Towards Semantics-Enhanced Pre-Training: Can Lexicon Definitions Help Learning Sentence Meanings? Appendix

Xuancheng Ren,[1] Xu Sun,[1,2*] Houfeng Wang,[1] Qun Liu[3]

[1] MOE Key Laboratory of Computational Linguistics, School of EECS, Peking University
[2] Center for Data Science, Peking University
[3] Huawei Noah's Ark Lab
{renxc, xusun, wanghf}@pku.edu.cn, qun.liu@huawei.com

Experiment Details

Semantics-Enhanced Pre-Training

Data Collection The training data is collected from the WordNet (Miller 1995) using the interface provided by the natural language toolkit (NLTK) (Bird, Loper, and Klein 2009) Python package and the version of the WordNet data should be 3.0. As our training instances consist of a pair of two semantic related words and their definitions, we do the following the transform the WordNet data originally organized in synsets:

1. Each synset is deconstructed into the words (or lemmas, as called in the NLTK interface) it contains. Each pair of the words in a synset is linked by a new type of synonymy relation.
2. The semantic relations in the WordNet defined on lemmas or words are kept and we resolve the semantic relations defined on synsets to words, and especially, we exclude the also see relation. This gives us 22 types of semantic relations and 1.4M word pairs, i.e., training instances.

The numbers of training instances in terms of semantic relations are shown in Table 1. The data is not split further into validation set and training set to achieve the maximum gain on data coverage.

Data Format Since the sentences are pre-processed with GPT2 BPE, a word may be split into multiple tokens, e.g., `d aff od ils`. Considering this, in addition to the special `[D]` we used to separate the word and its definition, we further surround the word with another special token `[R]`. The final data format of the spring-season example is like this:

> `[C] [R] [M] [R] [D]` the season of growth `[E]` `[S] [R]` season `[R] [D]` one of the natural `[M]` into which ... `[E]`

Note that each word-definition sequence is ended with a special token `[E]`, which is the practice of RoBERTa. In fairseq, `[S]` and `[E]` use the same actual symbol </s>. The masking practice follows Liu et al. (2019), except that we prevent the special tokens from being masked. We tried to mask only the content words not the functional words, but the influence is unclear because for the models that are finally used, each training instance is only seen once and the masking procedure may need more training steps to show effects.

Relation	#Instances
hyponym	329,086
hypernym	329,079
synonym	315,984
member holonym	60,458
member meronym	60,458
derivationally related form	55,296
similar to	50,736
part meronym	37,843
part holonym	37,843
instance hyponym	34,848
instance hypernym	34,848
topic domain	22,645
region domain	8,582
verb group	8,156
antonym	7,977
pertainym	7,746
usage domain	4,504
attribute	3,402
substance meronym	2,673
substance holonym	2,673
entailment	2,336
cause	1,038
Total	1,418,211

Table 1: Statistics of training data

Implementation We use the fairseq (Ott et al. 2019) package to implement our approach, since our chosen baseline is RoBERTa, and fairseq provides the functionality to train or fine-tune such models in multi-GPU settings. RoBERTa uses the BPE method from GPT2 (Radford et al. 2019), which we keep to allow the reuse of the model checkpoints pro-

[*] Corresponding Author

vided by Liu et al. (2019). We generally follow the original training procedure of RoBERTa, and the following is some notable considerations or differences:

