

Syntax Representation in Word Embeddings and Neural Networks – A Survey

Tomasz Limisiewicz and David Mareček

Institute of Formal and Applied Linguistics, Faculty of Mathematics and Physics, Charles University
{limisiewicz, marecek}@ufal.mff.cuni.cz

Abstract: Neural networks trained on natural language processing tasks capture syntax even though it is not provided as a supervision signal. This indicates that syntactic analysis is essential to the understating of language in artificial intelligence systems. This overview paper covers approaches of evaluating the amount of syntactic information included in the representations of words for different neural network architectures. We mainly summarize research on English monolingual data on language modeling tasks and multilingual data for neural machine translation systems and multilingual language models. We describe which pre-trained models and representations of language are best suited for transfer to syntactic tasks.

1 Introduction

Modern methods of natural language processing (NLP) are based on complex neural network architectures, where language units are represented in a metric space [23, 28, 29, 9, 30]. Such a phenomenon allows us to express linguistic features (i.e., morphological, lexical, syntactic) mathematically.

The method of obtaining such representation and their interpretations were described in multiple overview works. Almeida and Xexéo surveyed different types of static word embeddings [1], and Liu et al. [18] focused on contextual representations found in the most recent neural models. Belinkov and Glass [4] surveyed the strategies of interpreting latent representation. Best to our knowledge, we are the first to focus on the syntactic and morphological abilities of the word representations. We also cover the latest approaches, which go beyond the interpretation of latent vectors and analyze the attentions present in state-of-the-art Transformer models.

2 Vector Representations of Words

This section introduces several types of architectures that we will analyze in this work.

2.1 Static Word Embeddings

In the classical methods of language representation, each word is assigned a vector regardless of its current context. In the Latent Semantic Analysis [8], the representation was

obtained by counting word frequency across documents on distinct subjects.

In more recent approaches, a shallow neural network is used to predict each word based on context (Word2Vec [23]) or approximate the frequency of coocurrence for a pair of words (GloVe [28]). One explanation of the effectiveness of these algorithms is the distributional hypothesis [11]: "words that occur in the same contexts tend to have similar meanings".

2.2 Contextual Word Vectors in Recurrent Networks

The main disadvantage of the static word embeddings is that they do not take into account the context of words. This is especially an issue for languages rich in words that have multiple meanings.

The contextual embeddings introduced in [29] and [22] are able to encode both words and their contexts. They are based on recurrent neural networks (RNN) and are typically trained on language modeling or machine translation tasks using large text corpora. The outputs of the RNN layers are context-dependent representations that are proven to perform well when used as inputs for other NLP tasks with much less training data available.

Another improvement of context modeling was possible thanks to the attention mechanism [2]. It allowed passing the information from the most relevant part of the RNN encoder, instead of using only the contextual representation of the last token.

2.3 Contextual Representation in Transformers

The most recent and widely used architecture is the Transformer [32]. It consists of several (6 to 24) layers, and each token position in each layer has the ability to attend to any position in the previous layer using a self-attention mechanism. Training such architecture can be easily parallelized since individual tokens can be processed independently; their positions are encoded within the input embeddings. An example of visualization of attention distribution computed in Transformer trained for language modeling (BERT [9]) is presented in Figure 1.

In addition to vectors, Transformer includes latent representation in the form of self-attention weights, which are two-dimensional matrices. We summarize the research on the syntactic properties of attention weights in Section 5.

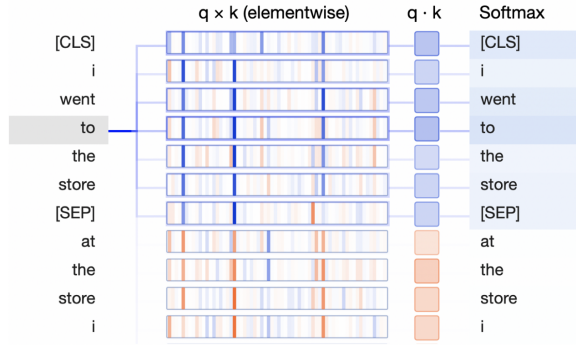


Figure 1: Visualization of attention mechanism in Transformer architecture. It shows which parts of the text are important to compute the representation for the word “to”. Created in BertViz framework [33].

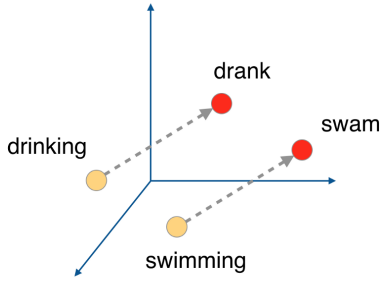


Figure 2: Spatial distribution of word embeddings depends on syntactic roles of words (visualization created by Ashutosh Singh).

3 Measures of Syntactic Information

This sections describes the metrics used to evaluate syntactic information captured by the word embeddings and latent representation.

3.1 Syntactic Analogies

In the recent revival of word embeddings[23, 28], a strong focus was put on examining the phenomenon of encoding analogies in multidimensional space. That is to say, the shift vector between pairs of analogous words is approximately constant, e.g., the pairs *drinking – drank*, *swimming – swam* in Figure 2.

Syntactic analogies of this type are particularly relevant for this overview. They include the following relations: adjective – adverb; singular – plural; adjective – comparative – superlative; verb – present participle – past participle. The syntactic analogy is usually evaluated on Google Analogy Test Set [23].¹

¹The test set is called syntactic by authors; nevertheless, it mostly focuses on morphological features.

