SLM: Learning a Discourse Language Representation with Sentence Unshuffling

Haejun Lee[♡] **Drew A. Hudson**[♣] **Kangwook Lee**[♡] **Christopher D. Manning**[♣]
♡ Samsung Research ♣ Stanford University
{haejun82.lee, kw.brian.lee}@samsung.com
{dorarad, manning}@cs.stanford.edu

Abstract

We introduce **S**entence-level **L**anguage **M**odeling, a new pre-training objective for learning a discourse language representation in a fully self-supervised manner. Recent pre-training methods in NLP focus on learning either bottom or top-level language representations: contextualized word representations derived from language model objectives at one extreme and a whole sequence representation learned by order classification of two given textual segments at the other. However, these models are not directly encouraged to capture representations of intermediate-size structures that exist in natural languages such as sentences and the relationships among them. To that end, we propose a new approach to encourage learning of a contextualized sentence-level representation by shuffling the sequence of input sentences and training a hierarchical transformer model to reconstruct the original ordering. Through experiments on downstream tasks such as GLUE, SQuAD, and DiscoEval, we show that this feature of our model improves the performance of the original BERT by large margins.

1 Introduction

Recent representation learning methods in NLP such as BERT (Devlin et al., 2019) have focused on learning two types of representations: the bottom-level – a contextual representation centered at a single word, trained by recovering randomly masked tokens or predicting previous and next words, and the top-level text, implicitly represented as a single [CLS] symbol and trained by predicting a relation between input segments, which usually consist of multiple sentences. However, natural language text has, in contrast, a very dominant hierarchical structure, with words grouped together into intermediate semantic units such as phrases and then sentences to convey the full meaning of a given text. Neither the transformer architecture nor the pre-training task leverages this hierarchy, treating the language as a flat sequence of tokens instead.

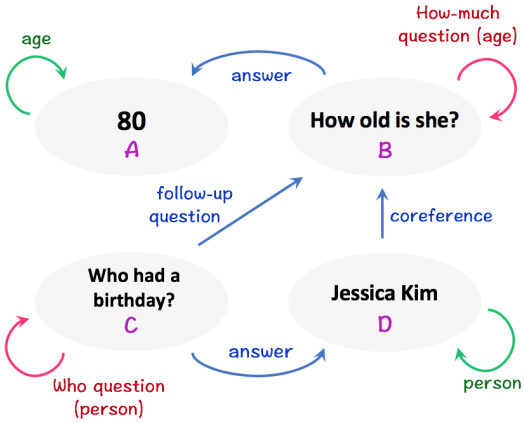

Figure 1: An example of a shuffled conversation and potential features that can aid in reconstructing the original sentence ordering, C → D → B → A.

Inspired by prior works about sentence-based representations for recurrent networks (Kiros et al., 2015; Hill et al., 2016; Gan et al., 2017; Jernite et al., 2017; Logeswaran et al., 2018; Gong et al., 2016; Chen et al., 2016), we seek to incorporate a more explicit hierarchy into the transformer by extending it with the capacity to learn contextualized sentence-level representations. Equipping the model with such a capacity allows it to learn languages at multiple levels of granularity, ranging from fine-grain connections across words to high-level discourse relations between sentences and paragraphs.

Predicting an original sequence of sentences requires deep understanding of natural language, including a variety of phenomena such as discourse relations, coreference, temporal dependencies, entailments, narratives, and so on. Figure 1 shows a set of statements from a conversation in the CoQA dataset (Reddy et al., 2019), and clues that can potentially be used to reconstruct their original ordering. To figure out the correct order, in this case

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 1551–1562,
November 16–20, 2020. ©2020 Association for Computational Linguistics

C-D-B-A, a model has to understand that D and A are answers to questions C and B respectively and that 'she' in B refers to D, and it may need to know that B is a follow-up question on C. Not only are the semantics of separate sentences important but the relationships between them are crucial in solving this task. We therefore seek to encourage the model to capture these vital properties by training it to reconstruct the original sentence ordering.

To achieve this, we propose a pre-training objective that extends the word-level language modeling to analogously learn representations for sentences. Typical word-level language models are trained by guessing the neighbors of given words or by predicting masked words based on their context. Applying the same idea at a higher level, our approach learns representations that support predicting the next sentence representation among shuffled sentences for given previous representations, based on the semantic relations between them.

To allow the model to be effectively trained with the new unshuffling objective, we propose a pointer-based neural module that is specialized to predict the sentence order, called the Sequence Reconstructor (SR). In the SR, each sentence is represented by a sentential token that is inserted in front of it. A pointer network layer stacked on the transformer decoder is trained to point at the next contextualized sentence representation based on the previous sentence representations.

We show that our method achieves robust improvements over standard BERT's performance on the following downstream NLP tasks: GLUE for Natural Language Inference (Wang et al., 2018), SQuAD for Question Answering (Rajpurkar et al., 2018), and DiscoEval (Chen et al., 2019) for discourse aware sentence representations. We match the score of Text-To-Text Transfer Transformer Base ($T5_{BASE}$) (Raffel et al., 2020), a state-of-the-art model that uses $BERT_{BASE}$ hyperparameters, while using only half the parameters, shorter training (3/8 tokens overall), and a fraction (1/37) of the data compared to T5. Moreover, we investigate the effect of the proposed objective through a qualitative analysis of the neighbor sentences of sentences that have similar sentential representations. We show that the results support our aim of enriching the transformer model with sentence-level language understanding.