- The batch size is set to 2048, distributed on 4 GPUs, and we keep the examples of at most 128 tokens, excluding 3,345 examples from training. For RoBERTa-base, we use the 11GB NVIDIA GeForce RTX 2080 Ti, while for RoBERTa-large, we use the 24GB NVIDIA Titan RTX.
- Using the batch size, each epoch contains 691 training steps. Following RoBERTa, we use a learning rate scheduler with linear warmup and linear decay. The learning rate will peak at the 295 training step and decay to 0 at the 6,910 training step, which is equal to 10 epochs. The peak learning rate is set to 2×10^{-5} and 5×10^{-5} for SemPre on RoBERTa-base and RoBERTa-large, respectively. Those values are based on the recommendation from the GitHub repository of RoBERTa [1] and our observation on the training loss.
- The Adam optimizer (Kingma and Ba 2015) is used for RoBERTa-base with $\beta_1 = 0.9$, $\beta_2 = 0.98$, and $\epsilon = 10^{-6}$. The LAMB optimizer (You et al. 2020) is used for RoBERTa-large with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-6}$, which we find is more stable in training. The optimizers are with decoupled weight decay of 0.01.
- Compared to the masked language pre-training used in RoBERTa, the only extra computation comes from the relation classification objective, which is almost negligible in parameter increase. Moreover, the introduced parameters are not needed in downstream evaluation, effectively meaning no extra cost. The number of parameters is 356M for RoBERTa-large with SemPre (compare to 355M for vanilla RoBERTa-large) and each epoch of RoBERTa-large with SemPre costs about 3.5 hours.
- We conduct a single trial for each model with a fixed random seed. The best model is selected based on validation loss on a part of the combined corpus of BookCorpus and Wikipedia used in Devlin et al. (2019), considering that the general language understanding ability of the pre-trained models should not be impaired. Please note that, once selected, the model is used in all further evaluations.

Due to various costs, we performed very little tuning for hyper-parameters in our preliminary experiments. The found hyper-parameters are directly used for training RoBERTa-base and -large with SemPre. It is possible that more hyper-parameter tuning may show better results.

Word Games

Data Collection This dataset is build by our own. The words are from the Oxford 3000 list[2], which is a curate list containing words that are fundamental with high frequency and most relevant to English learners. There are 3,275 word after filtering the functional words. We build two version of the dataset: one with the definitions in WordNet (WG1) and one with the definition from the Oxford Advanced Learner's Dictionary (WG2). In WG1, since the words in Oxford 3000 do not have sense information and only come with the part-of-speech tags, we retrieve the top-2 synsets of a words and use their definitions. In WG2, we use the accompanied definitions.

Templates
“ [word] means [definition]”
“ the definition of [word] is [definition]”
“ [word] : [definition]”
“ [word] is [definition]” *(for nouns only)*
“ [word] are [definition]”*(for nouns only)*
Form variations
original form, e.g., “`apple`”
original form + prepended space, e.g., “` apple`”
capitalized, e.g., “`Apple`”
capitalized + prepended space, e.g., “` Apple`”

Table 2: Templates and form variations for constructing test sentences in Word Games

Data Format The word game task aims to make the model predict the word by its definition. However, there are some limitations of the models in practice, if a relatively fair evaluation is pursued.

1. The vanilla RoBERTa is trained on syntactically correct sentences, which means it may not work properly on the data format we used in SemPre, and some words that have prominent occurrence in certain positions, e.g., `it` at the sentence start, will be preferred by the model. Ideally, we should manually make up a sentence for each word and its definition because it is hard to automatically generate a fluent and sensible sentence using the word and its definition. However, it can be labor-intensive. To deal with this problem, we construct multiple templates in the hope that a fluent and sensible sentence can by covered.
2. GPT2 BPE is case-sensitive and whitespace-sensitive, which means a common word can also be split if the form is not suitable, and it is hard for the RoBERTa models to predict multiple masked tokens at the same time. To mitigate the effect, we also try to vary the form of the word when we fill it in the template.

The two considerations, as detailed in Table 2, mean that a word-definition pair will correspond to multiple test sentences in the word games. The sentences where the target word is not split are further processed: the target word in those sentences are replaced with `[M]`, the sentences are prepended with `[C]` and appended with `[E]`, and the model ranks the masked word, with 0 meaning perfect rank. The best result among the sentences is reported for this specific word-definition pair.