An evaluation example consists of two word pairs represented by the embeddings: $(v_1, v_2), (u_1, u_2)$. We compute the analogy shift vector as the difference between embeddings of the first pair $s = v_2 - v_1$. The result is positive if the nearest word embedding to the vector $u_1 + s$ is u_2 .

$$WA = \frac{|\{(v_1, v_2, u_1, u_2) : u_2 \approx u_1 + v_2 - v_1\}|}{|\{(v_1, v_2, u_1, u_2)\}|} \quad (1)$$

3.2 Sequence Tagging

Sequence tagging is a multiclass classification problem. The aim is to predict the correct tag for each token of a sequence. A typical example is the part of speech (POS) tagging. The accuracy evaluation is straightforward: the number of correctly assigned tags is divided by the number of tokens.

3.3 Syntactic structure prediction

The inference of reasonable syntactic structures from word representations is the most challenging task covered in our survey. There are attempts to predict both the dependency[12, 31, 15, 7] and constituency trees [21, 13]. Dependency trees are evaluated using unlabeled attachment score (UAS) or its undirected variant (UUAS):

$$UAS = \frac{\#correctly_attached_words}{\#all_words} \quad (2)$$

The equation for Labeled Attachment Score is the same, but it requires predicting a dependency label for each edge. For constituency, trees we define precision (P) and recall (R) for correctly predicted phrases.

$$P = \frac{\#correct_phrases}{\#gold_phrases}, \quad R = \frac{\#correct_phrases}{\#predicted_phrases} \quad (3)$$

Usually, F1 is reported, which is a harmonic mean of precision and recall.

3.4 Attention’s Dependency Alignment

In Section 5 we describe the examination of syntactic properties of self-attention matrices. It can be evaluated using *Dependency Alignment* [34] which sums the attention weights at the positions corresponding to the pairs of tokens forming a dependency edge in the tree.

$$DepAl_A = \frac{\sum_{(i,j) \in E} A_{i,j}}{\sum_{i=1}^N \sum_{j=1}^N A_{i,j}} \quad (4)$$

Dependency Accuracy [35, 7, 15] is an alternative metric; for each dependency label it measures how often the relation’s governor/dependent is the most attended token by the dependent/governor.

$$DepAcc_{l,d,A} = \frac{|\{(i,j) \in E_{l,d} : j = \arg \max A_{i,j}\}|}{|E_{l,d}|} \quad (5)$$

Notation: E is a set of all dependency tree edges and $E_{l,d}$ is a subset of the edges with the label l and with direction d , i.e., in dependent to governor direction the first element of the tuple i is dependent of the relation and the second element j is the governor; A is a self-attention matrix and A_i denotes i^{th} row of the matrix; N is the sequence length.

4 Morphology and Syntax in Word Embeddings and Latent Vectors

In this section, we summarize the research on the syntactic information captured by vector representations of words. We devote a significant attention to POS tagging, which is a popular evaluation objective. Even though it is a morphological task, it is highly relevant to syntactic analysis.

4.1 Syntactic Analogies

The first wave of research on the vector representation of words focused on the statistical distribution of words across distinct topics – Latent Semantic Analysis [8]. It captured statistical properties of words, yet there were no positive results in syntactic analogies retrieval nor encoding syntax.

Google Analogy Test Set was released together with a popular word embedding algorithm Word2Vec [23]. One of the exceptional properties of this method was its high accuracy in the analogy tasks. In particular, the best configuration found the correct syntactic analogy in 68.9 % of cases.

The GloVe embeddings improved the results on syntactic analogies to 69.3% [28]. Much more significant improvement was reported for semantic analogies. They also outperform the variety of other vectorization methods.

In [24] a simple recurrent neural network was trained by language modeling objective. The word representation is taken from the input layer. The evaluation from [23] shows that Word2Vec performs better in syntactic analogy task. This observation is surprising because representations from RNN were proven effective in transfer to other syntactic tasks (we elaborate on that in Sections 4.2 and 4.3). We think that possible explanations could be: 1. the techniques of RNN training have crucially improved in recent years; 2. syntactic analogy focuses on particular words, while for other syntactic tasks, the context is more important.

4.2 Part of Speech Tagging

Measuring to what extent a linguistic feature such as POS is captured in word representations is usually performed by the method called *probing*. In probing, the parameters of the pretrained network are fixed, the output word representations are computed as in the inference mode and then fed to a simple neural layer. Only this simple layer is optimized for a new task.

The number of probing experiments rose with the advent of multilayer² RNNs trained for language modeling and machine translation.

Belinkov et al. [3] probe a recurrent neural machine translation (NMT) system with four layers to predict part of speech tags (along with morphological features). They use Arabic, Hebrew, French, German, and Czech to English pairs. They observe that adding a character-based representation computed by a convolutional neural network in addition to word-embedding input is beneficial, especially for morphologically rich languages.

In a subsequent study [4], the source language of translation now is English and the experiments are conducted solely for this language. It is noted that the most morphosyntactic representation is usually obtained in the middle layers of the network.

The influence of using a particular objective in pre-training RNN model is comprehensively analyzed by Blevins et al. [5]. They pre-train models on four objectives: syntactic parsing, semantic role labeling, machine translation, and language modeling. The two former objectives may reveal morphosyntactic information to a larger extent than other mentioned here settings. Particularly, the probe of RNN syntactic parser achieves near-perfect accuracy in part of speech tagging.