The contributions of our work are threefold: 1) we propose a new self-supervised pre-training objective that extends a word-level language modeling strategy to the sentence-level; 2) we propose a pointer-based neural module to train the model for this objective, and demonstrate it leads to significant improvements over diverse NLP tasks; and 3) we provide a qualitative analysis showing that the learned contextualized sentence representations embed more subtle and rich structural, semantic, and relational information.

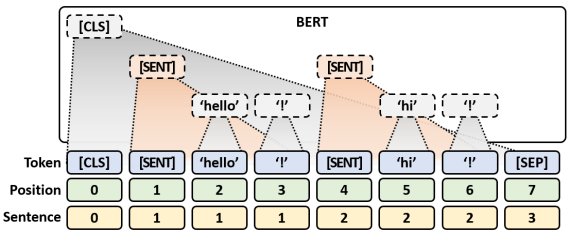

Figure 2: Intended representation scope of each contextualized sentence representation

2 Proposed Method

In this section, we describe the details of our proposed methods. A conventional representation model is extended with sentence representations to gather sentence information and a Sequence Reconstructor (SR) to predict their original ordering. Fine-tuning architectures are modified to inject sentence representation into downstream NLP tasks.

2.1 Pre-training with Sentence Unshuffling

Our model consists of an encoder for contextualizing input texts that uses a conventional transformer model like BERT, and a decoder that reconstructs the original ordering. We begin by splitting the input text into target segment units, sentences in this case, and shuffle them to remove their original ordering. We then add special tokens at the beginning of each segment, which will be used as sentence representations aggregating their meaning from their surroundings. We define a sentence-level language model (SLM) loss for reconstructing the original sentence ordering by making a sequence of predictions with a pointer network. Finally, the model is trained by sum of this loss and the standard masked language modeling (MLM) loss to pre-train the model. Through this pre-training objective, representations of sentences are properly contextualized by their neighboring sentence representations. Although we only consider sentence representations in this paper, our method can easily be extended to other syntactic levels such as

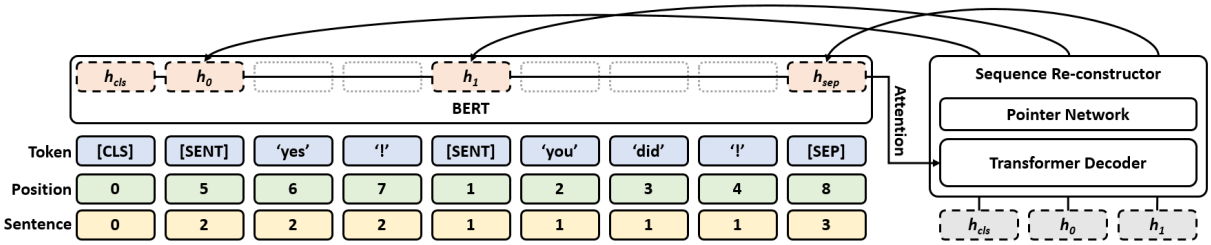

Figure 3: An overview of the architecture for the sentence unshuffling. Input sentences are randomly shuffled and the original ordering is reconstructed by our Sequence Reconstructor model using the sentence representations.

phrases or paragraphs. We provide further detail about each of these components in the followings.

Sentence Representation: We insert a sentence token, [SENT] in front of every sentence, that is meant to represent each sentence and aggregate its meaning from its surroundings. Similarly to positional and segment-level embeddings in the original BERT model, and to distinguish sentences and limit the scope of the sentence representations, we add trainable sentence embeddings indicating the index of each sentence to each of the word representations in order to support potential sentence-level aggregation as in ERNIE (Sun et al., 2020). Figure 2 shows trainable embeddings added to inputs and the scope of input tokens that are represented by each contextualized representation.

Sequence Shuffling: Input sentences are shuffled by manipulating their positional and sentence embeddings (as the positional encoding is the only clue to derive the actual sequences in a transformer architecture, unlike traditional RNNs). For example, the inputs of Figure 3 are a pair of swapped sentences. Position embeddings of the first sentence are 5 to 7 and those of the second sentence are 1 to 4. The sentence embedding are similarly swapped as well. Note that importantly the model can perform the unshuffling task only based on the semantics of the sentences and the relations between them, rather than by using the positional embeddings, since these are shuffled together with the sentences (namely, if sentence X and Y are swapped, so are their positional embeddings). For each iteration, only half of the batches are shuffled to allow the model to see some of the input in its natural ordering.

Sequence Reconstructor: SR is our conditional decoder consists of a pointer network and transformer decoder similar to Gong et al. (2016) and Logeswaran et al. (2018). It predicts the original ordering of the shuffled input as depicted in Figure 3. After encoding the shuffled inputs by BERT, we consider the contextualized embeddings: $C = [h_{cls}, h_0, ..., h_{N-1}, h_{sep}]$ that correspond to the N sentence tokens we have inserted into the input sequence, along with the first ([CLS]) and last ([SEP]) tokens. It is passed to the Sequence Reconstructor for the task of reconstructing the original sentence ordering.