[1] https://github.com/pytorch/fairseq/tree/v0.9.0/examples/roberta

[2] https://www.oxfordlearnersdictionaries.com/wordlists/oxford3000-5000

If the relation is	and the unseen word is	then replace it with
hypernym	head	hyponyms
hypernym	tail	hypernyms
hyponym	head	hypernyms
hyponym	tail	hyponyms

Table 3: Replacement rules for data expansion in selected model training for word games

Evaluation Metric The evaluation metric for this task is mean rank per definition per word. Conceptually speaking, the calculation is done as

$$\text{mean rank} = \mathop{\mathbb{E}}_{\text{word}} \mathop{\mathbb{E}}_{\text{definition}} \min_{\text{test sentences}} \text{rank}. \tag{1}$$

Model Training Since SemPre makes use of definitions in WordNet, it may gain favor on WG1. To alleviate this issue, we train specific models on only parts of the WordNet lexicon, for both of the two word game tasks.

Partial WordNet Specifically, we random select 1000 words from the Oxford 3000 list as the *unseen* words and exclude them from training, which means the model never sees the WordNet definitions of those words. However, as the unseen words are all common words, the training data is reduced dramatically if all relations containing the words are removed. To compensate this, we refine some types of the relations containing one unseen word by replacing the unseen word with its hypernyms or hyponyms. The rules are listed in Table 3. For example, `24-karat gold` and `gold` have the hypernym relation. However, `gold` should be an unseen word, which is also the tail of the relation. Then, the relation is expanded with the hypernyms of `gold`, e.g., `noble metal` as `24-karat gold` and `noble metal` have the hypernym relation. It should be noted that a relation may be expanded to multiple relations if there are multiple hypernyms or hyponyms to the unseen word. After expansion, the training data have 924,405 word pairs, which are about 65% of the original data.

Implementation We continue to use the batch size of 2048, which gives us 452 batches for an epoch. For RoBERTa-large, the sequences are truncated to at most 256 tokens, while for RoBERTa-base, no sequences are dropped or truncated. The common fine-tuning hyper-parameters are used because the models are only used for WGs and we do none hyper-parameter tuning. For both the base and the large version, we use the Adam optimizer with linear warmup and linear decay scheduled for 20 epochs, i.e., 9040 total training steps, peaked at the 904 training step, i.e., 10% of the total training steps, with a peak learning rate of 2×10^{-5}. The training is conducted on 4 NVIDIA GeForce RTX 2080 Ti GPUs with mixed precision and costs 0.2 hour and 2.5 hour per epoch for the base and the large model, respectively.

WG1	All	Seen	Unseen
RoBERTa-base	591	966	405
RoBERTa-base + SemPre	110	145	93
RoBERTa-large	331	537	229
RoBERTa-large + SemPre	85	117	69

Table 4: Fine-grained results on WG1

Dataset	#Instances	#Choices
Token-Level		
SM	1,877	2
WSC	283	2
CA	183	2
Sentence-Level		
SWAG	1,001	4
HellaSwag	1,000	4
SMR	2,021	3
ARCT1	444	2
ARCT2	888	2

Table 5: Statistics of CAT data

Detailed Results In the main paper, we show the WG results on all the words from the Oxford 3000 list. For WG2, all the definitions are not seen in SemPre training. However, for WG1, we can further calculate the mean rank on the unseen words and the seen words. The results are shown in Table 4. As we can see, the random selected unseen words are actually easier to guess by definition, comparing the seen words and the unseen words. Nonetheless, SemPre also significantly improves the results of the words whose definitions are not seen in proposed semantic-focused pretraining. It supports our notion that SemPre works by show the model a new way to interpret the knowledge existing but not adequately expressed in the general-purpose pretraining, which figuratively speaking, is similar to the idea that to teach someone how to fish is better than to just give someone a fish.

Please see the supplemented data for data and result examples.