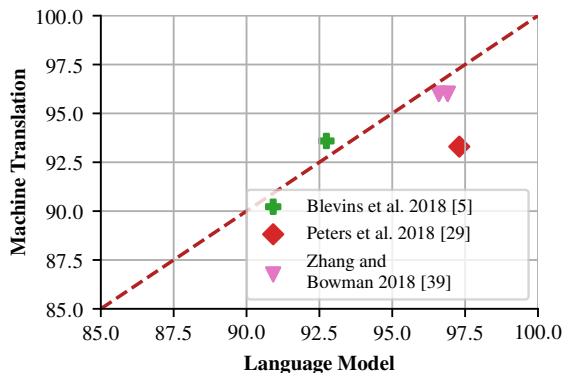
The introduction of ELMo [29] brought a remarkable advancement in transfer learning from the RNN language model to a variety of other NLP tasks. The authors examined POS capabilities of the representations and compared the results with the neural machine translation system CoVe [22], which also uses RNN architecture.

Zhang et al. [39] perform further experiments with CoVe and ELMo. They demonstrate that language modeling systems are better suited to capture morphology and syntax in the hidden states than machine translation, if comparable amounts of data are used to train both systems. Moreover, the corpora for language modeling are typically more extensive than for machine translation, which can further improve the results.

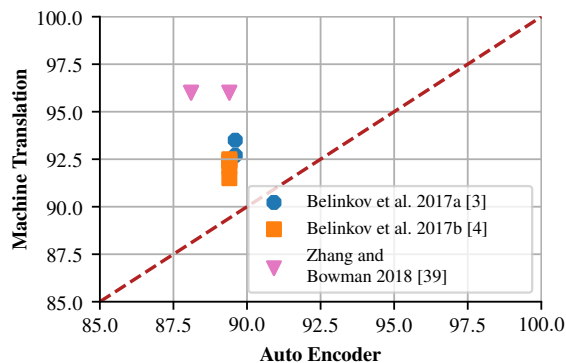
Another comprehensive evaluation of morphological and syntactic capabilities of language models was conducted by Liu et al. [17]. Probing was applied to a language model based on the Transformer architecture (BERT) and compared with ELMo and static word embeddings (Word2Vec). They observe that the hidden states of Transformer do not demonstrate a major increase in probed POS accuracy over the RNN model, even though it is more complex and consists of a larger number of parameters.

POS tag probing was also performed for languages other than English. For instance, Musil [25] trains translation systems (with RNN and Transformer architecture) from Czech to English and examines the learned input embeddings of the model and compares them to a Word2Vec model trained on Czech.

²Layer numbering in this work: We are numbering layers starting from one for the layer closest to the input. Please note that original papers may use different numbering.



(a) Neural machine translation compared with language modeling pre-training



(b) Neural machine translation compared with auto encoder pre-training

Figure 3: Accuracy of POS tag probing from RNN representation by the pre-training objective.

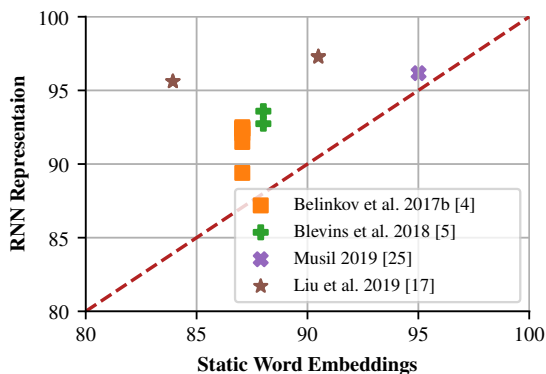


Figure 4: Accuracy of POS tag probing from RNN latent vectors compared with static word embeddings

In Figures 3 and 4, we present a comparison of different settings for POS tag probing. Each point denotes a pair of results obtained in the same paper and the same dataset, but with different types of embeddings or pretraining objectives. Therefore, we can observe that the setting plotted on the y-axis is better than the x-axis setting if the points are above identity function (red dashed line). We cannot say whether a method represented by another point performs better, as the evaluation settings differ.

Figure 4 clearly shows that the RNN contextualization helps in part of speech tagging. As expected, the information about neighboring tokens is essential to predict morphosyntactic functions of words correctly. It is especially true for the homographs, which can have various part of speech in different places in the text.

The influence of RNN’s pre-training task is presented in Figure 3. Machine translation captures much better POS information than auto-encoders, which can be interpreted as translation from and to the same language. It is likely that the latter task is straightforward and therefore does

not require to encode morphosyntax in the latent space. The difference between the results of machine translation and language modeling is small. Zhang et al. [39] show that using a larger corpus for pre-training improves the POS accuracy. The main advantage of language models is that monolingual data is much easier to obtain than parallel sentences necessary to train a machine translation system.

4.3 Syntactic Structure Induction

Extraction of dependency structure is more demanding because instead of prediction for single tokens, every pair of words need to be evaluated.

Blevins et al. [5] propose a feed-forward layer on top of a frozen RNN representation to predict whether a dependency tree edge connects a pair of tokens. They concatenate the vector representation of each of the words and their element-wise product. Such a representation is fed as an input to the binary classifier. It only looks on a pair of tokens at a time, therefore predicted edges may not form a valid tree.

Another approach, induction of the whole syntactic structures from latent representations was proposed by Hewitt and Manning [12]. Their syntactic probing is based on training a matrix which is used to transform the output of network’s layers (they use BERT and ELMo). The objective of the probing is to approximate dependency tree distances between tokens³ by the L2 norm of the difference of the transformed vectors. Probing produces the approximate syntactic pairwise distances for each pair of tokens. The minimum spanning tree algorithm is used on the distance matrix to find the undirected dependency tree. The best configuration employs the 15th layer of BERT large and induces treebank with 82.5% UAS on Penn Treebank with Stanford Dependency annotation (relation directions and punctuation were disregarded in the experiments). The

³Tree distance is the length of the tree path between two tokens

result for BERT is significantly higher than for ELMo, which gave 77.0% when the first layer was probed.

