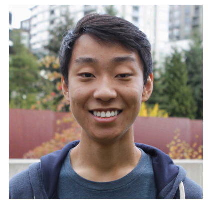
Linguistic Knowledge and Transferability of Contextual Representations





Nelson F. Liu | Matt Gardner | Yonatan Belinkov | Matthew E. Peters | Noah A. Smith

NAACL 2019—June 3, 2019

Thanks. Today, I'll be talking about the linguistic knowledge and transferability of contextual word representations.
This is joint work with Matt Gardner, Yonatan Belinkov, Matt Peters, and Noah Smith.

[McCann et al., 2017; Peters et al., 2018a; Devlin et al., 2019, *inter alia*]

Contextual Word Representations Are Extraordinarily Effective

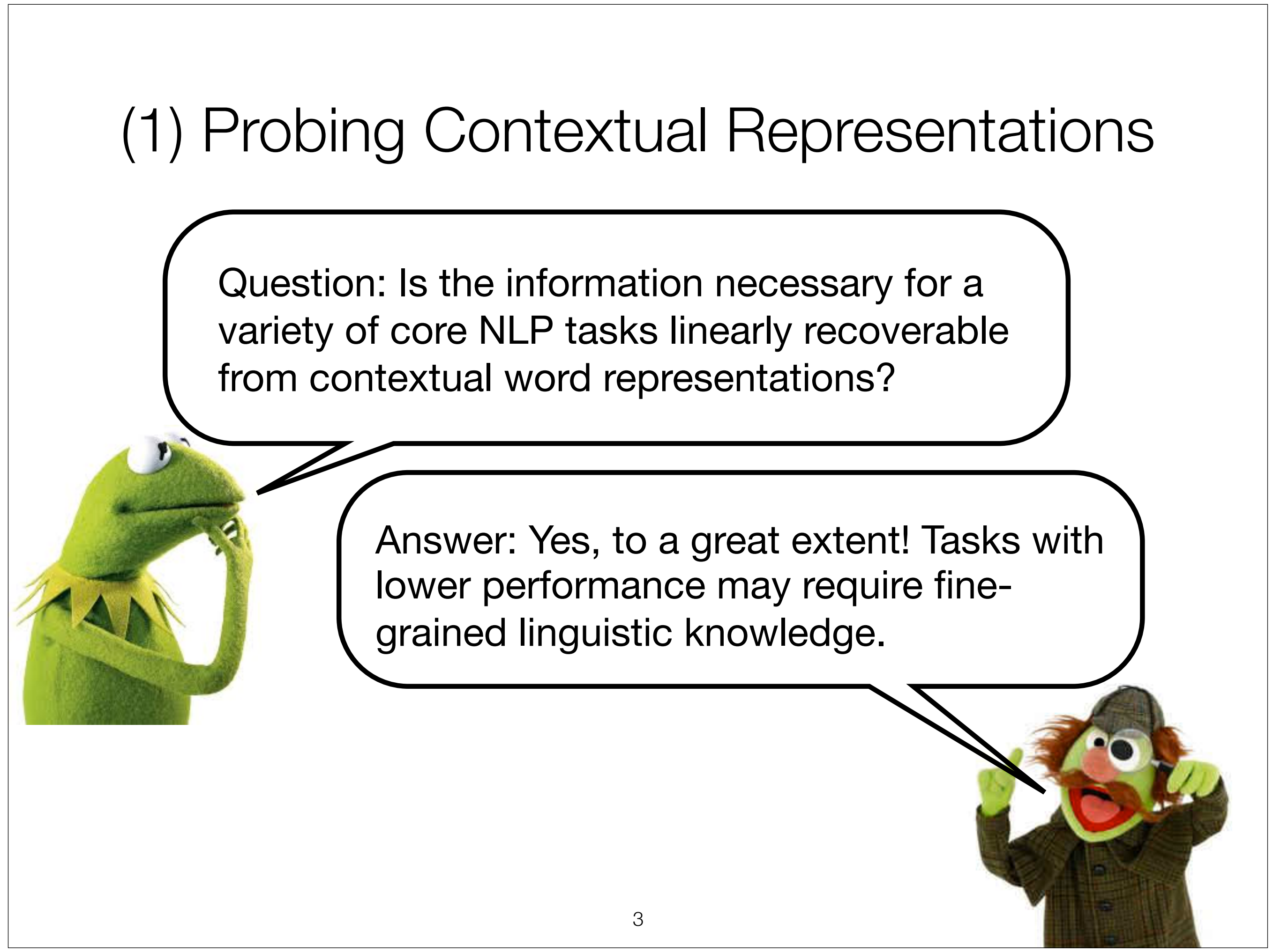
- Contextual word representations (from ***contextualizers*** like ELMo or BERT) work well on many NLP tasks.
- But **why** do they work so well?
- Better understanding enables principled enhancement.
- **This work:** studies a few questions about their generalizability and transferability.

Over the last year, contextual word representations from contextualizers like ELMo and BERT have pushed NLP to new heights across a diverse set of tasks.

However, it's still unclear *why* they work so well, or what their abilities and limitations are. Better understanding these models is a critical first step towards their principled enhancement.

In this work, we ask and answer a few questions about the generalizability and transferability of contextual word representations.

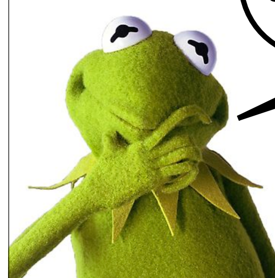
I'll start off the talk by giving a high-level summary of our findings, and I'll then dive deeper into more details.



One question we looked at is whether the information necessary for a variety of core NLP tasks is linearly recoverable from only contextual word representations.

At a first approximation, the answer appears to be yes! However, we find that performance on some tasks is lacking, perhaps because they require fine-grained linguistic knowledge.

(2) How Does Transferability Vary?

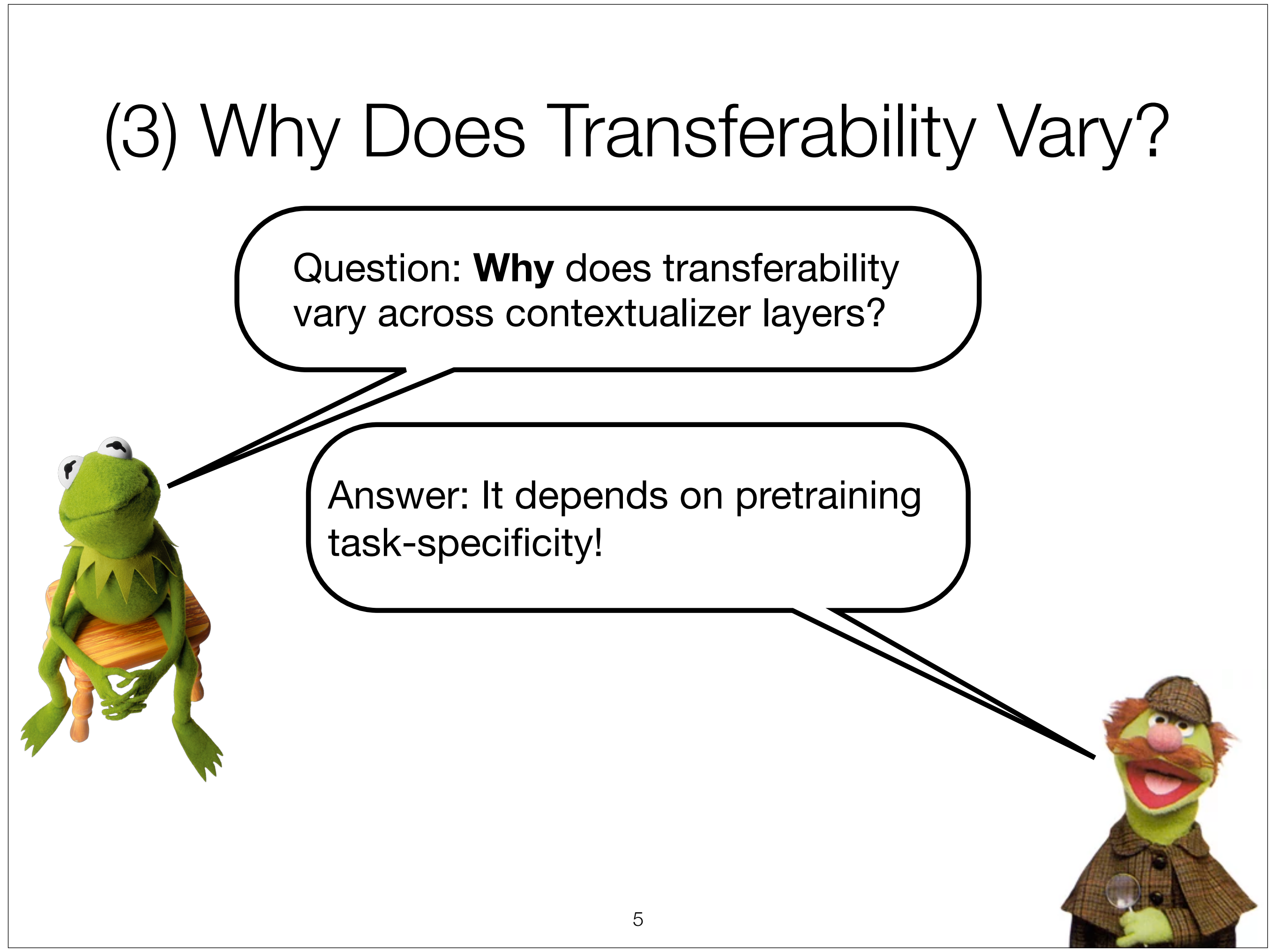


Question: How does transferability vary across contextualizer layers?

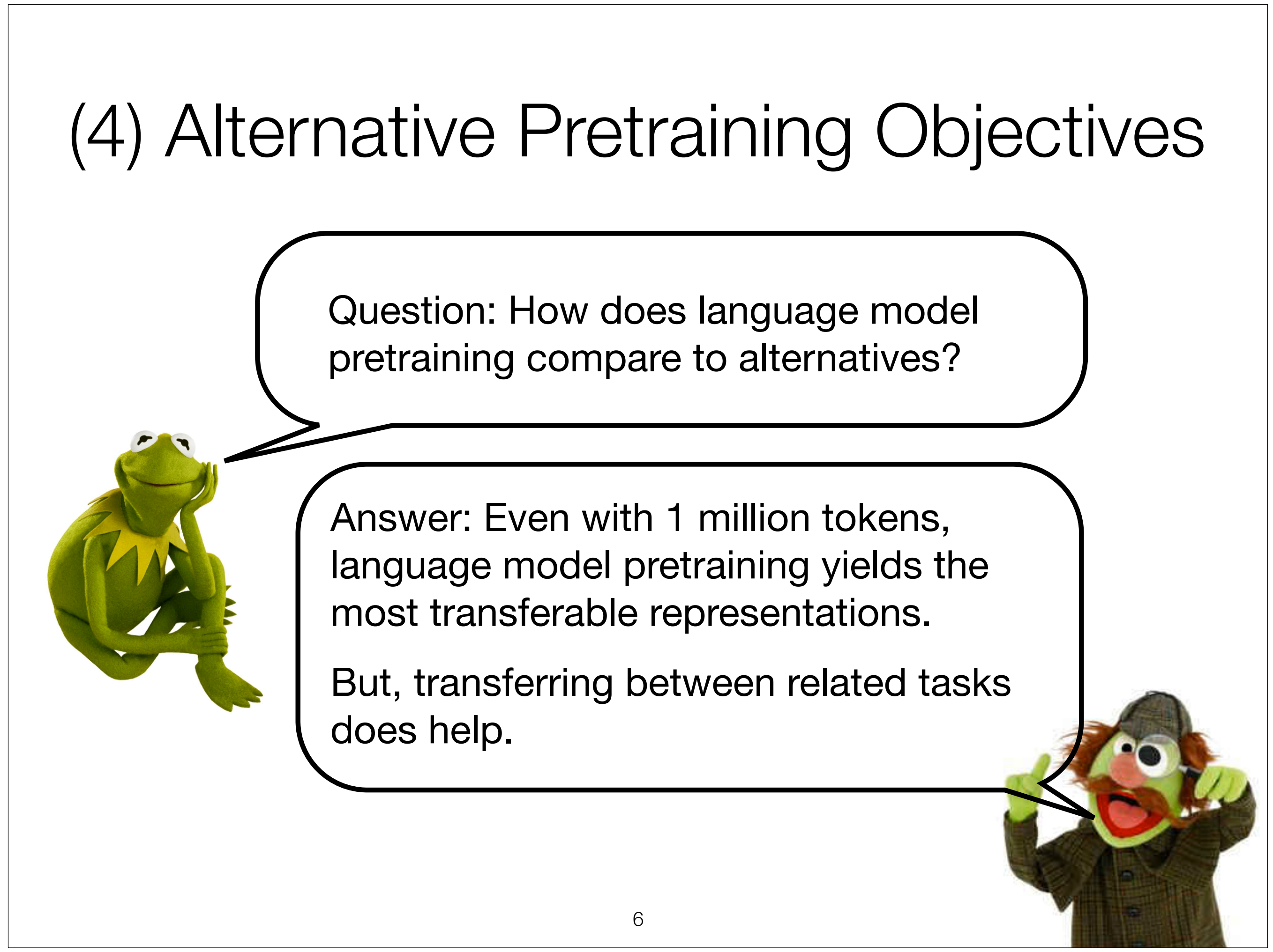
Answer: First layer in LSTMs is the most transferable. Middle layers for transformers.

We also studied how the transferability of contextual word representations varies across contextualizer layers.

We found that the first layer of LSTMs is consistently the most transferable. Transformers, on the other hand, have no such most-transferable layer---although the middle layers tend to be more transferable than others.



We also look at **why** transferability varies across contextualizer layers, and we find that higher layers in LSTMs are more task-specific and thus less transferable. Transformer layers do not show the same monotonic trend, but in both cases, the topmost layer is the most task-specific.



Lastly, we also looked into the source of the generalizability of language-model derived contextual word representations. In particular, do they work well only because they see a lot of data? Or is language modeling unto itself also a good objective?

We find that language modeling yields representations that are more transferable than eleven supervised alternatives that we studied. However, we do find that pretraining on related tasks helps.

[Shi et al., 2016; Adi et al., 2017]

Probing Models

Now that I've given a summary of the work, I'll go into more detail about each part.

I'll start by talking about probing models, which we use to study contextual word representations.

I'll walk through how we use them.

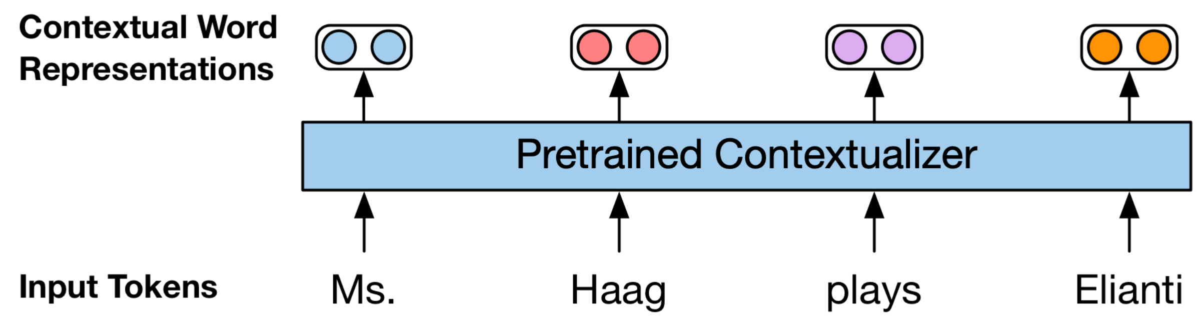
[Shi et al., 2016; Adi et al., 2017]

Probing Models

Input Tokens	Ms.	Haag	plays	Elianti

First, we start off with some input tokens.

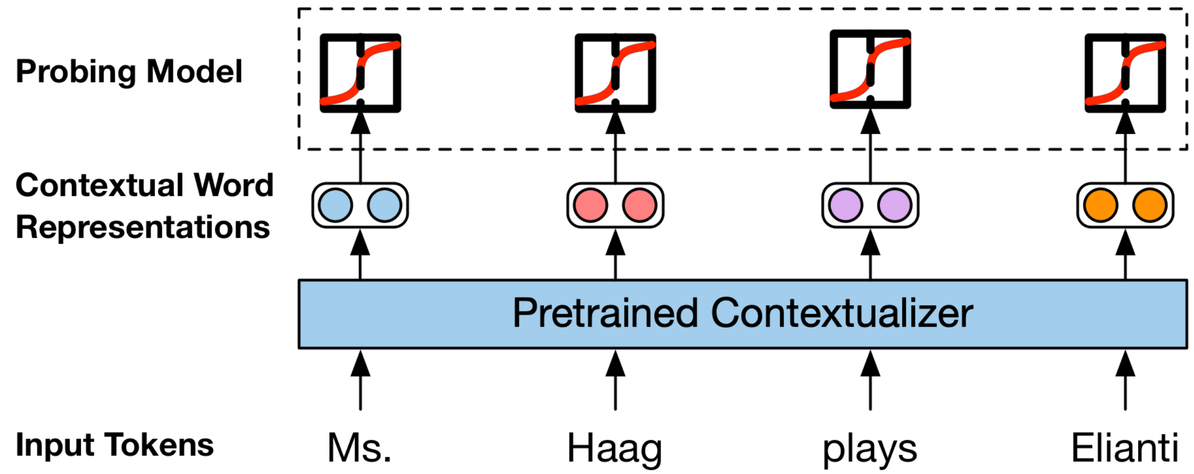
Probing Models



Then, we use some sort of pretrained contextualizer, like ELMo or BERT, to get contextual word representations for each token in our input.

[Shi et al., 2016; Adi et al., 2017]

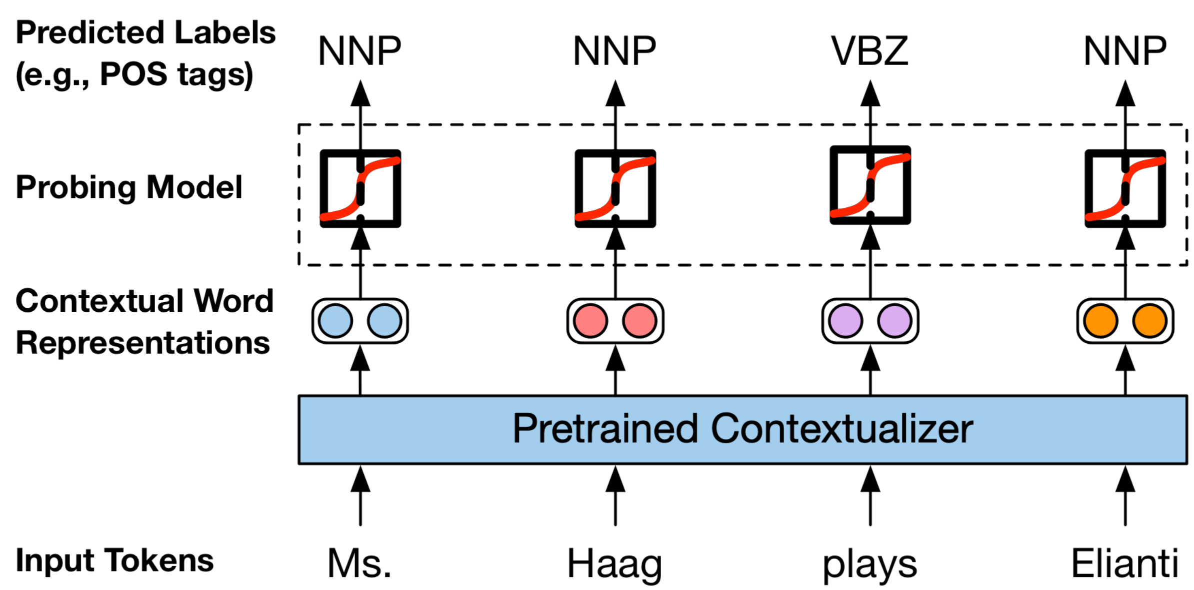
Probing Models



The probing model's input is the contextual word representation for a single *token.*

[Shi et al., 2016; Adi et al., 2017]

Probing Models



And it's trained to predict linguistic features of interest about that token from only its contextual word representation.

The key idea is that we can use the performance of the probing model as a proxy for how predictive our input representations are of the linguistic features of interest.

[Belinkov, 2018; Blevins et al., 2018; Tenney et al., 2019]

Pairwise Probing

We also looked at probing models that predict labels between pairs of words, which we call pairwise probing.