Commonsense Ability Tests

Data Collection We use the data provided by Zhou et al. (2020) at their GitHub repository[3], which is composed of 8 datasets. The datasets are (1) Sense Making (SM) (Wang et al. 2019c), which "tests whether a model can differentiate sense-making and non-sense-making statements", (2) Winograd Schema Challenge (WSC) (Levesque, Davis, and Morgenstern 2012), which is "recognized as one of the most difficult commonsense datasets", (3) Conjunction Acceptability (CA), which is extracted from WSC and paired with manual negative sample by Zhou et al. (2020), (4) SWAG

[3] https://github.com/XuhuiZhou/CATS

(Zellers et al. 2018), which "questions model's understanding towards the relationship between two physical scenes", (5) HellaSwag (Zellers et al. 2019), which is "an augmented version of SWAG" and Zhou et al. (2020) chose "only the instances coming from ActivityNet to make the results comparable to the original SWAG dataset", (6) Sense Making with Reasoning (SMR) (Wang et al. 2019c), which "focuses on identifying the reason behind a statment", and (7) Argument Reasoning Comprehension Task (ARCT) (Habernal et al. 2018), which is "to test a model's abductive reasoning ability" and is further materialized into two variants, where ARCT1 represents the original dataset by Habernal et al. (2018) and ARCT2 represents the augmented version by Niven and Kao (2019) to alleviate the statistical clues. The first three datasets have token-level differences among the positive and the negative examples, while the rest of the datasets have sentence-level differences. The statistics of the datsets are summarized in Table 5. We make a small change to the provided HellaSwag data and remove the `[SEP]` strings from the sentences, which seems to be a mistake by the original authors.

Data Format The sequences are presented to the model after BPE and `[C]` prepended to the start and `[E]` appended to the end.

Evaluation Metric The evaluation metric is accuracy. For an instance, there is one positive sentence and one or more negative sentences. If the sentence score of the positive sentence is better than others, the instance is recognized as correct.

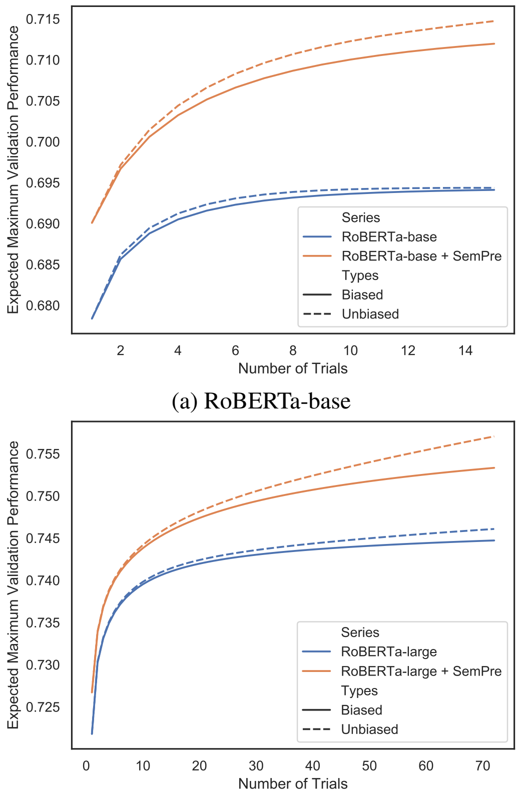
Implementation The CATs are based on sentence scores. The conventional sentence scores for language models are perplexity or equivalently negative log likelihood, which is the lower the better. However, as the masked language models can be bi-directional, which is not suitable for the conventional scores, Zhou et al. (2020) proposed to use the following calculation:

$$\text{Score}(S) = \frac{1}{n} \sum\nolimits_{k=1}^{n} \log\left(P_\theta(w_k|c_k)\right), \qquad (2)$$

where S is a sequence, n is the length of the sequence, w_k is the k-th token in the sequence, c_k is the sequence with the k-th token masked, and $\log\left(P_\theta(\cdot)\right)$ is the log likelihood with the parameter θ. The score is the higher the better. We re-implement the calculation to improve the efficiency and the results should be identical to the implementation by Zhou et al. (2020). However, the scores in the original paper is not fully reproducible, which is acknowledged by Zhou et al. (2020) in their GitHub repository. Nonetheless, the results in this paper are comparable because the implementation for testing the baseline and SemPre is the same.

