Perturbed Masking: Parameter-free Probing for Analyzing and Interpreting BERT

Zhiyong Wu[1], **Yun Chen**[2], **Ben Kao**[1], **Qun Liu**[3]
[1]The University of Hong Kong, Hong Kong, China
[2]Shanghai University of Finance and Economics, Shanghai, China
[3]Huawei Noah's Ark Lab, Hong Kong, China
{zywu,kao}@cs.hku.hk, yunchen@sufe.edu.cn, qun.liu@huawei.com

Abstract

By introducing a small set of additional parameters, a *probe* learns to solve specific linguistic tasks (e.g., dependency parsing) in a supervised manner using feature representations (e.g., contextualized embeddings). The effectiveness of such *probing tasks* is taken as evidence that the pre-trained model encodes linguistic knowledge. However, this approach of evaluating a language model is undermined by the uncertainty of the amount of knowledge that is learned by the probe itself. Complementary to those works, we propose a parameter-free probing technique for analyzing pre-trained language models (e.g., BERT). Our method does not require direct supervision from the probing tasks, nor do we introduce additional parameters to the probing process. Our experiments on BERT show that syntactic trees recovered from BERT using our method are significantly better than linguistically-uninformed baselines. We further feed the empirically induced dependency structures into a downstream sentiment classification task and find its improvement compatible with or even superior to a human-designed dependency schema.[1]

1 Introduction

Recent prevalent pre-trained language models such as ELMo (Peters et al., 2018b), BERT (Devlin et al., 2018), and XLNet (Yang et al., 2019) achieve state-of-the-art performance for a diverse array of downstream NLP tasks. An interesting area of research is to investigate the interpretability of these pre-trained models (i.e., the linguistic properties they capture). Most recent approaches are built upon the idea of *probing classifiers* (Shi et al., 2016; Adi et al., 2017; Conneau et al., 2018; Peters et al., 2018a; Hewitt and Manning, 2019; Clark et al., 2019; Tenney et al., 2019b; Jawahar et al., 2019). A *probe* is a simple neural network (with a small additional set of parameters) that uses the feature representations generated by a pre-trained model (e.g., hidden state activations, attention weights) and is trained to perform a supervised task (e.g., dependency labeling). The performance of a *probe* is used to measure the quality of the generated representations with the assumption that the measured quality is mostly attributable to the pre-trained language model.

One downside of such approach, as pointed out in (Hewitt and Liang, 2019), is that a probe introduces a new set of additional parameters, which makes the results difficult to interpret. Is it the pre-trained model that captures the linguistic information, or is it the probe that learns the downstream task itself and thus encodes the information in its additional parameter space?

In this paper we propose a parameter-free probing technique called *Perturbed Masking* to analyze and interpret pre-trained models. The main idea is to introduce the *Perturbed Masking* technique into the masked language modeling (**MLM**) objective to measure the impact a word x_j has on predicting another word x_i (Sec 2.2) and then induce the global linguistic properties (e.g., dependency trees) from this inter-word information.

Our contributions are threefold:

- We introduce a new parameter-free probing technique, *Perturbed Masking*, to estimate inter-word correlations. Our technique enables global syntactic information extraction.
- We evaluate the effectiveness of our probe over a number of linguistic driven tasks (e.g., syntactic parsing, discourse dependency parsing). Our results reinforce the claims of recent probing works, and further complement them by quantitatively evaluating the validity of their claims.
- We feed the empirically induced dependency

[1] https://github.com/LividWo/Perturbed-Masking

structures into a downstream task to make a comparison with a parser-provided, linguist-designed dependency schema and find that our structures perform on-par or even better (Sec 6) than the parser created one. This offers an insight into the remarkable success of BERT on downstream tasks.

2 Perturbed Masking

We propose the perturbed masking technique to assess the impact one word has on the prediction of another in MLM. The inter-word information derived serves as the basis for our later analysis.

2.1 Background: BERT

BERT[2] (Devlin et al., 2018) is a large Transformer network that is pre-trained on 3.3 billion tokens of English text. It performs two tasks: (1) Masked Language Modeling (**MLM**): randomly select and mask 15% of all tokens in each given sequence, and then predict those masked tokens. In masking, a token is (a) replaced by the special token [MASK], (b) replaced by a random token, or (c) kept unchanged. These replacements are chosen 80%, 10%, and 10% of the time, respectively. (2)Next Sentence Prediction: given a pair of sentences, predict whether the second sentence follows the first in an original document or is taken from another random document.

2.2 Token Perturbation

Given a sentence as a list of tokens $\mathbf{x} = [x_1, \ldots, x_T]$, BERT maps each x_i into a contextualized representation $H_\theta(\mathbf{x})_i$, where θ represents the network's parameters. Our goal is to derive a function $f(x_i, x_j)$ that captures the impact a context word x_j has on the prediction of another word x_i.