[Belinkov, 2018; Blevins et al., 2018; Tenney et al., 2019]

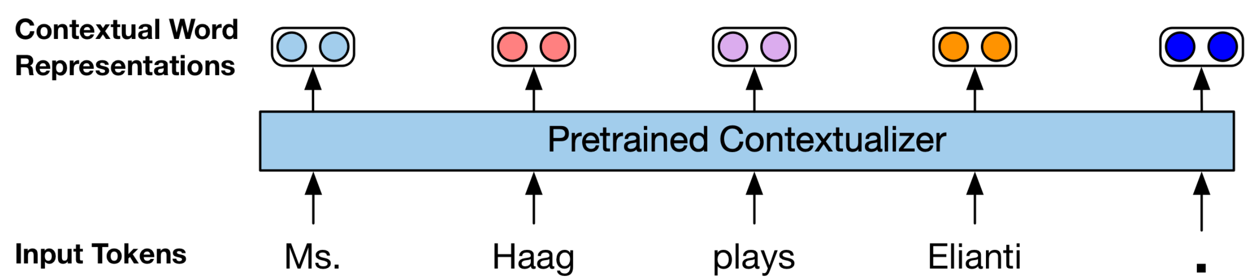
Pairwise Probing

Input Tokens Ms. Haag plays Elianti .

Again, we start off with a set of input tokens...

[Belinkov, 2018; Blevins et al., 2018; Tenney et al., 2019]

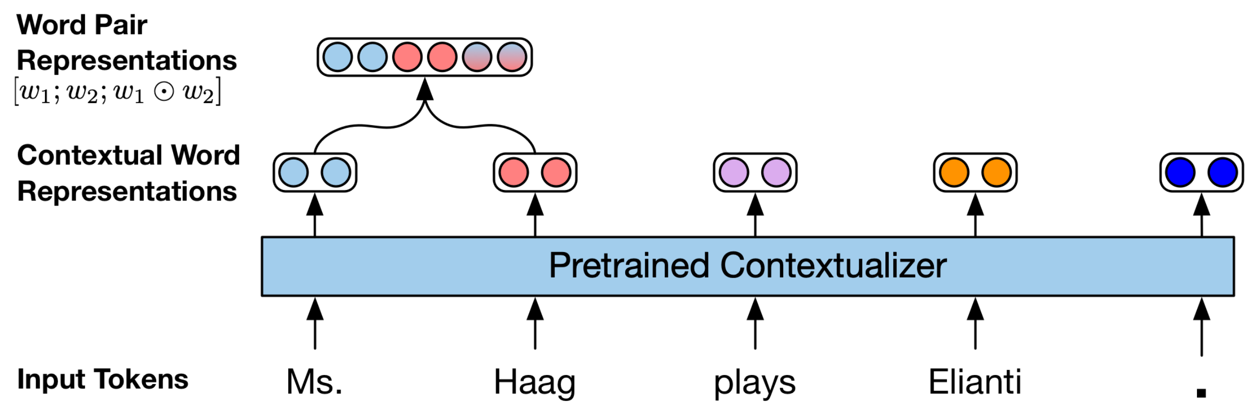
Pairwise Probing



...and we get contextual word representations for each of them.

[Belinkov, 2018; Blevins et al., 2018; Tenney et al., 2019]

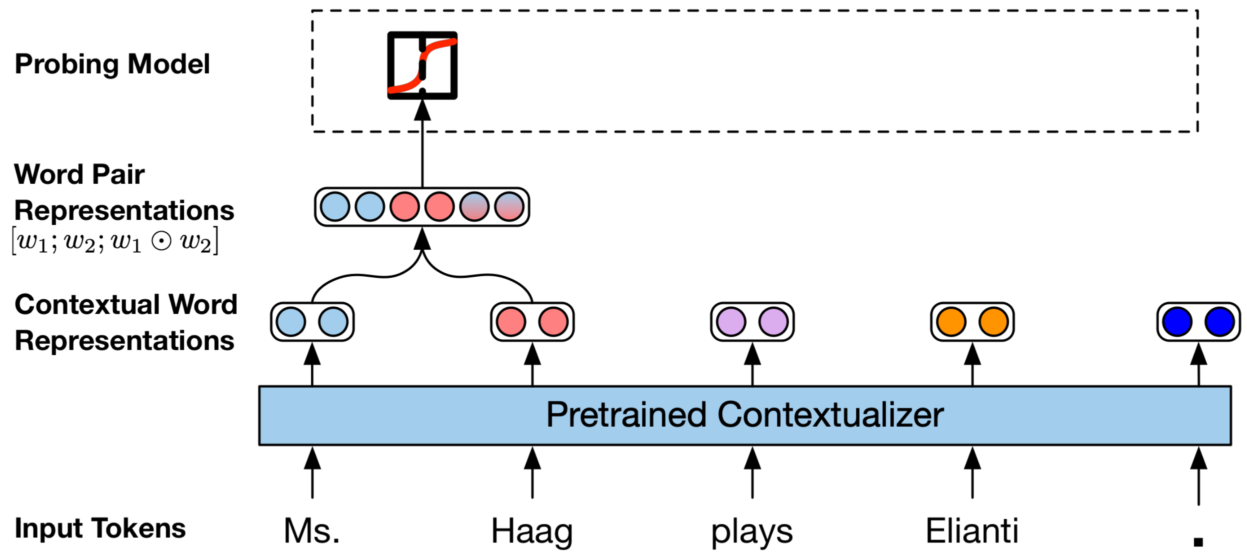
Pairwise Probing



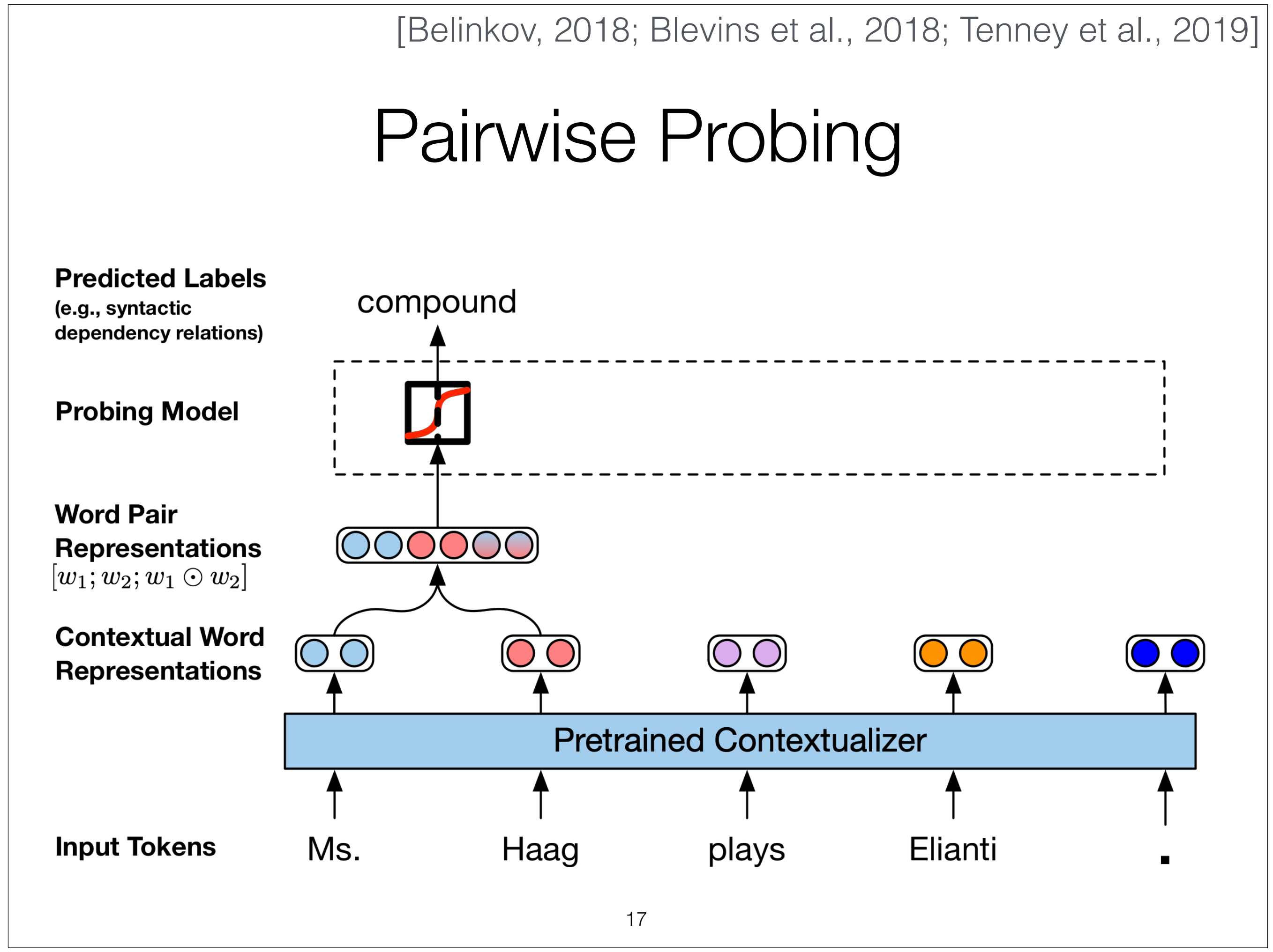
Now, to predict some linguistic feature between two tokens, we combine their contextual word representations. In this example, we're combining the contextual word representations of Ms. and Haag.

[Belinkov, 2018; Blevins et al., 2018; Tenney et al., 2019]

Pairwise Probing



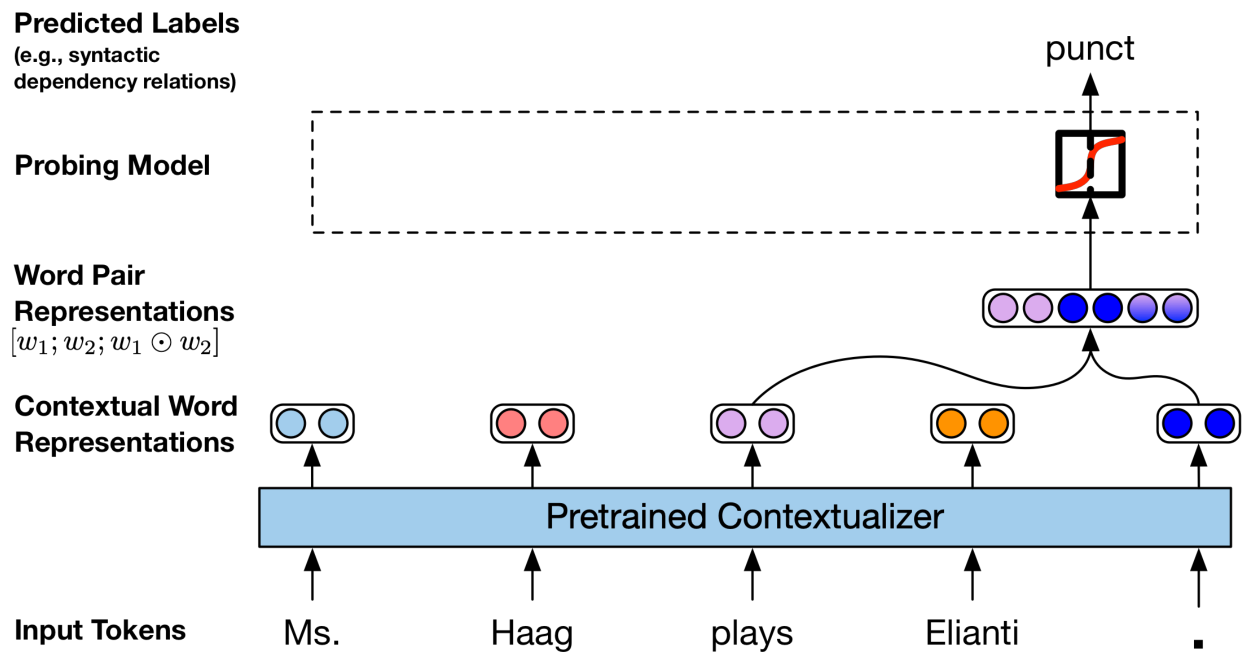
This **combined representation** is now the input to our probing model,



and the probing model is trained to predict information about the relationship between the tokens, for instance, their syntactic dependency relation.

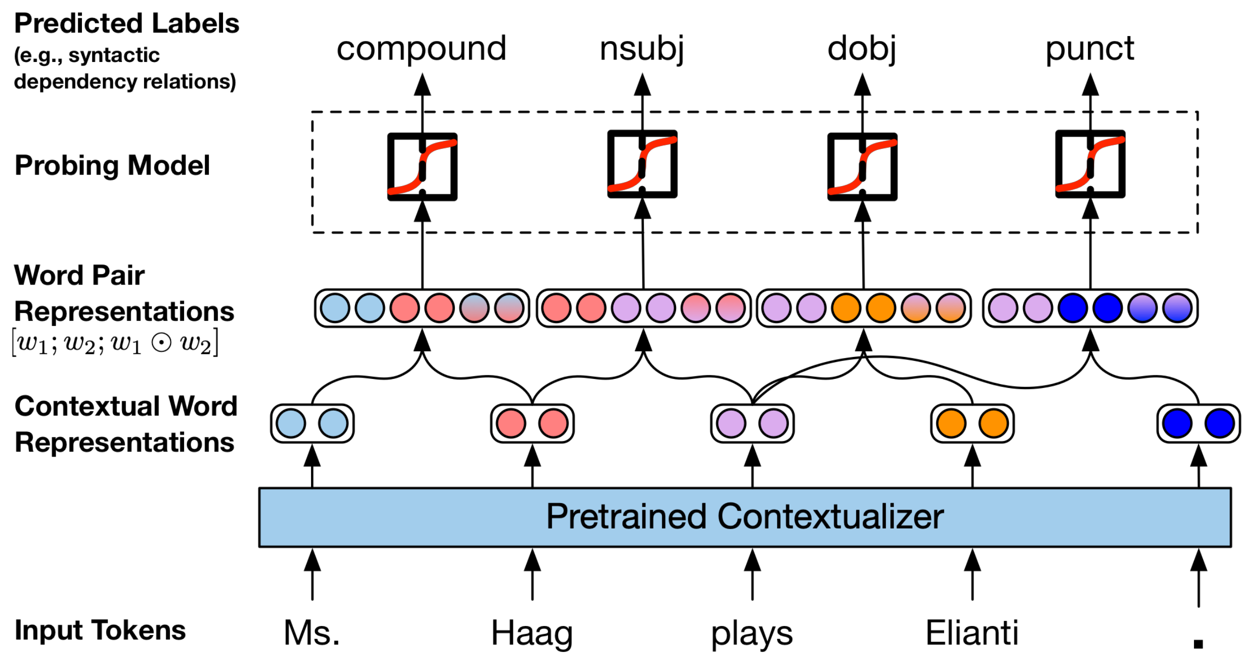
[Belinkov, 2018; Blevins et al., 2018; Tenney et al., 2019]

Pairwise Probing



We can probe arbitrary pairs like "plays" and the period here.

Pairwise Probing



To be clear, in this setting, the only trainable parameters are **still** in the probing model. When we combine contextual word representations to form word-pair representations, we do so without using any extra parameters.

Probing Model Setup

- Contextualizer weights are always frozen.
- Results are from the highest-performing contextualizer layer.
- We use a linear probing model.

In terms of probing model setup, we always freeze the contextualizer weights---only the probing model parameters are updated during training.

In addition, we probe each contextualizer layer, and the results are from the highest-performing layer (unless otherwise stated).

Lastly, we use a linear probing model, which limits its capacity and minimizes the number of external parameters used in our study.

Contextualizers Analyzed

With regard to the contextualizers that we study, we look at three main families.

[Peters et al., 2018a,b]

Contextualizers Analyzed

ELMo
Bidirectional language model (BiLM) pretraining on 1B Word Benchmark

2-layer LSTM (ELMo original)

4-layer LSTM (ELMo 4-layer)

6-layer Transformer (ELMo transformer)

First is ELMo, which is a bidirectional language model. We look at 3 ELMo models, one with a 2-layer LSTM, one with a 4-layer LSTM, and one with a 6-layer transformer.

[Peters et al., 2018a,b; Radford et al., 2018]

Contextualizers Analyzed

ELMo

Bidirectional language model (BiLM) pretraining on 1B Word Benchmark

2-layer LSTM (ELMo original)

4-layer LSTM (ELMo 4-layer)

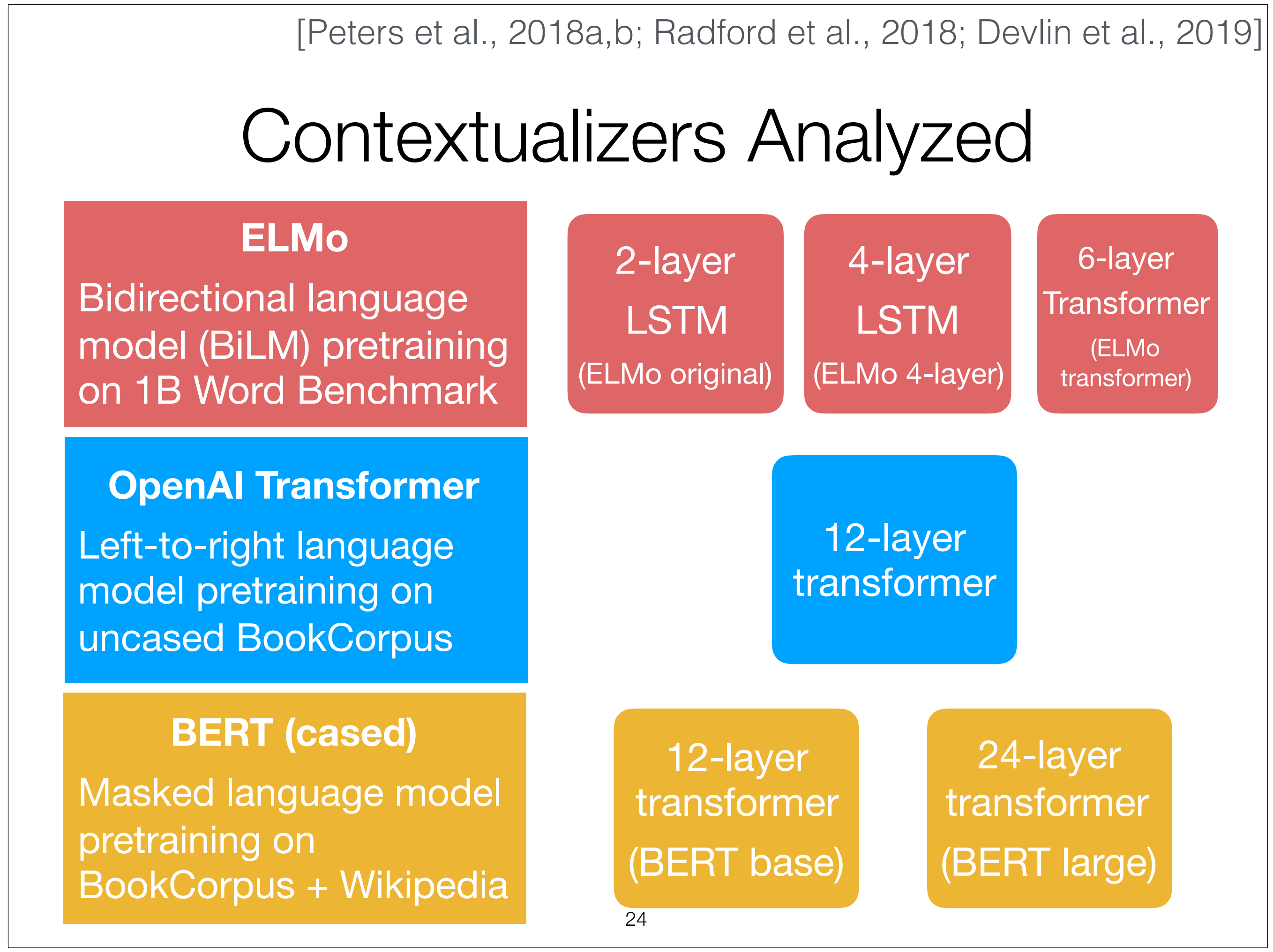
6-layer Transformer (ELMo transformer)

OpenAI Transformer

Left-to-right language model pretraining on uncased BookCorpus

12-layer transformer

We also look at the OpenAI transformer, which is a left-to-right language model. This is a 12-layer transformer, and it's also known as GPT version 1.



