Out of Order: How Important Is The Sequential Order of Words in a Sentence in Natural Language Understanding Tasks?

Thang M. Pham[1] thangpham@auburn.edu
Trung Bui[2] bui@adobe.com
Long Mai[2] malong@adobe.com
Anh Nguyen[1] anh.ng8@gmail.com

[1]Auburn University [2]Adobe Research

Abstract

Do state-of-the-art natural language understanding models care about word order? Not always! We found 75% to 90% of the correct predictions of BERT-based classifiers, trained on many GLUE tasks, remain constant after input words are randomly shuffled. Although BERT embeddings are famously contextual, the contribution of each individual word to classification is almost unchanged even after its surrounding words are shuffled. BERT-based models exploit superficial cues (e.g. the sentiment of keywords in sentiment analysis; or the word-wise similarity between sequence-pair inputs in natural language inference) to make correct decisions when tokens are randomly shuffled. Encouraging models to capture word order information improves the performance on most GLUE tasks and SQuAD 2.0. Our work suggests that many GLUE tasks are not challenging machines to understand the meaning of a sentence.

1 Introduction

Machine learning (ML) models recently achieved excellent performance on state-of-the-art benchmarks for evaluating natural language understanding (NLU). In July 2019, RoBERTa (Liu et al., 2019) was the first to surpass a human baseline on GLUE (Wang et al., 2019). Since then, 13 more methods have also outperformed humans on the GLUE leaderboard. Notably, at least 8 out of the 14 solutions are based on BERT (Devlin et al., 2019)—a transformer architecture that learns representations via a bidirectional encoder. Given their superhuman GLUE-scores, how do BERT-based models solve NLU tasks? How do their NLU capability differs from that of humans?

We shed light into these important questions by examining model sensitivity to the order of words. Word order is one of the key characteristics of a sequence and is tightly constrained by many linguistic factors including syntactic structures, subcategorization, and discourse (Elman, 1990). Thus, arranging a set of words in a correct order is considered a key problem in language modeling (Hasler et al., 2017; Zhang and Clark, 2015).

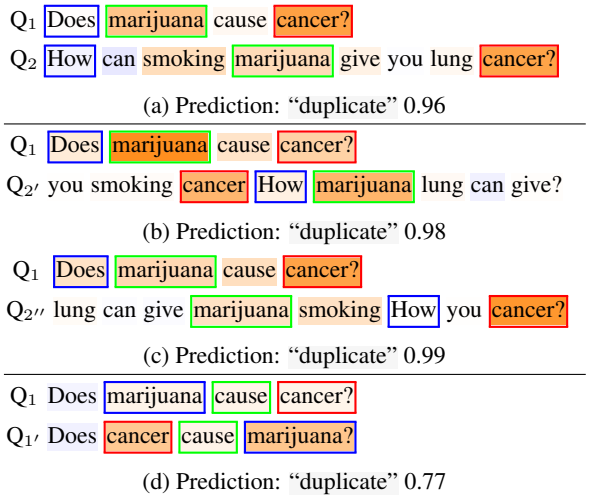


Figure 1: A RoBERTa-based model achieving a 91.12% accuracy on QQP, here, correctly labeled a pair of Quora questions "duplicate" (a). Interestingly, the predictions remain unchanged when all words in question Q_2 is randomly shuffled (b–c). QQP models also often incorrectly label a real sentence and its shuffled version to be "duplicate" (d). We found evidence that GLUE models rely heavily on words to make decisions e.g. here, "marijuana" and "cancer" (more important words are highlighted by LIME). Also, there exist self-attention matrices tasked explicitly with extracting word-correspondence between two input sentences regardless of the position of those words. Here, the top-3 pairs of words assigned the highest self-attention weights at (layer 0, head 7) are inside red, green, and blue rectangles, respectively.

Therefore, a natural question is: **Do BERT-based models trained on GLUE care about the order of words in a sentence?** Lin et al. (2019)

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 1145–1160
August 1–6, 2021.

found that pretrained BERT captures word-order information in the first three layers. However, it is unknown whether BERT-based classifiers actually use word order information when performing NLU tasks. Recently, Wang et al. (2020) showed that incorporating additional word-ordering and sentence-ordering objectives into BERT pretraining could lead to text representations (StructBERT) that enabled improved GLUE scores. However, StructBERT findings are inconclusive across different GLUE tasks and models. For example, in textual entailment (Wang et al., 2019, RTE), StructBERT improved the performance for $\text{BERT}_{\text{large}}$ but hurt the performance for RoBERTa (Table 2d).

Wang et al. 2020 motivated interesting questions: **Are state-of-the-art BERT-based models using word order information when solving NLU tasks? If not, what cues do they rely on?** To the best of our knowledge, our work is the first to study the above questions for an NLU benchmark (GLUE). We tested BERT-, RoBERTa-, and ALBERT-based (Lan et al., 2020) models on 7 GLUE tasks where the words of only one select sentence in the input text are shuffled at varying degrees. An ideal agent that truly understands language is expected to choose a "reject" option when asked to classify a sentence whose words are randomly shuffled. Alternatively, given shuffled input words, true NLU agents are expected to perform at random chance in multi-way classification that has no "reject" options (Fig. 1b). Our findings include:

1. 65% of the groundtruth labels of 5 GLUE tasks can be predicted when the words in one sentence in each example are shuffled (Sec. 3.1).

2. Although pretrained BERT embeddings are known to be contextual, in some GLUE tasks, the contribution of an individual word to classification is almost unchanged even after its surrounding words are shuffled (Sec. 3.3).

3. In sentiment analysis (SST-2), the polarity of a single salient word is $\geq 60\%$ predictive of an entire sentence's label (Sec. 3.4.1).

4. BERT-based models trained on sequence-pair GLUE tasks used a set of self-attention heads for finding similar tokens shared between the two inputs (Sec. 3.4).

5. Encouraging RoBERTa-based models to be more sensitive to word order improves the performance on SQuAD 2.0 and most GLUE tasks tested (i.e. except for SST-2) (Sec. 3.5).

Despite their superhuman scores, most GLUE-trained models behave similarly to Bag-of-Words (BOW) models, which are prone to naive mistakes (Fig. 1b–d). Our results also suggest that GLUE does not necessarily require syntactic information or complex reasoning.

2 Methods

2.1 Datasets

We chose GLUE because of three reasons: (1) GLUE is a common benchmark for NLU evaluation (Wang et al., 2019); (2) there exist NLU models (e.g. RoBERTa) that outperform humans on GLUE, making an important case for studying their behaviors; (3) it is unknown how sensitive GLUE-trained models are to word order and whether GLUE requires them to be sensitive (Wang et al., 2020).

Tasks Out of 9 GLUE tasks, we chose all 6 binary-classification tasks because they share the same random baseline of 50% accuracy and enable us to compare models' word-order sensitivity across tasks. Six tasks vary from acceptability (CoLA Warstadt et al. 2019), to natural language inference (QNLI Rajpurkar et al. 2016), RTE (Wang et al., 2019), paraphrase (MRPC Dolan and Brockett 2005, QQP Quora 2017), and sentiment analysis (SST-2 Socher et al. 2013).