We propose a two-stage approach to achieve our goal. First, we replace x_i with the [MASK] token and feed the new sequence $\mathbf{x}\backslash\{x_i\}$ into BERT. We use $H_\theta(\mathbf{x}\backslash\{x_i\})_i$ to denote the representation of x_i. To calculate the impact $x_j \in \mathbf{x}\backslash\{x_i\}$ has on $H_\theta(\mathbf{x}\backslash\{x_i\})_i$, we further mask out x_j to obtain the second corrupted sequence $\mathbf{x}\backslash\{x_i, x_j\}$. Similarly, $H_\theta(\mathbf{x}\backslash\{x_i, x_j\})_i$ denotes the new representation of token x_i.

We define $f(x_i, x_j)$ as:

$$f(x_i, x_j) = d\left(H_\theta(\mathbf{x}\backslash\{x_i\})_i, H_\theta(\mathbf{x}\backslash\{x_i, x_j\})_i\right)$$

where $d(\mathbf{x}, \mathbf{y})$ is the distance metric that captures the difference between two vectors. We experimented with two options for $d(\mathbf{x}, \mathbf{y})$:

- **Dist:** Euclidean distance between $\mathbf{x}$ and $\mathbf{y}$
- **Prob**: $d(\mathbf{x}, \mathbf{y}) = a(\mathbf{x})_{x_i} - a(\mathbf{y})_{x_i}$,

where $a(\cdot)$ maps a vector into a probability distribution among the words in the vocabulary. $a(\mathbf{x})_{x_i}$ represents the probability of predicting token x_i base on $\mathbf{x}$.

By repeating the two-stage perturbation on each pair of tokens $x_i, x_j \in \mathbf{x}$ and calculating $f(x_i, x_j)$, we obtain an **impact matrix** $\mathcal{F}$, where $\mathcal{F}_{ij} \in \mathbb{R}^{T \times T}$. Now, we can derive algorithms to extract syntactic trees from $\mathcal{F}$ and compare them with ground-truth trees that are obtained from benchmarks. Note that BERT uses byte-pair encoding (Sennrich et al., 2016) and may split a word into multiple tokens(subwords). To evaluate our approach on word-level tasks, we make the following changes to obtain inter-word impact matrices. In each perturbation, we mask all tokens of a split-up word. The impact *on* a split-up word is obtained by averaging[3] the impacts over the split-up word's tokens. To measure the impact exerted *by* a split-up word, we assume the impacts given by its tokens are the same; We use the impact given by the first token for convenience.

2.3 Span Perturbation

Given the token-level perturbation above, it is straightforward to extend it to span-level perturbation. We investigate how BERT models the relations between spans, which can be phrases, clauses, or paragraphs. As a preliminary study, we investigate how well BERT captures document structures.

We model a document D as N non-overlapping text spans $D = [e_1, e_2, \ldots, e_N]$, where each span e_i contains a sequence of tokens $e_i = [x_1^i, x_2^i, \ldots, x_M^i]$.

For span-level perturbation, instead of masking one token at a time, we mask an array of tokens in a span simultaneously. We obtain the span representation by averaging the representations of all the tokens the span contains. Similarly, we calculate the impact e_j has on e_i by:

$$f(e_i, e_j) = d\left(H_\theta(D\backslash\{e_i\})_i, H_\theta(D\backslash\{e_i, e_j\})_i\right)$$

where d is the **Dist** function.

[2] In our experiments, we use the base, uncased version from (Wolf et al., 2019).

[3] We also experimented with other alternatives, but observe no significant difference.

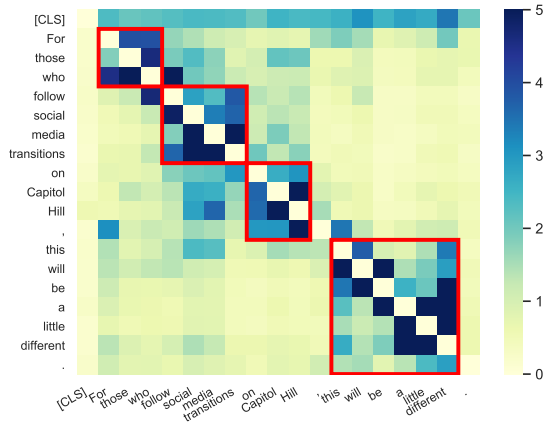

Figure 1: Heatmap of the impact matrix for the sentence "For those who follow social media transitions on Capitol Hill, this will be a little different."

3 Visualization with Impact Maps

Before we discuss specific syntactic phenomena, let us first analyze some example impact matrices derived from sample sentences. We visualize an impact matrix of a sentence by displaying a heatmap. We use the term "impact map" to refer to a heatmap of an impact matrix.

Setup. We extract impact matrices by feeding BERT with 1,000 sentences from the English Parallel Universal Dependencies (PUD) treebank of the CoNLL 2017 Shared Task (Zeman et al., 2017). We follow the setup and pre-processing steps employed in pre-training BERT. An example impact map is shown in Figure 1.

