Shallow Syntax in Deep Water

Swabha Swayamdipta♠[*] **Matthew Peters**♣
Brendan Roof♣ **Chris Dyer**♡ **Noah A. Smith**♢♣

♠Language Technologies Institute, Carnegie Mellon University
♣Allen Institute for Artificial Intelligence
♢Paul G. Allen School of Computer Science & Engineering, University of Washington
♡Google DeepMind
`{swabhas,matthewp,brendanr}@allenai.org`

Abstract

Shallow syntax provides an approximation of phrase-syntactic structure of sentences; it can be produced with high accuracy, and is computationally cheap to obtain. We investigate the role of shallow syntax-aware representations for NLP tasks using two techniques. First, we enhance the ELMo architecture (Peters et al., 2018b) to allow pretraining on predicted shallow syntactic parses, instead of just raw text, so that contextual embeddings make use of shallow syntactic context. Our second method involves shallow syntactic features obtained automatically on downstream task data. Neither approach leads to a significant gain on any of the four downstream tasks we considered relative to ELMo-only baselines. Further analysis using black-box probes from Liu et al. (2019) confirms that our shallow-syntax-aware contextual embeddings do not transfer to linguistic tasks any more easily than ELMo's embeddings. We take these findings as evidence that ELMo-style pretraining discovers representations which make additional awareness of shallow syntax redundant.

1 Introduction

The NLP community is revisiting the role of linguistic structure in applications with the advent of contextual word representations (CWRs) derived from pretraining language models on large corpora (Peters et al., 2018a; Radford et al., 2018; Howard and Ruder, 2018; Devlin et al., 2018). Recent work has shown that downstream task performance may benefit from explicitly injecting a syntactic inductive bias into model architectures (Kuncoro et al., 2018), even when CWRs are also used (Strubell et al., 2018). However, high quality linguistic structure annotation at a large scale remains expensive—a trade-off needs to be made

[*] Work done during an internship at the Allen Institute for Artificial Intelligence.

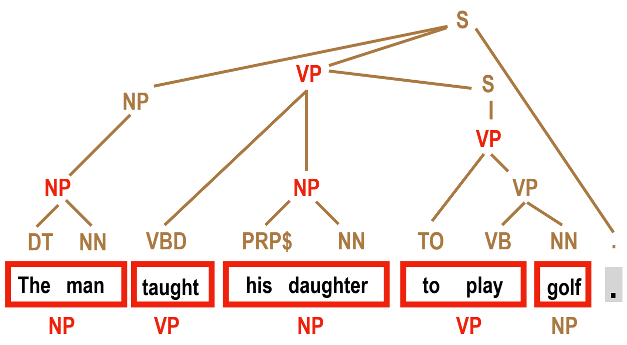


Figure 1: A sentence with its phrase-syntactic tree (brown) and shallow syntactic (chunk) annotations (red). Nodes in the tree which percolate down as chunk labels are in red. Not all tokens in the sentence get chunk labels; e.g., punctuation is not part of a chunk.

between the quality of the annotations and the computational expense of obtaining them. Shallow syntactic structures (Abney, 1991; also called chunk sequences) offer a viable middle ground, by providing a flat, non-hierarchical approximation to phrase-syntactic trees (see Fig. 1 for an example). These structures can be obtained efficiently, and with high accuracy, using sequence labelers. In this paper we consider shallow syntax to be a proxy for linguistic structure.

While shallow syntactic chunks are almost as ubiquitous as part-of-speech tags in standard NLP pipelines (Jurafsky and Martin, 2000), their relative merits in the presence of CWRs remain unclear. We investigate the role of these structures using two methods. First, we enhance the ELMo architecture (Peters et al., 2018b) to allow pretraining on predicted shallow syntactic parses, instead of just raw text, so that contextual embeddings make use of shallow syntactic context (§2). Our second method involves classical addition of chunk features to CWR-infused architectures for four different downstream tasks (§3). Shallow syntactic information is obtained automatically using a highly accurate model (97% F_1 on standard benchmarks). In both settings, we observe only modest gains on three of the four downstream tasks relative to ELMo-only baselines (§4).