We also performed our tests on STS-B (Cer et al., 2017)—a regression task of predicting the semantic similarity of two sentences.[1] While CoLA and SST-2 require single-sentence inputs, all other tasks require sequence-pair inputs.

Reject options For all binary-classification tasks (except SST-2), the negative label is considered the reject option (e.g. QQP models can choose "not duplicate" in Fig. 1b to reject shuffled inputs).

Metrics We use accuracy scores to evaluate the binary classifiers (for ease of interpretation) and Spearman correlation to evaluate STS-B regressors, following Wang et al. (2019).

2.2 Classifiers

We tested BERT-based models because (1) they outperformed humans on the GLUE leaderboard; and (2) the pretrained BERT was shown to capture word positional information (Lin et al., 2019).

[1] We did not choose WNLI (Levesque et al., 2012) as model performance is not substantially above random baseline.

Pretrained BERT encoders We tested three sets of classifiers finetuned from three different, pretrained BERT variants: BERT, RoBERTa, and ALBERT, downloaded from Huggingface (2020). The pretrained models are the "base" versions i.e. bidirectional transformers with 12 layers and 12 self-attention heads. The pretraining corpus varies from uncased (BERT, ALBERT) to case-sensitive English (RoBERTa).

Classifiers For each of the seven GLUE tasks, we added one classification layer on top of each of the three pretrained BERT encoders and finetuned the entire model. Unless otherwise noted, the mean performance per GLUE task was averaged over three classifiers. Each model's performance matches either those reported on Huggingface (2020) or the original papers (Table A6).

Hyperparameters Following Devlin et al. (2019), we finetuned classifiers for 3 epochs using Adam (Kingma and Ba, 2015) with a learning rate of 0.00002, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$. We used a batch size of 32, a max sequence length of 128, and dropout on all layers with a probability of 0.1.

2.3 Constructing sets of real and shuffled examples for experiments

Modifying one sentence As GLUE tasks vary in the number of inputs (one or two input sequences) and the sequence type per input (a sentence or a paragraph), we only re-ordered the words in one *sentence* from only one input while keeping the rest of the inputs unchanged. Constraining the modifications to a single sentence enables us to measure (1) the importance of word order in a single sentence; and (2) the interaction between the shuffled words and the unchanged, real context.

Random shuffling methods To understand model behaviors across varying degrees of word-order distortions, we experimented with three tests: randomly shuffling n-grams where n = $\{1, 2, 3\}$.

Shuffling 1-grams is a common technique for analyzing word-order sensitivity (Sankar et al., 2019; Zanzotto et al., 2020). We split a given sentence by whitespace into a list of n-grams, and re-combined them, in a random order, back into a "shuffled" sentence (see Table 1 for examples). The ending punctuation was kept intact. We re-sampled a new random permutation until the shuffled sentence was different from the original sentence.

As the label distributions, dev-set sizes, and the performance of models vary across GLUE tasks, to compare word-order sensitivity across tasks, we tested each model on two sets: (1) `dev-r` i.e. a subset of the original dev-set (Sec. 2.3.1); and (2) `dev-s` i.e. a clone of version of `dev-r` but that each example has one sentence with re-ordered words (Sec. 2.3.2).

How can smoking marijuana give you lung cancer?
Q_3 lung cancer | marijuana give you | How can smoking?
Q_2 smoking marijuana | lung cancer | give you | How can?
Q_1 marijuana | can | cancer | How | you | smoking | give | lung?
Q_s How can smoking | cancer | give you lung | marijuana?

Table 1: A real question on Quora (QQP dataset) and its three modified versions (Q_3 to Q_1) created by randomly shuffling 3-grams, 2-grams, and 1-grams, respectively. Q_s was created by swapping two random nouns.

2.3.1 Selecting real examples

For each pair of (task, classifier), we selected a subset of dev-set examples via the following steps:

1. For tasks with either a single-sequence or a sequence-pair input, we used examples where the input sequence to be modified has only one sentence[2] that has more than 3 tokens (for shuffling 3-grams to produce a sentence different from the original sentence).

2. We only selected the examples that were correctly classified by the classifier (to study what features were important for high accuracy).

3. We balanced the numbers of positive and negative examples by removing random examples from the larger-sized class.

That is, on average, we filtered out ∼34% of the original data. See Table A4 for the total number of examples remaining after each filtering step above.

2.3.2 Creating shuffled sets

For each task, we cloned the `dev-r` sets above and modified each example to create a "shuffled" set (a.k.a. `dev-s`) per shuffling method.

Specifically, a CoLA and SST-2 example contains only a single sentence and we modified that sentence. Each QQP, MRPC and STS-B example has two sentences and we modified the first sentence. An RTE example has a pair of (premise, hypothesis), and we modified the hypothesis since it

[2]We used NLTK sentence splitter (Bird et al., 2009) to detect text that has more than one sentence.

is a single sentence while premises are paragraphs. Each QNLI example contains a pair of (question, answer) and we modified the question, which is a sentence, while an answer is often a paragraph.

3 Experiments and Results

3.1 How much is word order information required for solving GLUE tasks?

GLUE has been a common benchmark for evaluating NLU progress. But, do GLUE tasks require models to use word order and syntactic information? We shed light into this question by testing model performance when word order is increasingly randomized.

If a task strictly requires words to form a semantically meaningful sentence, then randomly re-positioning words in correctly-classified sentences will cause model accuracy to drop from 100% to 50% (i.e. the random baseline b for binary-classification tasks with two balanced classes). Thus, to compare model-sensitivity across tasks, we use a Word-Order Sensitivity score (WOS):

$$s = (100 - p)/(100 - b) \quad (1)$$

where $p \in [50, 100]$ is the accuracy of a GLUE-trained model evaluated on a `dev-s` set (described in Sec. 2.3.2) and $s \in [0, 1]$. Here, $b = 50$.

Experiments For each GLUE task, we computed the mean accuracy and confidence score over three classifiers (BERT, RoBERTa, and ALBERT-based) on `dev-s` sets created by shuffling 1-grams, 2-grams, and 3-grams. The results reported in Table 2 were averaged over 10 random shuffling runs (i.e. 10 random seeds) per n-gram type, and then averaged over 3 models per task.

Results We found that for CoLA, i.e. detecting grammatically incorrect sentences, the model accuracy, on average, drops to near random chance i.e. between 50.69% and 56.36% (Table 2b) when n-grams are shuffled. That is, most of examples were classified into "unacceptable" after n-gram shuffling, yielding ∼50% accuracy (see Fig. A2 for qualitative examples).

Surprisingly, for the rest of the 5 out of 6 binary-classification tasks (i.e. except CoLA), between 75% and 90% of the originally correct predictions remain constant after 1-grams are randomly re-ordered (Table 2b; 1-gram). These numbers increase as the shuffled n-grams are longer (i.e. as n increases from 1→3), up to 95.32% (Table 2b; QNLI). Importantly, given an average dev-set accuracy of 86.35% for these 5 tasks, **at least** 86.35% × 75% ≈ **65% of the groundtruth labels of these 5 GLUE tasks can be predicted when all input words in one sentence are randomly shuffled.**

Additionally, on average over three n-gram types, models trained on these five GLUE tasks are from 2 to 10 times more *insensitive* to word-order randomization than CoLA models (Table 2c). That is, if not explicitly tasked with checking for grammatical errors, GLUE models mostly will not care about the order of words in a sentence (see qualitative examples in Figs. 1, A2–A4). Consistently, the confidence scores of BERT-based models for five non-CoLA tasks only dropped ∼2% when 1-grams are shuffled (Table 2).