The embeddings in C, except the last h_{sep}, are sequentially processed through a transformer decoder (Vaswani et al., 2017) to obtain output w_i while attending on the whole C and embeddings processed in the previous steps, $\{C_j\}_{j=0}^{i-1}$. Each processed output will be used to predict the sentence i in step i (Note that indexes of sentence representations in C are shifted by 1 due to h_{cls} inserted in front).

$$w_i = \text{TransformerDecoder}(C_i, \{C_j\}_{j=0}^{i-1}, C) \tag{1}$$

Then, we compute a probability distribution over the sentence representations using a pointer network. Each probability represents the likelihood of each sentence to come up next at step i, aiming to predict their original ordering:

$$\text{P}_i = \text{Softmax}(w_i C^T) \tag{2}$$

We calculate a cross-entropy loss to match for each contextualized sentence embedding the probability distribution of the predicting ordering with the golden positions. $\text{P}_i = \{p_{i,j}\}_{j=0}^{N+1}$ is the predicted probabilities of sentence j-1 (due to a shift by h_{cls}) appearing at position i, and o_i is a one hot ground-truth vector of the correct position at step i. Finally, we average the losses for all positions $i = 0, \ldots, N$:

$$\mathcal{L}_{\text{slm}} = -\frac{1}{N+1} \sum_{i=0}^{N} \sum_{j=1}^{N+1} \text{o}_{i,j} \log\left(p_{i,j}\right) \tag{3}$$

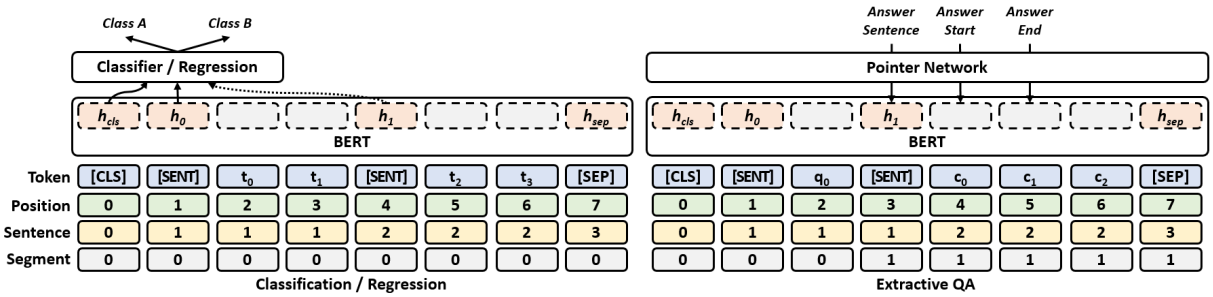

Figure 4: Fine-tuning Architectures. Sentence representations are concatenated to the [CLS] representation before being fed into the output layer for downstream classification (or regression) tasks and used to predict an answer sentence for extractive QA tasks.

Intuitively, the decoder learns how to transform a representation of sentence to the next sentence's representation recurrently by attending on the whole sentence representations from the encoder and previously transformed embeddings.

One important design choice here is that the Sequence Reconstructor only sees C, which are contextualized embeddings of [CLS], the sentence tokens, and [SEP] while other contextualized word representations are masked out. The model should find out the original sequence using only sentence representations and this will enforce the encoder to embed all necessary information into the contextualized sentence embedding instead of spreading them over all embeddings.

Overall, the whole model is trained by minimizing a sum of the standard masked language model loss as in the original BERT design and our new sentence-level language model loss.

$$\mathcal{L} = \mathcal{L}_{mlm} + \mathcal{L}_{slm} \quad (4)$$

Note that the SR module only adds about 7.1% overhead in computation time to the standard transformer encoder and only during the pre-training stage. It is lightweight compared to the full transformer stack in terms of computation and number of parameters, because it performs re-ordering over a small number of sentential tokens (20 in a 512 token sequence) using shallow decoder stacks (3 layers).

2.2 Fine-tuning

The original fine-tuning methods of BERT are slightly modified to encourage the model to use sentence representations learned during pre-training for downstream NLP tasks. Figure 4 shows the overall fine-tuning architecture for extractive QA and classification and regression tasks.

Classification and regression: The contextualized sentence embeddings of given sentences are concatenated to the [CLS] embedding. Depending on the sentence number of the task, one or two sentence tokens are concatenated before being fed into the output layer.

Extractive QA: For extractive tasks, we add an answer sentence prediction that finds the sentence token corresponding to the answer, using a pointer network to an existing answer span prediction. To distinguish the question and the context, segment embeddings as in BERT are used while other tasks use the same segment index for all inputs. The final loss is the sum of three independent pointer networks losses: 1) the answer start index, 2) the answer end index, and 3) the answer sentence index.

3 Experiments

3.1 Benchmark Datasets

To measure the excellence of our propose SLM on various downstream tasks, we conduct experiments with three well-known benchmarks: General Language Understanding Evaluation (GLUE), the Stanford Question Answering (SQuAD) v1.1 and v2.0, and Discourse Evaluation (DiscoEval) (Chen et al., 2019). We exclude the WNLI task from GLUE following BERT which always achieves 65.1 accuracy of the majority class. RST-DT task is also excluded from DiscoEval which requires a tree structure encoding. Metrics used are Matthew's correlation for CoLA, F1 for MRPC, Spearman's correlation for STS-B, Exact Match (EM) / F1 for SQuAD, and accuracy for rest of tasks. We report development set results for SQuAD to compare with other models, both development and test set scores for GLUE, and test set scores for DiscoEval.

