.5-4.75 hours of work, depending on their proficiency), students can earn 1% in extra credit of the overall course grade.

During label verification, we host our questions on Amazon Mechanical Turk. We design a HIT type for each RNPC task, which is also included in the Supplementary Materials. With 600 correctly answered questions (equivalent to 3.5-4 hours of work), students can earn 1% in extra credit of the overall course grade. We calculate the inter-annotator agreement using Krippendorff's alpha.[17] The agreement is 0.843 for SPTE, 0.575 for MPTE, and 0.933 for EPC.

M_1 / M_2	Int.	Sub.	Pri.
Int.	13	37	74
Sub.	138	109	162
Pri.	99	420	250

Table 6: Number of NPs in RNPC with different combinations of modifier category in the M_1 and M_2 position. Possible categories include intersective, subsective, and privative.

Task	Train	Test
SPTE	200	963
MPTE	200	863
EPC	200	1,279

Table 7: Number of examples in the training and testing split for each RNPC task in the finetuning experiment.

Dataset	Train	Dev	Test
MNLI	392,702	20,000	20,000
SNLI	550,152	10,000	10,000
MPE	8,000	1,000	1,000
ADEPT	12,892	1,611	1,612

Table 8: Number of examples in existing datasets of the same format used for finetuning.

A.3 Debiasing and Anonymization

The collected data does not contain any information that names or uniquely identifies individual people or offensive content. We ensure this by 1) manually reviewing the set of extracted NPs from corpora, and filtering out any NP that contains any sensitive/offensive information, 2) not requesting any personal information during human annotation, and 3) manually reviewing each RNPC example written by the human participants.

B Existing Benchmarks for Finetuning

We use the following benchmark datasets for finetuning. Each of them has the same format as one of our RNPC tasks. Table 8 shows the number of examples in each dataset.

MNLI. The Multi-Genre Natural Language Inference corpus (Williams et al., 2018) is a dataset of 433k textual entailment examples, labeled as entailment, contradiction, or neutral. It covers a

[17] https://pypi.org/project/krippendorff

range of genres of spoken and written text. The language in the dataset is English. The corpus is released under the OANC's license, the Creative Commons Share-Alike 3.0 Unported License, and the Creative Commons Attribution 3.0 Unported Licenses, depending on the portion.

SNLI. The Stanford Natural Language Inference corpus (Bowman et al., 2015) is a crowdsourced dataset of textual entailment examples, labeled as entailment, contradiction, or neutral. The sentences are written by humans doing a novel grounded task based on image captioning. The language in the dataset is English. The dataset is released under the Creative Commons Attribution-ShareAlike 4.0 International License.

MPE. Lai et al. (2017) introduce a Multiple Premise Entailment Task dataset. This is a novel textual entailment task that requires inference over multiple premise sentences. Each example consists of four premise sentences (captions from a FLICKR30K image), one hypothesis sentence (a simplified FLICKR30K caption), and one label (entailment, neutral, or contradiction) that indicates the relationship between the set of four premises and the hypothesis. The language in the dataset is English. The license of the dataset is unspecified.

ADEPT. Emami et al. (2021) introduce a dataset of the Adjective-Dependent Plausibility Task (ADEPT). Each example contains a base sentence, and a slightly modified sentence obtained by adding an adjective to a noun in the base sentence. The dataset is created to support explorations into how certain classes of adjectives might influence the plausibility of events depicted in natural language sentences. The textual data come from Wikipedia, the Common Crawl, and ConceptNet. The language of the dataset is English. ADEPT is released under the CC BY-SA 3.0 license. It is intended to be used only for research, exploratory evaluation, and auditing, which our use is consistent with.

C Probing Pretrained LMs

C.1 Motivation

When addressing question (a), we finetune pretrained LMs on existing benchmarks of the same format as each RNPC task, assuming that the finetuning process allows models to do textual inference in the required format. However, it is possible that this assumption does not hold, because LMs can overfit the finetuning data beyond just learning the format. Then even if the target knowledge is present in pretrained LMs, *catastrophic forgetting* (Kemker et al., 2018) can happen during finetuning.

C.2 Task Conversion

We complement Section 5 with another experiment, where we directly probe pretrained LMs using a prompting method inspired by the line of work on LMs as knowledge bases (Petroni et al., 2019). Specifically, we convert each RNPC task to a likelihood comparison task:

SPTE. Given the original formulation which has a premise and a hypothesis, we define L_{entail} as the **conditional likelihood** that the hypothesis is necessarily true given the premise, assigned by an LM. Contrarily, $L_{non-entail}$ stands for the conditional likelihood that the hypothesis is NOT necessarily true given the premise.[18] If $L_{entail} > L_{non-entail}$, the model is considered to predict `entailment`, and vice versa.