Dependency. We notice that the impact map contains many *stripes*, which are short series of vertical/horizontal cells, typically located along the diagonal. Take the word "*different*" as an example (which is illustrated by the second-to-last column in the impact matrix). We observe a clear vertical stripe above the main diagonal. The interpretation is that this particular occurrence of the word "*different*" strongly affects the occurrences of those words before it. These strong influences are shown by the darker-colored pixels seen in the second last column of the impact map. This observation agrees with the ground-truth dependency tree, which selects "*different*" as the head of all remaining words in the phrase "*this will be a little different*." We also observe similar patterns on "*transitions*" and "*Hill*". Such correlations lead us to explore the idea of extracting dependency trees from the matrices (see Section 4.1).

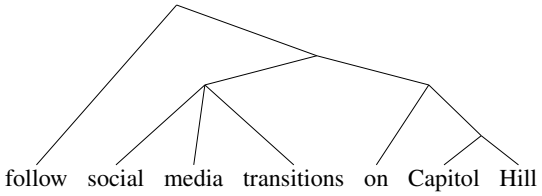

Figure 2: Part of the constituency tree.

Constituency. Figure 2 shows part of the constituency tree of our example sentence generated by Stanford CoreNLP (Manning et al., 2014). In this sentence, "*media*" and "*on*" are two words that are adjacent to "*transitions*". From the tree, however, we see that "*media*" is closer to "*transitions*" than "*on*" is in terms of syntactic distance. If a model is syntactically uninformed, we would expect "*media*" and "*on*" to have comparable impacts on the prediction of "*transitions*", and vice versa. However, we observe a far greater impact (darker color) between "*media*" and "*transitions*" than that between "*on*" and "*transitions*". We will further support this observation with empirical experiments in Section 4.2.

Other Structures. Along the diagonal of the impact map, we see that words are grouped into four contiguous chunks that have specific intents (e.g., a noun phrase – *on Capitol Hill*). We also observe that the two middle chunks have relatively strong inter-chunk word impacts and thus a bonding that groups them together, forming a larger verb phrase. This observation suggest that BERT may capture the compositionality of the language.

In the following sections we quantitatively evaluate these observations.

4 Syntactic Probe

We start with two syntactic probes – dependency probe and constituency probe.

4.1 Dependency Probe

With the goal of exploring the extent dependency relations are captured in BERT, we set out to answer the following question: Can BERT outperform linguistically uninformed baselines in unsupervised dependency parsing? If so, to what extent?

We begin by using the token-level perturbed masking technique to extract an impact matrix $\mathcal{F}$ for each sentence. We then utilize graph-based algorithms to induce a dependency tree from $\mathcal{F}$, and compare it against ground-truth whose annotations

are linguistically motivated.

Experiment Setup. We evaluate the induced trees on two benchmarks: (1) the PUD treebank described in Section 3. (2) the WSJ10 treebank, which contains 7,422 sentences (all less than 10 words after punctuation removal) from the Penn Treebank (PTB) (Marcus et al., 1993). Note that the original PTB does not contain dependency annotations. Thus, we convert them into Universal Dependencies using Stanford CoreNLP. We denote this set as WSJ10-U.

Next, two parsing algorithms, namely, the Eisner algorithm (1996) and Chu-Liu/Edmonds (CLE) algorithm (1965; 1967), are utilized to extract the projective and non-projective unlabeled dependency trees, respectively. Given that our impact matrices have no knowledge about the dependency root of the sentence, we use the gold root in our analysis. Introducing the gold root may artificially improve our results slightly. We thus apply this bias evenly across all baselines to ensure a fair comparison, as done in (Raganato and Tiedemann, 2018; Htut et al., 2019).

We compared our approach against the following baselines: (1) right-(left-) chain baseline, which always selects the next(previous) word as dependency head. (2) A *random* BERT baseline, with which we randomly initialize weights of the BERT model (Htut et al., 2019), then use our methods to induce dependency trees.

We measure model performance using Unlabeled Attachment Score (UAS). We note that UAS has been shown to be highly sensitive to annotation variations (Schwartz et al., 2011; Tsarfaty et al., 2011; Kübler et al., 2009). Therefore, it may not be a fair evaluation metric for analyzing and interpreting BERT. To reflect the real quality of the dependency structures that are retained in BERT, we also report Undirected UAS (UUAS) (Klein and Manning, 2004) and the Neutral Edge Direction (NED) scores (Schwartz et al., 2011).

Results. Tables 1 and 2 show the results of our dependency probes. From Table 1, we see that although BERT is trained without any explicit supervision from syntactic dependencies, to some extent the syntax-aware representation already exists in it. The best UAS scores it achieves (Eisner+Dist) are substantially higher than that of the random BERT baseline with respect to both WSJ10-U(+41.7) and PUD(+31.5). Moreover, the *Dist* method significantly outperforms the *Prob* method on both datasets we evaluated. We thus use *Dist* as the default distance function in our later discussion. We also note that the Eisner algorithm shows a clear advantage over CLE since English sentences are mostly projective. However, our best performing method does not go much beyond the strong right-chain baseline (with gold root modified), showing that the dependency relations learned are mostly those simple and local ones.