Lastly, we look at the two BERT cased models, which are pretrained on masked language modeling and next sentence prediction. We look at BERT base, which is a 12-layer transformer, and BERT-large, which is a 24-layer transformer.

Note that you can't make fair comparisons about pretraining strategies for contextualizers that aren't in the same row, because they aren't trained on the same data. So, it's fair to compare any of the three ELMo models against each other, but it's not fair to compare them to the BERT models.

(1) Probing Contextual Representations

Question: Is the information necessary for a variety of core NLP tasks linearly recoverable from contextual word representations?

Answer: Yes, to a great extent! Tasks with lower performance may require fine-grained linguistic knowledge.

Coming back to our first question, I'll talk about the results from probing.

Examined 17 Diverse Probing Tasks

- Part-of-Speech Tagging
- CCG Supertagging
- Semantic Tagging
- Preposition supersense disambiguation
- Event Factuality
- Syntactic Constituency Ancestor Tagging
- Syntactic Chunking
- Named entity recognition
- Grammatical error detection
- Conjunct identification
- Syntactic Dependency Arc Prediction
- Syntactic Dependency Arc Classification
- Semantic Dependency Arc Prediction
- Semantic Dependency Arc Classification
- Coreference Arc Prediction

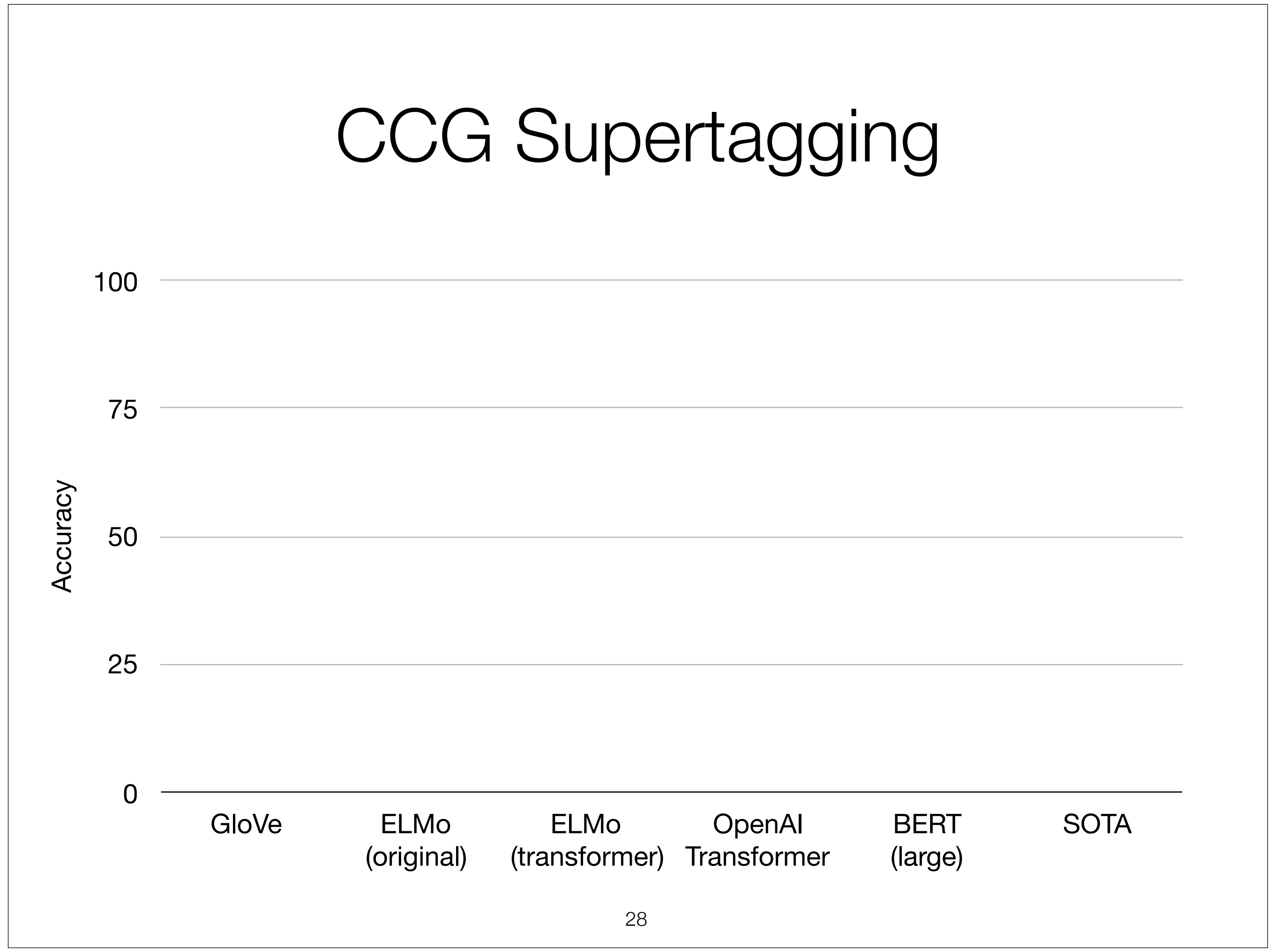
To better characterize the strengths and limitations of contextual word representations, we built a suite of 17 diverse probing tasks.

Linear Probing Models Rival Task-Specific Architectures

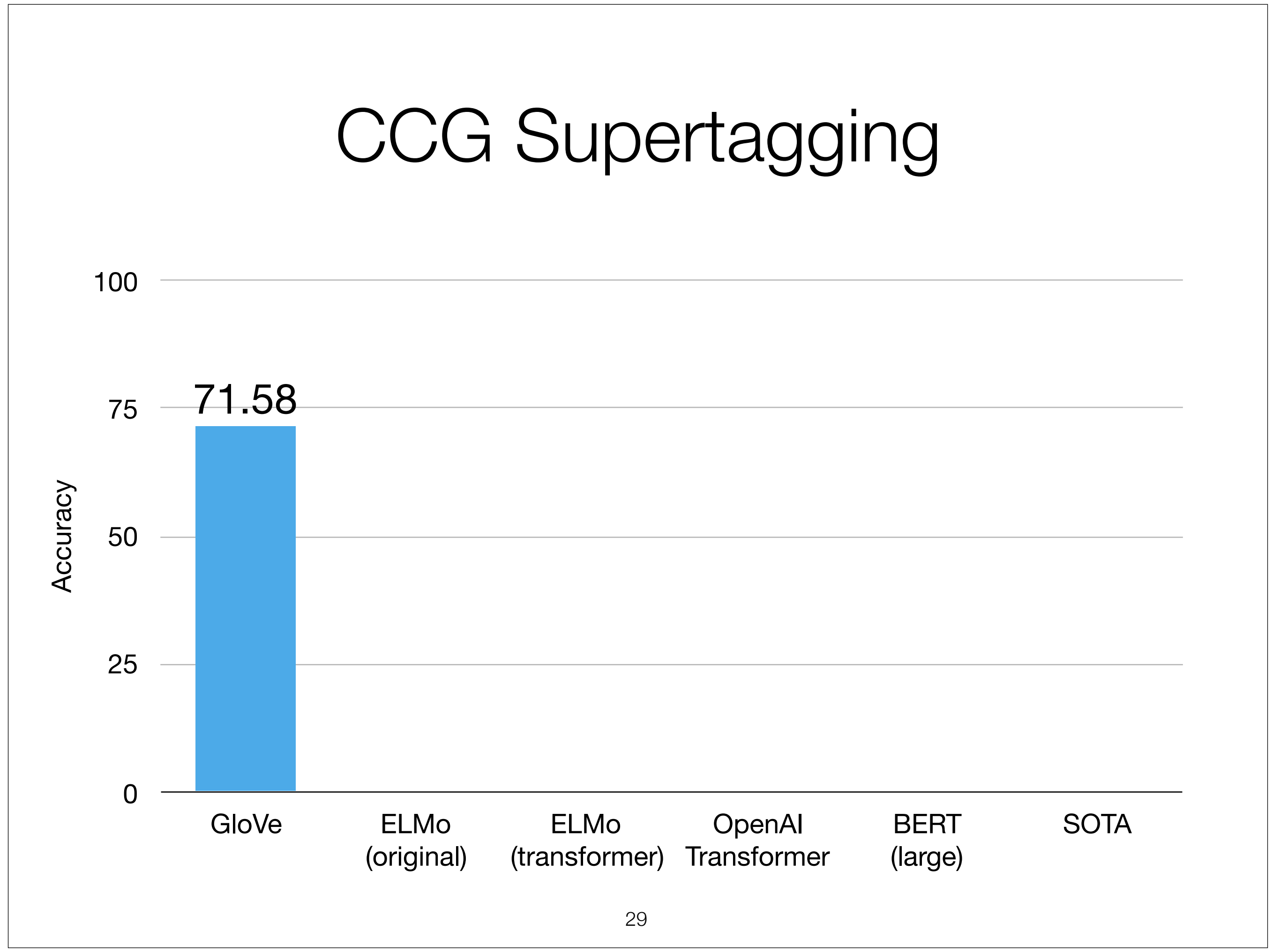
- Part-of-Speech Tagging
- CCG Supertagging
- Semantic Tagging
- Preposition supersense disambiguation
- Event Factuality
- Syntactic Constituency Ancestor Tagging
- Syntactic Chunking
- Named entity recognition
- Grammatical error detection
- Conjunct identification
- Syntactic Dependency Arc Prediction
- Syntactic Dependency Arc Classification
- Semantic Dependency Arc Prediction
- Semantic Dependency Arc Classification
- Coreference Arc Prediction

8 of these tasks are established tasks, in the sense that there's a prior state-of-the-art to compare to.

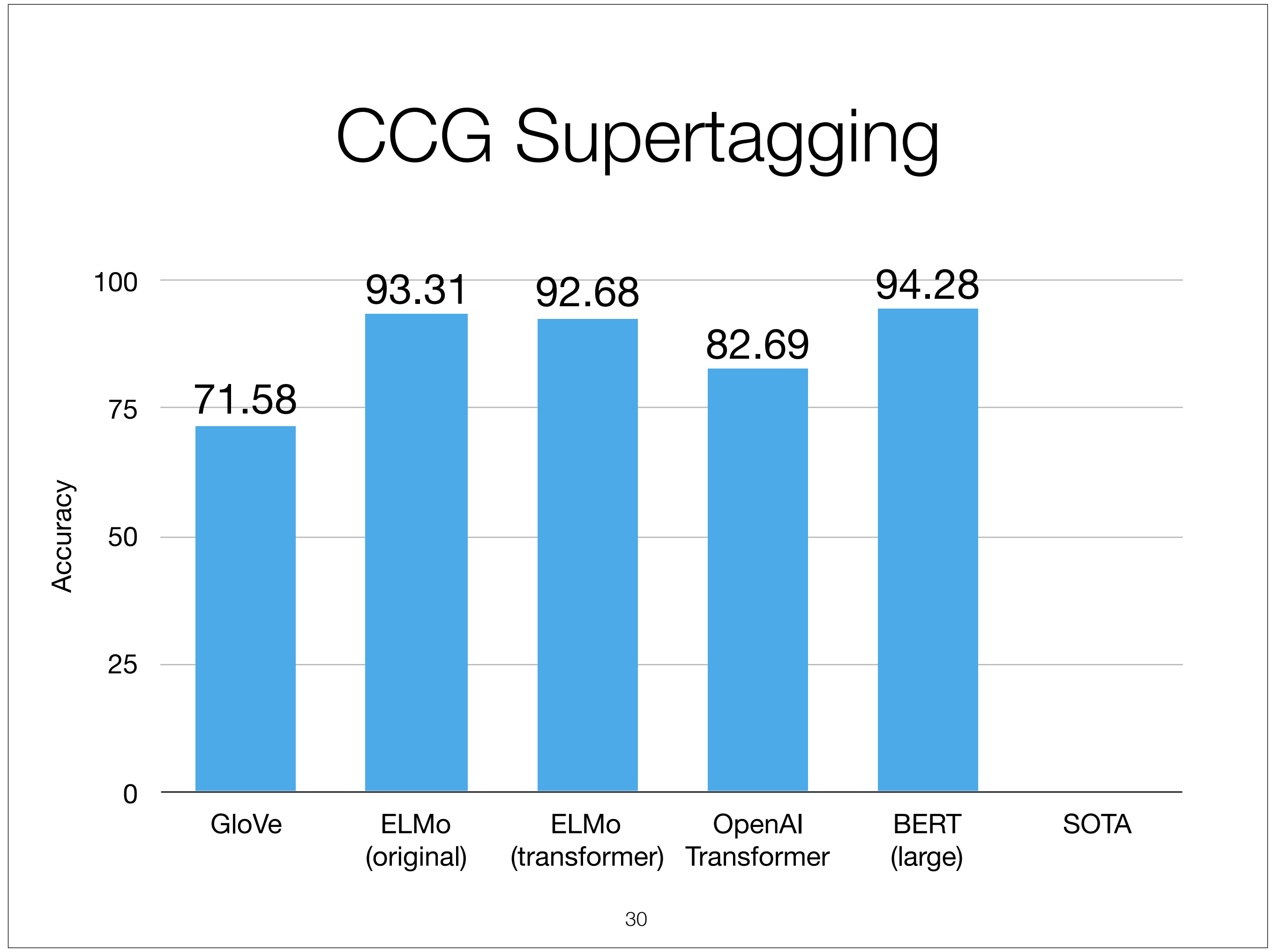
On 7 of these 8 tasks, we see that a linear probing model trained on top of frozen contextual word representations is competitive with prior state-of-the-art, task-specific models that don't use contextual word representations.



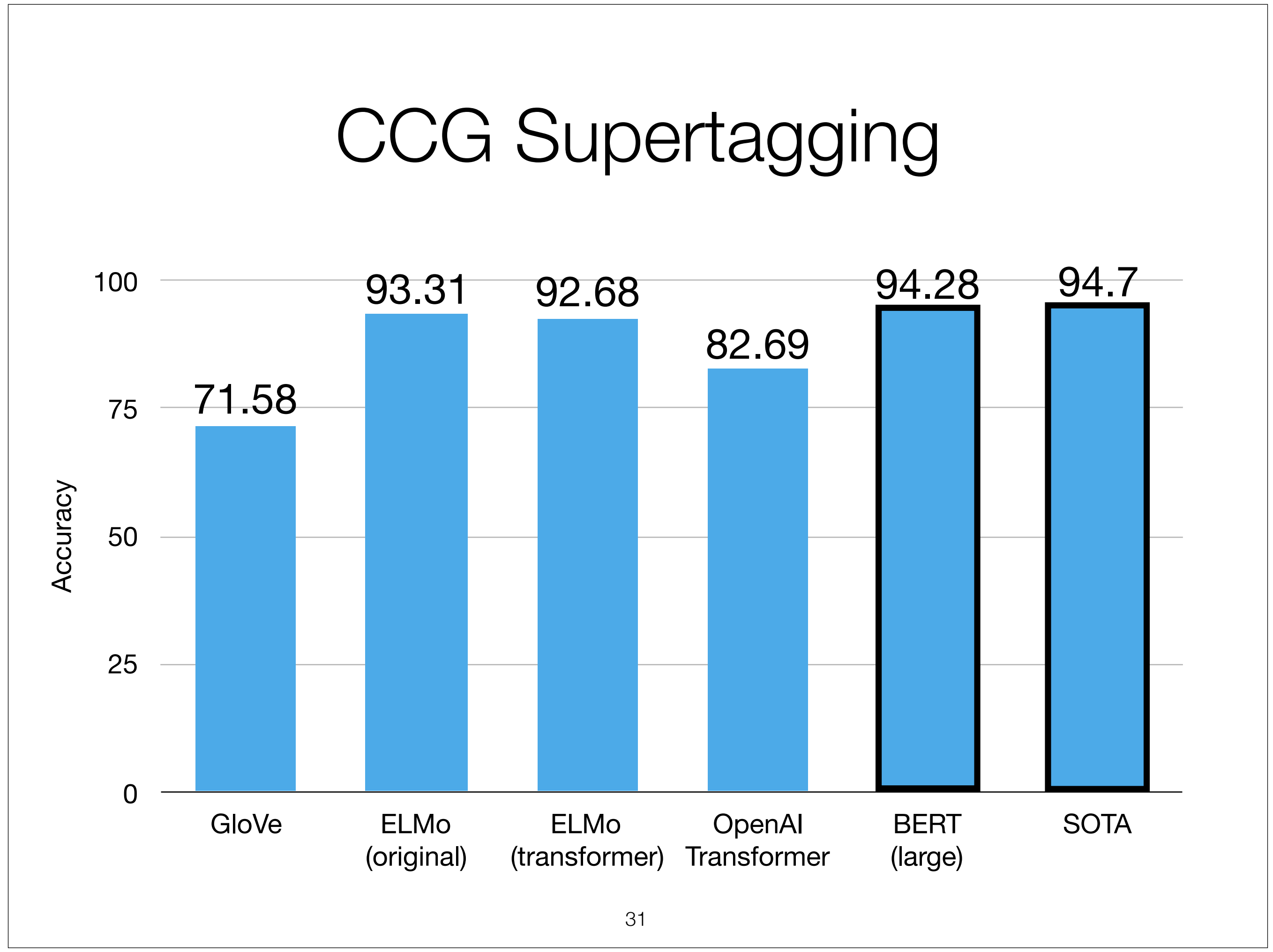
For instance, one of the tasks we looked at was CCG supertagging.



We train a probing model on top of GloVe vectors as a baseline.



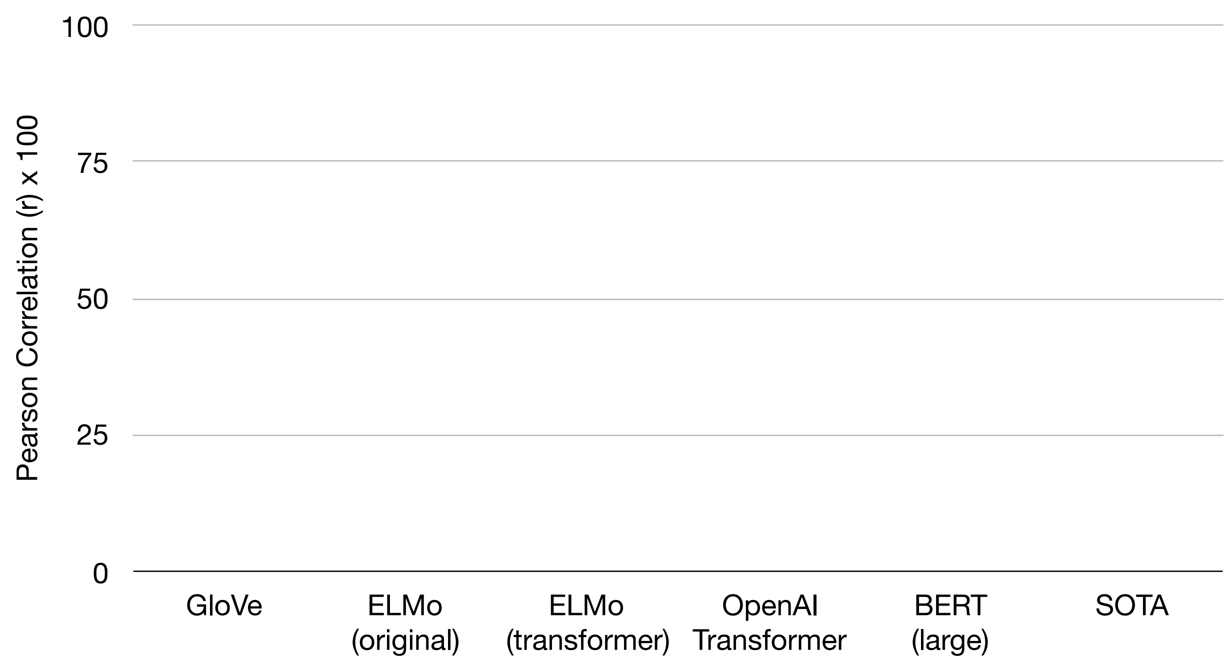
Probing models that use contextual word representations do much better than the GloVe baseline, with BERT large reaching an accuracy of 94.28.