MPTE. The conversion method is the same as that for SPTE, except that in the conditional likelihood computation, we now consider the concatenation of two premises as the given condition.

EPC. Given the original formulation with two events, Event 1 and Event 2, we define L_1 and L_2 as the **(unconditional) likelihood** of Event 1 and Event 2 assigned by an LM, respectively. We then choose a threshold θ,[19] and compare it to the absolute difference between L_1 and L_2. If the difference is smaller than θ, we consider the model prediction as `equally likely`. Otherwise, the model prediction is `more likely` if L_2 is higher, and `less likely` if L_1 is higher.

For Causal LMs (e.g., GPT), the likelihood is computed with standard left-to-right language modeling scores. For Masked LMs (e.g., BERT, RoBERTa, BART), the likelihood is computed with pseudo-log-likelihood scores (Salazar et al., 2020).

C.3 Sanity Check

Before evaluating LMs on the converted RNPC, we perform a sanity check to see if our formalization makes sense to LMs, i.e., whether they understand the meaning of *necessarily* and *not necessarily*.

[18] For example, if the original SPTE example has the premise *This is the second green ball* and the hypothesis *This is the second ball*, then L_{entail} equals to L(*This is necessarily the second ball* | *This is the second green ball*), and $L_{non-entail}$ equals to L(*This **isn't** necessarily the second ball* | *This is the second green ball*).

[19] In the range [0.1, 0.5, 1, 2, 3, 5], 0.5 is the empirical optimal.

Model	SPTE	MPTE	EPC
gpt2-base	59.4	52.8	33.4
gpt2-medium	**62.6**	53.6	33.6
gpt2-large	61.4	56.1	32.4
gpt2-xl	61.7	**56.9**	31.7
gpt3-ada	55.2	55.2	33.2

Table 9: Accuracy of SOTA pretrained models directly evaluated on RNPC tasks.

We write 50 sentence pairs for likelihood comparison, all consisting of simple commonsense knowledge. For example, comparing *A human being is necessarily female* and *A human being isn't necessarily female*, the second sentence should be more likely; while for *Humans are necessarily mortal* and *Humans aren't necessarily mortal*, the first sentence should be more likely. Such comparisons do not require any knowledge about recursive NPs, and involve only common entities and facts. If models understand *necessarily* and *not necessarily* correctly, they should find the task easy.

To our surprise, almost all Masked LMs we test (BERT-base/large, RoBERTa-base/large) fail the sanity check, mostly performing around chance (50 accuracy). However, most Causal LMs (GPT-2-base/medium/large/xl, GPT-3-ada) reasonably perform above chance, with accuracy scores ranging from 70 to 80. We suspect that pseudo-log-likelihood scores are not entirely suitable for our purposes; also, the task is harder than expected due to reporting bias, as the tested knowledge (e.g., *not all humans are female*) is potentially too obvious to be explicitly stated in the pretraining data.

C.4 Results

We evaluate LMs that pass the sanity check on the converted RNPC, and report their performance in Table 9. Despite the decent performance on the sanity check examples (70-80), the accuracy on RNPC is remarkably lower. Compared to our original results of probing the finetuned models, the optimal performance on SPTE and MPTE slightly improves, while accuracy on EPC decreases. However, the same patterns hold: most models perform around or slightly above chance, with a large difference from human performance. These findings further strengthen our answer to question (a), i.e. LMs do not inherently have the knowledge to interpret recursive NPs.

Model	Acc.	P	R	F_1
BERT-base (SNLI)	49.8	49.9	77.0	60.5
BERT-base (MNLI)	51.3	50.7	97.8	66.8
RoBERTa-large (MNLI)	**61.1**	56.3	99.1	71.9
BART-large (MNLI)	59.3	55.1	97.9	70.7

Table 10: Full results of SOTA models evaluated on SPTE. The finetuning dataset is in brackets.

Model	Acc.	P	R	F_1
BERT-base	47.2	48.0	44.0	45.9
BERT-large	41.5	34.2	16.3	22.1
RoBERTa-base	51.1	51.0	100.0	67.5
RoBERTa-large	50.9	50.9	100.0	67.5
GPT3-ada	52.0	51.5	97.0	67.3
GPT3-curie	**54.1**	52.6	97.4	**68.4**

Table 11: Full results of SOTA models evaluated on MPTE. The finetuning dataset is MPE for all models.