The paper also describes an alternative method of approximating the syntactic depth by the L2 norm of latent vector multiplied by a trainable matrix. The estimated depths allow prediction of the root of a sentence with 90.1% accuracy when representation from the 16th layer of BERT large is probed.

4.4 Multilingual Representations

The subsequent paper by Chi et al. [6] applies the setting from [12] to the multilingual language model mBERT. They train syntactic distance probes on 11 languages and compare UAS of induced trees in four scenarios: 1. training and evaluating on the same languages; 2. training on a single language, evaluating on a different one; 3. training on all languages except the evaluation one; 4. training on all languages, including the evaluation one. They demonstrate that the transfer is effective as the results in all the configurations outperform the baselines⁴. Even in the hardest case – zero-shot transfer from just one language, the result is at least 6.9 percent points above the baselines (for Chinese). Nevertheless, for all the languages, no transfer-learning setting can beat the training and evaluating a probe on the same language.

The paper includes analysis of intrinsic features of the BERT’s vectors transformed by a probe. Noticeably, the vector differences between the representations of words connected by dependency relation are clustered by relation labels, see figure 5.

Multilingual BERT embeddings are also analyzed by Wang et al. [36]. They show that even for the multilingual vectors, the results can be improved by projecting vector spaces across languages. They use Biaffine Graph-based Parser by Dozat and Manning [10], which consists of multiple RNN layers. Therefore, the experiment is not strictly comparable with probing as the most of syntactic information is captured by the parser, and not by the embeddings. The article compares different types of vector representations fed as an input to the parser. It is demonstrated that cross-lingual transformation on mBERT embedding improves the results significantly in LAS of parser trained on English and evaluated on 14 languages (including English); on average, from 60.53% to 63.54%. In comparison to other cross-lingual representations, the proposed method outperforms transformed static embeddings (Fast-Text with SVD) and also slightly outperforms contextual embeddings (XLM).

5 Syntax in Transformer’s Attention Matrices

Besides the vector representations of individual tokens, the Transformer architecture offers another representation

⁴There are two baselines: right-branching tree and probing on randomly initialized mBERT without pretraining

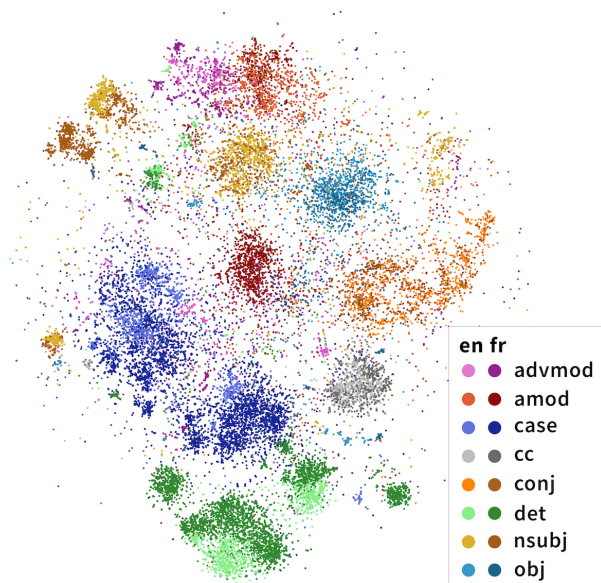


Figure 5: Two dimensional t-SNE visualization of probed mBERT embeddings from [6]. Analysis of the clusters shows that embeddings encode information about the type of dependency relations and, to a lesser extent, language.

with a possible syntactic interpretation – the weights of the self-attention heads. In each head, information can flow from each token to any other one. These connections may be easily analyzed and compared to syntactic relations proposed by linguists. In this section, we will summarize different approaches of extracting syntax from attention. We present the methods both for dependency and constituency structures.

5.1 Dependency Trees

Raganato and Tiedemann [31] induce dependency trees from self-attention matrices of a neural machine translation encoder. They use the maximum spanning tree algorithm to connect pairs of tokens with high attention. Gold root information is used to find the direction of the edges. Trees extracted in this way are generally worse than the right-branching baseline (35.08 % UAS on PUD) and outperform it slightly in a few heads. The maximum UAS is obtained when a dependency structure is induced from one head of the 5th layer of English to Chinese encoder - 38.87% UAS. Nevertheless, their approach assumes that the whole syntactic tree may be induced from just one attention head.

Recent articles focused on the analysis of features and classification of Transformer’s self-attention heads. Vig and Belinkov [34] apply multiple metrics to examine properties of attention matrices computed in a unidirectional language model (GPT-2 [30]). They showed that in some heads, the attentions concentrate on tokens representing specific POS tags and the pairs of tokens are more often attended one to another if an edge in the dependency tree