Recent work has probed the knowledge encoded in CWRs and found they capture a surprisingly large amount of syntax (Blevins et al., 2018; Liu et al., 2019; Tenney et al., 2019). We further examine the contextual embeddings obtained from the enhanced architecture and a shallow syntactic context, using black-box probes from Liu et al. (2019). Our analysis indicates that our shallow-syntax-aware contextual embeddings do not transfer to linguistic tasks any more easily than ELMo embeddings (§4.2).

Overall, our findings show that while shallow syntax can be somewhat useful, ELMo-style pretraining discovers representations which make *additional* awareness of shallow syntax largely redundant.

2 Pretraining with Shallow Syntactic Annotations

We briefly review the shallow syntactic structures used in this work, and then present a model architecture to obtain e**m**beddings from shallow **Syn**tactic **C**ontext (**mSynC**).

2.1 Shallow Syntax

Base phrase chunking is a cheap sequence-labeling–based alternative to full syntactic parsing, where the sequence consists of non-overlapping labeled segments (Fig. 1 includes an example.) Full syntactic trees can be converted into such shallow syntactic chunk sequences using a deterministic procedure (Jurafsky and Martin, 2000). Tjong Kim Sang and Buchholz (2000) offered a rule-based transformation deriving non-overlapping chunks from phrase-structure trees as found in the Penn Treebank (Marcus et al., 1993). The procedure percolates some syntactic phrase nodes from a phrase-syntactic tree to the phrase in the leaves of the tree. All overlapping embedded phrases are then removed, and the remainder of the phrase gets the percolated label—this usually corresponds to the head word of the phrase.

In order to obtain shallow syntactic annotations on a large corpus, we train a BiLSTM-CRF model (Lample et al., 2016; Peters et al., 2017), which achieves 97% F_1 on the CoNLL 2000 benchmark test set. The training data is obtained from the CoNLL 2000 shared task (Tjong Kim Sang and Buchholz, 2000), as well as the remaining sections (except §23 and §20) of the Penn Treebank, using the official script for chunk generation.[1] The standard task definition from the shared task includes eleven chunk labels, as shown in Table 1.

Label	% Occurrence
Noun Phrase (NP)	51.7
Verb Phrase (VP)	20.0
Prepositional Phrase (PP)	19.8
Adverbial Phrase (ADVP)	3.7
Subordinate Clause (SBAR)	2.1
Adjective Phrase (ADJP)	1.9
Verb Particles (PRT)	0.5
Conjunctive Phrase (CONJ)	0.06
Interjective Phrase (INTJ)	0.03
List Marker (LST)	0.01
Unlike Coordination Phrase (UCP)	0.002

Table 1: Shallow syntactic chunk phrase types from CoNLL 2000 shared task (Tjong Kim Sang and Buchholz, 2000) and their occurrence % in the training data.

2.2 Pretraining Objective

Traditional language models are estimated to maximize the likelihood of each word x_i given the words that precede it, $p(x_i \mid \boldsymbol{x}_{<i})$. Given a corpus that is annotated with shallow syntax, we propose to condition on both the preceding words *and* their annotations.

We associate with each word x_i three additional variables (denoted c_i): the indices of the beginning and end of the last completed chunk *before* x_i, and its label. For example, in Fig. 2, $c_4 = \langle 3, 3, \text{VP} \rangle$ for $x_4 =$ the. Chunks, $\boldsymbol{c}$ are only used as conditioning context via $p(x_i \mid \boldsymbol{x}_{<i}, \boldsymbol{c}_{\leqslant i})$; they are not predicted.[2] Because the $\boldsymbol{c}$ labels depend on the entire sentence through the CRF chunker, conditioning each word's probability on any $\boldsymbol{c}_i$ means that our model is, strictly speaking, not a language model, and it can no longer be meaningfully evaluated using perplexity.