Consistently across three different BERT "base" variants and a RoBERTa "large" model (Table A5), our results suggest that word order and syntax, in general, are not necessarily required to solve GLUE.

2-noun swaps Besides shuffled n-grams, we also repeated all experiments with more syntactically-correct modified inputs where only two random nouns in a sentence were swapped (Table 1; Q_s). This is a harder test for NLU models since the meaning of a sentence with two nouns swapped often changes while its syntax remains correct. We found the conclusions to generalize to this setting. That is, the models hardly changed predictions although the meanings of the original sentence and its swapped version are different (Table 2b; 2-noun swap vs. 1-gram).

3.2 How sensitive are models trained to predict the similarity of two sentences?

An interesting hypothesis is that models trained explicitly to evaluate the semantic similarity of two sentences should be able to tell apart real from shuffled examples. Intuitively, word order information is essential for understanding what an entire sentence means and, therefore, for predicting whether two sentences convey the same meaning.

We tested this hypothesis by analyzing the sensitivity of models trained on QQP and STS-B—two prominent GLUE tasks for predicting semantic similarity of a sentence pair. While QQP is a binary classification task, STS-B is a regression task where a pair of two sentences is given a score $\in [0, 5]$ denoting their semantic similarity.

Experiments We tested the models on `dev-r` and `dev-s` sets (see Sec. 2.3.2) where in each pair, the word order of the first sentence was randomized while the second sentence was kept intact.

QQP results Above 83% of QQP models' correct predictions on real pairs remained unchanged after word-order randomization (see Figs. 1a–c for examples).

STS-B results Similarly, STS-B model performance only drops marginally, i.e. less than 2 points from 89.67 to 87.80 in Spearman correlation (Table 2; STS-B). Since a STS-B model outputs a score $\in [0, 5]$, we binned the scores into 6 ranges. One might expect STS-B models to assign near-zero similarity scores to most modified pairs. However, the distributions of similarity scores for the modified and real pairs still closely match up (Fig. 2). In sum, **despite being trained explicitly on predicting semantic similarity of sentence pairs, QQP and STS-B are surprisingly insensitive to n-gram shuffling, exhibiting naive understanding of sentence meanings**.

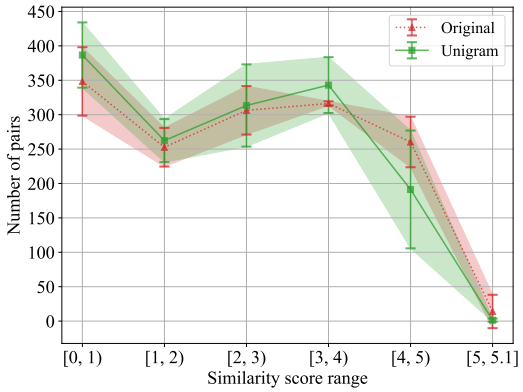


Figure 2: The distribution of similarity scores over 6 ranges for the (real, shuffled) pairs in `dev-s` (green) is highly similar to that for (real, real) STS-B pairs in `dev-r` (red). The statistics in each range were computed over 3 models (BERT, RoBERTa, and ALBERT).

3.3 How important are words to classification after their context is shuffled?

BERT representations for tokens are known to be highly contextual (Ethayarajh, 2019). However, after finetuning on GLUE, would the importance of a word to classification drop after its context is shuffled?

To answer the above question, we used LIME (Ribeiro et al., 2016) to compute word importance.

LIME computes a score $\in [-1, 1]$ for each token in the input denoting how much its presence contributes for or against the network's predicted label (Fig. 1; highlights). The importance score per word w is intuitively the mean confidence-score drop over a set of randomly-masked versions of the input when w is masked out.

Experiments We chose to study RoBERTa-based classifiers here because they have the highest GLUE scores among the three BERT variants considered. We observed that **62.5% (RTE) to 79.6% (QNLI) of the `dev-r` examples were consistently, correctly classified into the same labels in all 5 different random shuffles** (i.e. 5 different random seeds). We randomly sampled 100 such examples per binary-classification task and computed their LIME attribution maps to compare the similarity between the LIME heatmaps before and after unigrams are randomly misplaced.

Results On CoLA and RTE, the importance of words (i.e. mean absolute value of LIME-attribution per word), *decreased* substantially by 0.036 and 0.019, respectively. That is, the individual words become *less important* after their context is distorted—a behavior expected when CoLA and RTE have the highest WOS scores (Table 2). In contrast, for the other 4 tasks, word importance only changed marginally (by 0.008, i.e. 4.5× smaller than the 0.036 change in CoLA). That is, **except for CoLA and RTE models, the contribution of a word to classification is almost unchanged even after the context of each word is randomly shuffled** (Fig. 1a–c). This result suggests that the word embeddings after finetuning on GLUE became much less contextual than the pretrained BERT embeddings (Ethayarajh, 2019).

3.4 If not word order, then what do classifiers rely on to make correct predictions?

Given that all non-CoLA models are highly insensitive to word-order randomization, how did they arrive at correct decisions when words are shuffled?

We chose to answer this question for SST-2 and QNLI because they have the lowest WOS scores across all 6 GLUE tasks tested (Table 2) and they are representative of single-sentence and sequence-pair tasks, respectively.

Task	(a) Perf. on dev-r		(b) Performance on dev-s				(c) Word-Order Sensitivity			(d) StructBERT improvements		
	Models	Baseline	2-noun swap	1-gram	2-gram	3-gram	1-gram	2-gram	3-gram	$BERT_{base}$	$BERT_{large}$	RoBERTa
CoLA	100 (0.93)	50	71.75 (0.91)	50.69 (0.95)	53.98 (0.94)	56.36 (0.92)	**0.99**	**0.92**	**0.87**	**+4.9**	+4.8	+1.4
RTE	100 (0.81)	50	85.86 (0.81)	75.69 (0.80)	81.89 (0.80)	85.18 (0.79)	0.49	0.36	0.30	N/A	**+13.0**	–0.9
QQP	100 (0.98)	50	86.90 (0.96)	83.19 (0.96)	88.02 (0.96)	89.04 (0.96)	0.34	0.24	0.22	+0.7	+1.2	+0.5
MRPC	100 (0.91)	50	96.51 (0.91)	83.89 (0.89)	87.1 (0.90)	89.38 (0.90)	0.32	0.26	0.21	N/A	+3.9	**+1.7**
SST-2	100 (0.99)	50	97.78 (0.98)	84.04 (0.96)	88.35 (0.97)	90.56 (0.97)	0.32	0.23	0.19	+0.2	+0.3	+0.4
QNLI	100 (0.98)	50	94.31 (0.97)	89.42 (0.96)	93.85 (0.97)	95.32 (0.98)	0.21	0.12	0.09	N/A	+3.0	+0.3
STS-B	89.67	N/A	88.93	87.80	88.66	88.95	N/A	N/A	N/A	N/A	N/A	N/A

Table 2: All results (a–c) are reported on the GLUE dev-r sets i.e. 100% accuracy (a). Shuffling n-grams caused the accuracy to drop (b) the largest for CoLA and the least for QNLI. Each row is computed by averaging the results of 3 BERT-based models and 10 random shuffles. From top to bottom, the Word-Order Sensitivity (WOS) is sorted descendingly (c) and is consistent across three types of n-grams i.e. WOS scores decrease from top down and from left to right. In contrast, the StructBERT results (d), taken from Table 1 and 4 in Wang et al. 2020, showed inconsistent improvements across different tasks. STS-B results are in scaled Spearman correlation. In addition to small accuracy drops, the mean confidence scores of all classifiers—reported in parentheses e.g. "(0.93)"—also changed marginally after words are shuffled (a vs. b).