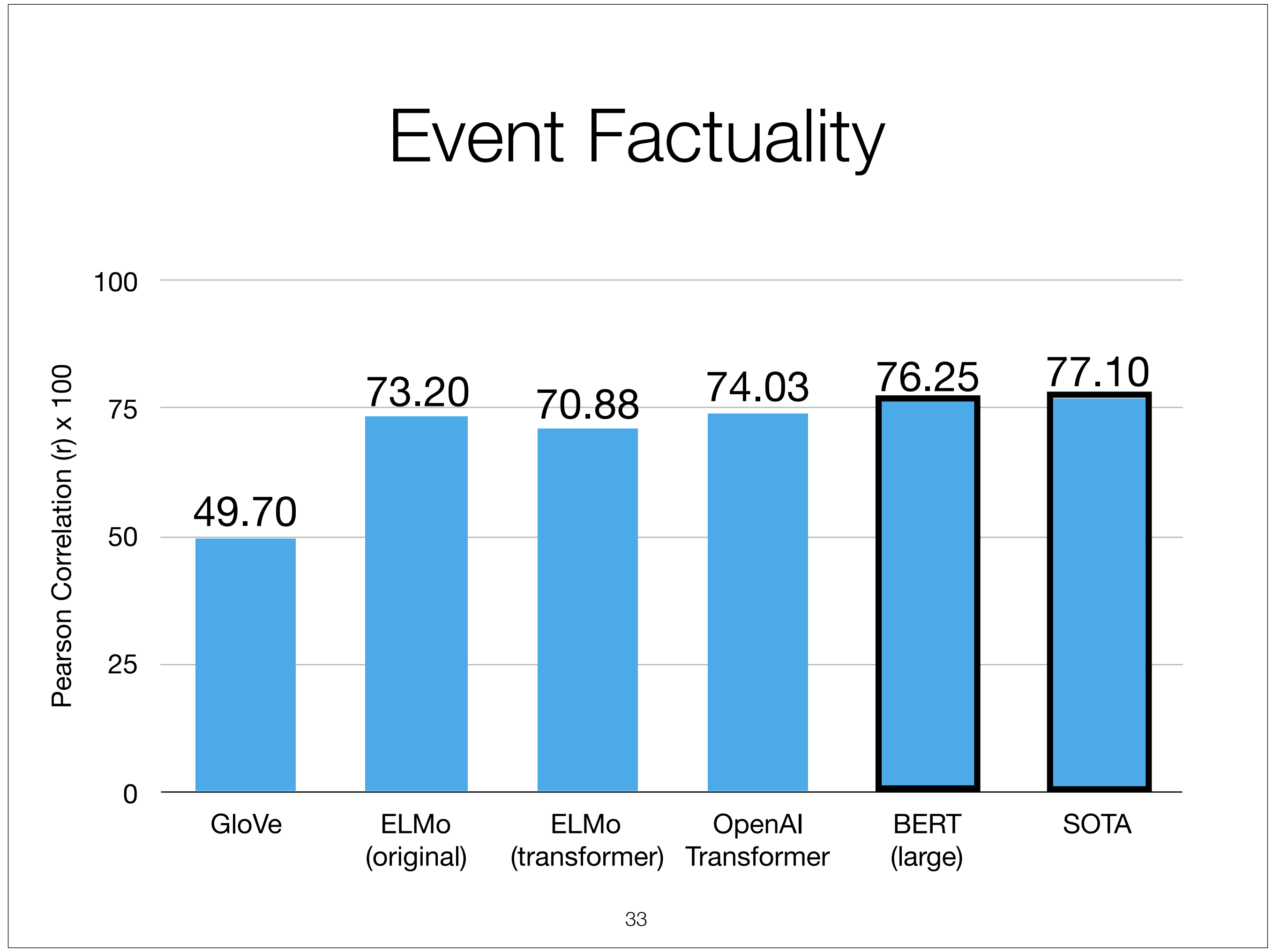


This is surprisingly close to the prior state-of-the-art without pretraining, which achieved an accuracy of 94.7 .

So, contextual word representations clearly contain features that are predictive of CCG supertags.

Event Factuality





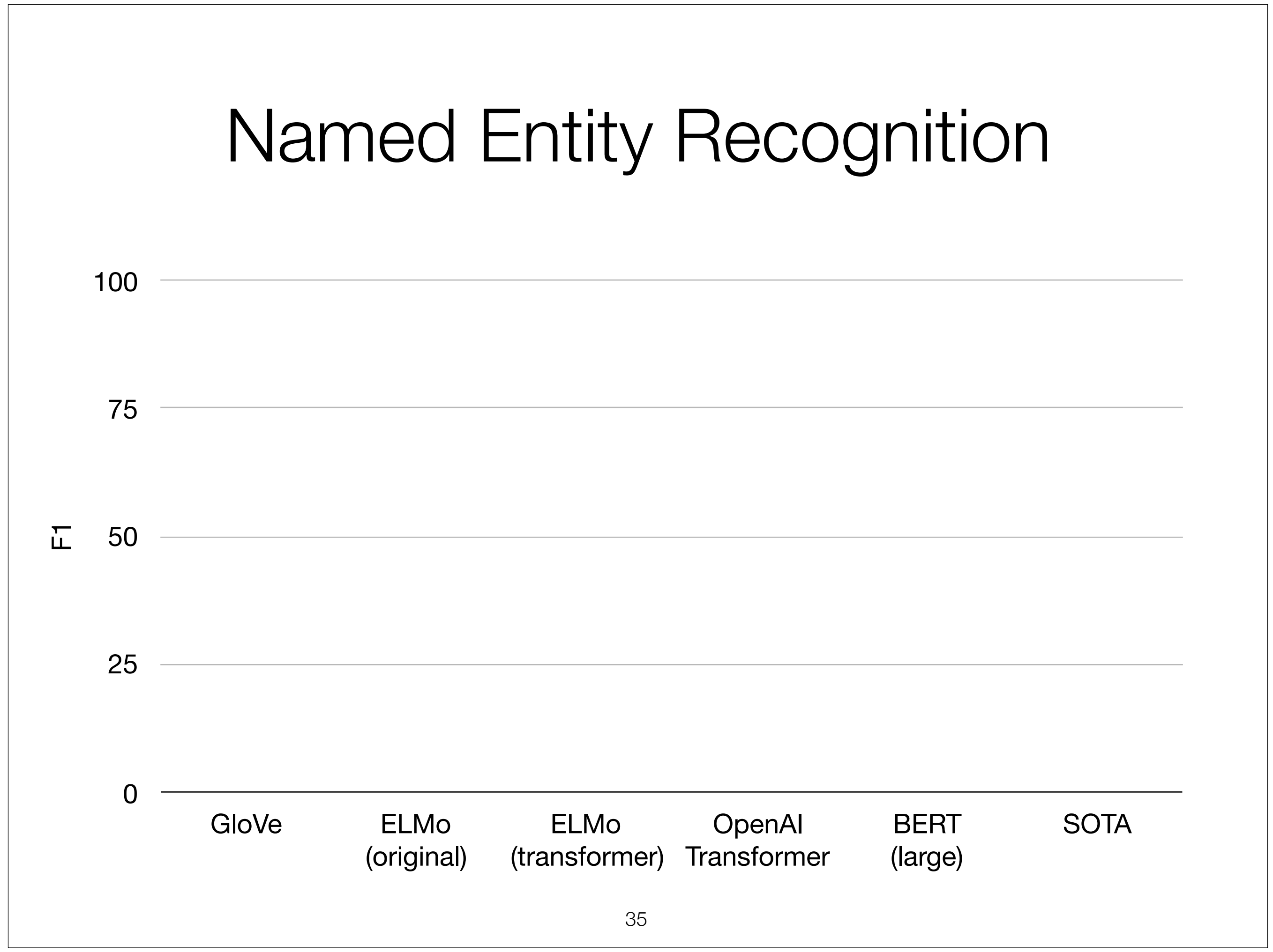
We reach a similar conclusion for the more semantic task of event factuality, where the model is given a predicate and must predict whether it happened or not.

But Linear Probing Models Underperform on Some Tasks

- Tasks that linear model + contextual word representation performs poorly may require more fine-grained linguistic knowledge.

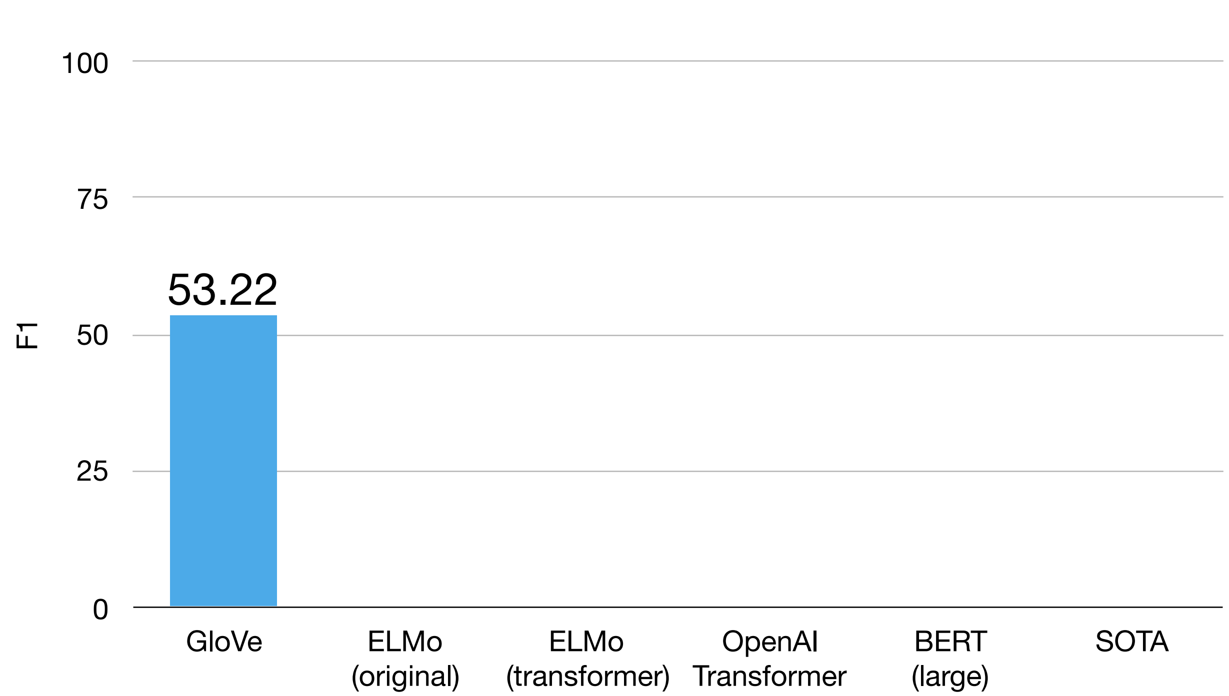
- In these cases, task-specific contextualization leads to especially large gains. See the paper for more details.

However, we also saw that, when linear probing models trained on top of contextual word representations failed to do well on tasks, they seem to require fine-grained linguistic knowledge.

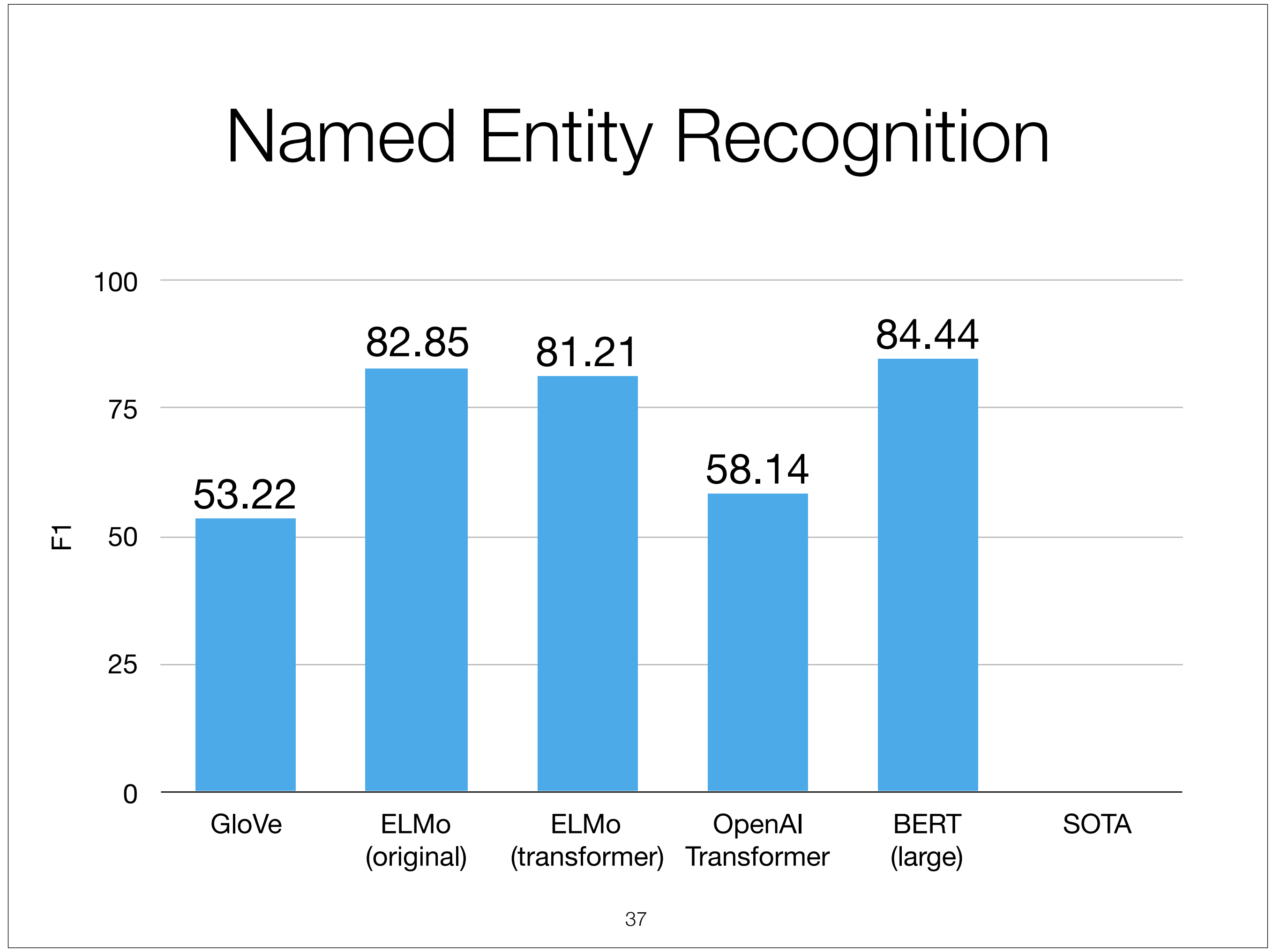


For instance, we looked at the task of named entity recognition, or NER.

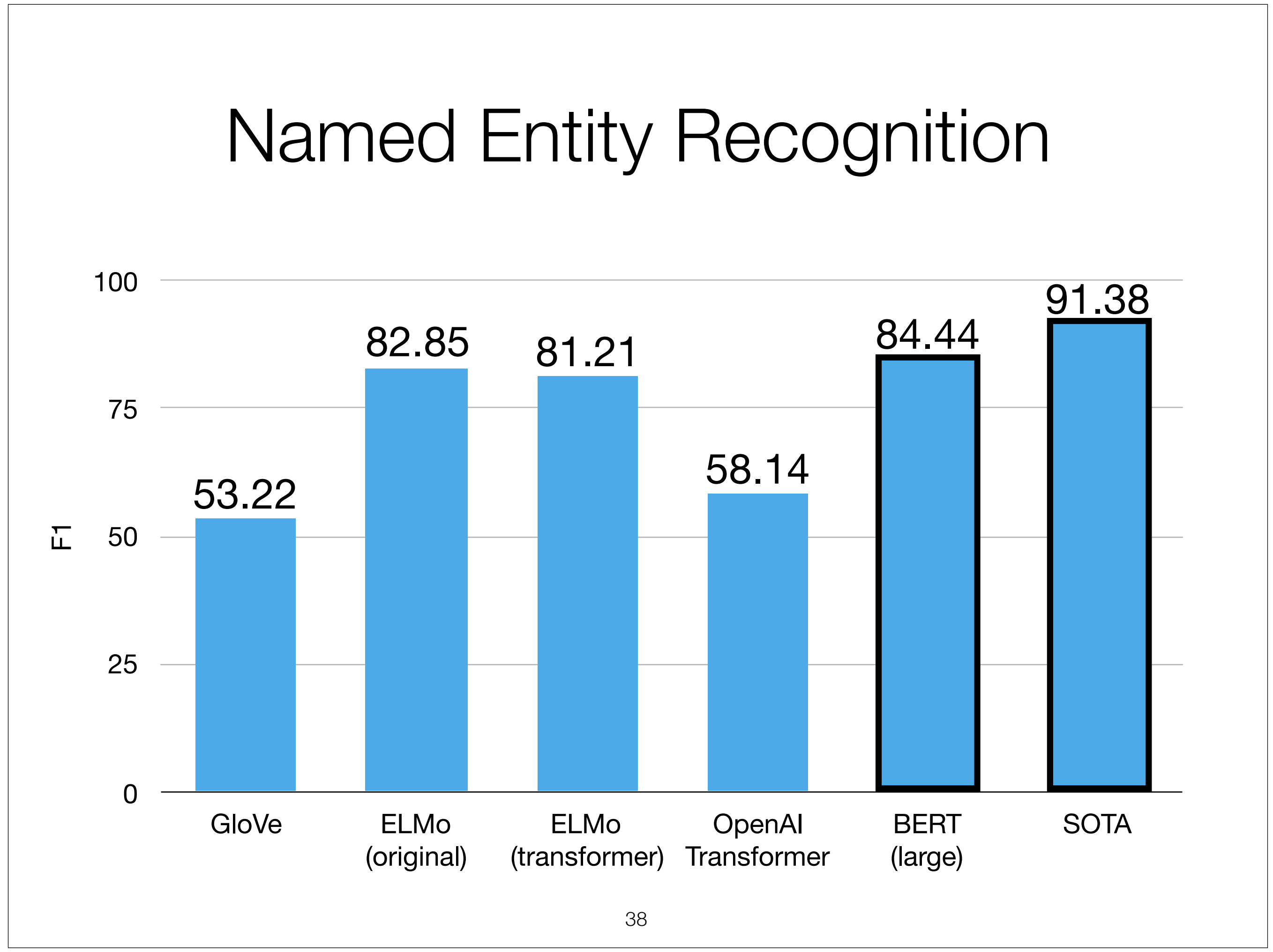
Named Entity Recognition



So, once again, we run our GloVe baseline.



We also see in this case that contextual word representations are significantly more predictive than GloVe.

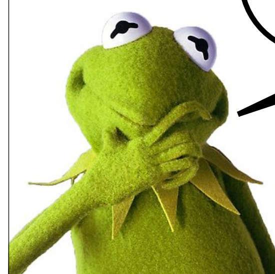


but these numbers are still quite far from the state-of-the-art.

We might see a larger gap here because entities are rather rare in text, and these contextual word representations probably don't capture too much about them simply because it isn't useful enough for their pretraining task of predicting the next word or a masked-out word.

(2) How Does Transferability Vary?

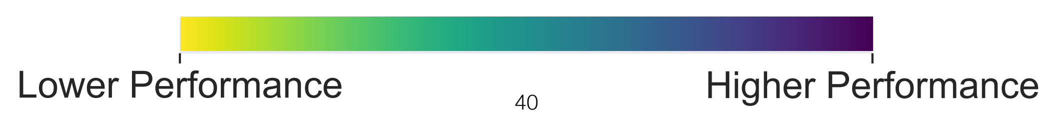
Question: How does transferability vary across contextualizer layers?



Answer: First layer in LSTMs is the most transferable. Middle layers for transformers.

We also looked at *layerwise* patterns in transferability---what sort of variation do we see, and why do we see it?