A right-to-left model is constructed analogously, conditioning on $\boldsymbol{c}_{\geqslant i}$ alongside $\boldsymbol{x}_{>i}$. Fol-

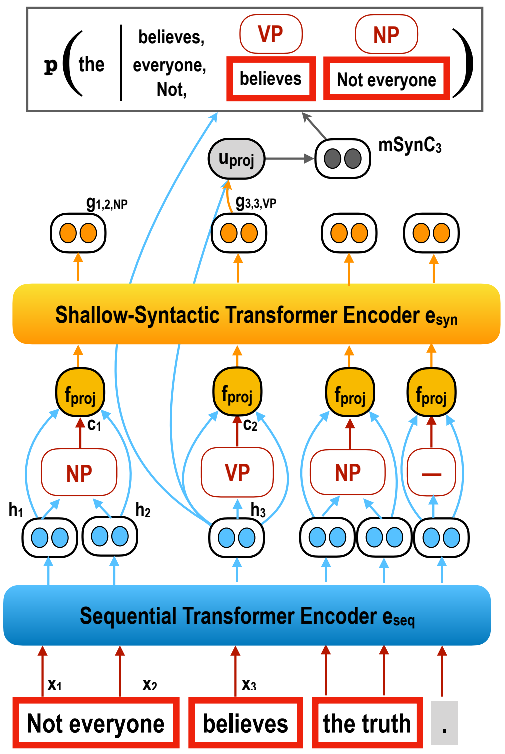
[1] https://www.clips.uantwerpen.be/conll2000/chunking/

[2] A different objective could consider predicting the next chunks, along with the next word. However, this chunker would have access to strictly less information than usual, since the entire sentence would no longer be available.

lowing Peters et al. (2018a), we use a joint objective maximizing data likelihood objectives in both directions, with shared softmax parameters.

2.3 Pretraining Model Architecture

Our model uses two encoders: e_{seq} for encoding the sequential history ($\boldsymbol{x}_{<i}$), and e_{syn} for shallow syntactic (chunk) history ($\boldsymbol{c}_{\leqslant i}$). For both, we use transformers (Vaswani et al., 2017), which consist of large feedforward networks equipped with multiheaded self-attention mechanisms.



p (the | believes, everyone, Not, VP believes NP Not everyone)
uproj
mSynC3
g1,2,NP
g3,3,VP
Shallow-Syntactic Transformer Encoder esyn
fproj
c1
c2
NP
VP
NP
—
h1
h2
h3
Sequential Transformer Encoder eseq
x1
x2
x3
Not everyone
believes
the truth
.

Figure 2: Model architecture for pretraining with shallow syntax. A sequential encoder converts the raw text into CWRs (shown in blue). Observed shallow syntactic structure (chunk boundaries and labels, shown in red) are combined with these CWRs in a shallow syntactic encoder to get contextualized representations for chunks (shown in orange). Both representations are passed through a projection layer to get **mSynC** embeddings (details shown only in some positions, for clarity), used both for computing the data likelihood, as shown, as well as in downstream tasks.

As inputs to e_{seq}, we use a context-independent embedding, obtained from a CNN character encoder (Kim et al., 2016) for each token x_i. The outputs $\boldsymbol{h}_i$ from e_{seq} represent words in context.

Next, we build representations for (observed) chunks in the sentence by concatenating a learned embedding for the chunk label with $\boldsymbol{h}$s for the boundaries and applying a linear projection (f_{proj}). The output from f_{proj} is input to e_{syn}, the shallow syntactic encoder, and results in contextualized chunk representations, $\boldsymbol{g}$. Note that the number of chunks in the sentence is less than or equal to the number of tokens.