3.4.1 SST-2: Salient words are highly predictive of sentence labels

As 84.04% of the SST-2 correct predictions did not change after word-shuffling (Table 2b), a common hypothesis is that the models might rely heavily on a few key words to classify an entire sentence.

S	the film 's performances are thrilling .	1.00
S_1	the film thrilling performances are 's .	1.00
S_2	's thrilling film are performances the .	1.00
S_3	's thrilling are the performances film .	1.00

Figure 3: An original SST-2 dev-set example (S) and its three shuffled versions (S_1 to S_3) were all correctly labeled "positive" by a RoBERTa-based classifier with high confidence scores (right column).

Experiments To test this hypothesis, we took all SST-2 dev-r examples whose all 5 randomly shuffled versions were all correctly labeled by a RoBERTa-based classifier (i.e. this "5/5" subset is ∼65% of the dev-set). We used LIME to produce a heatmap of the importance of words in each example.

We identified the polarity of each top-1 most important word (i.e. the highest LIME-attribution score) per example by looking it up in the Opinion Lexicon (Hu and Liu, 2004) of 2,006 positive and 4,783 negative words. ∼57% of these top-1 words were found in the dictionary and labeled either "positive" or "negative" (see Table A3).

Results We found that if the top-1 word has a positive meaning, then there is a 100% probability that the sentence's label is "positive". For example, the word "thrilling" in a movie review indicates a "positive" sentence (see Fig. 3). Similarly, the conditional probability of a sentence being labeled "negative" given a negative top-1 word is 94.4%. That is, given this statistics, the SST-2 label distribution and model accuracy, **at least 60% of the SST-2 dev-set examples can be correctly predicted from only a single top-1 salient word.**

We also reached similar conclusions when experimenting with ALBERT classifiers and the SentiWords dictionary (Gatti et al., 2015) (see Table A3).

3.4.2 Self-attention layers matching similar words in both the question and the answer

For sequence-pair tasks, e.g. QNLI, how can models correctly predict "entailment" when the question words are randomly shuffled (Fig. 4; Q_1) or when the question syntax is correct but its meaning

	QNLI sentence-pair inputs and their LIME attributions (negative -1, neutral 0, positive +1)	Confidence score
Q	How long did Phillips manage the Apollo missions?	1.00
A	Mueller agreed, and Phillips managed Apollo from January 1964, until it achieved the first manned landing in July 1969, after which he returned to Air Force duty.	
Q_1	Apollo the Phillips How missions long did manage?	0.96
A	Mueller agreed, and Phillips managed Apollo from January 1964, until it achieved the first manned landing in July 1969, after which he returned to Air Force duty.	
Q_2	Phillips long manage How missions the Apollo did?	0.97
A	Mueller agreed, and Phillips managed Apollo from January 1964, until it achieved the first manned landing in July 1969, after which he returned to Air Force duty.	
Q_s	How long did Apollo manage the Phillips missions?	0.99
A	Mueller agreed, and Phillips managed Apollo from January 1964, until it achieved the first manned landing in July 1969, after which he returned to Air Force duty.	

Figure 4: A RoBERTa-based model's correct prediction of "entailment" on the original input pair (Q, A) remains unchanged when the question is randomly shuffled (Q_1 & Q_2) or when two random nouns in the question are swapped (Q_s). The salient words in the questions e.g. manage and missions remain similarly important after their context has been shuffled. Also, the classifier harnessed self-attention to detect the correspondence between similar words that appear in both the question and the answer e.g. manage (Q) and managed (A). That is, the top-3 pairs of words that were assigned the largest question-to-answer weights in a self-attention matrix (layer 0, head 7) are inside in the red, green, and blue rectangles.

changes entirely (Fig. 4; Q_s). We hypothesize that inside the model, there might be a self-attention (SA) layer that extracts pairs of similar words that appear in both the question and the answer (e.g. "manage" vs. "managed" in Fig. 4).

Experiments To test this hypothesis, we analyzed the 5,000 QNLI `dev-r` examples (Table A4) of RoBERTa-based classifiers trained on QNLI. For each example, we identified one SA matrix (among all 144 as the base model has 12 layers & 12 heads per layer) that assigns the highest weights to pairs of similar words between the question and the answer, i.e. excluding intra-question and intra-answer attention weights (see the procedure in Sec. A).

Results First, in ∼58% of the examples, we found at least three pairs of words that *match* (i.e. the sum Levenshtein character-level edit-distance for all 3 pairs is ≤ 4). Second, we found, in total, 15 SA heads (out of the 144) which are explicitly tasked with capturing such question-to-answer word correspondence, *regardless* of word order (see Fig. 4).

Remarkably, **87% of the work of matching similar words that appear in both the QNLI question and the answer was handled by only 3 self-attention heads at (layer, head) of (0,7), (1,9), and (2,6)**.

We found consistent results when repeating the same analysis for other three sequence-pair tasks. That is, interestingly, **the three SA heads at exactly the same location of (0, 7), (1, 9), and (2, 6) account for 76%, 89%, and 83% of the "word-matching" task on QQP, RTE, and MRPC, respectively**. This coincidence is likely due to the fact that these classifiers were finetuned for different downstream tasks starting from the same pre-trained RoBERTa encoder. See Figs. 1, 4, A3–A4 for qualitative examples of these three tasks.

How important are the 15 word-matching attention heads to QNLI model performance? We found that zero-ing out 15 random heads had almost no effect to correctly-classified predictions–i.e. accuracy dropped marginally (-1% to -3%, Table 3) across different groups of examples. However, ablating the 15 word-matching heads caused the performance to drop substantially i.e. (a) by 9.6% on the 1,453 "positive" examples identified in Sec. A; (b) by 22.1% on a set of 2,906 random, examples including both "positive" and "negative" examples (at 50/50 ratio); and (c) by 24.5% on the entire QNLI 5,000-example `dev-r` set. That is, **the 15 SA heads that learned to detect similar words played an important role in solving QNLI, i.e. enabling at least ∼50% of the correct predictions** (Table 3d; accuracy dropped from 100% to 75.54% when the random chance is 50%). In sum, we found overlap between words in the question and answer of QNLI examples and strong evidence that QNLI models harnessed self-attention to exploit such overlap to make correct decisions in spite of a random word-order.