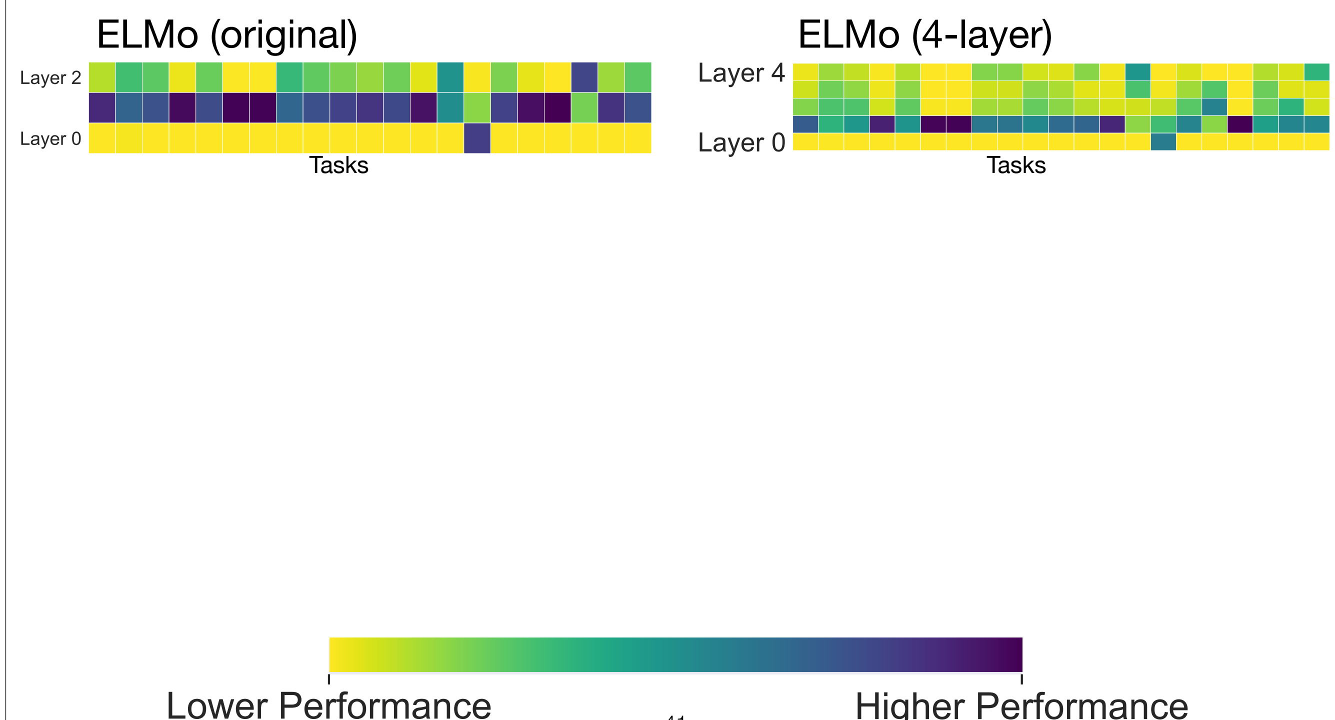
Layerwise Patterns in Transferability



So, starting off...

Layerwise Patterns in Transferability

LSTM-based Contextualizers

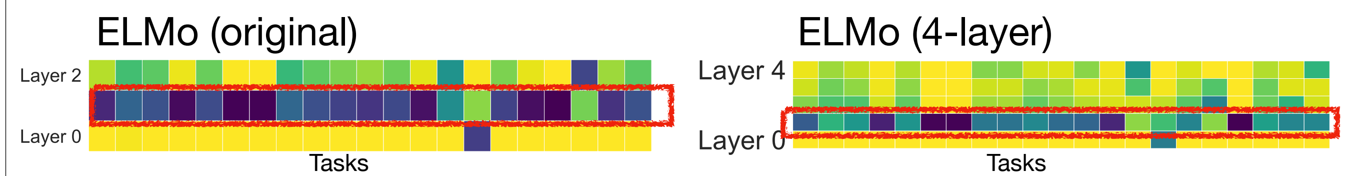


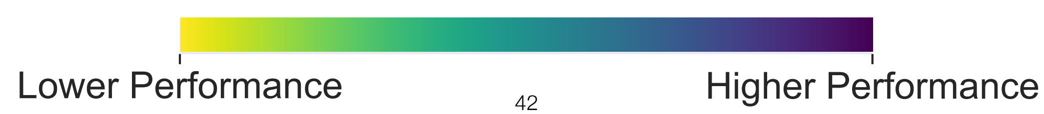
We looked at LSTM-based contextualizers, which are the 2-layer and 4-layer LSTM ELMo models.

In the heatmap, each row is a layer of the contextualizer, and each column is a probing task.

Layerwise Patterns in Transferability

LSTM-based Contextualizers





The heatmap for LSTM-based contextualizers shows a marked dark band across the 1st layer outputs---the 1st layer is consistently the strongest on probing tasks, and thus it's the most transferable.

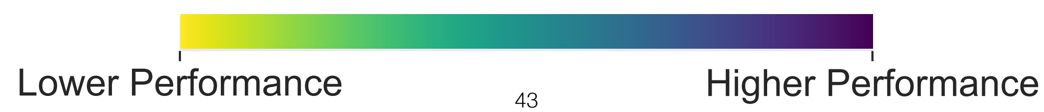
Layerwise Patterns in Transferability

LSTM-based Contextualizers

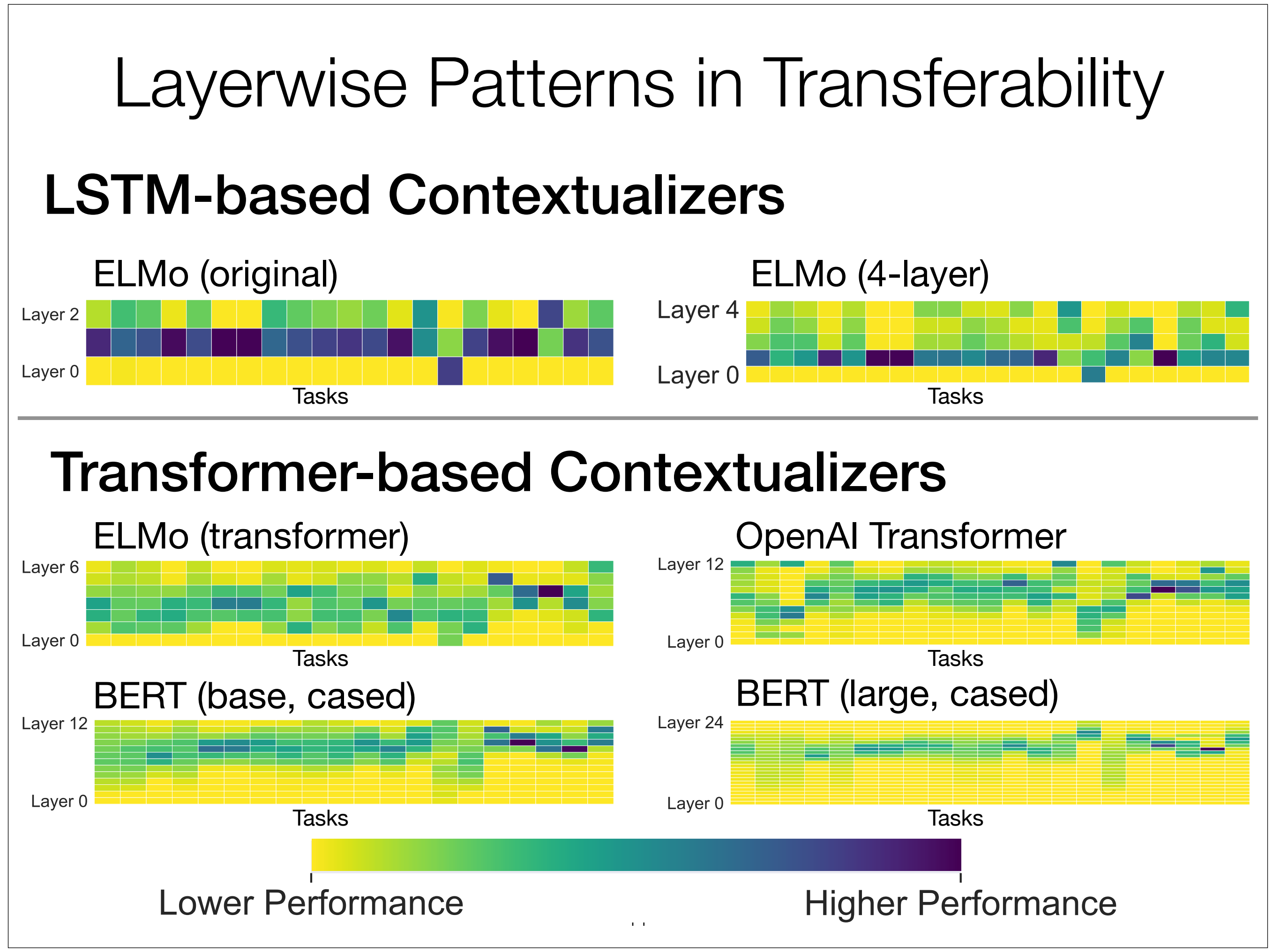
ELMo (original)

ELMo (4-layer)

Transformer-based Contextualizers



In contrast, if we look at transformer-based contextualizers...



We see that there is no such clear dark band---no one layer is the most transferable. Instead, we see wider dark bands around the middle layers.

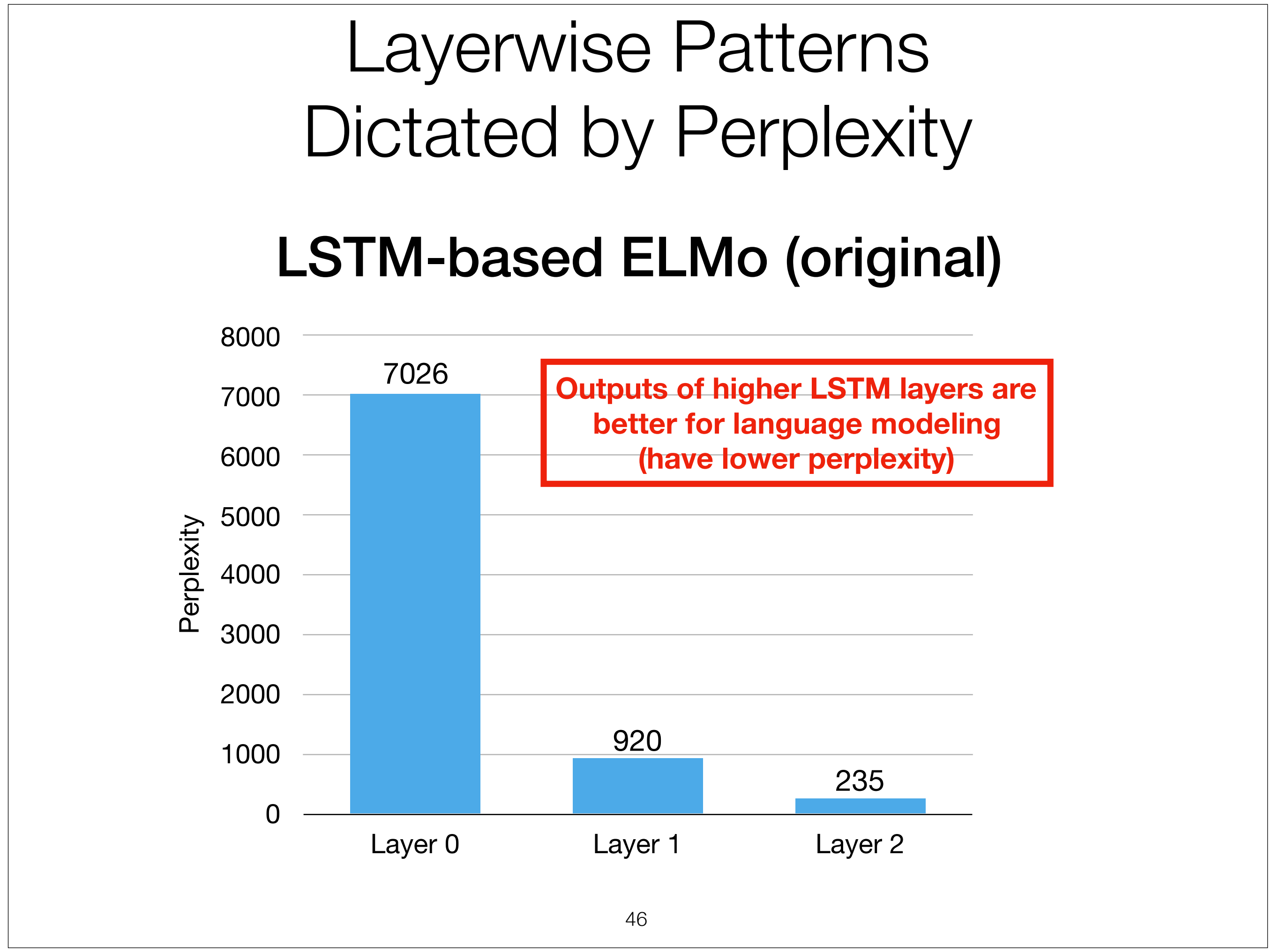
This points to concrete differences in how LSTMs and transformers store information, and this would be an interesting direction for future work.

(3) Why Does Transferability Vary?

Question: **Why** does transferability vary across contextualizer layers?

Answer: It depends on pretraining task-specificity!

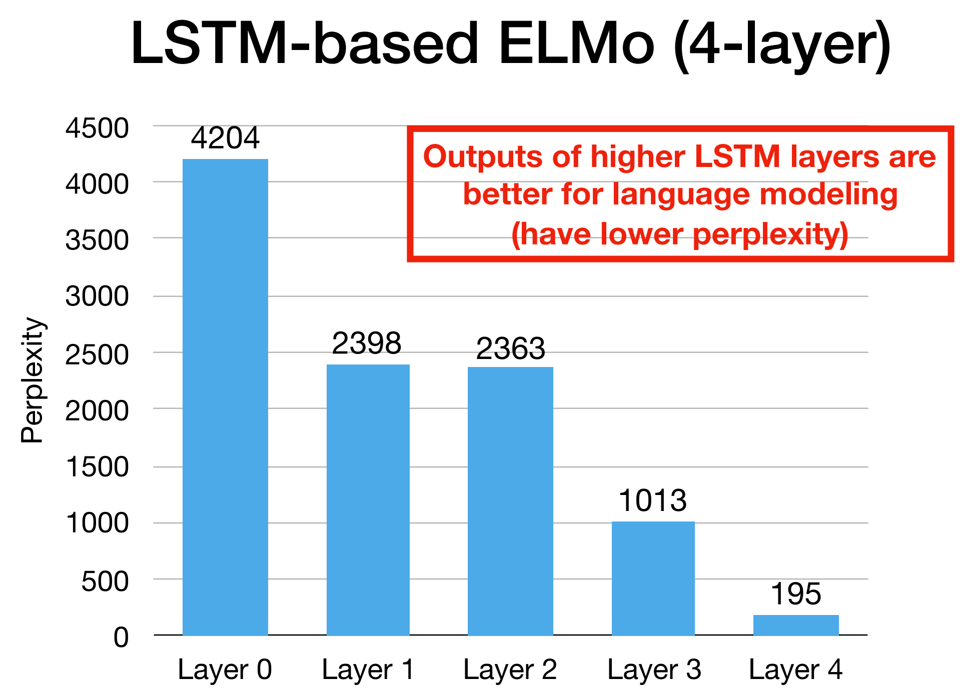
Looking at layerwise performance on the tasks, we also didn't see higher layers doing better with higher-level semantics. So, what dictates these patterns?



In short, we see perplexity---since these contextualizers are pretrained on language modeling, the higher layers are tuned toward optimizing perplexity.

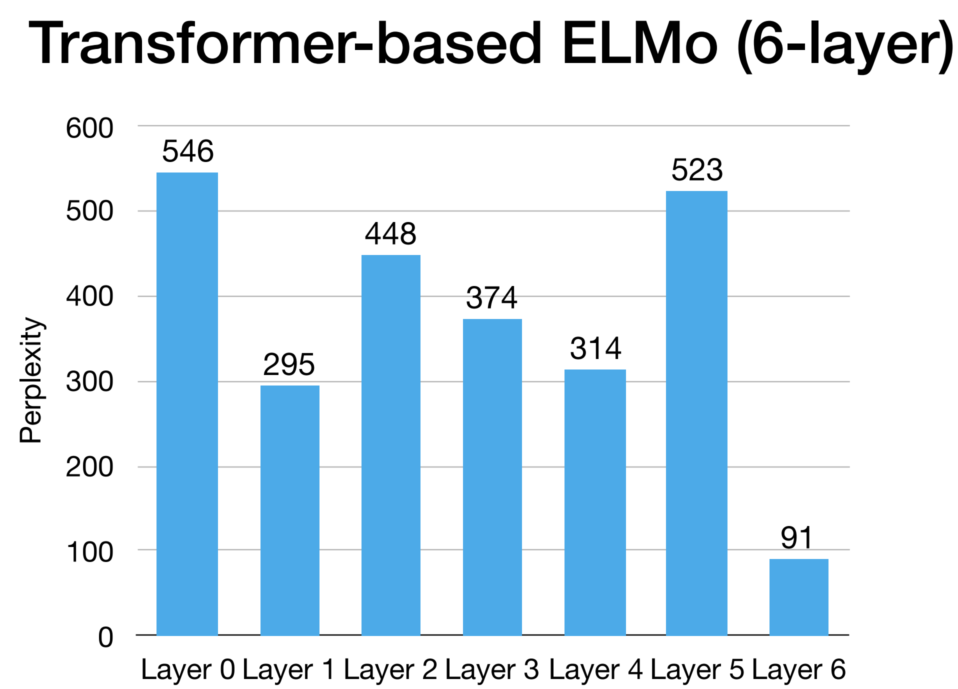
Past work has shown that representations from higher-level layers seem to do better on higher-level tasks. Instead, it seems likely that higher-level layers simply focus on encoding what's useful for their pretraining task. It just happens that certain high-level semantic phenomena are *incidentally* useful for the contextualizer's pretraining task, leading to their presence in higher layers.

Layerwise Patterns Dictated by Perplexity



We see a similar trend in 4-layer LSTMs

Layerwise Patterns Dictated by Perplexity



Although transformers do not show the same monotonic trend.

(4) Alternative Pretraining Objectives

Question: How does language model pretraining compare to alternatives?

Answer: Even with 1 million tokens, language model pretraining yields the most transferable representations.

But, transferring between related tasks does help.

Lastly, to better understand what makes language-model derived contextual word representations so transferable, we study alternatives to language model pretraining.

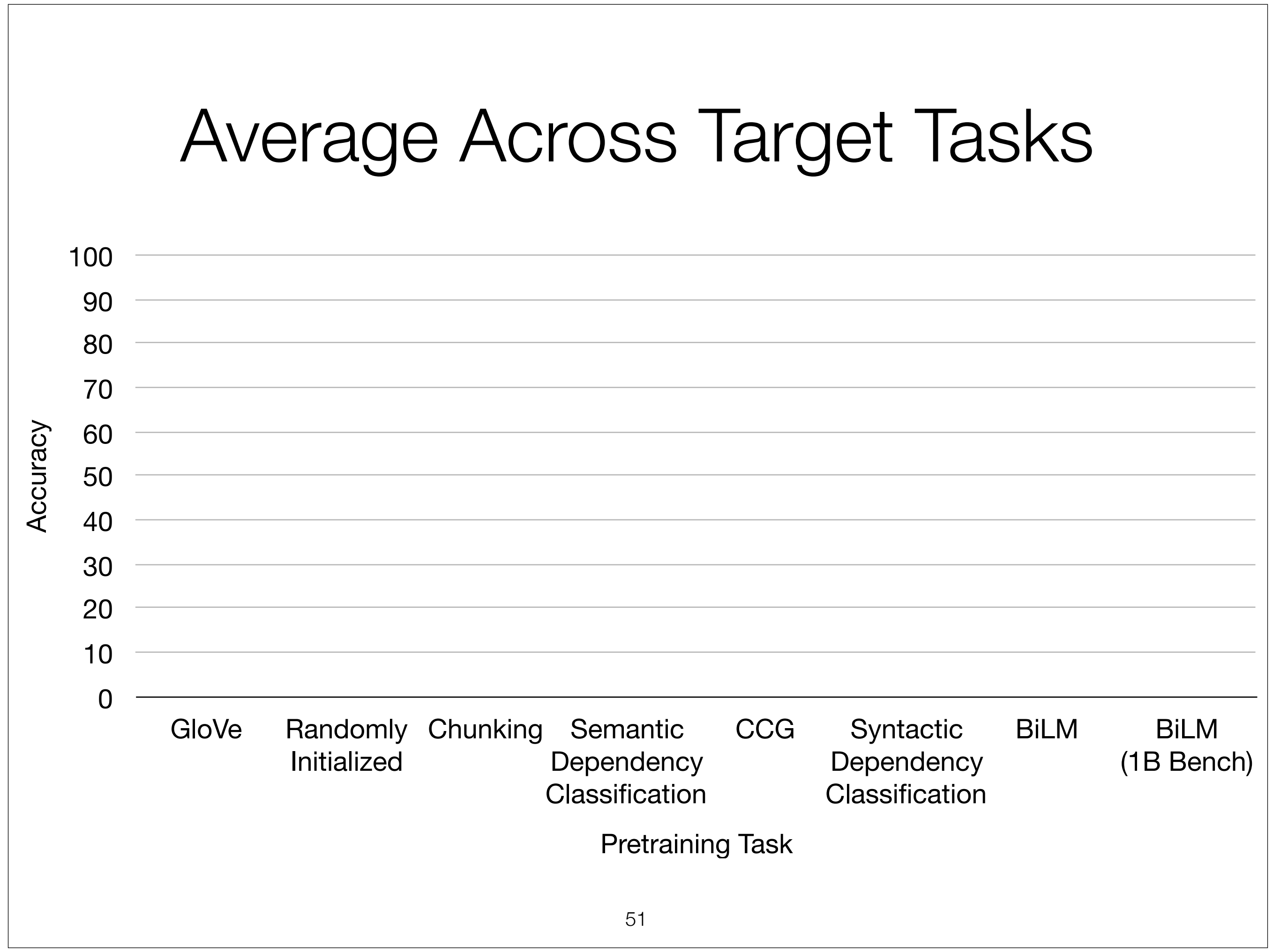
Investigating Alternatives to Language Model Pretraining

- How does the language modeling as a pretraining objective compare to explicitly supervised tasks?
- Pretrain ELMo (original)-architecture contextualizer on the Penn Treebank, with a variety of different objectives.
- Evaluate how well the resultant representations transfer to target (held-out) tasks.