Research	Transformer Model	Type of tree	Syntactic evaluation	Evaluation data	Percentage of syntactic heads
Raganato and Tiedemann 2019 [31]	NMT Encoder (6 layers 8 heads)	Dependency	Tree induction	PUD [27]	0% - 8% ⁵
Vig and Belinkov 2019 [34]	LM (GPT-2)	Dependency	Dependency Alignment	Wikipedia (automatically annotated)	—
Clark et al. 2019 [7]	LM (BERT)	Dependency	Dependency Accuracy, Tree induction	WSJ Penn Treebank [20]	—
Voita et al. 2019 [35]	NMT Encoder (6 layers 8 heads)	Dependency	Dependency Accuracy	WMT, OpenSubtitles [16] (both automatically annotated)	15% - 19%
Limisiewicz et al. 2020 [15]	LMs (BERT, mBERT)	Dependency	Dependency Accuracy, Tree induction	PUD [27], EuroParl [14] (automatically annotated)	46%
Mareček and Rosa 2019 [21]	NMT Encoder (6 layers 16 heads)	Constituency	Tree induction	EuroParl [14] (automatically annotated)	19% - 33%
Kim et al. 2019 [13]	LMs (BERT, GPT2, RoBERTa, XLNet)	Constituency	Tree induction	WSJ Penn Treebank [20], MNL [37]	—

Table 1: Summary of syntactic properties observed in Transformer’s self-attention heads

connects them, i.e., dependency alignment is high. They observe that the strongest dependency alignment occurs in the middle layers of the model – 4th and 5th. They also point that different dependency types (labels) are captured in different places of the model. Attention in upper layers aligns more with subject relations whereas in the lower layer with modifying relations, such as auxiliaries, determiners, conjunctions, and expletives.

Voita et al. [35] also observed alignment with dependency relations in the encoders of neural machine translation systems from English to Russian, German, or French. They have evaluated dependency accuracy for four dependency labels: noun subject, direct object, adjective modifier, and adverbial modifier. They separately address the cases where a verb attends to a dependent subject, and subject attends to governor verb. The heads with more than 10% improvement over a positional baseline are identified as syntactic ⁶. Such heads are found in all encoder layers except the first one. In further experiments, the authors propose the algorithm to prune the heads from the model with a minimal decrease in translation performance. During pruning, the share of syntactic heads rises from 17% in the original model to 40% when 75% heads are cut out, while a change in translation score is negligible. These results support the claim that the model’s ability to capture syntax is essential to its performance in non-syntactic tasks.

A similar evaluation of dependency accuracy for the BERT language model was conducted by Clark et al. [7].

⁵A head is syntactic when the tree extracted from it surpasses the right-branching chain in terms of UAS. It is a strong baseline for syntactic trees in English. Thus only a few heads are recognized as syntactic.

⁶In the positional baseline, the most frequent offset is added to the index of relation’s dependent/governor to find its governor/dependent, e.g., for adjective to noun relations the most frequent offset is +1 in English

They identify syntactic heads that significantly outperform positional baseline for the following labels: prepositional object, determiner, direct object, possession modifier, auxiliary passive, clausal component, marker, phrasal verb particle. The syntactic heads are found in the middle layers (4th to 8th). However, there is no single head that would capture the information for all the relations.

In another experiment, Clark et al. [7] induce a dependency tree from attentions. Instead of extracting structure from each head [31] they use probing to find the weighted average of all heads. The maximum spanning tree algorithm is used to induce the dependency structure from the average. This approach produces trees with 61% UAS and can be improved to 77% by making weights dependent on the static word representation (fixed GloVe vectors). Both the numbers are significantly higher than right branching baseline 27%.

A related analysis for English (BERT) and the multilingual variant (mBERT) was conducted by Limisiewicz et al. [15]. We have observed that the information about one dependency type is split across many self-attention heads and in other cases, the opposite happens - many heads have the same syntactic function. They extract labeled dependency trees from the averaged heads and achieves 52% UAS and show that in the multilingual model (mBERT) specific relation (noun subject, determines) are found in the same heads across typologically similar languages.

5.2 Constituency trees

There are fewer papers devoted to deriving constituency syntax tree structures.

Mareček and Rosa [21] examined the encoder of the machine translation system for translation between English, French, and German. We observed that in some

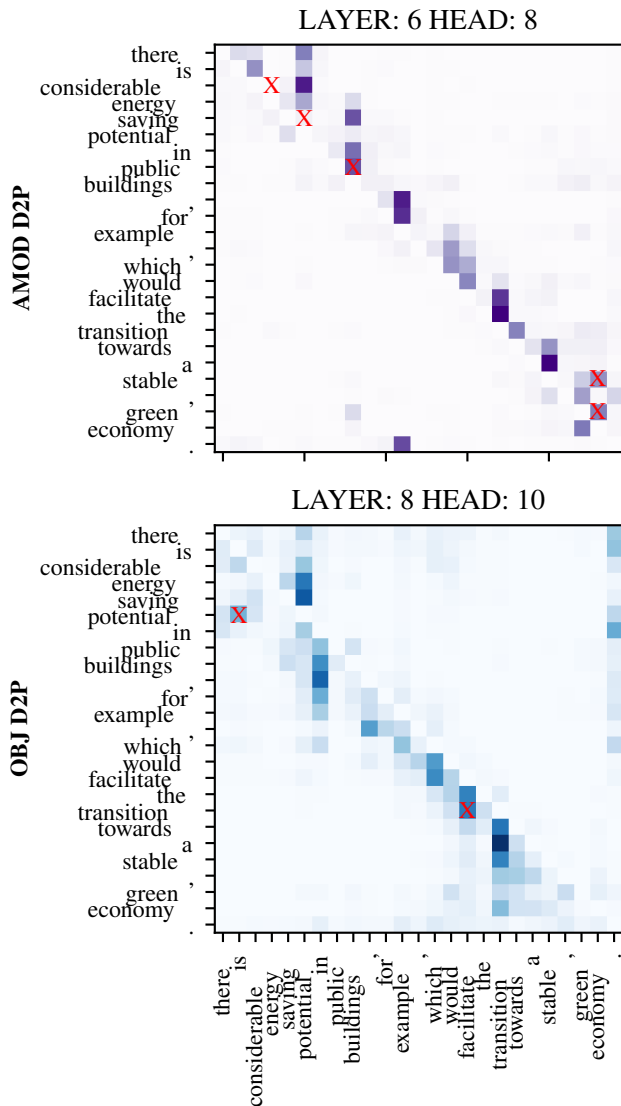


Figure 6: Self-attentions in particular heads of a language model (BERT) aligns with dependency relation adjective modifiers and objects. The gold relations are marked with Xs.