QNLI `dev-r` examples	Full network	Zero-out 15 attention matrices	
		Random	Ours
a. 1,453 selected 0/5 (+) examples	100	99.31	90.43
b. 1,453 random 0/5 (+) examples	100	99.24	91.05
c. 1,453 random 0/5 (+) examples & 1,453 random 0/5 (-) examples	100	98.18	77.91
d. (+/-) All 5,000 examples	100	96.96	75.54

Table 3: Zero-ing out a set of 15 "word-matching" self-attention matrices (identified via the procedure in Sec. 3.4.2) caused a substantial drop of ∼25% in accuracy (d) while the random baseline is 50%. These 15 matrices played an important role in QNLI because ablating 15 random matrices only caused a ∼1-3% drop in accuracy.

3.5 Does increasing word-order sensitivity lead to higher model performance?

Here, we test whether encouraging BERT representations to be more sensitive to word order (i.e. more syntax-aware) would improve model performance on GLUE & SQuAD 2.0 (Rajpurkar et al., 2018). We performed this test on the five GLUE binary-classification tasks (i.e. excluding CoLA because its WOS score is already at 0.99; Table 2).

Experiments Inspired by the fact that CoLA models are highly sensitive to word order, we finetuned the pretrained RoBERTa on a *synthetic*, CoLA-like task first, before finetuning the model on downstream tasks.

The synthetic task is to classify a single sentence into "real" vs. "fake" where the latter is formed by taking each real sentence and swapping two random words in it. For every downstream task (e.g. SST-2), we directly used its original training and dev sets to construct a balanced, 2-class, synthetic dataset. After finetuning the pretrained RoBERTa on this synthetic binary classification task, we reinitialized the classification layer (keeping the rest unchanged) and continued finetuning it on a downstream task.

For both finetuning steps, we trained 5 models per task and followed the standard BERT finetuning procedure (described in Sec. 2.2).

Results After the first finetuning on synthetic tasks, all models obtained a ∼99% training-set accuracy and a ∼95% dev-set accuracy. **After the second finetuning on downstream tasks, we observed that all models were substantially more sensitive to word order**, compared to the baseline models (which were only finetuned on the downstream tasks). That is, we repeated the 1-gram shuffling test (Sec. 3.1) and found a ∼1.5 to 2× increase in the WOS scores of all models (see Table 4a vs. b).

GLUE `dev-s`	(a) RoBERTa		(b) Ours	
	Accuracy	WOS	Accuracy	WOS
`RTE`	80.76	0.38	64.01	**0.72** (+189%)
`MRPC`	83.86	0.32	72.88	**0.54** (+169%)
`SST-2`	84.26	0.31	76.97	**0.46** (+148%)
`QQP`	87.66	0.25	77.11	**0.46** (+184%)
`QNLI`	91.09	0.18	82.44	**0.35** (+194%)

Table 4: With finetuning on synthetic tasks, all of our models (b) have a larger drop in accuracy on shuffled `dev-s` examples, compared to the standard RoBERTa-based classifiers (a). That is, our models are substantially more sensitive to word-order randomization (i.e. +148% to +194% in WOS scores).

GLUE On GLUE dev sets, on average over 5 runs, our models outperformed the RoBERTa baseline on all tasks except for SST-2 (Table 5). The highest improvement is in RTE (from 72.2% to 73.21% on average, and to 74.73% for the best single model), which is consistent with the fact that RTE has the highest WOS score among non-CoLA tasks (Sec. 3.1).

SQuAD 2.0 Our models also outperformed the RoBERTa baseline on the SQuAD 2.0 dev set, with the highest F1 gain from 80.62% to 81.08% (Table 5).

In sum, leveraging the insights that the original BERT-based models are largely word-order invariant, we showed that increasing model sensitivity via a simple extra finetuning step directly improves GLUE and SQuAD 2.0 performance.

4 Related Work

Pretrained BERT Lin et al. (2019) found that positional information is encoded in the first there layers of $\text{BERT}_{\text{base}}$ and fades out starting layer 4. Ettinger (2020) found that BERT heavily relies on word order when predicting missing words in masked sentences from the CPRAG-102 dataset. That is, shuffling words in the context sentence caused the word-prediction accuracy to drop by ∼1.3 to 2×. While all above work studied the

	RTE (Acc)	QQP (Acc)	MRPC (Acc)	SST-2 (Acc)	QNLI (Acc)	SQuAD (F1)
RoBERTa	72.20	91.12	87.25	**94.50**	92.57	80.62
Our best model	**74.73** **+2.53**	**91.31** **+0.19**	**88.73** **+1.48**	94.50 +0	**93.08** **+0.51**	**81.08** **+0.46**
Average (5 runs)	**73.21** **+1.01**	**91.19** **+0.07**	**87.31** **+0.06**	94.22 -0.28	**92.71** **+0.14**	**80.75** **+0.13**

Table 5: Finetuning the pretrained RoBERTa on synthetic tasks (before finetuning on the downstream tasks) improved model dev-set performance on SQuAD 2.0 (b) and all the tested tasks in GLUE (a), except SST-2.

pretrained BERT, we instead study BERT-based models *finetuned* on downstream tasks.

Word-ordering as an objective In text generation, Elman (1990) found that recurrent neural networks were sensitive to regularities in word order in simple sentences. Language models (Mikolov et al., 2010) with long short-term memory (LSTM) units (Hochreiter and Schmidhuber, 1997) were able to recover the original word order of a sentence from randomly-shuffled words even without any explicit syntactic information (Schmaltz et al., 2016). Wang et al. 2020 also observed an increase in GLUE performance after pretraining BERT with two additional objectives of word-ordering and sentence-ordering. Their work differs from ours in three points: (1) they did not study the importance of word order alone; (2) StructBERT improvements were inconsistent across tasks and models (Table 2d) and motivated us to compare the word-order importance between GLUE tasks; and (3) we proposed to improve model performance by finetuning not pretraining.

Word-order insensitivity in other NLP tasks ML models have been shown to be insensitive to word order in several NLP tasks such as reading comprehension (Si et al., 2019; Sugawara et al., 2020), dialog (Sankar et al., 2019), natural language inference (Parikh et al., 2016; Sinha et al., 2020), and essay scoring (Parekh et al., 2020). Zanzotto et al. 2020 found that for several text classification tasks, syntactic information was not always required. In word prediction, LSTMs and pre-trained BERT were found to exhibit a certain degree of insensitivity when the context words are randomly shuffled (Khandelwal et al., 2018; Mitchell and Bowers, 2020; Ettinger, 2020). Compared to the prior work, we are the first to perform a word-order analysis on a NLU benchmark and to contrast this sensitivity across the tasks.

Humans can also be word-order invariant A recent human study interestingly showed that sentences with scrambled word orders elicit a response as high as that elicited by original sentences as long as the local mutual information among words is high enough (Mollica et al., 2020). Gibson et al. 2013 found that humans can also exhibit word-order-invariance effects, especially when one interpretation is much more semantically plausible. Our work therefore documents an important similarity between humans and advanced NLU models.

Invariance to patch-order in computer vision In computer vision, the accuracy of state-of-the-art image classifiers was found to only drop marginally when the patches in an image were randomly shuffled (Chen et al., 2020; Zhang and Zhu, 2019).

5 Discussion and Conclusion

Consistently across three BERT variants and two model sizes, we found that GLUE-trained BERT-based models are often word-order invariant unless explicitly asked for (e.g. in CoLA).