Word-in-Context

Data Collection We use the data provided by Wang et al. (2019a), which has the same content with Pilehvar and Camacho-Collados (2019) but has a different format. It has 5,428 training instances, 638 validation instances, and 1,400 test instances.



(a) RoBERTa-base

(b) RoBERTa-large

Figure 1: Expected maximum validation performance on WiC as a function of the number of hyper-parameter trials.

Data Format Each instance has two sentences or fragments, which we concatenated with special tokens. For example, the validation instance of the word `thing` is transformed as

`[C]` A thing of the spirit. `[E]` `[S]` Things of the heart. `[E]`

Note, sentences are processed by BPE, which is not reflected in the example. We truncate the sequence to at most 128 tokens.

Evaluation Metric The evaluation metric is accuracy.

Implementation Following Wang et al. (2019a); Liu et al. (2019), the concatenation of the representations of the `[C]` token and the two target words are used for predicting the label. If the target word is split into multiple tokens, the average of the tokens are used for the target word. The classification network is the same as the one used in RoBERTa, containing a hidden layer and an output layer. Especially, SemPre does not cause differences in fine-tuning compared to the RoBERTa baseline. The fine-tuning procedure follows Liu et al. (2019) using the Adam optimizer with linear warmup and linear decay peaked at 10% of the total training steps. The batch size is set to 32. We did a basic hyper-parameter search following RoBERTa:

- Learning rate: $[1 \times 10^{-5}, 2 \times 10^{-5}, 3 \times 10^{-5}]$
- Max epochs: $[10, 50]$

Each configuration is run with different random start for 12 trials for the large model and 5 trials for the small model on a single NVIDIA GeForce 1080 Ti GPU with early stopping on validation accuracy. Note that for the base model, we didn't try max epochs of 50, because the difference is already obvious after tuning learning rates. The model with the best validation accuracy is submitted to online testing and the corresponding results are reported in the paper. The best configurations are

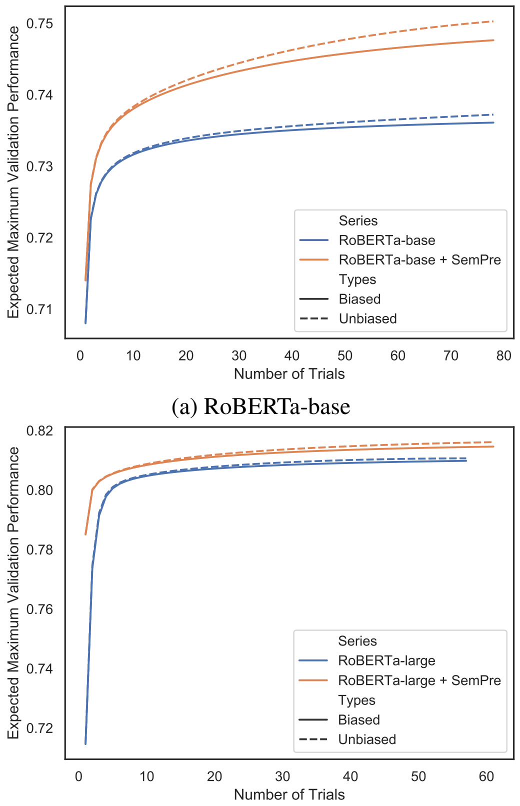
- RoBERTa-base: learning rate as 3×10^{-5} and 10 max epochs
- RoBERTa-base + SemPre: learning rate as 3×10^{-5} and 10 max epochs
- RoBERTa-large: learning rate as 1×10^{-5} and 50 max epochs
- RoBERTa-large + SemPre: learning rate as 1×10^{-5} and 50 max epochs