heads, stretches of words attend to the same token forming shapes similar to balustrades (Figure 7). Furthermore, those stretches usually overlap with syntactic phrases. This notion is employed in the new method for constituency tree induction. In their algorithm, the weights for each stretch of tokens are computed by summing the attention focused on the balustrades and then inducing a constituency tree with CKY algorithm [26]. As a result, we produce trees that achieve up to 32.8% F1 score for English sentences, 43.6% for German and 44.2% for French.⁷ The results can be improved by selecting syntactic heads and using only them in the algorithm. This approach requires a sample of 100 annotated sentences for head selection and raises F1

⁷The evaluation was done on 1000 sentences for each language parsed with supervised Stanford Parser

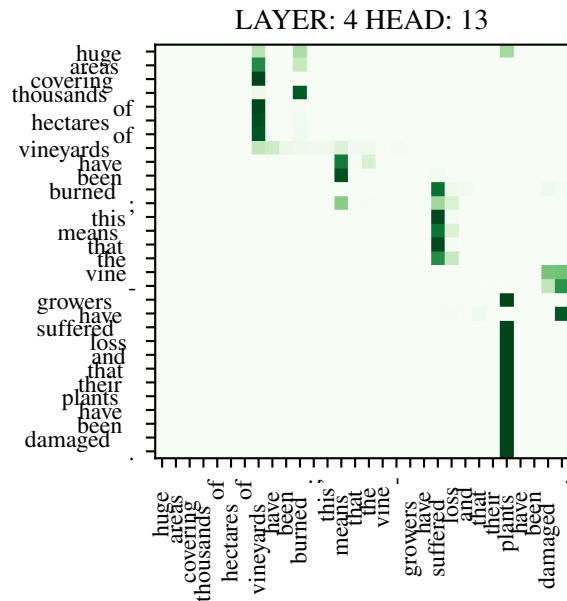


Figure 7: Balustrades observed in NMT's encoder tend to overlap with syntactic phrases.

by up to 8.10 percent points in English.

The extraction of constituency trees from language models was described by Kim et al. [13]. They present a comprehensive study that covers nine types of pre-trained networks: BERT (base, large), GPT-2 [30] (original, medium), RoBERTa [19] (base, large), XLNet [38] (base, large). Their approach is based on computing distance between each pair of subsequent words. In each step, they are branching the tree in the place where the distance is the highest. The authors try three distance measures on the vector outputs of the encoder layer (cosine, L1, and L2 distances for pairs of vectors) and two distance measures on the distributions of token's attention (Jason-Shannon and Hellinger distances for pairs of distribution). In the former case, distances are computed only per layer and in the latter case for each head and average of heads in one layer. The best setting achieves 40.1% F1 score on WSJ Penn Treebank. It uses XLNet-base and Helinger distance on averaged attentions in the 7th layer. Generally, attention distribution distances perform better than vector ones. Authors also observe that models trained on regular language modeling objective (i.e., next word prediction in GPT, XLNet) captured syntax better than masked language models (BERT, RoBERTa). In line with the previous research, the middle layers tend to be more syntactic.

5.3 Syntactic information across layers

Figure 8 summarizes the evaluation of syntactic information across layers for different approaches. In Transformer-based language models: BERT, mBERT, and GPT-2, the middle layers are the most syntactic. In neural machine translation models, the top layers of the encoder are the most syntactic. However, it is important to note that the