Each $\boldsymbol{h}_i$ is now concatentated with $\boldsymbol{g}_{c_i}$, where $\boldsymbol{g}_{c_i}$ corresponds to c_i, the last chunk before position i. Finally, the output is given by $\mathbf{mSynC}_i = u_{proj}(\boldsymbol{h}_i, \boldsymbol{g}_{c_i}) = \boldsymbol{W}^\top[\boldsymbol{h}_i; \boldsymbol{g}_{c_i}]$, where $\boldsymbol{W}$ is a model parameter. For training, $\mathbf{mSynC}_i$ is used to compute the probability of the next word, using a sampled softmax (Bengio et al., 2003). For downstream tasks, we use a learned linear weighting of all layers in the encoders to obtain a task-specific **mSynC**, following Peters et al. (2018a).

Staged parameter updates Jointly training both the sequential encoder e_{seq}, and the syntactic encoder e_{syn} can be expensive, due to the large number of parameters involved. To reduce cost, we initialize our sequential CWRs $\boldsymbol{h}$, using *pretrained* embeddings from ELMo-transformer. Once initialized as such, the encoder is fine-tuned to the data likelihood objective (§2.2). This results in a staged parameter update, which reduces training duration by a factor of 10 in our experiments. We discuss the empirical effect of this approach in §4.3.

3 Shallow Syntactic Features

Our second approach incorporates shallow syntactic information in downstream tasks via token-level chunk label embeddings. Task training (and test) data is automatically chunked, and chunk boundary information is passed into the task model via BIOUL encoding of the labels. We add randomly initialized chunk label embeddings to task-specific input encoders, which are then fine-tuned for task-specific objectives. This approach does not require a shallow syntactic encoder or chunk annotations for pretraining CWRs, only a chunker. Hence, this can more directly measure the impact of shallow syntax for a given task.[3]

4 Experiments

Our experiments evaluate the effect of shallow syntax, via contextualization (**mSynC**, §2) and features (§3). We provide comparisons with four baselines—ELMo-transformer (Peters et al.,

[3]In contrast, in §2, the shallow-syntactic encoder itself, as well as predicted chunk quality on the large pretraining corpus could affect downstream performance.

2018b), our reimplementation of the same, as well as two CWR-free baselines, with and without shallow syntactic features. Both ELMo-transformer and **mSynC** are trained on the 1B word benchmark corpus (Chelba et al., 2013); the latter also employs chunk annotations (§2.1). Experimental settings are detailed in Appendix §A.1.

4.1 Downstream Task Transfer

We employ four tasks to test the impact of shallow syntax. The first three, namely, coarse and fine-grained named entity recognition (NER), and constituency parsing, are *span-based*; the fourth is a sentence-level sentiment classification task. Following Peters et al. (2018a), we do not apply fine-tuning to task-specific architectures, allowing us to do a controlled comparison with ELMo. Given an identical base architecture across models for each task, we can attribute any difference in performance to the incorporation of shallow syntax or contextualization. Details of downstream architectures are provided below, and overall dataset statistics for all tasks is shown in the Appendix, Table 5.

NER We use the English portion of the CoNLL 2003 dataset (Tjong Kim Sang and De Meulder, 2003), which provides named entity annotations on newswire data across four different entity types (`PER`, `LOC`, `ORG`, `MISC`). A bidirectional LSTM-CRF architecture (Lample et al., 2016) and a BIOUL tagging scheme were used.

Fine-grained NER The same architecture and tagging scheme from above is also used to predict fine-grained entity annotations from OntoNotes 5.0 (Weischedel et al., 2011). There are 18 fine-grained NER labels in the dataset, including regular named entitities as well as entities such as date, time and common numerical entries.

Phrase-structure parsing We use the standard Penn Treebank splits, and adopt the span-based model from Stern et al. (2017). Following their approach, we used predicted part-of-speech tags from the Stanford tagger (Toutanova et al., 2003) for training and testing. About 51% of phrase-syntactic constituents align exactly with the predicted chunks used, with a majority being single-width noun phrases. Given that the rule-based procedure used to obtain chunks only propagates the phrase type to the head-word and removes all overlapping phrases to the right, this is expected. We did not employ jack-knifing to obtain predicted chunks on PTB data; as a result there might be differences in the quality of shallow syntax annotations between the train and test portions of the data.