Model	CoLA	SST-2	MRPC	STS-B	QQP	MNLI-m/mm	QNLI	RTE	Avg.
Development set results. The best score from hyperparameter searches using parameters in Section 3.2.									
$\text{BERT}_{\text{BASE}}$	57.3	92.9	89.0	88.6	91.4	84.8 / 84.9	91.8	71.5	83.6
$\text{BERT}_{\text{LARGE}}$	63.1	93.2	88.0	89.5	91.7	86.6 / 86.6	92.3	74.0	85.0
$\text{BERT}_{\text{LARGE-WWM}}$	64.1	94.7	90.0	90.4	91.7	87.8 / 87.7	94.0	77.3	86.4
SLM_{BASE}(1M steps)	62.1	93.7	90.0	90.3	91.6	86.6 / 86.4	93.0	81.2	86.1
SLM_{BASE}(3M steps)	62.4	94.2	90.4	90.9	91.7	87.4 / 87.5	93.8	83.0	86.8
Test set results. Other models' scores are taken from the GLUE leaderboard or their papers									
$\text{BERT}_{\text{BASE}}$	52.1	93.5	88.9	87.1	89.2	84.6 / 83.4	90.5	66.4	81.7
ERNIE 2.0_{BASE}	55.2	95.0	89.9	86.5	**89.8**	86.1 / 85.5	92.9	74.8	84.0
T5_{BASE}	51.1	**95.2**	**90.7**	**88.6**	89.4	87.1 / 86.2	93.7	**80.1**	84.7
$\text{BERT}_{\text{LARGE}}$	60.5	94.9	89.3	87.6	89.3	86.7 / 85.9	92.7	70.1	84.1
$\text{XLNet}_{\text{LARGE}}$	63.6	95.6	89.2	91.8	91.8	89.8 / -	93.9	83.8	87.4
$\text{ELECTRA}_{\text{LARGE}}$	68.2	94.8	89.6	91.0	90.1	90.1 / -	95.4	83.6	87.9
T5_{LARGE}	61.2	96.3	92.4	89.9	89.9	89.9 / 89.6	94.8	87.2	87.9
SLM_{BASE}(3M steps)	**55.3**	95.1	90.0	88.3	89.6	**87.3 / 86.8**	**93.9**	78.5	**85.0**

Table 1: GLUE results. BERT dev scores are produced by same hyperparameter searches using the official pre-trained models. Bolded are the highest scores among Base-size models. XLNet and ELECTRA's scores are averages except MNLI-mm that are not reported in the papers.

3.2 Experimental Setup

For the experiments, we follow $\text{BERT}_{\text{BASE}}$'s hyperparameters and corpus. Our model is trained with 512 length-256 batch size using Wikipedia dumps and BookCorpus (Zhu et al., 2015). We split inputs into sentences using the NLTK toolkit (Loper and Bird, 2002), which are then re-shuffled for every epoch. In terms of data preprocessing, when each input is fed into the model, we set the maximum count M of sentences to process as 20 in our experiments. If exceeded, we randomly merge pairs of adjacent sentences and treat them as a single sentence until the required sentence count is reached. We perform this preprocessing step in order to deal with an uncommonly large number of sentences. For the masked language model objective, we randomly mask up to three continuous word tokens using a geometric distribution, $Geo(p = 0.2)$ following the findings of T5 (Raffel et al., 2020) and SpanBERT (Joshi et al., 2020), but sentence tokens are not masked. The models are pre-trained for 1M and 3M steps and optimized by Adam with linear weight decaying using a learning rate 1.5e-4. Any other configuration not mentioned here is the same as the original BERT model.

Fine-tuning is done by a hyperparameter search of learning rate: {1e-5, 3e-5, 5e-5, 1e-4} and epoch between 2 to 15 depends on the tasks. We select the run with the best development set score among five runs for each parameter combination for more stable results. For GLUE test set results, predictions from models with the best development score are submitted. All results of downstream NLP tasks in this paper are results of a single model trained with a single task. For DiscoEval(Chen et al., 2019) tasks, we freeze the encoder and only fine-tune the output layer following Chen et al. (2019).

Although the recent state-of-the-art models such as ALBERT (Lan et al., 2019) or T5 use extremely large model and training data, we follow $\text{BERT}_{\text{BASE}}$'s model size because of practical limitations of computation resources. We believe that the BASE settings allow for a robust comparison to the current leading approaches.

4 Results

We compare our SLM model with the original BERT, XLNet (Yang et al., 2019), ELECTRA (Clark et al., 2019), ERNIE 2.0 (Sun et al., 2020), CONPONO (Iter et al., 2020), BART (Lewis et al., 2020) and T5 which are the current state-of-the-art for the benchmark datasets we use. We mainly compare with models using $\text{BERT}_{\text{BASE}}$ hyperparameters.

4.1 GLUE results

Table 1 shows the performance of our method on the GLUE dataset for all tasks except WNLI. Our method significantly improves the downstream NLP tasks in the GLUE dataset. While most tasks are improved from the original BERT, improvement of RTE by 12.1 points is the most significant. We assume this is because our model learns the relationship of sentences thanks to our objective and it is the most effective on the entailment task with relatively small data like RTE. While our average test score is the highest among Base-size models that is 0.3 points higher than T5_{BASE}, some scores

Model	SQuAD v1.1 EM / F1	SQuAD v2.0 EM / F1
$\text{BERT}_{\text{BASE}}$	80.8 / 88.5	75.2 / 78.3
$\text{XLNet}_{\text{BASE}}$	-	78.5 / 81.3
$\text{BART}_{\text{BASE}}$	- / 90.8	-
$\text{ELECTRA}_{\text{BASE}}$*	84.5 / 90.8	80.5 / 83.3
T5_{BASE}	**85.4** / 92.1	-
$\text{BERT}_{\text{LARGE}}$	84.1 / 90.9	78.7 / 81.9
$\text{BERT}_{\text{LARGE-WWM}}$	86.7 / 92.8	82.6 / 85.4
SLM_{BASE}(1M)	84.6 / 91.5	80.7 / 83.7
SLM_{BASE}(3M)	85.3 / **92.2**	**81.9 / 84.9**