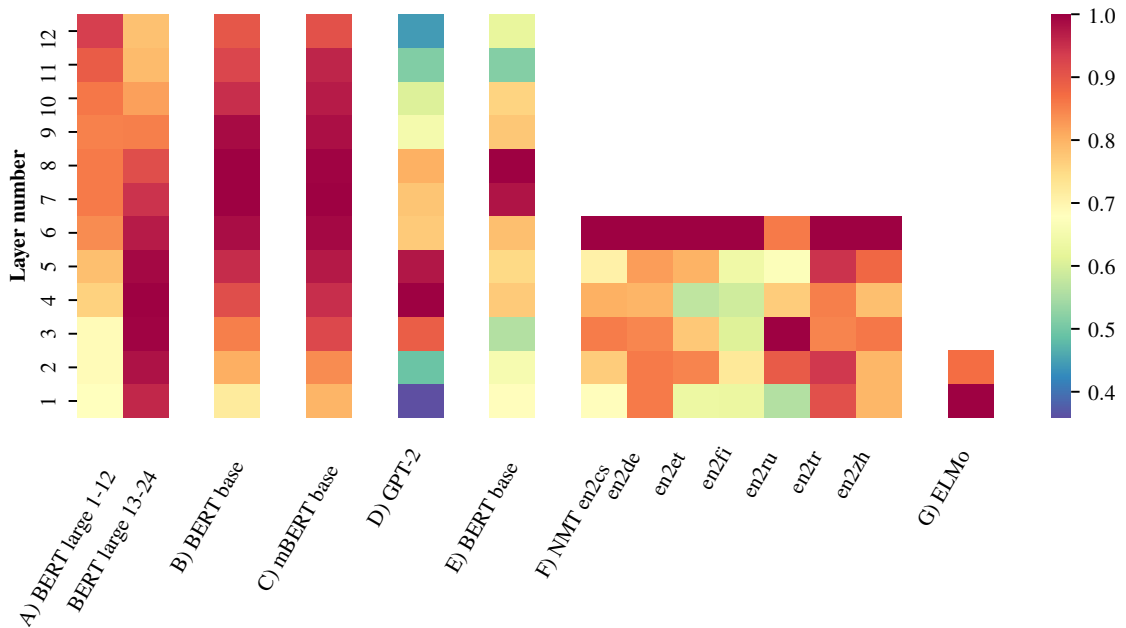


Figure 8: Relative syntactic information across attention models and layers. The values are normalized so that the best layer for each method has 1.0. The methods A), B), C), and G) show undirected UAS trees extracted by probing the n -th layer [12, 6]. The method D) shows the dependency alignment averaged across all heads in each layer [34]. The methods E) and F) show UAS of trees induced from attention heads by the maximum spanning tree algorithm [31, 15]. The results for the best layer (corresponding to value 1.0 in the plot) are: A) 82.5; B) 79.8; C) 80.1; D) 22.3; E) 24.3; F) en2cs: 23.9, en2de: 20.9, en2et: 22.1, en2fi: 24.0, en2ru: 22.4, en2tr: 17.5, en2zh: 21.6; G) 77.0

NMT Transformer encoder is only the first half of the whole translation architecture, and therefore the most syntactic layers are, in fact, in the middle of the process. In RNN language model (ELMo) the first layer is more syntactic than the second one.

We conjecture that the initial Transformer’s layers capture simple relations (e.g., attending to next or previous tokens) and the last layers mostly capture task-specific information. Therefore, they are less syntactic.

We also observe that in supervised probing [12, 6], better results are obtained from initial and top layers than in unsupervised structure induction [31, 15], i.e., the distribution across layers is smoother.

6 Conclusion

In this overview, we survey that syntactic structures are latently learned by the neural models for natural language processing tasks. We have compared multiple approaches of others and described the features that affect the ability to capture the syntax. The following aspects tend to improve the performance on syntactic tasks such as POS tagging:

1. Using contextual embeddings from RNNs or Transformer outperforms static word embeddings (Word2Vec, GloVe).

2. Pretraining on tasks with masked input (language modeling or machine translation) produces better syntactic representation than auto encoding.
3. The advantage of language modeling over machine translation is the fact that larger corpora are available for pretraining.

Our meta-analysis of latent states showed that the most syntactic representation could be found in the middle layers of the model. They tend to capture more complex relations than initial layers, and the representations are less dependent on the pretraining objectives than in the top layers.

We have shown to what extent systems trained for a non-syntactic task can learn grammatical structures. The question we leave for further research is whether providing explicit syntactic information to the model can improve its performance on other NLP tasks.

Acknowledgments

This work has been supported by the grant 18-02196S of the Czech Science Foundation. It has been using language resources and tools developed, stored and distributed by the LINDAT/CLARIAH-CZ project of the Ministry of Education, Youth and Sports of the Czech Republic (project LM2018101).

References

- [1] Felipe Almeida and Geraldo Xexéo. Word embeddings: A survey. *CoRR*, abs/1901.09069, 2019.
- [2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2015.
- [3] Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada, July 2017. Association for Computational Linguistics.
- [4] Yonatan Belinkov, Lluís Màrquez, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. Evaluating layers of representation in neural machine translation on part-of-speech and semantic tagging tasks. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing.
- [5] Terra Blevins, Omer Levy, and Luke Zettlemoyer. Deep RNNs encode soft hierarchical syntax. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 14–19, Melbourne, Australia, July 2018. Association for Computational Linguistics.
- [6] Ethan A. Chi, John Hewitt, and Christopher D. Manning. Finding universal grammatical relations in multilingual BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5577, Online, July 2020. Association for Computational Linguistics.
- [7] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does BERT look at? An analysis of BERT’s attention, 2019.
- [8] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.
- [9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.
- [10] Timothy Dozat and Christopher D. Manning. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings*, 2017.
- [11] Zellig Harris. Distributional structure. *Word*, 10(23):146–162, 1954.
- [12] John Hewitt and Christopher D. Manning. A structural probe for finding syntax in word representations. In *NAACL-HLT*, 2019.
- [13] Taeuk Kim, Jihun Choi, Daniel Edmiston, and Sang-goo Lee. Are Pre-trained Language Models Aware of Phrases? Simple but Strong Baselines for Grammar Induction. In *International Conference on Learning Representations*, January 2020.
- [14] Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. 5, 11 2004.
- [15] Tomasz Limisiewicz, Rudolf Rosa, and David Mareček. Universal dependencies according to BERT: both more specific and more general. *ArXiv*, abs/2004.14620, 2020.
- [16] Pierre Lison, Jörg Tiedemann, and Milen Kouylekov. OpenSubtitles2018: Statistical rescoring of sentence alignments in large, noisy parallel corpora. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA).
- [17] Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. Linguistic knowledge and transferability of contextual representations. In *NAACL-HLT*, 2019.
- [18] Qi Liu, Matt J. Kusner, and Phil Blunsom. A survey on contextual embeddings. *ArXiv*, abs/2003.07278, 2020.
- [19] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
- [20] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.
- [21] David Mareček and Rudolf Rosa. From balustrades to pierre vinken: Looking for syntax in transformer self-attentions. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 263–275, Florence, Italy, August 2019. Association for Computational Linguistics.