Model	Acc.	P	R	F_1
BERT-base	31.6	29.2	31.6	22.4
BERT-large	32.2	27.7	32.2	23.7
RoBERTa-base	31.0	46.8	31.0	22.3
RoBERTa-large	**39.5**	54.1	39.5	**32.7**
GPT3-ada	35.2	40.2	35.2	28.3
GPT3-curie	38.7	69.9	38.7	32.8

Table 12: Full results of SOTA models evaluated on EPC. The finetuning dataset is ADEPT for all models.

D Full Results

In Section 5, we evaluate SOTA LMs on RNPC tasks. In addition to accuracy, we also report precision, recall, and F-1 score here. Tables 10, 11 and 12 show the full results for each task, respectively.

E Implementation Details

E.1 Models Finetuned on Existing Benchmarks

In Section 5, we evaluate SOTA LMs finetuned on existing benchmarks of the same format on RNPC. We use four different pretrained models, BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019b), BART (Lewis et al., 2020), and GPT3 (Brown et al., 2020), in different sizes. The first three are implemented with HuggingFace Transformers[20], and the last is from OpenAI's standard API[21].

The pretrained model checkpoints we use include: `bert-base-uncased` (110M parameters), `bert-large-uncased` (336M parameters), `roberta-base` (125M param-

[20] https://github.com/huggingface/transformers

[21] https://beta.openai.com/docs/api-reference

eters), `roberta-large` (335M parameters), `facebook/bart-large` (406M parameters), `GPT3-ada` (350M parameters), and `GPT3-curie` (6.7B parameters).[22] Their licenses include Apache License 2.0 (BERT and BART), GNU General Public License v2.0 (RoBERTa), and MIT license (GPT3).

Due to the size of MNLI and SNLI, we use existing checkpoints available on the Huggingface Transformers model hub. For all other datasets, we finetune the pretrained models using the `SequenceClassification` pipeline on Huggingface, or the standard prompt completion finetuning API on OpenAI.[23] The finetuning scripts are adapted from the `text-classification` example in the HuggingFace Transformers repository.[24] We performed hyperparameter search in the following range:

- batch size: [4, 8, 16, 32]
- learning rate: [1e-5, 1e-6]
- number of epochs: [2, 3, 5]
- max sequence length: [64, 128]

The optimal hyperparameter values and finetuned models are available on the HuggingFace model hub.

We run our finetuning experiments on an NVIDIA GeForce RTX 2080 Ti GPU, with half-precision floating point format (FP16). The finetuning takes 2 to 5 hours depending on the task.

E.2 Models Finetuned on RNPC

In Section 6, we address the question of whether LMs can learn the meaning of recursive NPs. We finetune each model from Section E.1 on an increasing number of examples of each RNPC task. The model architectures, the pipelines used, the range of hyperparameter search, and the computing resources used are all the same as in the previous subsection. After being finetuned on 200 examples, the best performing models are RoBERTa-large (MNLI) for SPTE, RoBERTa-base (MPE) for MPTE, and RoBERTA-large (ADEPT) for EPC. The optimal hyperparameter values and finetuned models on the full 200 examples of each RNPC task are available on the HuggingFace model hub.

[22] All models above are available at https://huggingface.co/transformers/v4.8.2/pretrained_models.html or https://beta.openai.com

[23] https://beta.openai.com/docs/api-reference/fine-tunes

[24] https://github.com/huggingface/transformers/tree/master/examples/legacy

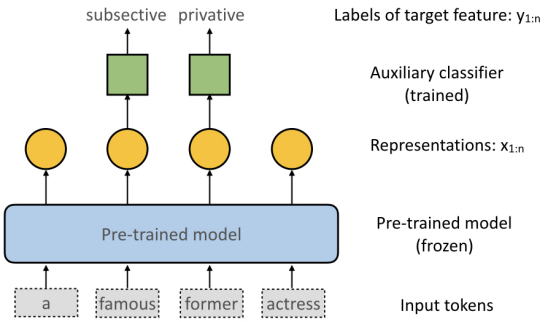

Figure 6: An illustration of the Edge Probing method. Figure adapted from Tenney et al. (2019).

E.3 The "Edge Probing" Method

In Section 7, we adopt the Edge Probing technique from Tenney et al. (2019) to investigate if the modifier category feature can be learned from our tasks.

To reintroduce the general idea of this method, consider the following setup: we have data $D = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, where $(x_1, x_2, ..., x_n)$ are the model representations to be probed and $(y_1, y_2, ..., y_n)$ are the labels of a linguistic feature we are interested in probing for. The goal is to see if $(x_1, x_2, ..., x_n)$ encodes $(y_1, y_2, ..., y_n)$.