Model	Parsing UAS	
	WSJ10-U	PUD
Right-chain	49.5	35.0
Left-chain	20.6	10.7
Random BERT	16.9	10.2
Eisner+Dist	**58.6**	**41.7**
Eisner+Prob	52.7	34.1
CLE+Dist	51.5	33.2

Table 1: UAS results of BERT on unsupervised dependency parsing.

Model	UAS	UUAS	NED
Eisner+Dist	41.7	52.1	69.6
Right-chain	35.0	39.9	41.2

Table 2: Performance on PUD when evaluated using UAS, UUAS, and NED.

For reference, the famous unsupervised parser – DMV (Klein and Manning, 2004) achieves a 43.2 UAS on WSJ10 with Collins (1999) conventions. Note that the DMV parser utilizes POS tags for training while ours start with the gold root. The results are therefore not directly comparable. By putting them together, however, we see potential room for improvement for current neural unsupervised dependency parsing systems in the BERT era.

From Table 2, we see that although BERT only outperforms the right-chain baseline modestly in terms of UAS, it shows significant improvements on UUAS (+12.2) and NED (+28.4). We also make similar observation with WSJ10-U. This suggests that BERT does capture inter-word dependencies despite that it may not totally agree with one specific human-designed governor-dependent schema. We manually inspect those discrepancies and observe that they can also be syntactically valid. For instance, consider the sen-

tence "It closed on Sunday.". For the phrase "on Sunday", our method selects the functional word "on" as the head while the gold-standard annotation uses a lexical head ("Sunday")[4].

The above findings prove that BERT has learned its own syntax as a by-product of self-supervised training, not by directly copying any human design. However, giving the superior performance of BERT on downstream tasks, it is natural to ask if BERT is learning an empirically useful structure of language. We investigate this question in Sec 6.

4.2 Constituency Probe

We now examine the extent BERT learns about the constituent structure of sentences. We first present the algorithm for unsupervised constituent parsing, which executes in a top-down manner by recursively splitting larger constituents into smaller ones.

Top-Down Parsing. Given a sentence as a sequence of tokens $\mathbf{x} = [x_1, \ldots, x_T]$ and the corresponding impact matrix $\mathcal{F}$. We start by finding the best splitting position k that will separate the sentence into constituents $((\mathbf{x}_{<k}), (x_k, (\mathbf{x}_{>k})))$, where $\mathbf{x}_{<k} = [x_1, \ldots, x_{k-1}]$. The best splitting position ensures that each constituent has a large average impact between words within it (thus those words more likely to form a constituent) while at the same time the impact between words of different constituents are kept as small as possible (thus they are unlikely to be in the same constituent). Mathematically, we decide the best k for the constituent $\mathbf{x} = [x_i, x_{i+1}, \ldots, x_j]$ by the following optimization:

$$\begin{aligned} \arg\max_k \quad & \mathcal{F}^{i,\ldots,k}_{i,\ldots k} + \mathcal{F}^{k+1,\ldots,j}_{k+1,\ldots,j} \\ & - \mathcal{F}^{k+1,\ldots,j}_{i,\ldots,k} - \mathcal{F}^{i,\ldots,k}_{k+1,\ldots,j} \end{aligned} \tag{1}$$

where $\mathcal{F}^{i,\ldots,k}_{i,\ldots k} = \frac{\sum_{a=i}^{k}\sum_{b=i}^{k} f(x_a, x_b)}{|\theta|}$, and $|\theta|$ is the number of off-diagonal elements in the corresponding impact matrix $\begin{bmatrix} x_{i,i} & \ldots & x_{i,k} \\ \vdots & \ddots & \vdots \\ x_{k,i} & \ldots & x_{k,k} \end{bmatrix}$. We recursively split $(\mathbf{x}_{<k})$ and $(\mathbf{x}_{>k})$ until only single words remain. Note that this top-down strategy is similar to that of ON-LSTM (Shen et al., 2019) and PRPN (Shen et al., 2018), but differs from them in that ON-LSTM and PRPN decide the splitting position based on a "syntactic distance vector" which is explicitly modeled by a special network component. To distinguish our approach from the others, we denote our parser as **MART** (**MA**t**R**ix-based **T**op-down parser)

Experiment Setup. We follow the experiment setting in Shen et al (2019; 2018) and evaluate our method on the 7,422 sentences in WSJ10 dataset and the PTB23 dataset (the traditional PTB test set for constituency parsing).

Results. Table 3 shows the results of our constituency probes. From the table, we see that BERT outperforms most baselines on PTB23, except for the second layer of ON-LSTM. Note that all these baselines have specifically-designed architectures for the unsupervised parsing task, while BERT's knowledge about constituent formalism emerges purely from self-supervised training on unlabeled text.