Table 2: Fine-tuning results on the SQuAD v1.1 and v2.0 development sets. The scores of BERT on SQuAD v2.0 are produced using the official pre-trained models and others are taken from their papers. Bolded are the highest scores among Base-size models. *mean scores over 10 runs.

are even comparable with large models trained by more total tokens and data. Our average score is 0.9 points higher than $\text{BERT}_{\text{LARGE}}$. Our SST-2 score is higher than BERT and ELECTRA's scores and MRPC score is higher than BERT, XLNet, and ELECTRA.

4.2 SQuAD results

As shown in Table 2, our method significantly improves the performance on both SQuAD v1.1 and v2.0. It exceeds the original $\text{BERT}_{\text{BASE}}$ model by 3.0 and 5.4 points F1 and 3.8 and 5.5 points EM for v1.1 and v2.0 respectively when it is equally trained for 1M steps. When it is sufficiently trained for 3M steps, additional gains are achieved by 0.7 and 1.2 points EM/F1. Finally, it achieves a tie score with the T5_{BASE} model in SQuAD v1.1. This is an impressive result because T5 is an encoder-decoder model which uses twice as many parameters and is trained by 2.7-times more total tokens and astonishingly 37-times more data.

Model	SP	BSO	DC	SSP	PDTB-E/I
$\text{BERT}_{\text{BASE}}$	53.1	68.5	58.9	80.3	41.9 / 42.4
$\text{BERT}_{\text{LARGE}}$	53.8	69.3	59.6	80.4	44.3 / 43.6
$\text{RoBERTa}_{\text{BASE}}$	38.7	58.7	58.4	79.7	39.4 / 40.6
$\text{CONPONO}_{\text{BASE}}$	60.7	76.8	72.9	80.4	42.9 / 44.9
SLM_{BASE}(1M)	72.4	84.1	75.4	**81.5**	45.9 / 46.3
SLM_{BASE}(3M)	**73.4**	**84.5**	**76.1**	**81.5**	**46.4 / 47.8**

Table 3: Test set results of DiscoEval datasets except RST-DT. Scores of other models are from Chen et al. (2019) and Iter et al. (2020)

4.3 DiscoEval Results

Table 3 shows the test set results on DiscoEval. Our method achieves improvements with a large margin on all tasks over the previous state-of-the-art model, CONPONO (Iter et al., 2020). This result reveals

Model	SQuAD v1.1 EM / F1	SQuAD v2.0 EM / F1
Original Sentences	82.7 / 89.8	76.5 / 79.4
Shuffled Sentences	**83.2 / 90.1**	**77.4 / 80.5**

Table 4: The results on SQuAD datasets of models trained with original and shuffled sentences.

that our method learns discourse relations better with our sentence unshuffling objective. Especially, an improvement on SP, finding the original position of a given sentence, is the most significant by 12.7 points higher than CONPONO. This is because sentence positions for a given sentence can be found by finding the next sentence which means the SP task is a reduced problem of our sequence reconstruction task. The improvements on SSP, predicting whether the given sentence belongs to the Abstract section, is relatively little because it has less in common with the SLM objective.

5 Analysis

5.1 Sentence representation in BERT

To gain motivation for our approach, we begin by running a simple experiment to measure the level of hierarchical understanding of the standard BERT model using the masked language model objective. We train two models: one with sentences in the original ordering and another with the sentences shuffled. The latter model is expected to lose the hierarchical semantics at the discourse level as the shuffling breaks the relations implied by the natural ordering. In practice, the SQuAD results detailed in Table 4 show a model trained with the shuffled sentences in fact outperforms the model trained with the sentences in the original ordering. These results indicate that BERT does not take advantage of the global, sentence-level, structure of the text but rather apparently captures a shallower understanding that does not translate into an improvement in reading comprehension performance.

5.2 Effect of our proposed method

In this section, we perform ablation studies to show the impact of our proposed methods. We trained 5 models with different configurations discarding different aspects of the model. We use SQuAD and four tasks from GLUE which are large enough to derive stable comparisons.

Shuffled Batch: Amount of shuffled batches that need to be balanced between learning by seeing

Shuf. Batch	Seq. Recon.	Sent. Repr.	SQuAD v1.1 EM / F1	SQuAD v2.0 EM / F1	MNLI m / mm	QNLI	SST-2	QQP	Avg.
50%	✓	✓	**84.6 / 91.5**	**80.7 / 83.7**	**86.6 / 86.4**	**93.0**	**93.7**	**91.6**	**88.0**
100%	✓	✓	84.2 / 91.2	80.3 / 83.4	86.1 / 86.2	92.9	93.5	91.4	87.7
100%		✓	83.7 / 90.7	77.3 / 80.4	85.4 / 85.6	92.3	93.0	91.3	86.6
0%		✓	82.8 / 90.1	76.8 / 79.8	85.3 / 85.5	92.2	93.0	91.1	86.3
0%			82.7 / 89.8	76.5 / 79.4	84.7 / 84.6	91.7	92.8	91.1	85.9

Table 5: Ablation study on our proposed methods

and predicting the original sequence.