Expected Validation Performance The expected maximum validation performance as a function of the number of hyper-parameter trials are shown in Table 1. Due to the scientific debate about the calculation and the efficacy of such way of reporting (Tang et al. 2020), we adopt two calculation methods, i.e., the biased version in Dodge et al. (2019), which under-estimates the maximum performance when $n > 1$, and the unbiased version in Tang et al. (2020), which reaches the observed maximum performance when $n = N$, where N is the number of total trials. Since SemPre makes no difference in the fine-tuning procedure, which means the training time should be the same with the baseline, we use the number of trials as the x-axis, equivalent to training time budget. As we can see, the advantage of the proposed SemPre is consistent with respect to the number of hyper-parameter search trials. For RoBERTa-base, the reported result almost reaches the best performance, while for RoBERTa-base with SemPre, it seems that the performance can be further improved with more hyper-parameter trials. For RoBERTa-large, if the growth trend is reliable, SemPre might generate even better results that those reported in this paper.

Physical Interaction: Question Answering

Data Collection We use the data provided by Bisk et al. (2020). It has 16,113 training instances, 1,838 validation instances, and 3,084 test instances.

Data Format Each instance has a goal sentence and two solution sentences. We concatenate each solution sentence to the goal sentence with special tokens. For example, the validation instance of the goal *a shelf* is transformed as



(a) RoBERTa-base

(b) RoBERTa-large

Figure 2: Expected maximum validation performance on PIQA as a function of the number of hyper-parameter trials.

`[C]` a shelf `[E]` can hold a book `[E]`

and

`[C]` a shelf `[E]` can hold milk `[E]`

Note, sentences are processed by BPE, which is not reflected in the example. We exclude instances that have sentences more than 128 tokens, that is, 189 training instances are excluded.

Evaluation Metric The evaluation metric is accuracy.

Implementation Following Bisk et al. (2020); Liu et al. (2019), we use a two-way classification objective, that is, the representation of the `[C]` token in each goal-solution sequence is passed to a neural network to generate a score and two scores for the two sequences in a training instances are normalized using softmax, representing the possibility of each sequence being the correct choice. Then, the we use the cross-entropy loss to fine-tune the model. The classification network is the same as the one used in RoBERTa, containing a hidden layer and an output layer. The rest is the same with the fine-tuning of WiC. The batch size is set to 32. We did a basic hyper-parameter search following RoBERTa:

- Learning rate: $[1 \times 10^{-5}, 2 \times 10^{-5}, 3 \times 10^{-5}]$

Model (Metric)	CoLA (MCC)	SST-2 (Acc)	MRPC (F1/Acc)	STS-B (PCC/SCC)	QQP (F1/Acc)	MNLI-m (Acc)	MNLI-mm (Acc)	QNLI (Acc)	RTE (Acc)	WNLI (Acc)	AX (MCC)	GLUE -
Validation Results												
RoBERTa-base (ours)	64.9	95.6	91.2/93.5	91.3/91.0	92.0/89.2	87.8	-	93.0	79.4	-	-	-
RoBERTa-base + SemPre	63.8	95.4	90.7/93.3	91.3/91.1	92.0/89.3	87.7	-	92.9	80.5	-	-	-
Test Results												
RoBERTa-base (ours)	59.1	95.6	90.1/86.7	89.7/89.0	72.4/89.5	87.5	87.3	49.3	72.4	65.1	40.8	76.4
RoBERTa-base + SemPre	60.0	95.2	91.5/88.3	89.6/88.9	72.6/89.5	87.5	87.2	49.4	71.6	65.1	41.4	76.5

Table 6: Results on GLUE benchmark datasets. MCC denotes Matthew's Correlation Coefficient; Acc denotes accuracy; PCC denotes Pearson's Correlation Coefficient; and SCC denotes Spearman's Correlation Coefficient. The GLUE score is calculated without AX, which is the diagnostics dataset using models fine-tuned on MNLI.