It is also worth noting that recent results (Dyer et al., 2019; Li et al., 2019a) have suggested that the parsing algorithm used by ON-LSTM (PRPN) is biased towards the right-branching trees of English, leading to inflated F1 compared to unbiased parsers. To ensure a fair comparison with them, we also introduced this right-branching bias. However, our results show that our method is also robust without this bias (e.g., only 0.9 F1 drops on PTB23).

To further understand the strengths and weaknesses of each system, we analyze their accuracies by constituent tags. In Table 3, we show the accuracies of five most common tags in PTB23. We find that the success of PRPN and ON-LSTM mainly comes from the accurate identification of NP (noun phrase), which accounts for 38.5% of all constituents. For other phrase-level tags like VP (verb phrase) and PP (prepositional phrase), the accuracies of BERT are competitive. Moreover, for clause level tags, BERT significantly outplays ON-LSTM. Take SBAR (clause introduced by a subordinating conjunction) for example, BERT achieves an accuracy of 51.9%, which is about 3.4 times higher than that of ON-LSTM. One possible interpretation is that BERT is pre-trained on long contiguous sequences extracted from a document-level corpus. And the masking strategy (randomly mask 15% tokens) utilized may allow BERT to learn to model a sequence of words (might form a clause).

[4]This specific choice is actually agreed with the YM (Yamada and Matsumoto, 2003) schema.

Model	Parsing F1		Accuracy on PTB23 by Tag				
	WSJ10	PTB23	NP	VP	PP	S	SBAR
PRPN-LM	70.5	37.4	63.9	-	24.4	-	-
ON-LSTM 1st-layer	42.8	24.0	23.8	15.6	18.3	48.1	16.3
ON-LSTM 2nd-layer	66.8	49.4	61.4	51.9	55.4	54.2	15.4
ON-LSTM 3rd-layer	57.6	40.4	57.5	13.5	47.2	48.6	10.4
300D ST-Gumbel w/o Leaf GRU	-	25.0	18.8	-	9.9	-	-
300D RL-SPINN w/o Leaf GRU	-	13.2	24.1	-	14.2	-	-
MART	58.0	42.1	44.6	47.0	50.6	66.1	51.9
Right-Branching	56.7	39.8	25.0	71.8	42.4	74.2	68.8
Left-Branching	19.6	9.0	11.3	0.8	5.0	44.1	5.5

Table 3: Unlabeled parsing F1 results evaluated on WSJ10 and PTB23.

5 Discourse Probe

Having shown that clause-level structures are well-captured in BERT using the constituency probe, we now explore a more challenging probe – probing BERT's knowledge about the structure of a document. A document contains a series of coherent text spans, which are named Elementary Discourse Units (EDUs) (Yang and Li, 2018; Polanyi, 1988). EDUs are connected to each other by discourse relations to form a document. We devise a discourse probe to investigate how well BERT captures structural correlations between EDUs. As the foundation of the probe, we extract an EDU-EDU impact matrix for each document using span-level perturbation.

Setup. We evaluate our probe on the discourse dependency corpus SciDTB (Yang and Li, 2018). We do not use the popular discourse corpora RST-DT (Carlson et al., 2003) and PDTB (Prasad et al.) because PDTB focuses on local discourse relations but ignores the whole document structure, while RST-DT introduces intermediate nodes and does not cover non-projective structures. We follow the same baseline settings and evaluation procedure in Sec 4.1, except that we remove gold root from our evaluation since we want to compare the accuracy by syntactic distances.

Results. Table 4 shows the performance of our discourse probes. We find that both Eisner and CLE achieve significantly higher UAS (+28) than the random BERT baseline. This suggests that BERT is aware of the structure of the document it is given. In particular, we observe a decent accuracy in identifying discourse relations between adjacent EDUs, perhaps due to the "next sentence prediction" task in pre-training, as pointed out in (Shi and Demberg, 2019). However, our probes fall behind the left-chain baseline, which benefits from its strong structural prior[5] (principal clause mostly in front of its subordinate clause). Our finding sheds some lights on BERT's success in downstream tasks that have paragraphs as input (e.g., Question Answering).

Model	UAS	Accuracy by distance			
		0	1	2	5
Right-chain	10.7	20.5	-	-	-
Left-chain	**41.5**	**79.5**	-	-	-
Random BERT	6.3	20.4	7.5	3.5	0.0
Eisner+Dist	34.2	61.6	7.3	**7.6**	**12.8**
CLE+Dist	34.4	63.8	3.3	3.5	2.6

Table 4: Performance of different discourse parser. The distance is defined as the number of EDUs between head and dependent.

6 BERT-based Trees VS Parser-provided Trees

Our probing results suggest that although BERT has captured a certain amount of syntax, there are still substantial disagreements between the syntax BERT learns and those designed by linguists. For instance, our constituency probe on PTB23 significantly outperforms most baselines, but it only roughly agree with the PTB formalism (41.2% F1). However, BERT has already demonstrated its superiority in many downstream tasks. An interesting question is whether *BERT is learning an*

[5]For reference, a supervised graph-based parser (Li et al., 2014) achieves an UAS of 57.6 on SciDTB

empirically useful or even better structure of a language.