Sequence Reconstructor: Whether the model is trained with a loss from sentence-level objective and the sequence reconstructor model.

Sentence Representation: Whether the model uses sentence representation tokens and embeddings.

While all features increase performance individually, the biggest gains stem from the sequence reconstruction objective with both the shuffled and original orderings. This combination increases SQuAD v1.1 EM / F1 by 1.9 / 1.7 points and v2.0 by 4.2 / 4.3 points respectively. MNLI-m/mm accuracies are similarly improved by 1.9 and 1.8 points. Other tasks are also slightly improved. Notably, our best performing model is achieved by shuffling the sentence ordering only for half of the input batches, to allow the model experiencing also the natural ordering. Removing the ordering from all batches rather than half of them leads to slight reductions of 0.1-0.5 points for most tasks, which might result from the model's lack of exposure to natural input ordering that would be necessary for some tasks.

One interesting point is that just adding a sentence representation or shuffling the input are each individually increasing the performance even without the sentence-level objective. We conjecture that some of the sentence representations can be learned from the standard word-level prediction loss without the specific objective based on the explicit sentence representation. Shuffling input might help by adding regularization effects and making the model more robust against noisy permutations as well.

5.3 What information is learned by SLM?

For a more intuitive understanding of what is learned from the unshuffling objective, we search the closest sentences to query sentences using a cosine distance of sentence representations from our pre-trained model. Sentences with typical discourse labels from samples in Jernite et al. (2017) are used as queries to see whether our model captures a variety of different discourses and 1M sentence pairs from Gutenberg BookCorpus (Lahiri, 2014) are used as targets to search. Table 6 shows the Top 3 closest sentences for each query sentence with discourse labels. We additionally pick one of the highly ranked results to show diversities of similarities between queries and retrievals. We mark query and retrieved sentences in bold and show previous sentences together. Semantically similar phrases are marked in blue while potential clues to discourse relations are in red. Results that do not match the query's discourse label are marked by $\times$.

As we can see from the table, most of the retrieved sentences share a mixture of syntactic, semantic and discourse-level aspects with the given queries, especially in regard to their relationships with their surrounding context. Even without any fine-tuning and filtering of the retrieved results, most of the highly ranked ones share a similar discourse relation with their prior sentences to that of the query sentences. Some discourse relations are commonly indicated by conjunctive words, and then the retrieved sentences have conjunctions of the relevant discourse sense, e.g. *"in fact"*, and *"soon"* for *strengthen*, and words indicating similar temporal relations for *temporal*. In other cases, when the results differ in the conjunctive words, they still hold high-level structural similarities to the query sentences. This is especially true for the *elaboration* and *list* relations. Finally, we notice that semantically similar phrases are also captured by the retrieved results. In the *contrast* example, results contain phrases like *"all was still"*, and *"no other sound came from"* which correspond to the word *"silence"* in the query. It is noticeable that overall our model seems to focus more on the relational aspect than the semantic and syntactic aspects. This serves as a qualitative evidence for the effectiveness of the new unshuffling objective: while all aspects are needed for sentence understanding, we conjecture that the relational component is the one most vital for the un-shuffling task as it naturally demands an understanding of the

Discourse	Sources and Nearest Neighbors
Contrast	**Q** : The waterwheel hammered on. **Otherwise there was silence.**
×	1) A portion of the burning log fell on to the hearth. **Then there was silence**
	2) The bees still worked on, and the butterflies did not rest from roving, their smallness seeming to shield them from the stagnating effect that this turning moment of day had on larger creatures. **Otherwise all was still.**
	3) A class in spelling, big boys and little girls, toed a crack in front of the waster's desk. **The rest of the school droned away on appointed tasks in the drowsy interlude.**
×	4) Fra Girolamo, give her-the Crucifix, said the voice of Fra Girolamo. **No other sound came from the dying lips.**
Strengthen	**Q** : It doesn't hurt at all. **In fact it's exhilarating.**
	1) We have no great political guns aboard. **On the contrary, the majority of the passengers are Americans.**
	2) I don't believe in the faults. **They're just a joyous softening of the outline - more beautiful than perfection.**
	3) I never regarded Ealer's readings as educational. **Indeed, they were a detriment to me.**
	7) We have no sympathy at all with the moral indignation of our time against M. Zola. **It is simply the indignation of Tartuffe on being exposed.**
Elaboration	**Q** : Then I became a woman. **A strong one at that.**
	1) Christianity was the last great religious synthesis. **It is the one nearest to us.**
	2) I wanted you to care for me so that I could influence you. **It wasn't easy.**
	3) It got on my nerves - the women I saw. **Worse than any man.**
	9) He's past his seventy now, - ever so much; but he's just as modest as a young girl. **A deal more modest than some of them.**
Return	**Q** : He had a point. **Still for good measure, I pouted.**
	1) Logically - if not legally - there is apparently an inference of the interchange of matrimonial consent here. **I stick to my own opinion, nevertheless.**
	2) To him I was a squeezed lemon. **Nevertheless I took his hint.**
	3) It was almost dark. **Yet I must walk away.**
	9) He spoke not a word. **I pitied him from the bottom of my heart.**
Temporal	**Q** : It limped closer at a slow pace. **Soon it stopped in front of us.**
	1) The current carried them on and on, but not so swiftly as it was carrying the tree. **Soon they were approaching the bend.**
	2) Then he got down from his post and loafed along the sidewalk, still observing and occasionally commenting. **Presently he dropped into my wake and followed along behind.**
	3) Slowly, patiently, watchfully, the hunter followed. **After a while he stopped with a satisfied grin.**
	12) The mass fell into columns by threes and fours to accommodate itself to the narrow road, and strode briskly along southward in the wake of the leaders. **In a few minutes the Hogan cabin was reached.**
List	**Q** : I saw flowers on the ground. **I heard birds in the trees.**
×	1) He saw a garden. **We saw a wilderness.**
	2) I heard the pulse of the besieging sea throb far away all night. **I heard the wind fly crying, and convulse tumultuous palms.**
×	3) I took several long walks while collecting objects of natural history. **The country is pleasant for exercise.**
	5) My maid is a treasure. **My dressmaker is charming.**

Table 6: Nearest neighbors of contextualized sentence representations.

relationships between the sentences.