Sentiment analysis We consider fine-grained (5-class) classification on Stanford Sentiment Treebank (Socher et al., 2013). The labels are `negative`, `somewhat_negative`, `neutral`, `positive` and `somewhat_positive`. Our model was based on the biattentive classification network (McCann et al., 2017). We used all phrase lengths in the dataset for training, but test results are reported only on full sentences, following prior work.

Results are shown in Table 2. Consistent with previous findings, CWRs offer large improvements across all tasks. Though helpful to span-level task models without CWRs, shallow syntactic features offer little to no benefit to ELMo models. **mSynC**'s performance is similar. This holds even for phrase-structure parsing, where (gold) chunks align with syntactic phrases, indicating that task-relevant signal learned from exposure to shallow syntax is *already* learned by ELMo. On sentiment classification, chunk features are slightly harmful on average (but variance is high); **mSynC** again performs similarly to ELMo-transformer. Overall, the performance differences across all tasks are small enough to infer that shallow syntax is not particularly helpful when using CWRs.

4.2 Linguistic Probes

We further analyze whether awareness of shallow syntax carries over to other linguistic tasks, via probes from Liu et al. (2019). Probes are linear models trained on frozen CWRs to make predictions about linguistic (syntactic and semantic) properties of words and phrases. Unlike §4.1, there is minimal downstream task architecture, bringing into focus the transferability of CWRs, as opposed to task-specific adaptation.

4.2.1 Probing Tasks

The ten different probing tasks we used include CCG supertagging (Hockenmaier and Steedman, 2007), part-of-speech tagging from PTB (Marcus et al., 1993) and EWT (Universal Depedencies Silveira et al., 2014), named entity recognition (Tjong Kim Sang and De Meulder, 2003),

	NER	Fine-grained NER	Constituency Parsing	Sentiment
Baseline (no CWR)	88.1 ± 0.27	78.5 ± 0.19	88.9 ± 0.05	51.6 ± 1.63
+ shallow syn. features	88.6 ± 0.22	78.9 ± 0.13	90.8 ± 0.14	51.1 ± 1.39
ELMo-transformer (Peters et al., 2018b)	91.1 ± 0.26	—	93.7	—
ELMo-transformer (our reimplementation)	91.5 ± 0.25	85.7 ± 0.08	94.1 ± 0.06	53.0 ± 0.72
+ shallow syn. features	91.6 ± 0.40	85.9 ± 0.28	94.3 ± 0.03	52.6 ± 0.54
Shallow syn. contextualization (**mSynC**)	91.5 ± 0.19	85.9 ± 0.20	94.1 ± 0.07	53.0 ± 1.07

Table 2: Test-set performance of ELMo-transformer (Peters et al., 2018b), our reimplementation, and **mSynC**, compared to baselines without CWR. Evaluation metric is F_1 for all tasks except sentiment, which reports accuracy. Reported results show the mean and standard deviation across 5 runs for coarse-grained NER and sentiment classification and 3 runs for other tasks.

	CCG	PTB POS	EWT POS	Chunk	NER	Sem. Tagging	Gramm. Err. D	Prep. Role	Prep. Func.	Event Fact.
ELMo-transformer	92.68	97.09	95.13	92.18	81.21	93.78	30.80	72.81	82.24	70.88
mSynC	92.03	96.91	94.64	96.89	79.98	93.03	30.86	70.83	82.67	70.39

Table 3: Test performance of ELMo-transformer (Peters et al., 2018b) vs. **mSynC** on several linguistic probes from Liu et al. (2019). In each case, performance of the best layer from the architecture is reported. Details on the probes can be found in §4.2.1.

base-phrase chunking (Tjong Kim Sang and Buchholz, 2000), grammar error detection (Yannakoudakis et al., 2011), semantic tagging (Bjerva et al., 2016), preposition supersense identification (Schneider et al., 2018), and event factuality detection (Rudinger et al., 2018). Metrics and references for each are summarized in Table 6. For more details, please see Liu et al. (2019).