- [22] Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6297–6308, 2017.
- [23] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, July 2013.
- [24] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia, June 2013. Association for Computational Linguistics.
- [25] Tomáš Musil. Examining Structure of Word Embeddings with PCA. In *Text, Speech, and Dialogue*, pages 211–223. Springer International Publishing, 2019.
- [26] H. Ney. Dynamic programming parsing for context-free grammars in continuous speech recognition. *IEEE Transactions on Signal Processing*, 39(2):336–340, 1991.
- [27] Joakim Nivre, Željko Agić, Lars Ahrenberg, Lene Antonsen, Maria Jesus Aranzabe, Masayuki Asahara, Luma Ateyah, Mohammed Attia, Aitziber Atutxa, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, John Bauer, Kepa Bengoetxea, Riyaz Ahmad Bhat, Eckhard Bick, Cristina Bosco, Gosse Bouma, Sam Bowman, Aljoscha Burchardt, Marie Candito, Gauthier Caron, Gülşen Cebiroğlu Eryiğit, Giuseppe G. A. Celano, Savas Cetin, Fabricio Chalub, Jinho Choi, Yongseok Cho, Silvie Cinková, Çağrı Çöltekin, Miriam Connor, Marie-Catherine de Marneffe, Valeria de Paiva, Arantza Diaz de Ilarraza, Kaja Dobrovoljc, Timothy Dozat, Kira Drohanova, Marhaba Eli, Ali Elkahky, Tomaz Erjavec, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökırmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta González Saavedra, Matias Gironi, Normunds Grūzītis, Bruno Guillaume, Nizar Habash, Jan Hajič, Jan Hajič jr., Linh Hà Mỹ, Kim Harris, Dag Haug, Barbora Hladká, Jaroslava Hlaváčová, Petter Hohle, Radu Ion, Elena Irimia, Anders Johannsen, Fredrik Jørgensen, Hüner Kaşıkara, Hiroshi Kanayama, Jenna Kanerva, Tolga Kayadelen, Václava Kettnerová, Jesse Kirchner, Natalia Kotsyba, Simon Krek, Sookyoung Kwak, Veronika Laippala, Lorenzo Lambertino, Tatiana Lando, Phương Lê Hồng, Alessandro Lenci, Saran Lertpradit, Herman Leung, Cheuk Ying Li, Josie Li, Nikola Ljubešić, Olga Loginova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Ruli Manurung, Cătălina Măranduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Gustavo Mendonça, Anna Missilä, Verginica Mititelu, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Shunsuke Mori, Bohdan Moskalevskyi, Kadri Muischnek, Nina Mustafina, Kaili Müürisep, Pinkey Nainwani, Anna Nedoluzhko, Lương Nguyễn Thị, Huyèn Nguyễn Thị Minh, Vitaly Nikolaev, Rattima Nitisaroj, Hanna Nurmi, Stina Ojala, Petya Osenova, Lilja Øvrelid, Elena Pascual, Marco Passarotti, Cemel-Augusto Perez, Guy Perrier, Slav Petrov, Jussi Piitulainen, Emily Pitler, Barbara Plank, Martin Popel, Lauma Pretkalinča, Prokopis Prokopidis, Tiina Puolakainen, Sampo Pyysalo, Alexandre Rademaker, Livy Real, Siva Reddy, Georg Rehm, Larissa Rinaldi, Laura Rituma, Rudolf Rosa, Davide Rovati, Shadi Saleh, Manuela Sanguinetti, Baiba Saulīte, Yanin Sawanakunanon, Sebastian Schuster, Djamel Seddah, Wolfgang Seeker, Mojgan Seraji, Lena Shakurova, Mo Shen, Atsuko Shimada, Muh Shohibussirri, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Aaron Smith, Antonio Stella, Jana Strnadová, Alane Suhr, Umut Sulubacak, Zolt Szántó, Dima Taji, Takaaki Tanaka, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Zdeňka Urešová, Larraitz Uribe, Hans Uszkoreit, Gertjan van Noord, Viktor Varga, Veronika Vincze, Jonathan North Washington, Zhuoran Yu, Zdeněk Žabokrtský, Daniel Zeman, and Hanzhi Zhu. Universal dependencies 2.0 – CoNLL 2017 shared task development and test data, 2017. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.
- [28] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.
- [29] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

- [30] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.
- [31] Alessandro Raganato and Jörg Tiedemann. An analysis of encoder representations in transformer-based machine translation. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 287–297, Brussels, Belgium, November 2018. Association for Computational Linguistics.
- [32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.
- [33] Jesse Vig. A multiscale visualization of attention in the transformer model. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28 - August 2, 2019, Volume 3: System Demonstrations*, pages 37–42. Association for Computational Linguistics, 2019.
- [34] Jesse Vig and Yonatan Belinkov. Analyzing the Structure of Attention in a Transformer Language Model. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 63–76, Florence, Italy, August 2019. Association for Computational Linguistics.
- [35] Elena Voita, David Talbot, Fedor Moiseev, Rico Senrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy, July 2019. Association for Computational Linguistics.
- [36] Yuxuan Wang, Wanxiang Che, Jiang Guo, Yijia Liu, and Ting Liu. Cross-lingual bert transformation for zero-shot dependency parsing. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.
- [37] Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.
- [38] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*, 2019.
- [39] Kelly W. Zhang and Samuel R. Bowman. Language modeling teaches you more syntax than translation does: Lessons learned through auxiliary task analysis. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, November 2018.