In particular, language modeling is useful because it doesn't require any labeled data, so you can pretrain on massive datasets. But, is its self-supervised nature the only benefit? Or is language modeling just a good pretraining objective unto itself, disregarding the fact that we can get lots of data for it?

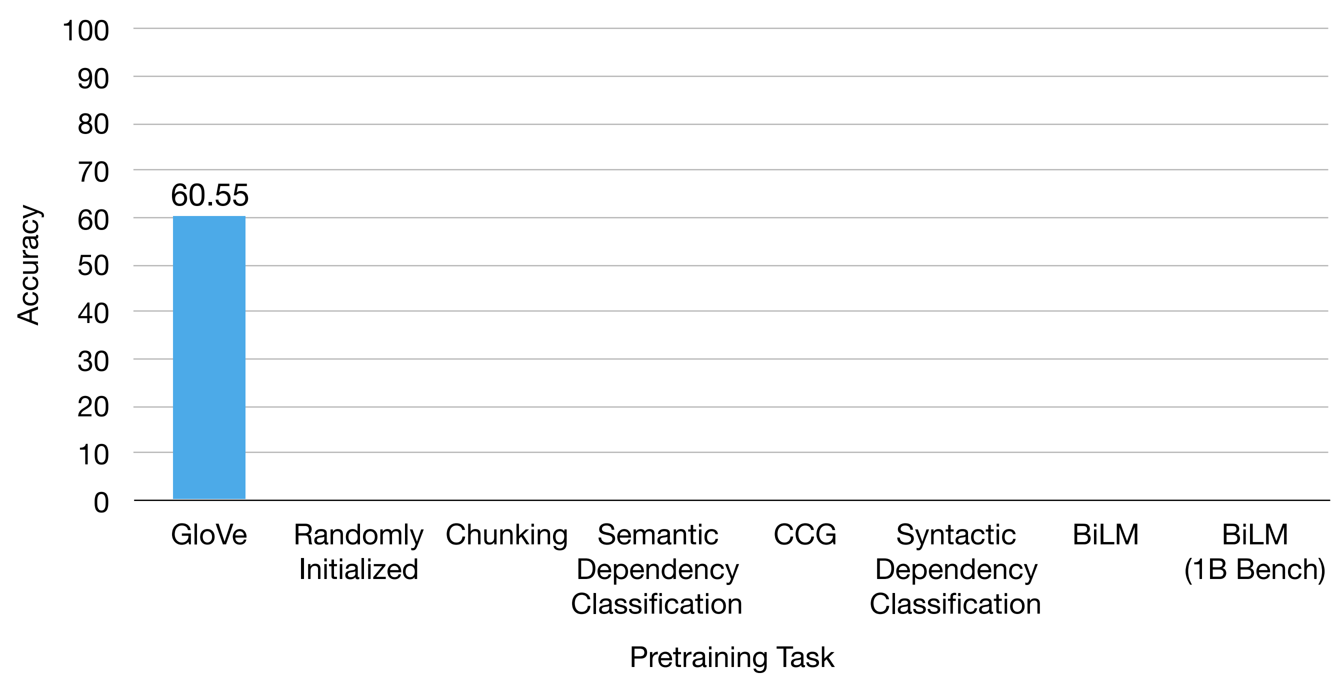
To test this, we pretrain 2-layer LSTM contextualizers on the Penn Treebank, with a variety of different objectives. We then evaluate how well each of the resultant representations transfers to target held-out tasks to compare language modeling to eleven supervised alternatives.

To be clear, this is a controlled experiment because we use the same pretraining method, contextualizer architecture, and dataset. The ***only*** thing that changes between experiments is the type of supervision that the contextualizer is pretrained on.

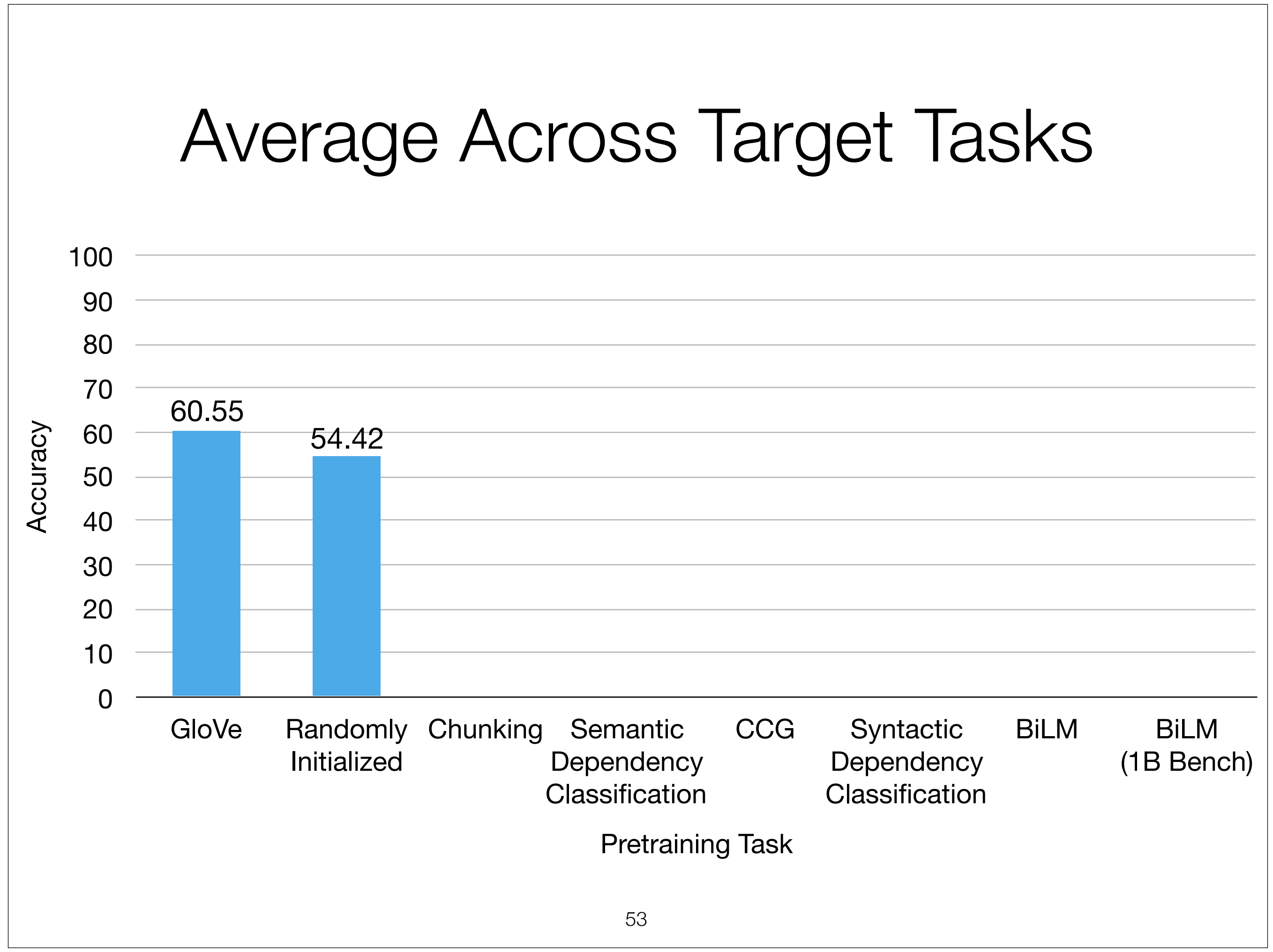


So moving to our results, I'll first show the average performance across target tasks when pretraining on a variety of objectives. See the paper for the full results.

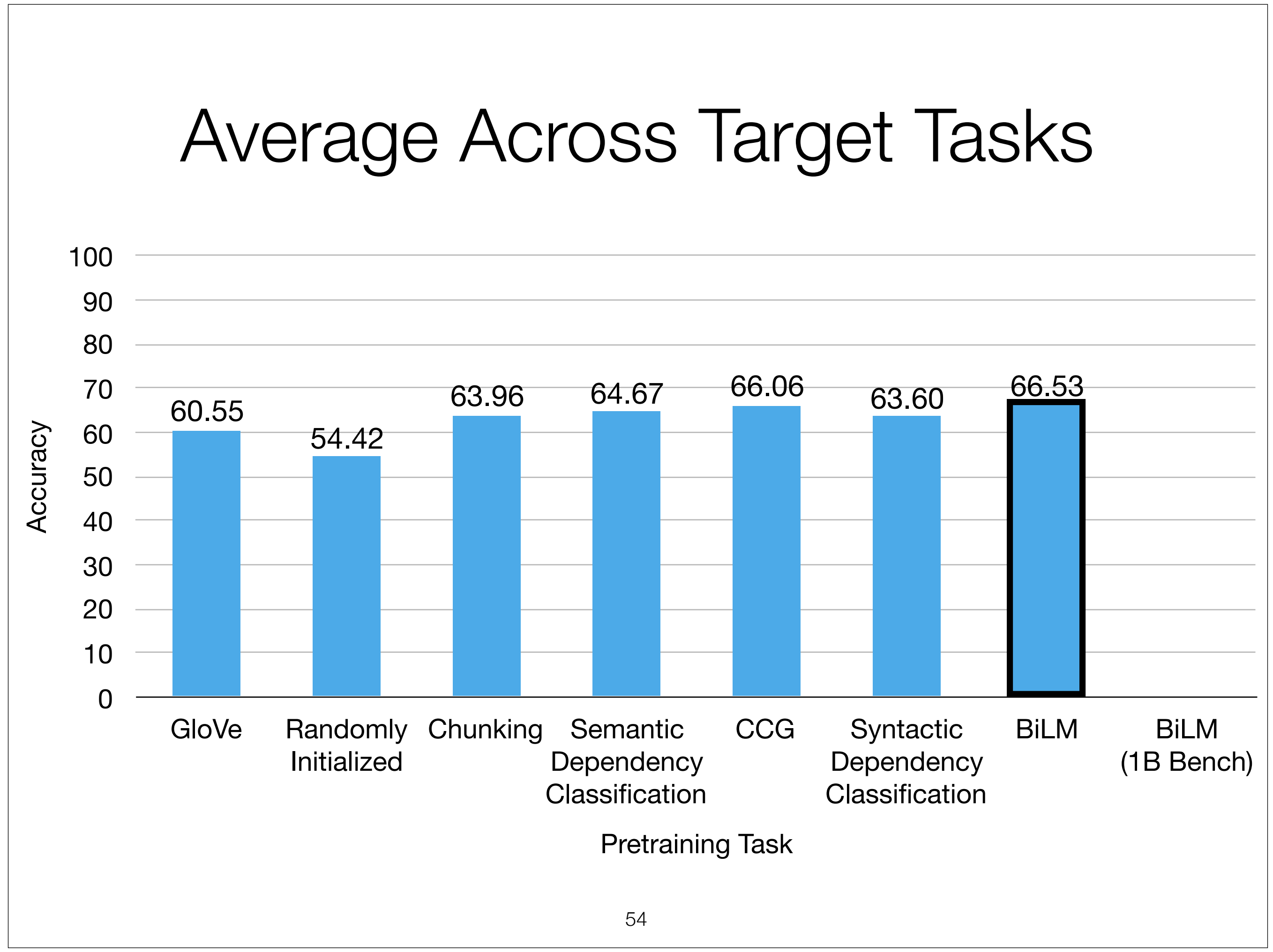
Average Across Target Tasks



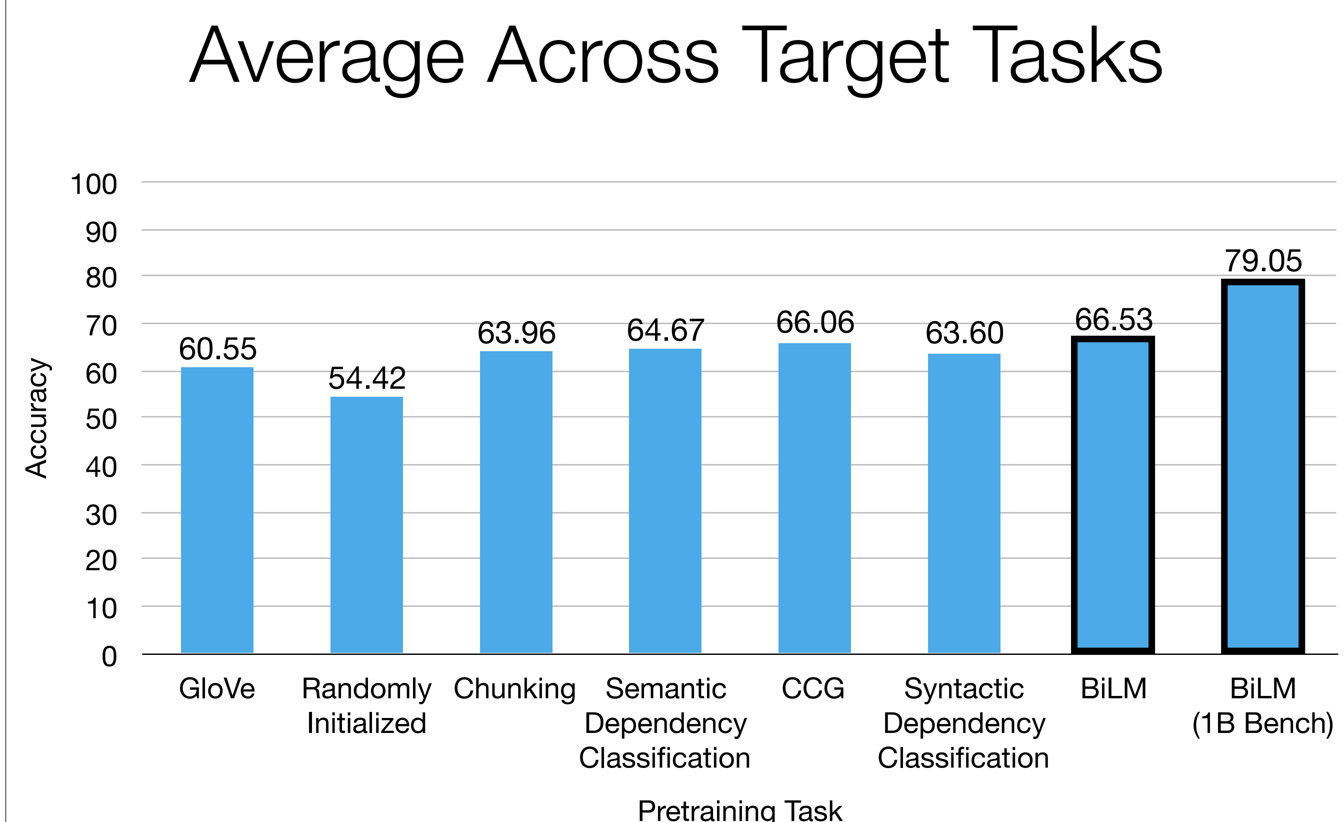
As a baseline, we took the average performance when we train our probing model on top of GloVe.



Inspired by recent work showing that contextualizers with random weights actually do quite well, our second baseline is a randomly-initialized, untrained model.



Now, when we pretrain on the Penn Treebank with different supervision signals, we see that any sort of pretraining does better than the GloVe and randomly initialized baselines. Among the eleven supervised pretraining tasks that we considered, bidirectional language modeling was the most transferable on average.

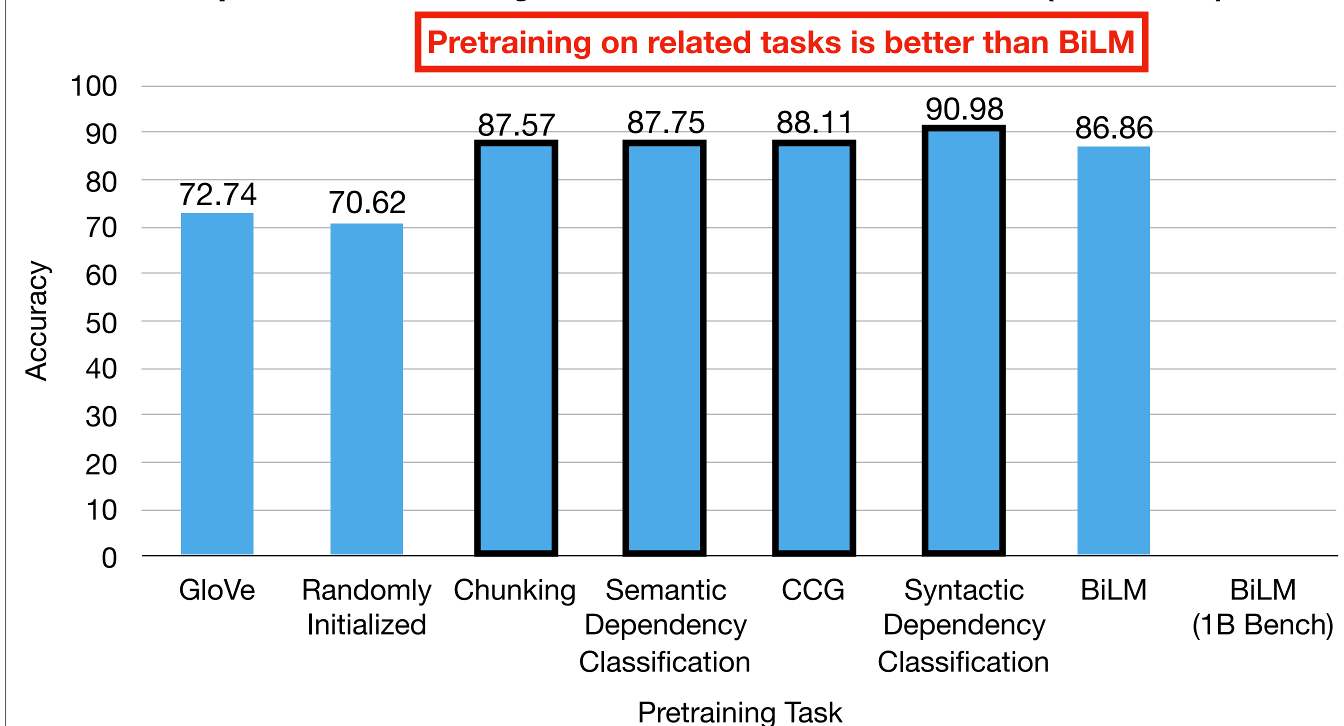


See Wang et al. (ACL 2019) "How to Get Past Sesame Street: Sentence-Level Pretraining Beyond Language Modeling" for more tasks + multitasking.

Just for reference, here's the performance when you pretrain the bidirectional language model on the 1 Billion Word Benchmark.