We present a reflection on the progress of NLU by studying GLUE—a benchmark where humans have been surpassed by many models in the past 18 months. As suggested by our work, these models; however, may neither use syntactic information nor complex reasoning. We revealed how self-attention, a key building block in modern NLP, is being used to extract superficial cues to solve sequence-pair GLUE tasks even when words are out of order.

Adversarial NLI We also replicated our shuffling experiments on ANLI (Nie et al., 2020), a task considered challenging to existing models, and where RoBERTa-based models only obtained a 56% accuracy. We found RoBERTa-based models to remain not always sensitive to word-order randomization on ANLI (Table A2; WOS of 0.63), suggesting a common issue in existing benchmarks.

Acknowledgments

We thank Michael Alcorn, Qi Li, and Peijie Chen for helpful feedback on the early results. We are grateful for valuable feedback from Sam Bowman, Ernest Davis, and Melanie Mitchell. AN is supported by the National Science Foundation under Grant No. 1850117, and donations from the NaphCare Charitable Foundation, and Nvidia.

References

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly, Beijing.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Peijie Chen, Chirag Agarwal, and Anh Nguyen. 2020. The shape and simplicity biases of adversarially robust imagenet-trained cnns. *arXiv e-prints*, pages arXiv–2006.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Jeffrey L Elman. 1990. Finding structure in time. *Cognitive science*, 14(2):179–211.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. 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 55–65, Hong Kong, China. Association for Computational Linguistics.

Allyson Ettinger. 2020. What bert is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Lorenzo Gatti, Marco Guerini, and Marco Turchi. 2015. Sentiwords: Deriving a high precision and high coverage lexicon for sentiment analysis. *IEEE Transactions on Affective Computing*, 7(4):409–421.

Edward Gibson, Leon Bergen, and Steven T Piantadosi. 2013. Rational integration of noisy evidence and prior semantic expectations in sentence interpretation. *Proceedings of the National Academy of Sciences*, 110(20):8051–8056.

GLUE leaderboard. Glue benchmark. https://gluebenchmark.com/leaderboard. (Accessed on 01/29/2021).

Eva Hasler, Felix Stahlberg, Marcus Tomalin, Adrià de Gispert, and Bill Byrne. 2017. A comparison of neural models for word ordering. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 208–212, Santiago de Compostela, Spain. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, page 168–177, New York, NY, USA. Association for Computing Machinery.

Huggingface. 2020. Pretrained models — transformers 3.3.0 documentation. huggingface.co/transformers/pretrained_models.html. (Accessed on 09/30/2020).

Urvashi Khandelwal, He He, Peng Qi, and Dan Jurafsky. 2018. Sharp nearby, fuzzy far away: How neural language models use context. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 284–294, Melbourne, Australia. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*. Citeseer.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019. Open sesame: Getting inside bert's linguistic knowledge. *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*.

Jeff Mitchell and Jeffrey Bowers. 2020. Priorless recurrent networks learn curiously. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5147–5158.

Francis Mollica, Matthew Siegelman, Evgeniia Diachek, Steven T Piantadosi, Zachary Mineroff, Richard Futrell, Hope Kean, Peng Qian, and Evelina Fedorenko. 2020. Composition is the core driver of the language-selective network. *Neurobiology of Language*, 1(1):104–134.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

NLTK. nltk.metrics.distance — nltk 3.5 documentation. https://www.nltk.org/_modules/nltk/metrics/distance.html. (Accessed on 12/25/2020).

Swapnil Parekh, Yaman Kumar Singla, Changyou Chen, Junyi Jessy Li, and Rajiv Ratn Shah. 2020. My teacher thinks the world is flat! interpreting automatic essay scoring mechanism. *arXiv preprint arXiv:2012.13872*.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas. Association for Computational Linguistics.

Quora. 2017. (1) first quora dataset release: Question pairs - data @ quora - quora. https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs. (Accessed on 09/30/2020).

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Chinnadhurai Sankar, Sandeep Subramanian, Chris Pal, Sarath Chandar, and Yoshua Bengio. 2019. Do neural dialog systems use the conversation history effectively? an empirical study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 32–37, Florence, Italy. Association for Computational Linguistics.

Allen Schmaltz, Alexander M. Rush, and Stuart" Shieber. 2016. Word ordering without syntax. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2319–2324, Austin, Texas. Association for Computational Linguistics.

Chenglei Si, Shuohang Wang, Min-Yen Kan, and Jing Jiang. 2019. What does bert learn from multiple-choice reading comprehension datasets? *arXiv preprint arXiv:1910.12391*.

Koustuv Sinha, Prasanna Parthasarathi, Joelle Pineau, and Adina Williams. 2020. Unnatural language inference. *arXiv preprint arXiv:2101.00010*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Saku Sugawara, Pontus Stenetorp, Kentaro Inui, and Akiko Aizawa. 2020. Assessing the benchmarking capacity of machine reading comprehension datasets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8918–8927.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Wei Wang, Bin Bi, Ming Yan, Chen Wu, Jiangnan Xia, Zuyi Bao, Liwei Peng, and Luo Si. 2020. Structbert: Incorporating language structures into pre-training for deep language understanding. In *International Conference on Learning Representations*.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Fabio Massimo Zanzotto, Andrea Santilli, Leonardo Ranaldi, Dario Onorati, Pierfrancesco Tommasino, and Francesca Fallucchi. 2020. KERMIT: Complementing transformer architectures with encoders of explicit syntactic interpretations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 256–267, Online. Association for Computational Linguistics.

Tianyuan Zhang and Zhanxing Zhu. 2019. Interpreting adversarially trained convolutional neural networks. In *International Conference in Machine Learning*.

Yue Zhang and Stephen Clark. 2015. Discriminative syntax-based word ordering for text generation. *Computational linguistics*, 41(3):503–538.

A Self-attention layers that match question-words to similar words in the answer

QNLI models being so insensitive to word shuffling (i.e. 89.4% of the correct predictions remain correct) suggests that inside the finetuned BERT, there might be a self-attention (SA) layer that extract pairs of similar words that appear in both the question and answer.

We started by analyzing all 2,500 "positive" `dev-r` examples (Table A4) of RoBERTa-based classifiers trained on QNLI because there were fewer and more consistent ways for labeling a sentence "positive" than for the "negative" (shown in Sec. 3.3).

Experiment There were 1,776 (out of 2,500) examples whose predictions did not change in 5 random shufflings (a.k.a 5/5 subset). For each such example, we followed the following 4 steps to identify one SA matrix (among all 144 as the base model has 12 layers & 12 heads per layer) that captures the strongest attention connecting the question and answer words.

1. Per example x, we created its shuffled version $\hat{x}$ by randomly shuffling words in the question and fed $\hat{x}$ into the classifier.

2. For each SA matrix obtained, we identified the top-3 highest-attention weights that connect the shuffled question tokens and the real answer tokens (i.e. excluding attention weights between question tokens or answer tokens only).

3. For each shuffled example $\hat{x}$, we identified one matrix M whose the top-3 word pairs are the nearest in Levenshtein character-level edit-distance (NLTK). For instance, the distance is 1 between manage and managed (Fig. 4).

4. For each matrix M identified for $\hat{x}$, we fed the corresponding real example x through the network and re-computed the edit-distance for each of the top-3 word pairs.