Model	Word	Def.	Pair	CA	WSC	SM	SMR	SWAG	HellaSwag	ARCT1	ARCT2	Average
(a) RoBERTa-large	○	○	○	96.2	69.3	79.2	47.3	**76.1**	**48.9**	54.3	60.0	66.4
(b) + Def.	○	●	○	**96.7**	70.7	80.4	48.3	73.8	46.1	59.2	60.2	66.9
(c) + Word Pair	●	○	●	95.1	72.4	75.1	46.1	72.5	45.2	52.7	55.5	64.3
(d) + Word-Def.	●	●	○	96.2	71.7	80.6	48.3	75.0	48.4	59.2	61.4	67.6
(e) + SemPre	●	●	●	**96.7**	73.5	80.4	**48.4**	75.9	48.5	58.3	60.9	**67.8**
(f) - Word Mask	∗	●	●	**96.7**	72.4	80.8	47.8	74.5	47.8	56.8	59.9	67.1
(g) - Def. Mask	●	∗	●	**96.7**	**74.6**	77.9	**48.4**	75.5	46.9	58.8	60.5	67.4
(h) - Rel. Pred.	●	●	∗	96.2	72.4	**81.4**	48.0	75.5	46.9	**59.5**	**61.5**	67.7

Table 7: Results of the ablation study on CATs. ○, ∗, and ● denote the respective information is not used, used only as inputs, and learned with an objective.

- Max epochs: $[10, 50]$

Each configuration is run with different random start for 12 trials on a single NVIDIA GeForce 2080 Ti GPU with mixed precision training and early stopping on validation accuracy. The model with the best validation accuracy is reported in the paper. The best configurations are

- RoBERTa-base: learning rate as 2×10^{-5} and 50 max epochs
- RoBERTa-base + SemPre: learning rate as 3×10^{-5} and 50 max epochs
- RoBERTa-large: learning rate as 1×10^{-5} and 50 max epochs
- RoBERTa-large + SemPre: learning rate as 1×10^{-5} and 50 max epochs

For online testing, we submitted only the best model, since it has a public leaderboard and some submission requirements.

Expected Validation Performance The expected maximum validation performance as a function of the number of hyper-parameter trials are shown in Table 2. As we can see, on the PIQA task, the advantage of the proposed SemPre is also consistent with respect to the number of hyper-parameter search trials. For RoBERTa-base, SemPre might experience more improvement with more search trials, while for RoBERTa-large, it seems that both models have reached the maximum.

Detailed GLUE Results

We used the data provided by Wang et al. (2019b) and preprocessed the data as Liu et al. (2019). The fine-tuning procedure and hyper-parameter search space also follows Liu et al. (2019). We did limited fine-tuning using only the RoBERT-base model and each hyper-parameter configuration was run twice or thrice with different random start. The model with the best valuation accuracy is reported and its results are submitted to online testing. The WNLI (Levesque, Davis, and Morgenstern 2012; White et al. 2017; Wang et al. 2019b) results in testing are not produced by the fine-tuned models but the baseline results provided by Wang et al. (2019b). The detailed results are shown in Table 6.

As we can see, the averaged GLUE scores do not show substantial differences and the differences on individual tasks are mixed. SemPre seems to improve the test performance of CoLA (Warstadt, Singh, and Bowman 2019), MRPC (Dolan and Brockett 2005), QQP (Iyer, Dandekar, and Csernai 2017), and QNLI (Rajpurkar et al. 2016; White et al. 2017; Demszky, Guu, and Liang 2018; Wang et al. 2019b) but not the validation performance, which could mean that SemPre entails slightly better generalization performance for certain tasks in GLUE. Particularly, while the results on MNLI (Williams, Nangia, and Bowman 2018; Bowman et al. 2015; Wang et al. 2019b) are very similar, the results on AX (Wang et al. 2019b) are quite different, although they all use the same models fine-tuned on MNLI. The results suggest that the ability enhanced by SemPre is not fully reflected by the tasks in the GLUE benchmark datasets and thus, we do not regard the GLUE datasets as