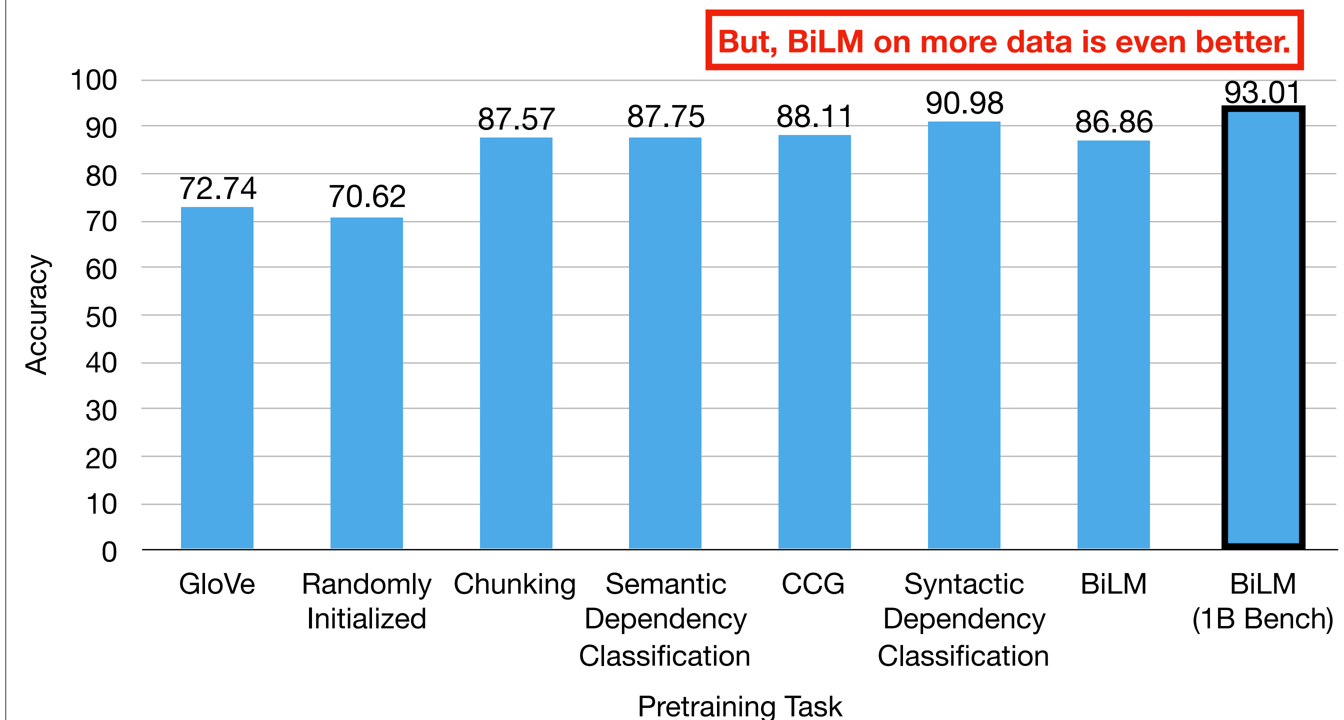
So, while training on more data is a large part of why language-model derived contextual word representations work well, bidirectional language modeling unto itself is also just a reasonably good task, at least compared to the alternatives we considered. Alex Wang et al have a paper at ACL this year that also supports the use of language modeling, and they see further gains from multitasking training as well.

Target Task: Syntactic Dependency Classification (EWT)

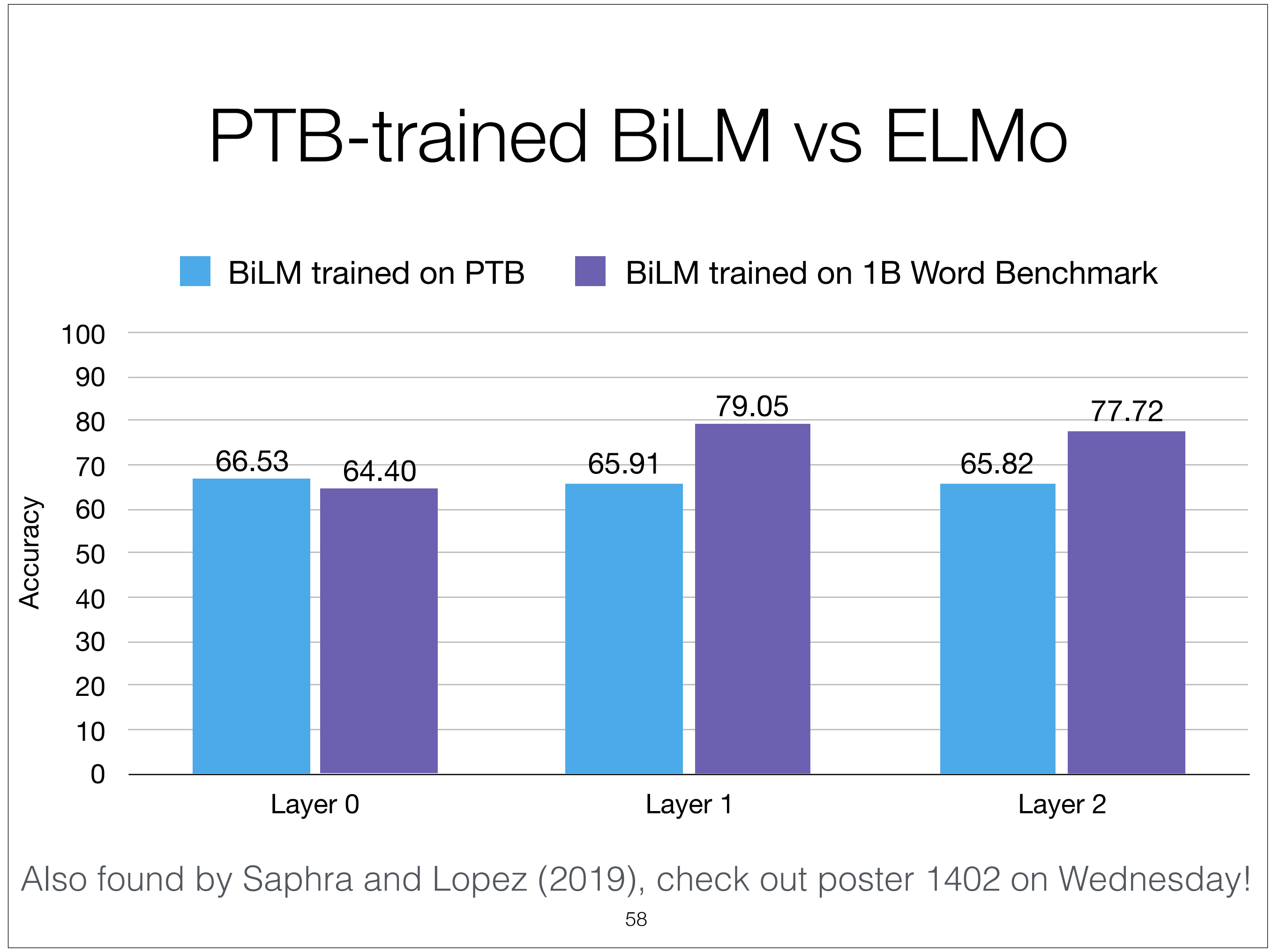


Looking at one task in particular, syntactic dependency arc classification, we see that pretraining on related tasks, which are the bolded bars, yields better performance than bidirectional language modeling. So, related task-transfer does help.

Target Task: Syntactic Dependency Classification (EWT)



However, just pretraining your bidirectional language model on more data yields even larger gains.



We also saw that layer 0 of the bidirectional language model is surprisingly performant---it does better than all other layers from all other pretraining tasks, and even surpasses layer 0 of a bidirectional language model trained on the 1 Billion Word Benchmark, which is orders of magnitude more data. This indicates that bidirectional language models learn lexical information first, and this drives its generalizability.

Naomi Saphra came to the same conclusion in her work, which studies the learning dynamics of language models---be sure to check out poster 1402 on Wednesday.

Online at: bit.ly/cwr-analysis-related

Some Related Work at NAACL

Wed. June 5, 10:30 – 12:00. ML & Syntax, Hyatt Exhibit Hall:

Understanding Learning Dynamics Of Language Models with SVCCA. Naomi Saphra and Adam Lopez.

Structural Supervision Improves Learning of Non-Local Grammatical Dependencies. Ethan Wilcox et al.

Analysis Methods in Neural Language Processing: A Survey. Yonatan Belinkov and James Glass.

Wed. June 5, 16:15–16:30. Machine Learning, Nicollet B/C:

A Structural Probe for Finding Syntax in Word Representations. John Hewitt and Christopher D. Manning.

Beyond the Saphra paper that I just mentioned, there's a bunch of other related work at NAACL that share our goal of understanding language models and their derived contextual word representations. If you found this talk interesting, you might find these presentations interesting as well.

There's a link to this slide above.

Takeaways

- Features from pretrained contextualizers are sufficient for high performance on a broad set of tasks.
- Tasks with lower performance might require fine-grained linguistic knowledge.
- Layerwise patterns in transferability exist. Dictated by how task-specific each layer is.
- Even on PTB-size data, BiLM pretraining yields the most general representations.
  - Pretraining on related tasks helps
  - *More data* helps even more!

Code: `http://nelsonliu.me/papers/contextual-repr-analysis`

So, in terms of takeaways, in this study we found that:

1) Features from contextual word representations are sufficient for high performance on a broad set of tasks, but fine-grained linguistic knowledge is not linearly recoverable.
2) Furthermore, patterns in layerwise transferability exist, and they can be explained by variations in how task-specific each of the layers are. We also find that higher-level layers don't necessarily encode higher-level semantic information, but instead encode things that are useful for their pretraining task.
3) Lastly, even on Penn Treebank-size data, bidirectional language model pretraining yields representations that are the most transferable on average. We do see that pretraining on related tasks gives the best results for individual target tasks, but ultimately training on more data yields even bigger gains.

Takeaways

Thanks! Questions?

- Features from pretrained contextualizers are sufficient for high performance on a broad set of tasks.
- Tasks with lower performance might require fine-grained linguistic knowledge.
- Layerwise patterns in transferability exist. Dictated by how task-specific each layer is.
- Even on PTB-size data, BiLM pretraining yields the most general representations.
  - Pretraining on related tasks helps
  - *More data* helps even more!

Code: `http://nelsonliu.me/papers/contextual-repr-analysis`

And that concludes my talk. Thanks for listening, and I'll take questions now.

Repeat the question when you get it!

Bonus Slides

Probing Task Examples

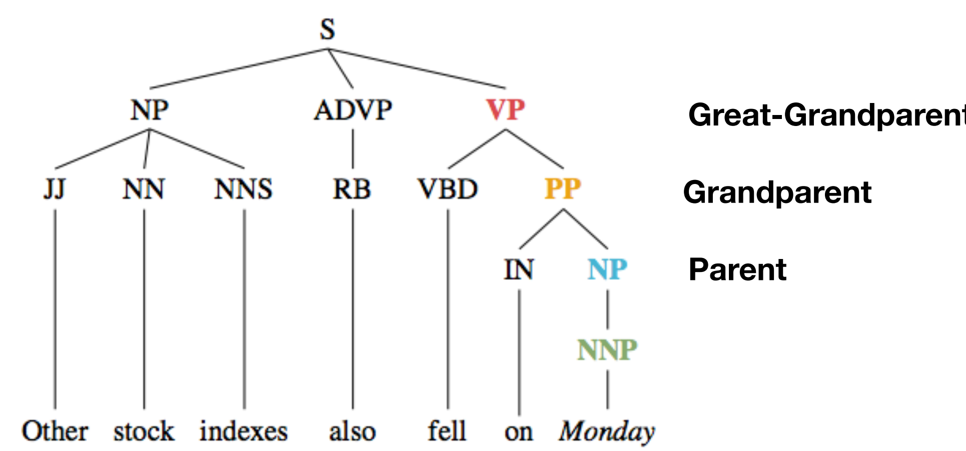
Part-of-Speech Tagging

Soon	she	was	running	the	office
RB	PRP	VBD	VBG	DT	NN

CCG Supertagging

$$
\begin{array}{cccccc}
\underline{\text{Soon}} & \underline{\text{she}} & \underline{\text{was}} & \underline{\text{running}} & \underline{\text{the}} & \underline{\text{office}} \\
S/S & NP & (S\backslash NP)/NP & NP & NP/N & N \\
\cline{3-4} \cline{5-6}
 & & \multicolumn{2}{c}{(S\backslash NP)/NP \quad >} & \multicolumn{2}{c}{NP \quad >} \\
\cline{3-6}
 & & \multicolumn{4}{c}{S\backslash NP \quad >} \\
\cline{2-6}
 & \multicolumn{5}{c}{S \quad <} \\
\cline{1-6}
\multicolumn{6}{c}{S \quad >}
\end{array}
$$

Syntactic Constituency Ancestor Tagging



Semantic Tagging

- Semantic tags abstract over redundant POS distinctions and disambiguate useful cases within POS tags.
- (1) Sarah bought **herself** a book
- (2) Sarah **herself** bought a book
- Same POS tag (Personal Pronoun), but different semantic function. (1) reflexive function, (2) emphasizing function

Preposition Supersense Disambiguation

- Classify a preposition's lexical semantic contribution (function), or the semantic role / relation it mediates (role).
- Specialized kind of word sense disambiguation.

Preposition Supersense Disambiguation

(1) I was booked **for**/DURATION 2 nights **at**/LOCUS this hotel **in**/TIME Oct 2007 .

(2) I went **to**/GOAL ohm **after**/EXPLANATION⇝TIME reading some **of**/QUANTITY⇝WHOLE the reviews .

(3) It was very upsetting to see this kind **of**/SPECIES behavior especially **in_front_of**/LOCUS **my**/SOCIALREL⇝GESTALT four year_old .

Event Factuality

- Label predicates with the factuality of events they describe.

Event "leave" did not happen.

(3) a. Jo didn’t remember to **leave**.
b. Jo didn’t remember **leaving**.

Event "leaving" happened.

Syntactic Chunking

[NP He] [VP reckons] [NP the current account deficit] [VP will narrow] [PP to] [NP only # 1.8 billion] [PP in] [NP September] .

Named Entity Recognition

[ORG U.N.] official [PER Ekeus] heads for [LOC Baghdad] .

Grammatical Error Detection

```
+ +    +  -       +   +      +   +    +    -      +
I like to playing the guitar and sing very louder .
```

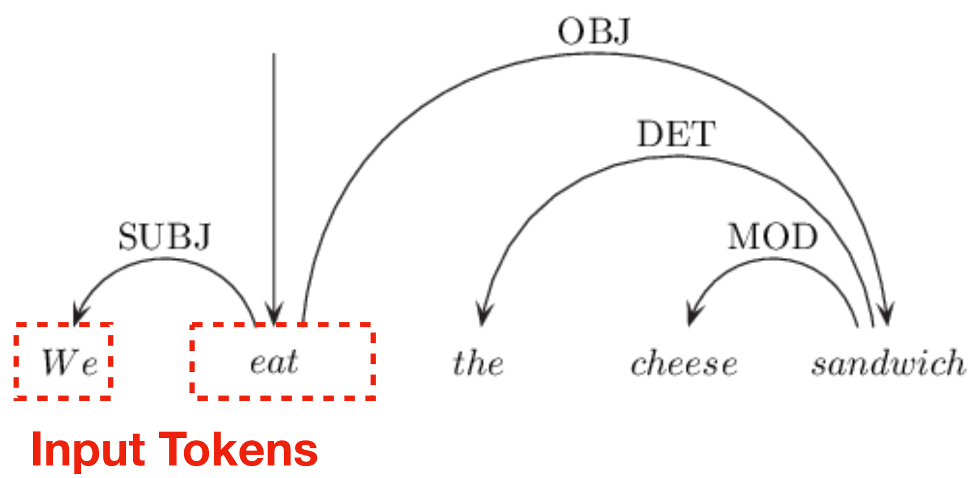
Conjunct Identification

- And the city decided to treat its guests more like **[royalty]** or **[rock stars]** than factory owners.

Two Types of Pairwise Relations

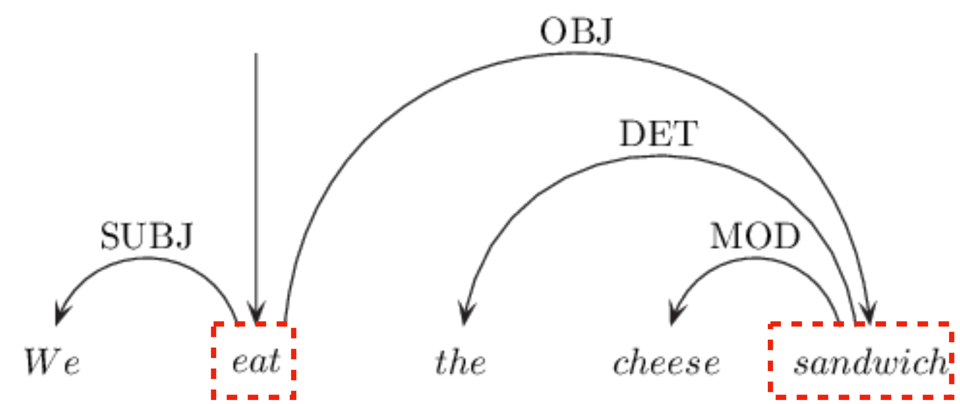
- **Arc prediction tasks:** Given two random tokens, identify **whether** a relation exists between them.
- **Arc classification tasks:** Given two tokens that are known to be related, **identify what** the relation is.