Results in Table 3 show ten probes. Again, we see the performance of baseline ELMo-transformer and **mSynC** are similar, with **mSynC** doing slightly worse on 7 out of 9 tasks. As we would expect, on the probe for predicting chunk tags, **mSynC** achieves 96.9 F_1 vs. 92.2 F_1 for ELMo-transformer, indicating that **mSynC** is indeed encoding shallow syntax. Overall, the results further confirm that explicit shallow syntax does not offer any benefits over ELMo-transformer.

4.3 Effect of Training Scheme

We test whether our staged parameter training (§2.3) is a viable alternative to an end-to-end training of both e_{syn} and e_{seq}. We make a further distinction between fine-tuning e_{seq} vs. not updating it at all after initialization (frozen). Downstream validation-set F_1 on fine-grained NER, reported in Table 4, shows that the end-to-end strategy lags behind the others, perhaps indicating the need to train longer than 10 epochs. However, a single epoch on the 1B-word benchmark takes 36 hours on 2 Tesla V100s, making this prohibitive. Interestingly, the frozen strategy, which takes the least amount of time to converge (24 hours on 1 Tesla V100), also performs almost as well as fine-tuning.

	Model	Fine-grained NER F_1
end-to-end	ELMo	86.90 ± 0.11
	mSynC end-to-end	86.89 ± 0.04
staged	**mSynC** frozen	87.36 ± 0.02
	mSynC fine-tuned	87.44 ± 0.07

Table 4: Validation F_1 for fine-grained NER across syntactic pretraining schemes, with mean and standard deviations across 3 runs.

5 Conclusion

We find that exposing CWR-based models to shallow syntax, either through new CWR learning architectures or explicit pipelined features, has little effect on their performance, across several tasks. Linguistic probing also shows that CWRs aware of such structures do not improve task transferability. Our architecture and methods are general enough to be adapted for richer inductive biases, such as those given by full syntactic trees (RNNGs; Dyer et al., 2016), or to different pretraining objectives, such as masked language modeling (BERT; Devlin et al., 2018); we leave this pursuit to future work.

References

Steven P Abney. 1991. Parsing by chunks. In *Principle-based parsing*, pages 257–278. Springer.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, pages 1137–1155.

Johannes Bjerva, Barbara Plank, and Johan Bos. 2016. Semantic tagging with deep residual networks. In *Proc. of COLING*.

Terra Blevins, Omer Levy, and Luke Zettlemoyer. 2018. Deep rnns encode soft hierarchical syntax. In *Proc. of ACL*.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. ArXiv:1312.3005.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. ArXiv:1810.04805.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A Smith. 2016. Recurrent neural network grammars. In *Proc. of NAACL-HLT*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. AllenNLP: A deep semantic natural language processing platform. ArXiv:1803.07640.

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

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proc. of ACL*.

Daniel Jurafsky and James H. Martin. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, 1st edition. Prentice Hall PTR, Upper Saddle River, NJ, USA.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. 2016. Character-aware neural language models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2741–2749. AAAI Press.

Adhiguna Kuncoro, Chris Dyer, John Hale, Dani Yogatama, Stephen Clark, and Phil Blunsom. 2018. Lstms can learn syntax-sensitive dependencies well, but modeling structure makes them better. In *Proc. of ACL*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proc. of NAACL-HLT*.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proc. of NAACL-HLT*.

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

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Proc. of NeurIPS*, pages 6294–6305.

Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *Proc. of ACL*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018a. Deep contextualized word representations. In *Proc. of NAACL-HLT*.