Results At step 3, there were 1,590 SA matrices (out of 1,776) whose the top-3 SA weights connected three pairs of *matching* words (i.e. the total edit-distance for 3 pairs together is ≤ 4)[3] that appear in both the shuffled question and original answer (see example top-3 pairs in Fig. 4). At step 4, this number only dropped slightly to 1,453 matrices when replacing the shuffled question by the original one (see Table A1 for detailed statistics).

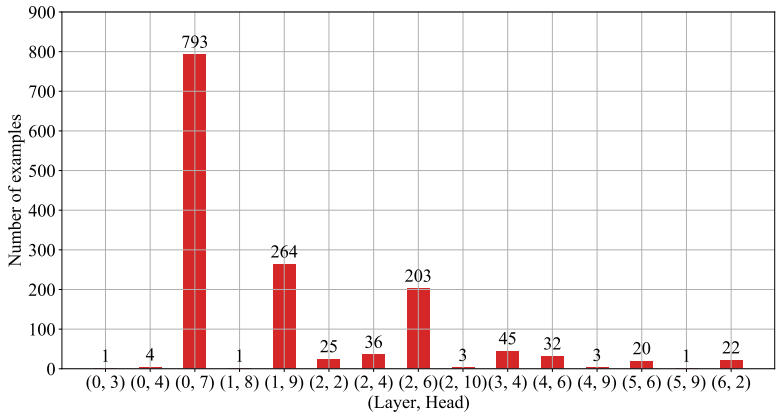
Sum distance	(a) `dev-s` alone		(b) `dev-s` & `dev-r`	
	# examples	%	# examples	%
≤ 0	749	42.17	392	24.65
≤ 1	1,253	70.55	1,071	67.36
≤ 2	1,440	81.08	1,283	80.69
≤ 3	1,543	86.88	1,391	87.48
≤ 4	**1,590**	**89.53**	**1,453**	**91.38**
≤ 15	1,776	100.00	1,574	98.99
Total	1,776	100.00	1,590	100.00

Table A1: The number of QNLI examples where we found $\geq$ one self-attention matrix that the most strongly attends to three pairs of *matching* words when given the `dev-s` examples i.e. (modified question, real answer) (a) or when given both the shuffled and real examples (b). In other words, **the numbers in (b) denote the number of examples where (1) there exist $\geq$ 3 words, regardless of its word order, in the question that can be found in the accompanying real answer; and (2) these correspondences are captured by at least one self-attention matrix.** The sum edit-distance for all 3 pairs of words are less than N where $N = \{0, 1, 2, 3, 4, 15\}$ (left column).

However, there are only 15 unique, RoBERTa self-attention matrices in these 1,453 examples (see Fig. A1). Also at step 4, 83% of the same word pairs remained within the top-3 of the same SA matrices, after question replacement, i.e. 17% of attention changed to different pairs e.g. from ("managed", "manage") to ("it", "it").

First, our results showed that there is a set of 15 self-attention heads explicitly tasked with capturing question-to-answer word correspondence *regardless of word order*. Second, **for ∼58% (i.e. 1,453 / 2,500) of QNLI "positive" examples: (1) there exist $\geq$ 3 words in the question that can be found in the accompanying answer; and (2) these correspondences are captured by at least one of the 15 SA matrices.** We also found similar results for 2,500 "negative" `dev-r` examples (data not shown).

[3] 4 is a tight budget to account for minor typos or punctuation differences e.g. "Amazon" vs. "Amazon's".



(a) Histogram of self-attention matrices

(Layer, Head)	# Examples	%
(0, 3), (0, 4), (0, 7)	798	54.9%
(1, 8), (1, 9)	265	18.2%
(2, 2), (2, 4), (2, 6), (2, 10)	267	18.4%
(3, 4)	45	3.1%
(4, 6), (4, 9)	35	2.4%
(5, 6), (5, 9)	21	1.5%
(6, 2)	22	1.5%
Total	1,453	100%

(b) Layer-wise comparison

Figure A1: Among 144 self-attention matrices in the RoBERTa-based classifier finetuned for QNLI, there are 15 "word-matching" matrices (a) that explicitly attend to pairs of similar words that appear in both questions and answers regardless of the order of words in the question (see example pairs in Fig. 4). For each QNLI example, we identified one such matrix that exhibits the matching behavior the strongest (a). 92% of the task of attending to duplicate words is mostly handled in the first three layers (b).

Model	Task	(a) Perf. on dev-r		(b) Perf. on dev-s				(c) Word-Order Sensitivity			
		Models	Baseline	2-noun swap	1-gram	2-gram	3-gram	2-noun swap	1-gram	2-gram	3-gram
RoBERTa$_{base}$	ANLI	100	33.33	74.26	57.74	66.63	69.04	0.39	0.63	0.50	0.46
	A1	100	33.33	81.46	63.31	71.52	75.37	0.28	0.55	0.43	0.37
	A2	100	33.33	70.83	54.61	64.73	67.02	0.44	0.68	0.53	0.49
	A3	100	33.33	70.50	55.29	63.63	64.73	0.55	0.67	0.55	0.53
RoBERTa$_{large}$	ANLI	100	33.33	70.41	54.87	64.11	68.76	0.44	0.68	0.54	0.47
	A1	100	33.33	78.06	60.31	70.57	75.86	0.33	0.6	0.44	0.36
	A2	100	33.33	67.88	51.44	60.64	66.31	0.48	0.73	0.59	0.51
	A3	100	33.33	65.30	52.85	61.11	64.10	0.52	0.71	0.58	0.54

Table A2: All results (a–c) of RoBERTa$_{base}$ and RoBERTa$_{large}$ models finetuned on the combination of NLI datasets (SNLI, MNLI, FEVER and ANLI) are reported on the ANLI dev-r sets (i.e. 100% accuracy) which includes A1, A2 and A3 (a). The accuracies for RoBERTa$_{base}$ and RoBERTa$_{large}$ on ANLI are 51.19% and 56.98%, respectively. Each row is computed by averaging the results of 10 random shuffles. Word-Order Sensitivity (WOS) of ANLI and its subsets (c). Since ANLI is 3-way classification task, the baseline is 33.33% (as described in Sec 2.3.1).

Dictionary	Opinion Lexicon (Hu and Liu, 2004)		SentiWords (Gatti et al., 2015)	
	(a) RoBERTa	(b) ALBERT	(c) RoBERTa	(d) ALBERT
Total examples in subset 5/5 ("positive" / "negative")	523 (278 / 245)	506 (228 / 278)	523 (278 / 245)	506 (228 / 278)
Not found in dictionary	223 / 523 (42.64%)	217 / 506 (42.89%)	110 / 523 (21.03%)	104 / 506 (20.55%)
Found in dictionary	300 / 523 (57.36%)	289 / 506 (57.11%)	413 / 523 (78.97%)	402 / 506 (79.45%)
P ("positive" sentence | positive top-1 word)	174 / 174 (100.00%)	143 / 144 (99.31%)	222 / 258 (86.05%)	186 / 215 (86.51%)
P ("negative" sentence | negative top-1 word)	119 / 126 (94.44%)	136 / 145 (93.79%)	145 / 155 (93.55%)	177 / 187 (94.65%)

Table A3: **If the top-1 most important word in an SST-2 5/5 example has a positive meaning, then there is a 100% chance that the sentence is labeled "positive" in SST-2.** Similarly, the conditional probability of a sentence being labeled "negative" given a negative most important word (by LIME Ribeiro et al. 2016) is 94.44%.