Syntactic Dependency Arc Prediction



Label: True, there exists a relation

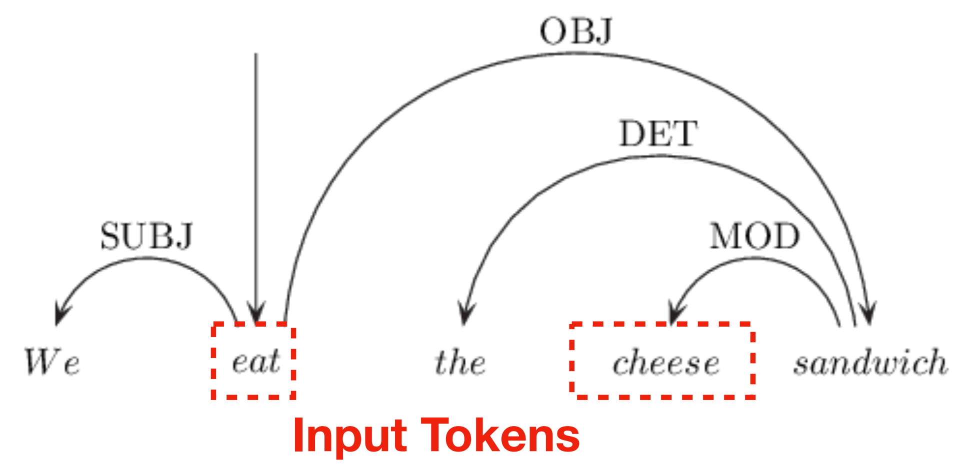
Syntactic Dependency
Arc Prediction



Input Tokens

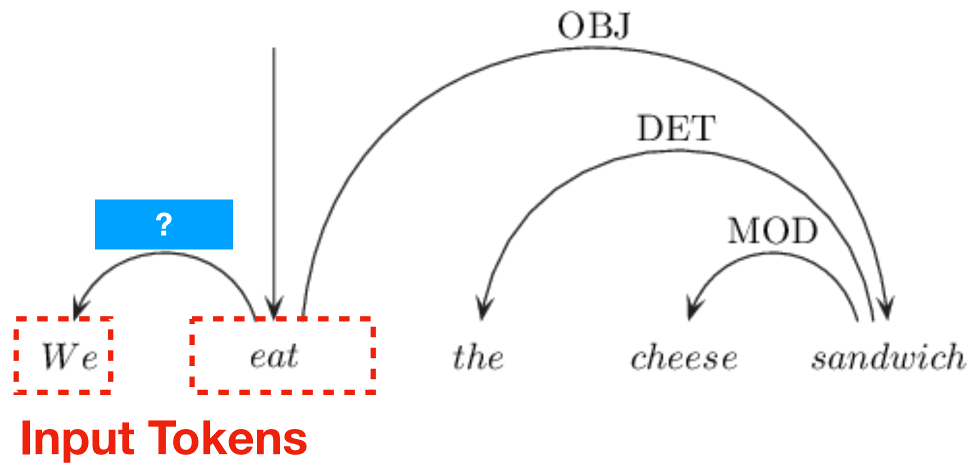
Label: True, there exists a relation

Syntactic Dependency Arc Prediction

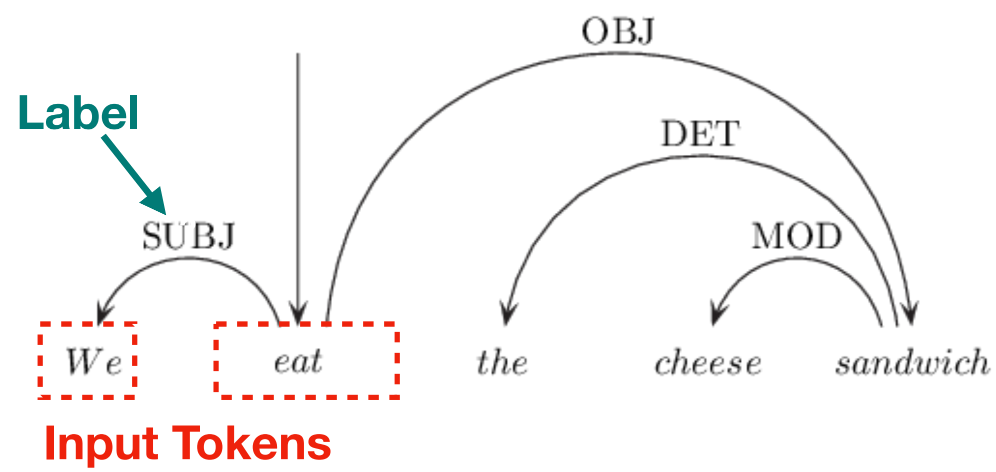


Label: False, there does not exist a relation

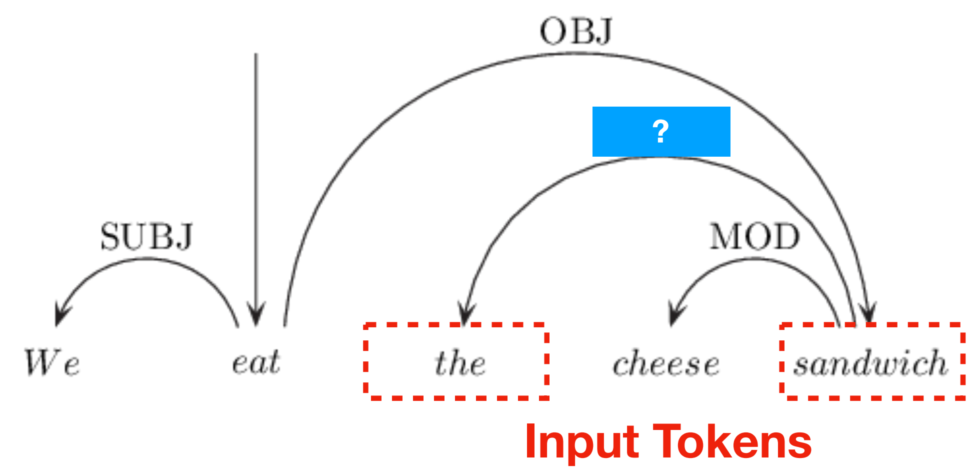
Syntactic Dependency Arc Classification



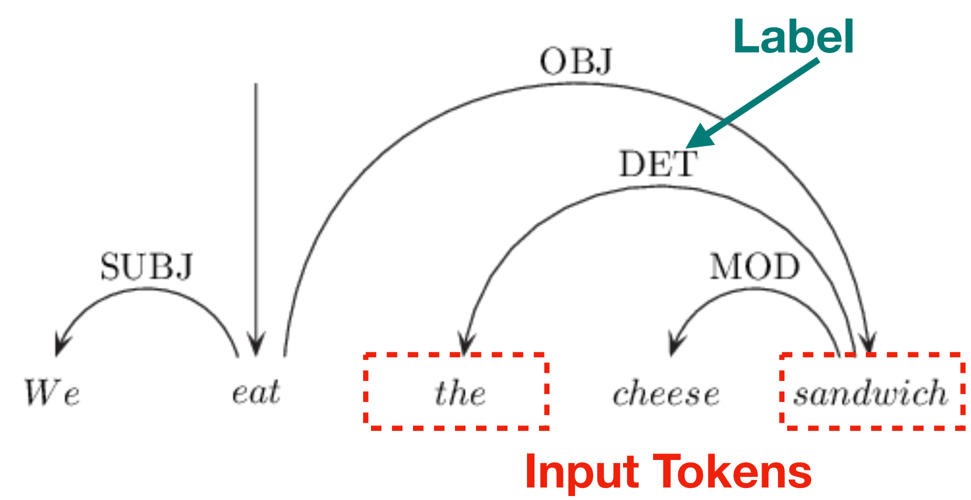
Syntactic Dependency Arc Classification



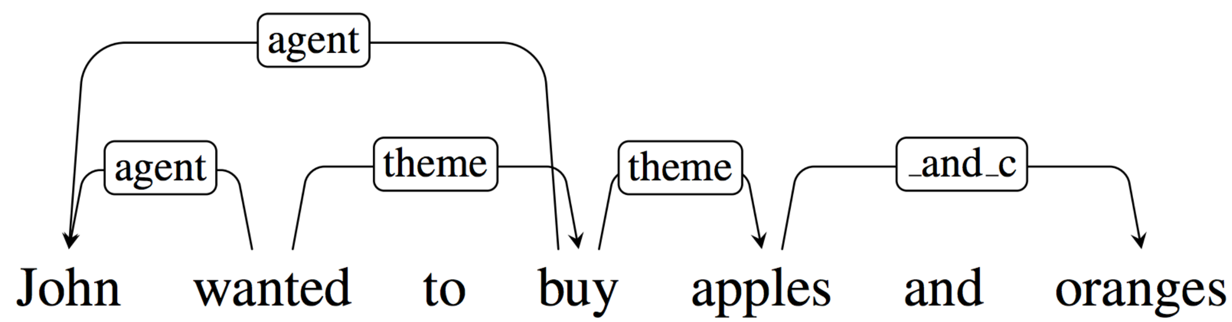
Syntactic Dependency Arc Classification



Syntactic Dependency Arc Classification



Semantic Dependencies

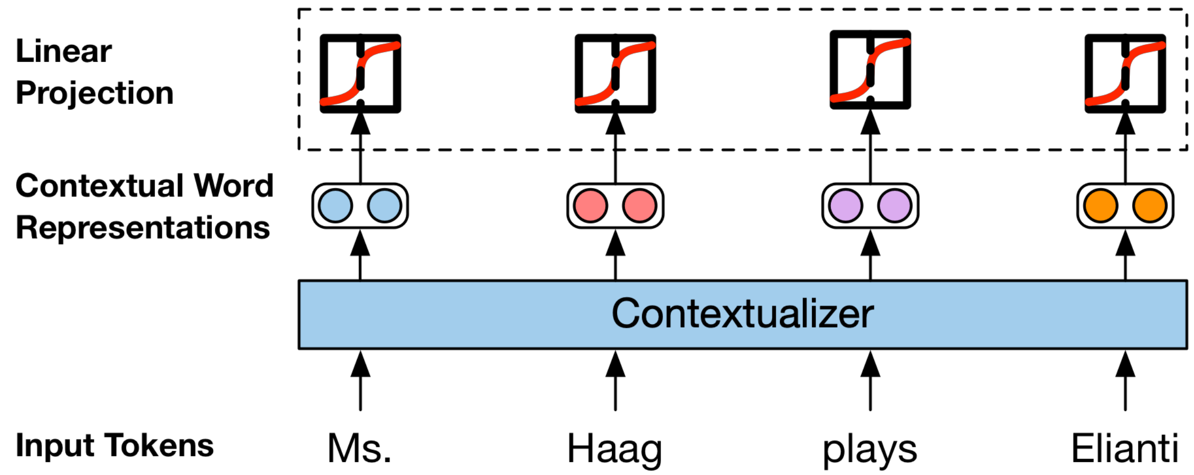


Coreference Relations

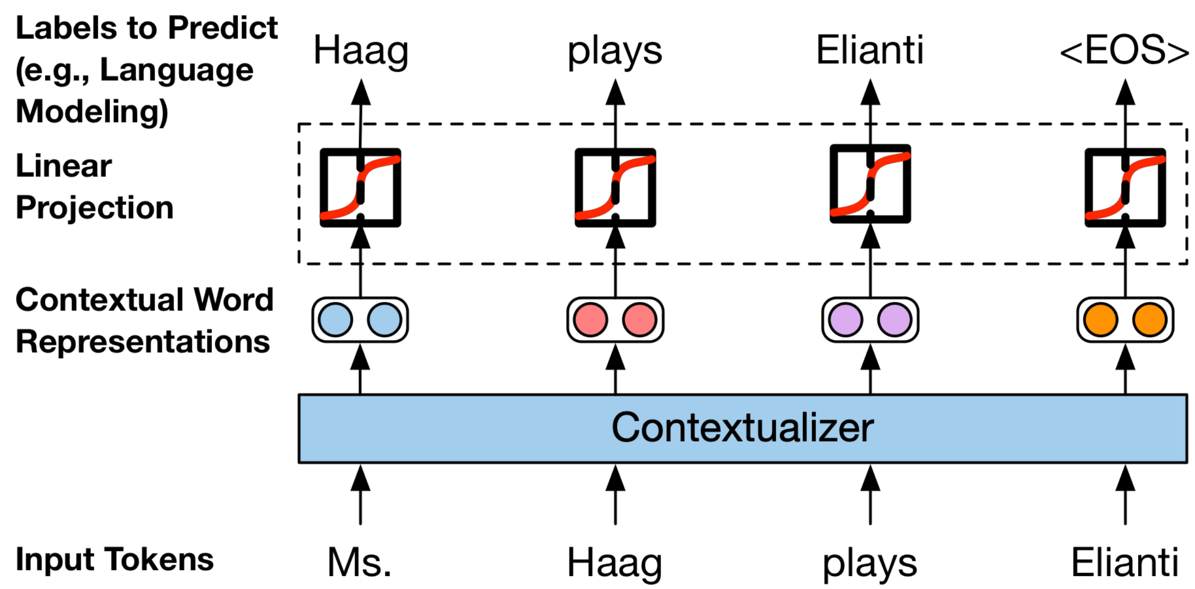
"I voted for Nader because he was most aligned with my values," she said.

Setting Up Alternative Pretraining Objectives

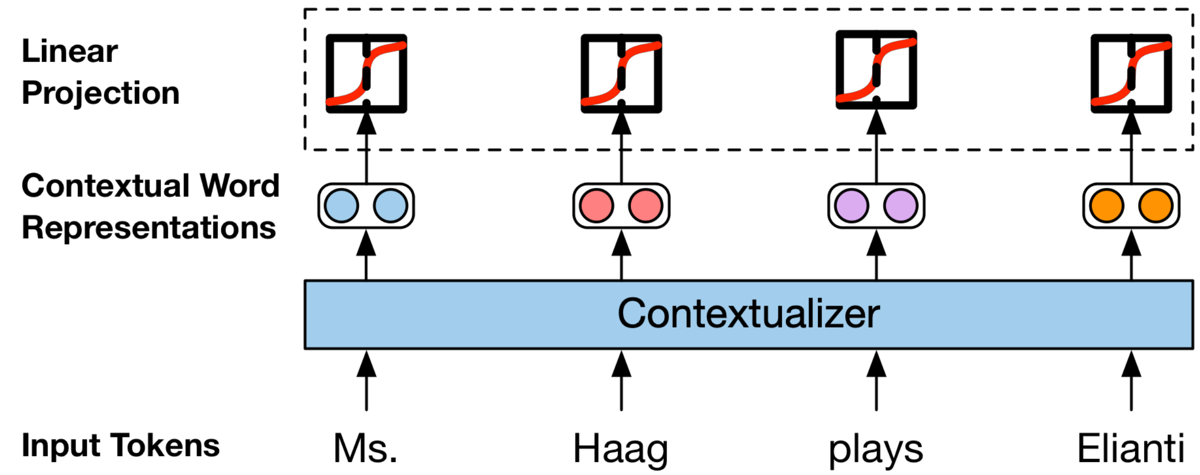
Language Model Pretraining



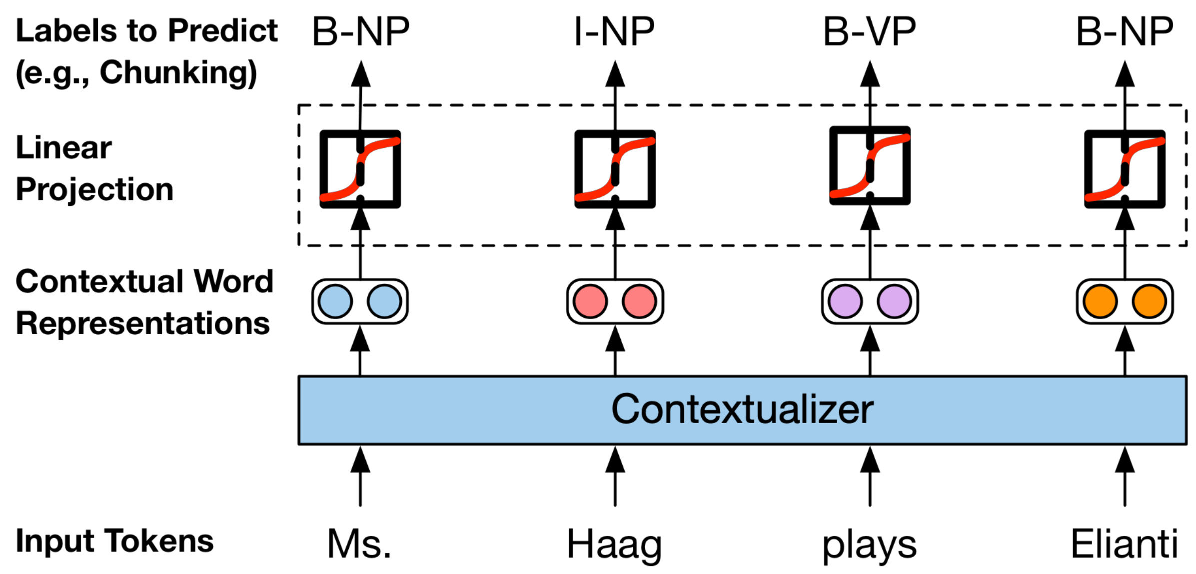
Language Model Pretraining



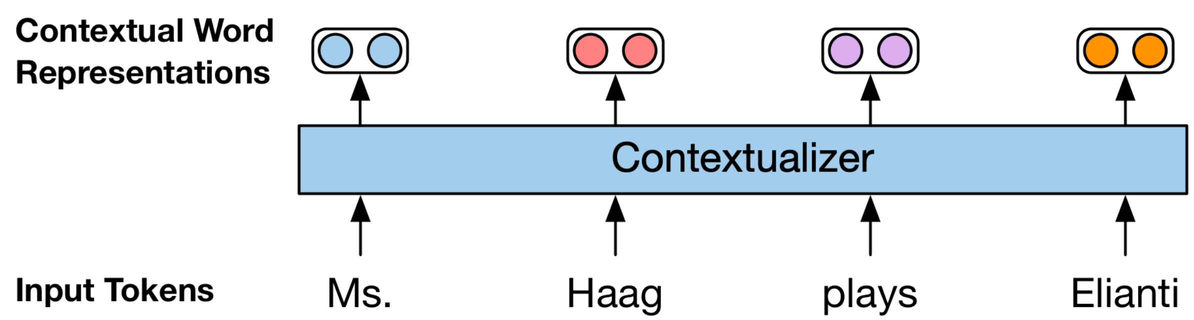
Chunking Pretraining

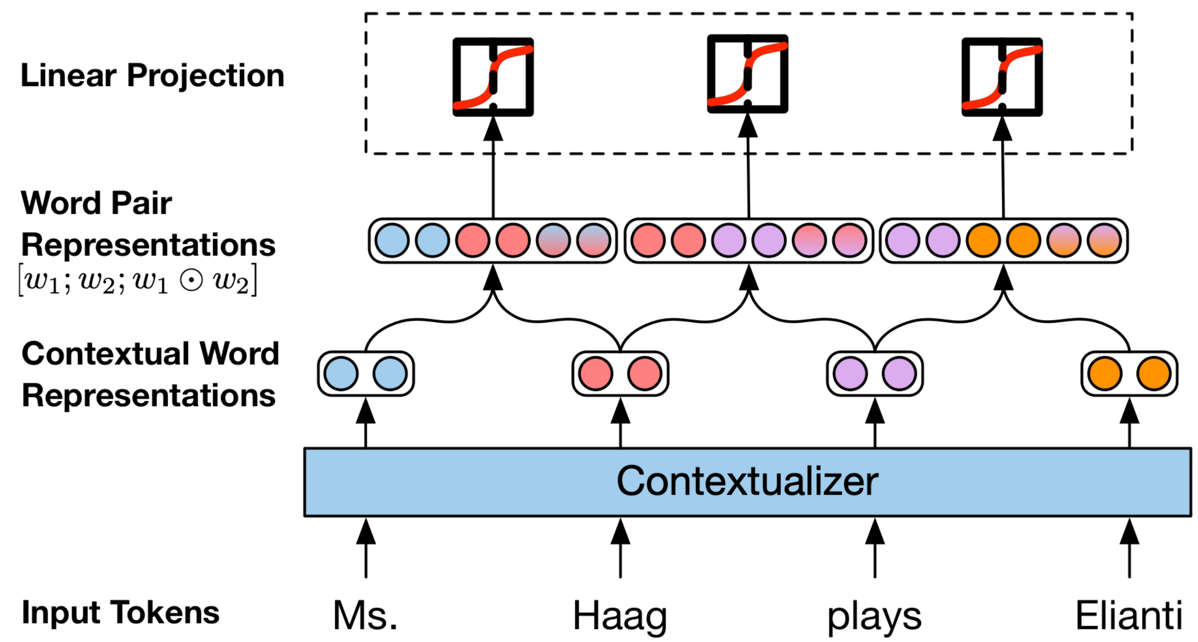


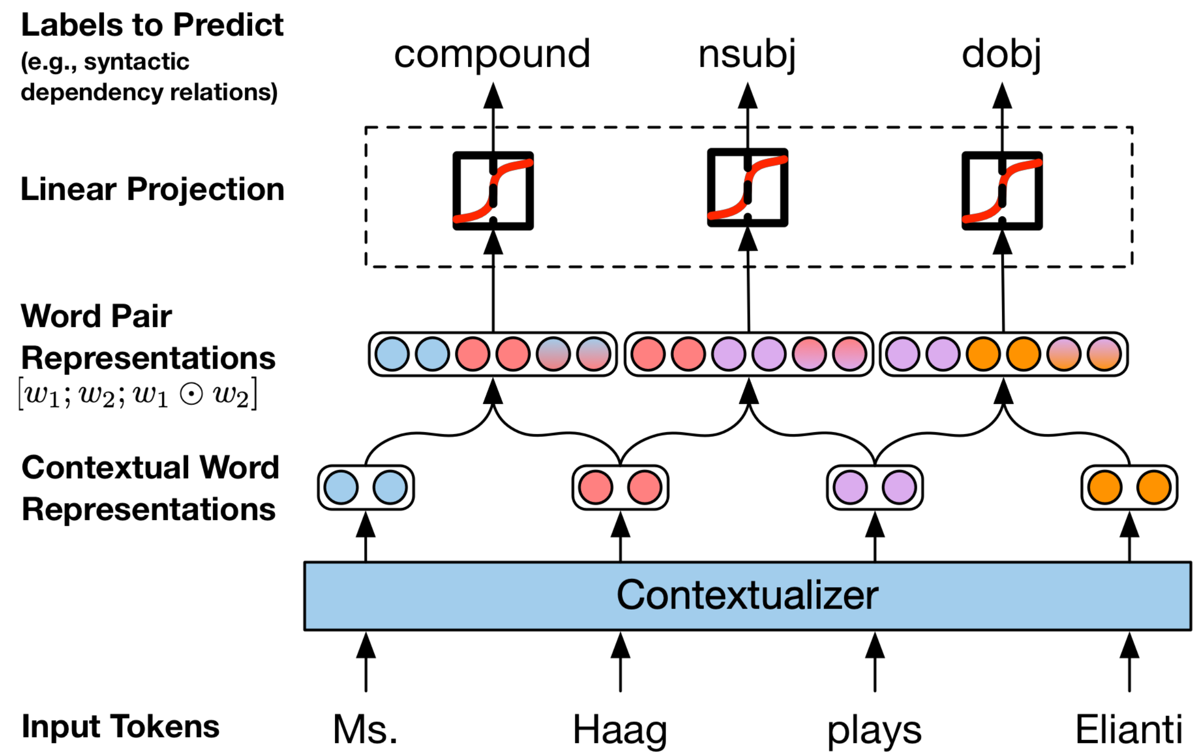
Chunking Pretraining



Flexible Paradigm, Use Any Task!




Linear Projection
Word Pair Representations
$[w_1; w_2; w_1 \odot w_2]$
Contextual Word Representations
Contextualizer
Input Tokens
Ms.
Haag
plays
Elianti


Labels to Predict
(e.g., syntactic dependency relations)
compound
nsubj
dobj
Linear Projection
Word Pair Representations
$[w_1; w_2; w_1 \odot w_2]$
Contextual Word Representations
Contextualizer
Input Tokens
Ms.
Haag
plays
Elianti