Matthew Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018b. Dissecting contextual word embeddings: Architecture and representation. In *Proc. of EMNLP*, pages 1499–1509.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Rachel Rudinger, Aaron Steven White, and Benjamin Van Durme. 2018. Neural models of factuality. In *Proc. of ACL*.

Nathan Schneider, Jena D Hwang, Vivek Srikumar, Jakob Prange, Austin Blodgett, Sarah R Moeller, Aviram Stern, Adi Bitan, and Omri Abend. 2018. Comprehensive supersense disambiguation of English prepositions and possessives.

Natalia Silveira, Timothy Dozat, Marie-Catherine De Marneffe, Samuel R Bowman, Miriam Connor, John Bauer, and Christopher D Manning. 2014. A gold standard dependency corpus for English. In *Proc. of LREC*, pages 2897–2904.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proc. of EMNLP*.

Mitchell Stern, Jacob Andreas, and Dan Klein. 2017. A minimal span-based neural constituency parser. In *Proc. of ACL*.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proc. of EMNLP*.

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

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 shared task: Chunking. In *Proc. of CoNLL*.

Erik F Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proc. of NAACL*. Association for Computational Linguistics.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proc. of NAACL*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. of NeurIPS*, pages 5998–6008.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. OntoNotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*.

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Craig Greenberg, Eduard Hovy, Robert Belvin, et al. 2011. OntoNotes Release 4.0. *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium*.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading esol texts. In *Proc. of ACL*.

A Supplemental Material

A.1 Hyperparameters

ELMo-transformer Our baseline pretraining model was a reimplementation of that given in Peters et al. (2018b). Hyperparameters were generally identical, but we trained on only 2 GPUs with (up to) 4,000 tokens per batch. This difference in batch size meant we used 6,000 warm up steps with the learning rate schedule of Vaswani et al. (2017).

mSynC The function f_{seq} is identical to the 6-layer biLM used in ELMo-transformer. f_{syn}, on the other hand, uses only 2 layers. The learned embeddings for the chunk labels have 128 dimensions and are concatenated with the two boundary $\boldsymbol{h}$ of dimension 512. Thus f_{proj} maps $1024 + 128$ dimensions to 512. Further, we did not perform weight averaging over several checkpoints.

Shallow Syntax The size of the shallow syntactic feature embedding was 50 across all experiments, initialized uniform randomly.

All model implementations are based on the `AllenNLP` library (Gardner et al., 2017).

Task	**Train**	**Heldout**	**Test**
CoNLL 2003 NER (Tjong Kim Sang and De Meulder, 2003)	23,499	5,942	5,648
OntoNotes NER (Weischedel et al., 2013)	81,828	11,066	11,257
Penn TreeBank (Marcus et al., 1993)	39,832	1,700	2,416
Stanford Sentiment Treebank (Socher et al., 2013)	8,544	1,101	2,210

Table 5: Downstream dataset statistics describing the number of train, heldout and test set instances for each task.

Task	Dataset	Metric
CCG Supertagging	CCGBank (Hockenmaier and Steedman, 2007)	Accuracy
PTB part-of-speech tagging	PennTreeBank (Marcus et al., 1993)	Accuracy
EWT part-of-speech tagging	Universal Dependencies (Silveira et al., 2014)	Accuracy
Chunking	CoNLL 2000 (Tjong Kim Sang and Buchholz, 2000)	F_1
Named Entity Recognition	CoNLL 2003 (Tjong Kim Sang and De Meulder, 2003)	F_1
Semantic Tagging	(Bjerva et al., 2016)	Accuracy
Grammar Error Detection	First Certificate in English (Yannakoudakis et al., 2011)	F_1
Preposition Supersense Role	STREUSLE 4.0 (Schneider et al., 2018)	Accuracy
Preposition Supersense Function	STREUSLE 4.0 (Schneider et al., 2018)	Accuracy
Event Factuality Detection	UDS It Happened v2 (Rudinger et al., 2018)	Pearson R

Table 6: Dataset and metrics for each probing task from Liu et al. (2019), corresponding to Table 3.