6 Related Work

Contextualized Representation learning for NLP: Self-supervised representation learning became popular after contextualization methods were introduced. ELMo (Peters et al., 2018) dynamically contextualizes representations of adjacent words using bi-directional recurrent encoders. BERT (Devlin et al., 2019) adopted a deep transformer encoder and has proposed the masked language modeling objective incorporating a bi-directional context. After the success of BERT, researchers started exploring various new pre-training objectives such as span boundary representations (Joshi et al., 2020), reordering of local permutations (Wang et al., 2019), detecting incorrectly replaced tokens (Clark et al., 2019), combining multiple tasks (Sun et al., 2020), a decoder-based masked word prediction (Song et al., 2019), and so on.

Another line of works tried to scale up the model with more parameters, training data, and computation resources. RoBERTa (Liu et al., 2019) trains $\text{BERT}_{\text{LARGE}}$ model with 10x more data for 16x more iterations. ALBERT (Lan et al., 2019) uses the xxlarge model whose parameters are reduced by weight sharing and factorizations. T5 proposes a multi-task encoder-decoder architecture that uses up-to about 33x weight parameters, 37x data size compared to the original $\text{BERT}_{\text{LARGE}}$ model.

Sentence Representation Models: Several works tried to learn sentence representations using adja-

cent sentences, especially in recurrent networks. SkipThoughts (Kiros et al., 2015) and FastSent (Hill et al., 2016) proposed an encoder-decoder architecture that encodes a sentence and generates the next and previous sentences. These works focus on independent representations of single independent sentences and do not consider the dynamic contextualization using neighboring sentences.

We believe that, similar to words, sentence representations should also be properly contextualized in order to embed richer meaning including relationships to other sentences. HLSTM (Chang et al., 2019) considered contextualization of sentence representations by incorporating previous step's sentence representation for word prediction. HIBERT (Zhang et al., 2019) proposed a hierarchical transformer encoder trained by recovering masked sentences, focusing in particular on summarization. However, both of these models are trained by performing prediction at the word level, wheres our sentence unshuffling approach demands fine understanding of the relations among the sentences at the more global discourse level.

Learning from Sentence Ordering: There are prior works about learning contextualized sentence representations by recovering the original sequence of sentences. Gong et al. (2016) and Logeswaran et al. (2018) proposed RNN-based pointer networks for reconstructing the sequence order. However, they utilized hierarchical models that encode each sentence separately without any access to other sentences. This fact not only restricts long-term contextualization of words and sentences but also limits downstream tasks due to the dedicated architectures. On the other hand, our method encodes multiple sentence representations at the same time with both inner and inter-sentence contextualization.

The effect of predicting textual segment order in pretrained language models has been widely investigated as well. BERT proposed the Next Sentence Prediction task (NSP) which predicts whether two given text segments are from the same documents or not. The results of SpanBERT (Joshi et al., 2020) question the value of NSP, suggesting this might be due to noise from merging 2 unrelated texts from different documents. Consequently, ALBERT (Lan et al., 2019) and StructBERT (Wang et al., 2019) added sentence ordering objectives by predicting the order of text segments. BART (Lewis et al., 2020) proposed the Sentence Permutation task which is similar with ours. It predicts the original sequence of sentences using an auto-regressive decoder, which reconstructs the whole sentences by word prediction. However, their approach does not provide any representation of each sentence. Moreover, while SLM shows strong task improvements, that is not the case for these models.

7 Conclusions

In this paper, we proposed the Sentence-level Language Modeling objective for contextualized sentence representation learning. Our approach extends a word-level language modeling strategy to the sentence-level by reconstructing the original order of shuffled sentences. In addition, we designed a special Sequence Reconstructor (SR) module to learn to perform the sentence re-ordering. It reconstructs the original order by pointing to the next sentence among encoded sentence representations using a pointer network and a transformer decoder. We evaluated the effect of the proposed idea on three benchmarks, GLUE, SQuAD, and DiscoEval, and showed consistent improvements over the previous approaches. We matched performance with the state-of-the-art model with the same model size using much fewer parameters, computation, and data. Through a qualitative analysis, we showed that our model can embed not only semantic but also relational features of sentences. We are excited about future work that could extend our motivation and further aim at incorporating stronger hierarchy into the language model architectures and the pre-training tasks.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Ming-Wei Chang, Kristina Toutanova, Kenton Lee, and Jacob Devlin. 2019. Language model pre-training

for hierarchical document representations. *arXiv preprint arXiv:1901.09128*.

Mingda Chen, Zewei Chu, and Kevin Gimpel. 2019. Evaluation benchmarks and learning criteria for discourse-aware sentence representations. 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 649–662.

Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2016. Neural sentence ordering. *arXiv preprint arXiv:1607.06952*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2019. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

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.

Zhe Gan, Yunchen Pu, Ricardo Henao, Chunyuan Li, Xiaodong He, and Lawrence Carin. 2017. Learning generic sentence representations using convolutional neural networks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2390–2400.

Jingjing Gong, Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2016. End-to-end neural sentence ordering using pointer network. *arXiv preprint arXiv:1611.04953*.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1367–1377.

Dan Iter, Kelvin Guu, Larry Lansing, and Dan Jurafsky. 2020. Pretraining with contrastive sentence objectives improves discourse performance of language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4859–4870.

Yacine Jernite, Samuel R Bowman, and David Sontag. 2017. Discourse-based objectives for fast unsupervised sentence representation learning. *arXiv preprint arXiv:1705.00557*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302.

Shibamouli Lahiri. 2014. Complexity of word collocation networks: A preliminary structural analysis. In *Proceedings of the Student Research Workshop at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 96–105.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

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*.

Lajanugen Logeswaran, Honglak Lee, and Dragomir R Radev. 2018. Sentence ordering and coherence modeling using recurrent neural networks. In *AAAI*.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, pages 63–70.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

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.

Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Alexander Sergeev and Mike Del Balso. 2018. Horovod: fast and easy distributed deep learning in TensorFlow. *arXiv preprint arXiv:1802.05799*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. In *International Conference on Machine Learning*, pages 5926–5936.

Y. Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, H. Wu, and Haifeng Wang. 2020. Ernie 2.0: A continual pre-training framework for language understanding. In *AAAI*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Wei Wang, Bin Bi, Ming Yan, Chen Wu, Jiangnan Xia, Zuyi Bao, Liwei Peng, and Luo Si. 2019. Structbert: Incorporating language structures into pre-training for deep language understanding. In *International Conference on Learning Representations*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019. Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

A Training Details

We provide details of hyperparameters and corpus we used for pre-training and fine-tunings. Table 7 shows all the detailed hyperparamters. We mainly used BERT-Base size (12 layers, 12 attention headers, 768 hidden size, and 110M parameters). Pre-training a BERT-Large sized model for 3M iterations is expected to take more than a month in our experiment environments, so we stick to BERT-Base size. All other hyperparameters except the learning rate during pre-training are the same as the original BERT. We found that LR 1.5e-4 results in slightly better compare to LR 1e-4 used in the original BERT.

Parameter	Pre-training	Fine-tuning
Encoder layers	12	12
Decoder layers	3	-
Attention heads	12	12
Hidden size	768	768
Sequence length	512	128 / 512
Sentence tokens	20	20
Vocab Size	30522	30522
Batch size	256	32
Step / Epoch	1M/3M	$2 \sim 15$
Warm-up	10K	10%
Learning Rate	1.5e-4	1/3/5e-5/1e-4
Adam epsilon	1e-6	1e-6
Dropout	0.1	0.1
Attention Dropout	0.1	0.1
Precision	Float16	Float16

Table 7: Training Hyperparameters.

We use English Wikipedia and Bookcorpus(Zhu et al., 2015) for pre-training. Since Bookcorpus is no longer available online, we collected our own version of BookCorpus using publicly available crawling code (https://github.com/soskek/bookcorpus). Texts are extracted from Wikipedia dumps using WikiExtractor (https://github.com/attardi/wikiextractor). Simple heuristic preprocesses are applied to clean the corpus which removed sentences that do not contain enough words. The final size of processed corpus are 9.2GB and 5.7 GB for Wikipedia and BookCorpus respectively. The total size is slightly smaller compare to the total corpus used to train the original BERT because some links are unavailable when we collected the BookCorpus. We used 8 NVIDIA v100 GPUs for pretraining and it took about 5 days to train for 1M iteration. We tokenize the corpus with WordPiece (Wu et al., 2016) tokenizer using uncased vocabulary of Google's official BERT release.

For all fine-tuning tasks, we use batch size 32 with 512 sequence length for SQuAD and 128 for GLUE tasks. Hyperparameter searches are mainly done for epoch {2, 3, 4, 5} and learning rate { 1e-5, 3e-5, 5e-5 }. We run 5 runs for each parameter combinations to get a stable dev set results. GLUE test set scores are achieved by submitting test set predictions of the best dev score models to the glue evaluation server. We report the best hyperparameters for each task in Table 8.

Task	Epoch	Learning Rate
SQuAD v1.1	2	5e-5
SQuAD v2.0	2	3e-5
CoLA	4	1e-5
SST-2	3	3e-5
MRPC	3	3e-5
STS-B	4	1e-5
QQP	4	3e-5
MNLI	2	3e-5
QNLI	2	3e-5
RTE	5	1e-5
SP	13	1e-4
BSO	13	5e-5
DC	15	5e-5
SPP	10	1e-4
PDTB-E	15	1e-4
PDTB-I	15	1e-4

Table 8: Best hyperparameters for NLP tasks.

B Implementation Details

We use Tensorflow(Abadi et al., 2015) for all of our experiments. Our implementations are based on the NVIDIA's tensorflow implementation of BERT which supports multi GPU using Horovod (Sergeev and Balso, 2018) and half precision training (https://github.com/NVIDIA/DeepLearningExamples/tree/master/TensorFlow/LanguageModeling/BERT). We adapt an implementation of transformer decoder from official tensorflow's transformer implementation (https://github.com/tensorflow/models/tree/master/official/transformer)