To answer this question, we turn to neural networks that adopt dependency parsing trees as the explicit structure prior to improve downstream tasks. We replace the ground-truth dependency trees those networks used with ones induced from BERT and approximate the effectiveness of different trees by the improvements they introduced.

We conduct experiments on the Aspect Based Sentiment Classification (**ABSC**) task (Pontiki et al., 2014). ABSC is a fine-grained sentiment classification task aiming at identifying the sentiment expressed towards each aspect of a given target entity. As an example, in the following comment of a restaurant, "I hated their fajitas, but their salads were great", the sentiment polarities for aspect *fajitas* is negative and that of *salads* is positive. It has been shown in Zhang et al. (2019) that injecting syntactic knowledge into neural networks can improve ABSC accuracy. Intuitively, given an aspect, a syntactically closer context word should play a more important role in predicting that aspect's sentiment. They integrate the distances between context words and the aspect on a dependency tree into a convolution network and build a Proximity-Weighted Convolution Network (PWCN). As a naive baseline, they compare with network weighted by relative position between aspect and context words.

Setup. We experimented on two datasets from SemEval 2014 (Pontiki et al., 2014), which consist of reviews and comments from two categories: LAPTOP and RESTAURANT. We adopt the standard evaluation metrics: Accuracy and Macro-Averaged F1. We follow the instructions of Zhang et al. (2019) to run the experiments 5 times with random initialization and report the averaged performance. We denote the original PWCN with relative position information as PWCN-Pos, and that utilizes dependency trees constructed by SpaCy[6] as PWCN-Dep. SpaCy has reported an UAS of 94.5 on English PTB and so it can serve as a good reference for human-designed dependency schema. We also compare our model against two trivial trees (left-chain and right-chain trees). For our model, we feed the corpus into BERT and extract dependency trees with the best performing setting: Eisner+Dist. For parsing, we introduce an inductive bias to favor short dependencies (Eisner and Smith, 2010). To ensure a fair comparison, we induce the root word from the impact matrix $\mathcal{F}$ instead of using the gold root. Specifically, we select the root word x_k based on the simple heuristic $\arg\max_i \sum_{j=1}^{T} f(x_i, x_j)$.

[6]https://spacy.io/

Model	Laptop		Restaurant	
	Acc	Macro-F1	Acc	Macro-F1
LSTM	69.63	63.51	77.99	66.91
PWCN				
+Pos	75.23	71.71	81.12	71.81
+Dep	76.08	72.02	80.98	72.28
+Eisner	75.99	72.01	**81.21**	**73.00**
+right-chain	75.64	71.53	81.07	72.51
+left-chain	74.39	70.78	80.82	72.71

Table 5: Experimental results of aspect based sentiment classification.

Results. Table 5 presents the performance of different models. We observe that the trees induced from BERT is either on-par (LAPTOP) or marginally better (RESTAURANT) in terms of downstream task's performance when comparing with trees produced by SpaCy. LAPTOP is considerably more difficult than RESTAURANT due to the fact that the sentences are generally longer, which makes inducing dependency trees more challenging. We also see that the Eisner trees generally perform better than the right-/left- chain baselines. It is also worth noting that the right-chain baseline also outperforms PWCN+Dep on RESTAURANT, which leads to an exciting future work that investigates how encoding structural knowledge can help ABSC.

Our results suggest that although the tree structures BERT learns can disagree with parser-provided-linguistically-motivated ones to a large extent, they are also empirically useful to downstream tasks, at least to ABSC. As future work, we plan to extend our analysis to more downstream tasks and models, like those reported in Shi (2018).

7 Related Work

There has been substantial research investigating what pre-trained language models have learned about languages' structures.

One rising line of research uses probing classifiers to investigate the different syntactic properties captured by the model. They are generally referred to as "probing task" (Conneau et al., 2018), "diagnostic classifier" (Giulianelli et al., 2018),

and "auxiliary prediction tasks" (Adi et al., 2017). The syntactic properties investigated range from basic ones like sentence length (Shi et al., 2016; Jawahar et al., 2019), syntactic tree depth (Jawahar et al., 2019), and segmentation (Liu et al., 2019) to challenging ones like syntactic labeling (Tenney et al., 2019a,b), dependency parsing (Hewitt and Manning, 2019; Clark et al., 2019), and constituency parsing (Peters et al., 2018a). However, when a probe achieves high accuracy, its difficult to differentiate if it is the representation that encodes targeted syntactic information, or it is the probe that just learns the task (Hewitt and Liang, 2019).

In line with our work, recent studies seek to find correspondences between parts of the neural network and certain linguistic properties, without explicit supervision.