In our case, given an NP of the form $\mathbf{Det\ M_1\ M_2\ N}$, $(x_1, x_2, ..., x_n)$ are the token representations of the best-performing models after being finetuned on each RNPC task, as mentioned in Section E.2, and $(y_1, y_2, ..., y_n)$ are the semantic categories of $\mathbf{M_1}$ and $\mathbf{M_2}$.

We freeze the representations $(x_1, x_2, ..., x_n)$, and attach a simple auxiliary classifier (or probing classifier) on top of them. The auxiliary classifier is trained to predict the semantic category y_i corresponding to every modifier token representation x_i. In our case, we choose a linear classifier as the probing classifier. The assumption is that the expressivity of the auxiliary classifier should be limited, or otherwise it might learn the target feature itself. Based on this assumption, the accuracy of predicting y indicates the extent to which the model representations encode the target feature. The method is illustrated in Figure 6.

Specifically, the linear classifier is an `SGDClassifier` implemented with scikit-learn.[25] We perform hyperparameter search in the following range:

[25] https://scikit-learn.org

- loss: [hinge, log, modified_huber, squared_hinge, perceptron]
- warm_start: [True, False]
- penalty: [l2, l1, elasticnet]
- alpha: [0.0001, 0.001, 0.01, 0.1]
- n_jobs: [-1]
- max_iter: [500, 1000, 2000]
- random_state: [0]
- eta0: [0.1, 0.01, 0.001, 0.0001]
- learning_rate: [optimal, invscaling, adaptive]

The optimal hyperparameter values for the best-performing models of each RNPC task from Section E.2 are available on the HuggingFace model hub.

E.4 Models Used for Extrinsic Evaluation

In Section 8, one of the models we compare to is GPT3. We exploit three versions of different sizes, `GPT3-ada`, `GPT3-curie`, and `GPT3-davinci`, using the same API described in Section E.1. The models are evaluated in a zero-shot setting, where we only provide the following task description:

> Given a "how-to" query, determine if the task is harmful to a teenager. The answer should be Yes or No.

Each input question is then formatted as:

> Question: Is "{how-to query}" harmful?
> Answer:

For every how-to query in our test set, we use the concatenation of the task description and the instantiated input question as the prompt, and let the model generate a one-token continuation. The top generated token is always *Yes* or *No*, implying that GPT3 has a good understanding of the task format.

F Ethical Considerations

F.1 Limitations

Assumptions. One assumption we make in answering question (a) is that LMs finetuned on existing benchmarks can learn the required format without overfitting the specific domains of the finetuning data. Suppose this assumption does not hold, then even if the target knowledge is present in pretrained LMs, they can "forget" it during finetuning. Therefore, the finetuning process does not allow us to elicit the target knowledge from pretrained LMs. To address this issue, we complement the behavioral test probing method with another experiment to directly probe the pretrained LMs via likelihood scoring. See Section C for details.

Another assumption occurs in our answer to question (d). We assume that a query is harmful if it contains a harmful entity. However, in practice, there can be queries like *How to prevent a fire*, which does contain a harmful entity (*fire*) but is precautionary instead of harmful. Our model does not take into account factors like predicates in context, and will therefore identify all such cases as false positives.

Scope of claims. Our first three claims (i.e. answers to question (a)-(c)) are only verified to hold on the RNPC dataset, which 1) is in English and 2) mainly consists of NPs in the news domain. Our last claim (i.e. answer to question (d)) is only verified to hold on the harm detection dataset we collect, which 1) is also in English, 2) consists of how-to queries in the domain of human activities, and 3) is annotated based on a non-exhaustive keyword list of harmful entities.

Moreover, part of our answer to question (b) (i.e. LMs have learned the feature of modifier semantic category from RNPC) is qualitative. The absolute increase in the probing accuracy after finetuning is limited, so it is likely not the entire picture. Quantifying to what extent LMs have learned this feature is an interesting direction for future work.

F.2 Risks

The risks associated with the study are minimal.

Harm detection models. Our harm detection models are intended for research purposes only. They are designed for specific types of harmful queries, i.e. those with harmful entities. One should not deploy them directly in real life since they are by no means applicable under all scenarios.

Data collection. Our human participants may experience slight discomfort due to boredom during data collection. To minimize this, we make sure that it is entirely voluntary to participate and discontinue at any time.

F.3 Intended Use

Our models and data should be used for research purposes only. They should not be deployed in the real world as anything other than a research prototype, especially commercially.