LIME attributions (negative -1, neutral 0, positive +1)		
CoLA example. Groundtruth: "acceptable"		
S	Medea denied poisoning the phoenix.	"acceptable" 0.99
S_1	poisoning the phoenix denied Medea.	"acceptable" 0.53
S_2	phoenix Medea denied the poisoning.	"acceptable" 0.99
S_3	Medea the poisoning phoenix denied.	"unacceptable" 0.95
S_4	phoenix Medea denied the poisoning.	"unacceptable" 0.99
S_5	Medea phoenix poisoning the denied.	"unacceptable" 0.96

Figure A2: Each CoLA example contains a single sentence. Here, we shuffled the words in the original sentence (S) five times to create five new sentences (S_1 to S_5) and fed them to a RoBERTa-based classifier for predictions. Words that are important for or against the prediction (by LIME Ribeiro et al. 2016) are in orange and blue, respectively. Most of the shuffled examples were classified into "unacceptable" label (i.e. grammatically incorrect) with even higher confidence score than the original ones.

MRPC example. Groundtruth: "equivalent"		
A	My decision today is not based on any one event . "	"equivalent" 0.99
B	Governor Rowland said his decision was " not based on any one event . "	
A_1	event any is one decision based on My today not . "	"equivalent" 0.98
B	Governor Rowland said his decision was " not based on any one event . "	
A_2	one based today not any My on event is decision . "	"equivalent" 0.98
B	Governor Rowland said his decision was " not based on any one event . "	

Figure A3: Each MRPC example contains a pair of sentences i.e. (A, B). Here, we shuffled the words in the original sentence (A) to create modified sentences (A_1 & A_2) and fed them together with the original second sentence (B) to a RoBERTa-based classifier for predictions. Also, the classifier harnessed self-attention to detect the correspondence between similar words that appear in both sentences. That is, the top-3 pairs of words that were assigned the largest cross-sentence weights in a self-attention matrix (layer 0, head 7) are inside in the red, green, and blue rectangles.

RTE example. Groundtruth: "entailment"		
P	About 33.5 million people live in this massive conurbation. I would guess that 95% of the 5,000 officially foreign-capital firms in Japan are based in Tokyo.	"entailment" 0.90
H	About 33.5 miilion people live in Tokyo.	
P	About 33.5 million people live in this massive conurbation. I would guess that 95% of the 5,000 officially foreign-capital firms in Japan are based in Tokyo.	"entailment" 0.79
H_1	people in miilion 33.5 live Tokyo About.	
P	About 33.5 million people live in this massive conurbation. I would guess that 95% of the 5,000 officially foreign-capital firms in Japan are based in Tokyo.	"entailment" 0.80
H_2	33.5 in people About live Tokyo miilion.	

Figure A4: Each RTE example contains a pair of premises and hypotheses i.e. (P, H). We shuffled the words in the original hypothesis H to create modified hypotheses (H_1 & H_2) and fed them together with the original premise (P) to a RoBERTa-based classifier for predictions. Also, the classifier harnessed self-attention to detect the correspondence between similar words that appear in both the premise and hypothesis. That is, the top-3 pairs of words that were assigned the largest premise-to-hypothesis weights in a self-attention matrix (layer 0, head 7) are inside in the red, green, and blue rectangles.

Task Name	Task Type	Label	GLUE dev-set processing				dev-r
			(a) dev set	(b) step 1	(c) step 2	(d) step 3	Total
CoLA	Acceptability	"unacceptable"	322	287	154	154	308
		"acceptable"	721	675	638	154	
RTE	NLI	"not entailment"	131	131	72	72	144
		"entailment"	146	145	127	72	
QQP	Paraphrase	"not duplicate"	25,545	22,907	20,943	12,683	25,366
		"duplicate"	14,885	14,000	12,683	12,683	
MRPC	Paraphrase	"not equivalent"	129	129	101	101	202
		"equivalent"	279	279	255	101	
SST-2	Sentiment	"negative"	428	427	402	402	804
		"positive"	444	443	420	402	
QNLI	NLI	"not entailment"	2,761	2,741	2,500	2,500	5,000
		"entailment"	2,702	2,690	2,527	2,500	
STS-B	Similarity	N/A	1,500	1,498	N/A	N/A	1,498

Table A4: The number of examples per class before (a) and after each of the three filtering steps to produce dev-r sets (described in Sec. 2.3.2) for RoBERTa-based classifiers. For each task, we repeated the same procedure for three sets of classifiers, for BERT-, RoBERTa-, ALBERT-based classifiers, respectively.

Model	Task	dev-r	dev-s	dev-s performance				Word-Order Sensitivity			
		performance	baseline	2-noun swap	1-gram	2-gram	3-gram	2-noun swap	1-gram	2-gram	3-gram
$\text{RoBERTa}_{\text{large}}$	CoLA	100	50	70.80	51.40	55.62	57.98	**0.58**	**0.97**	**0.89**	**0.84**
	RTE	100	50	82.29	73.85	80.42	83.75	0.35	0.52	0.39	0.33
	SST-2	100	50	98.24	83.71	88.16	90.43	0.04	0.33	0.24	0.19
	MRPC	100	50	98.54	85.53	88.64	90.49	0.03	0.29	0.23	0.19
	QQP	100	50	87.13	86.84	90.65	92.60	0.26	0.26	0.19	0.15
	QNLI	100	50	95.26	91.12	95.20	96.46	0.09	0.18	0.10	0.07
	STS-B	90.43	N/A	88.95	85.47	87.20	87.98	N/A	N/A	N/A	N/A

Table A5: Accuracy of all models on dev-s examples (created by shuffling n-grams and swapping 2 nouns) and their Word-Order Sensitivity scores ($\in [0, 1]$) across seven GLUE tasks. STS-B is a regression task and thus not comparable in word-order sensitivity with the other tasks, which are binary classification.

Task	GLUE dev set						
	CoLA (Acc)	RTE (Acc)	QQP (Acc)	MRPC (Acc)	SST-2 (Acc)	QNLI (Acc)	STS-B (Spearman Corr)
$\text{RoBERTa}_{\text{base}}$	82.56	72.20	91.12	87.25	94.50	92.57	90.17
$\text{ALBERT}_{\text{base}}$	81.21	72.20	90.25	87.99	91.40	91.78	90.82
$\text{BERT}_{\text{base}}$	81.89	64.25	90.81	85.54	92.09	91.38	88.49
$\text{RoBERTa}_{\text{large}}$	65.30	80.87	91.62	88.48	96.44	94.45	90.44
Average	82.78	72.38	90.95	87.32	93.61	92.55	89.98

Table A6: The dev-set performance of models finetuned from three different BERT "base" variants (12 self-attention layers and 12 heads) and one RoBERTa "large" model (24 self-attention layers and 16 heads) on seven GLUE tasks. These results match either those reported by original papers, Huggingface 2020 or GLUE leaderboard.