Most of them focus on analyzing attention mechanism, by extracting syntactic tree for each attention head and layer individually (Raganato and Tiedemann, 2018; Clark et al., 2019). Their goal is to check if the attention heads of a given pre-trained model can track syntactic relations better than chance or baselines. In particular, Raganato and Tiedemann (2018) analyze a machine translation model's encoder by extracting dependency trees from its self-attention weights, using Chu-Liu/Edmonds algorithm. Clark et al. (2019) conduct a similar investigation on BERT, but the simple head selection strategy they used does not guarantee a valid dependency tree. Mareček and Rosa (2018) propose heuristic methods to convert attention weights to syntactic trees. However, they do not quantitatively evaluate their approach. In their later study (Mareček and Rosa, 2019), they propose a bottom-up algorithm to extract constituent trees from transformer-based NMT encoders and evaluate their results on three languages. Htut et al. (2019) reassess these works but find that there are no generalist heads that can do holistic parsing. Hence, analyzing attention weights directly may not reveal much of the syntactic knowledge that a model has learned. Recent dispute about attention as explanation (Jain and Wallace, 2019; Serrano and Smith, 2019; Wiegreffe and Pinter, 2019) also suggests that the attention's behavior does not necessarily represent that of the original model.

Another group of research examine the outputs of language models on carefully chosen input sentences (Goldberg, 2019; Bacon and Regier, 2019). They extend previous works (Linzen et al., 2016; Gulordava et al., 2018; Marvin and Linzen, 2018) on subject-verb agreement test (generating the correct number of a verb far away from its subject) to provide a measure of the models syntactic ability. Their results show that the BERT model captures syntax-sensitive agreement patterns well in general. However, subject-verb agreement cannot provide more nuanced tests of other complex structures (e.g., dependency structure, constituency structure), which are the interest of our work.

Two recent works also perturb the input sequence for model interpretability (Rosa and Mareček, 2019; Li et al., 2019b). However, these works only perturb the sequence once. Rosa and Mareček (2019) utilize the original MLM objective to estimate each word's "reducibility" and import simple heuristics into a right-chain baseline to construct dependency trees. Li et al. (2019b) focus on evaluating word alignment in NMT, but unlike our two-step masking strategy, they only replace the token of interest with a zero embedding or a randomly sampled word in the vocabulary.

8 Discussion & Conclusion

One concern shared by our reviewers is that performance of our probes are underwhelming: the induced trees are barely closer to linguist-defined trees than simple baselines (e.g., rightbranching) and are even worse in the case of discourse parsing. However, this does not mean that supervised probes are wrong or that BERT captures less syntax than we thought. In fact, there is actually no guarantee that our probe will find a strong correlation with human-designed syntax, since we do not introduce the human-designed syntax as supervision. What we found is the "natural" syntax inherent in BERT, which is acquired from self-supervised learning on plain text. We would rather say our probe complements the supervised probing findings in two ways. First, it provides a lower-bound (on the unsupervised syntactic parsing ability of BERT). By improving this lower-bound, we could uncover more "accurate" information to support supervised probes' findings. Second, we show that when combined with a down-stream application (sec 6), the syntax learned by BERT might be empirically helpful despite not totally identical to the human design.

In summary, we propose a parameter-free probing technique to complement current line of work on interpreting BERT through probes. With carefully designed two-stage perturbation, we obtain impact matrices from BERT. This matrix mirrors the function of attention mechanism that captures inter-word correlations, except that it emerges through the output of BERT model, instead of from intermediate representations. We devise algorithms to extract syntactic trees from this matrix. Our results reinforce those of (Hewitt and Manning, 2019; Liu et al., 2019; Jawahar et al., 2019; Tenney et al., 2019b,a) who demonstrated that BERT encodes rich syntactic properties. We also extend our method to probe document structure, which sheds lights on BERT's effectiveness in modeling long sequences. Finally, we find that feeding the empirically induced dependency structures into a downstream system (Zhang et al., 2019) can further improve its accuracy. The improvement is compatible with or even superior to a human-designed dependency schema. This offers an insight into BERT's success in downstream tasks. We leave it for future work to use our technique to test other linguistic properties (e.g., coreference) and to extend our study to more downstream tasks and systems.

9 Acknowledgement

We would like to thank Lingpeng Kong from DeepMind for his constructive feedback of the paper. This research is supported by Hong Kong Research Grant Council GRF grants 17254016.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. Fine-grained Analysis of Sentence Embeddings Using Auxiliary Prediction Tasks. In *ICLR '17*.

Geoff Bacon and Terry Regier. 2019. Does BERT agree? Evaluating knowledge of structure dependence through agreement relations. *ArXiv*.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2003. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Current and new directions in discourse and dialogue*, pages 85–112. Springer.

Yoeng-Jin Chu. 1965. On the shortest arborescence of a directed graph. *Scientia Sinica*, 14:1396–1400.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *BlackboxNLP@ACL*, pages 276–286.

Michael Collins. 1999. *HEAD-DRIVEN STATISTICAL MODELS FOR NATURAL LANGUAGE PARSING*. Ph.D. thesis, University of Pennsylvania.

Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805.

Chris Dyer, Gábor Melis, and Phil Blunsom. 2019. A critical analysis of biased parsers in unsupervised parsing. *arXiv preprint arXiv:1909.09428*.

Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the national Bureau of Standards B*, 71(4):233–240.

Jason Eisner and Noah A Smith. 2010. Favor short dependencies: Parsing with soft and hard constraints on dependency length. In *Trends in Parsing Technology*, pages 121–150. Springer.

Jason M Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pages 340–345. Association for Computational Linguistics.

Mario Giulianelli, Jack Harding, Florian Mohnert, Dieuwke Hupkes, and Willem Zuidema. 2018. Under the hood: Using diagnostic classifiers to investigate and improve how language models track agreement information. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 240–248.

Yoav Goldberg. 2019. Assessing BERT's Syntactic Abilities. *ArXiv*, pages 2–5.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless Green Recurrent Networks Dream Hierarchically. In *NAACL-HLT*, pages 1195–1205.

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.

John Hewitt and Christopher D. Manning. 2019. A Structural Probe for Finding Syntax in Word Representations. In *Naacl*, pages 4129–4138.

Phu Mon Htut, Jason Phang, Shikha Bordia, and Samuel R. Bowman. 2019. Do attention heads in bert track syntactic dependencies?

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.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What Does BERT Learn about the Structure of Language? In *ACL*, pages 3651–3657.

Dan Klein and Christopher D Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 478. Association for Computational Linguistics.

Sandra Kübler, Ryan McDonald, and Joakim Nivre. 2009. Dependency parsing. *Synthesis Lectures on Human Language Technologies*, 1(1):1–127.

Bowen Li, Lili Mou, and Frank Keller. 2019a. An imitation learning approach to unsupervised parsing. *arXiv preprint arXiv:1906.02276*.

Sujian Li, Liang Wang, Ziqiang Cao, and Wenjie Li. 2014. Text-level discourse dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 25–35.

Xintong Li, Guanlin Li, Lemao Liu, Max Meng, and Shuming Shi. 2019b. On the Word Alignment from Neural Machine Translation. In *ACL*, pages 1293–1303.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the Ability of LSTMs to Learn Syntax-Sensitive Dependencies. *TACL*, 4(1990):521–535.

Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank.

David Mareček and Rudolf Rosa. 2018. Extracting Syntactic Trees from Transformer Encoder Self-Attentions. In *BlackboxNLP@EMNLP*, pages 347–349.

David Mareček and Rudolf Rosa. 2019. From Balustrades to Pierre Vinken: Looking for Syntax in Transformer Self-Attentions. In *BlackboxNLP@ACL*, pages 263–275.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202.

Matthew Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018a. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke S. Zettlemoyer. 2018b. Deep contextualized word representations. *ArXiv*, abs/1802.05365.

Livia Polanyi. 1988. A formal model of the structure of discourse. *Journal of pragmatics*, 12(5-6):601–638.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K Joshi, and Bonnie L Webber. The penn discourse treebank 2.0. Citeseer.

Alessandro Raganato and Jörg Tiedemann. 2018. An Analysis of Encoder Representations in Transformer-Based Machine Translation. In *BlackboxNLP@EMNLP*, pages 287–297.

Rudolf Rosa and David Mareček. 2019. Inducing Syntactic Trees from BERT Representations. In *BlackboxNLP@ACL*.

Roy Schwartz, Omri Abend, Roi Reichart, and Ari Rappoport. 2011. Neutralizing linguistically problematic annotations in unsupervised dependency parsing evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 663–672. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy.

Yikang Shen, Zhouhan Lin, Chin wei Huang, and Aaron Courville. 2018. Neural language modeling by jointly learning syntax and lexicon. In *International Conference on Learning Representations*.

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2019. Ordered neurons: Integrating tree structures into recurrent neural networks.

Haoyue Shi, Hao Zhou, Jiaze Chen, and Lei Li. 2018. On tree-based neural sentence modeling. In *International Conference on Learning Representations*.

Wei Shi and Vera Demberg. 2019. Next sentence prediction helps implicit discourse relation classification within and across domains. 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 5794–5800.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does String-Based Neural MT Learn Source Syntax? In *EMNLP*, Table 2, pages 1526–1534.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 49th Annual Meeting of the 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. 2019b. What do you learn from context? Probing for sentence structure in contextualized word representations. In *ICLR*, pages 1–17.

Reut Tsarfaty, Joakim Nivre, and Evelina Ndersson. 2011. Evaluating dependency parsing: robust and heuristics-free cross-nnotation evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 385–396. 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.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of the Eighth International Conference on Parsing Technologies*, pages 195–206.

An Yang and Sujian Li. 2018. Scidtb: Discourse dependency treebank for scientific abstracts. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 444–449.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *ArXiv*, abs/1906.08237.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajic, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, et al. 2017. Conll 2017 shared task: Multilingual parsing from raw text to universal dependencies. In *CoNLL 2017 Shared Task*, pages 1–19. Association for Computational Linguistics.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019. Syntax-aware aspect-level sentiment classification with proximity-weighted convolution network. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1145–1148. ACM.