Model	CA	WSC	SM	SMR	SWAG	HellaSwag	ARCT1	ARCT2	WiC	Avg
RoBERTa-base	95.6	62.5	75.0	40.0	**69.1**	41.3	50.0	53.7	69.4	65.2
+ Word-Def. (n. only)	96.7	62.9	76.8	39.2	68.5	42.0	54.3	56.3	72.4	67.2
+ Word-Def. (v. only)	**97.8**	60.4	77.0	40.3	67.3	**42.8**	53.6	55.9	71.6	66.7
+ Word-Def. (adj. and adv.)	**97.8**	62.2	77.5	**40.5**	67.8	41.6	**56.3**	57.2	**73.2**	**67.9**
+ Word-Def.	96.7	62.2	76.9	38.7	68.6	41.7	54.7	**56.4**	70.5	66.2
Model	CA	WSC	SM	SMR	SWAG	HellaSwag	ARCT1	ARCT2	WiC	Avg
RoBERTa-large	96.2	69.3	79.2	47.3	**76.1**	**48.9**	54.3	60.0	74.6	70.5
+ Word-Def. (n. only)	94.5	72.0	80.4	47.4	74.4	48.6	**59.7**	61.6	73.7	70.5
+ Word-Def. (v. only)	**97.8**	70.7	**82.2**	49.8	74.4	48.1	59.5	**62.3**	73.0	70.6
+ Word-Def. (adj. and adv.)	96.2	71.4	81.0	**50.1**	74.6	48.1	59.2	**62.3**	73.5	70.7
+ Word-Def.	96.2	71.7	80.6	48.3	75.0	48.4	59.2	61.4	**74.8**	**71.2**

Table 8: Results of ablation study in terms of part-of-speech tags.

a proper evaluation of the proposed approach or as the main results of the paper.

Ablation Study Details

Implementation For SemPre without propagating through relations, the pre-training procedure uses a different learning rate schedule compared to the complete version of SemPre, i.e., using the Adam optimizer with linear warmup and linear decay scheduled for 10 epochs peaked at 10% of the total training steps. The final models are taken for evaluation. The models are trained on a single GPU. The rest is kept the same with the complete version of SemPre. As some of the ablated SemPre models are trained much longer, i.e., each training instances are seen ten times compared to once in the full version of SemPre, it is possible that some specific word definitions are more sufficiently trained so that for zero-shot evaluation longer SemPre training may be in favor.

CATs Results The full results are shown on Table 7. The average is the marco average, which means the small datasets, particularly CA, are relatively considered more and take more portion of the final score. As we can see, the models trained with word-definitions, including model (d), (e), (f), (g), and (h), perform generally well on CATs. For the ablated model with word-definition pairs but without relation prediction (h), its average performance is promoted substantially with ARCT1 and ARCT2. It conforms to the trend that models with paired inputs show limited or no improvements over those without paired inputs, suggesting that the paired input and the relation prediction objective may hinder the learning of the knowledge required by the argument reasoning comprehension tasks.

Ablation in Terms of POS Tags Part-of-speech (POS) tags are representative distributional characteristics of words and by training only with words of a certain POS tag, we may gain a better understanding of how general pre-training prioritizes the learning of words with different distributional properties, that is, if applying SemPre with a certain POS tag induces significant improvements, such type of words are not well modeled in general pre-training. After categorization, there are 146k nouns, 24k verbs, and 36k adjectives and adverbs, which roughly describe things, actions, and states, respectively. For this ablation in terms of training data, since relations can cross word groups, we train without using paired word definitions and the results are reported in Table 8. While the improvements are in line with the semantic knowledge tested by the datasets, e.g., noun definitions benefit WSC, overall, training with adjectives and adverbs bring the most improvements. It indicates that adjectives and adverbs are hard to comprehend purely based on contexts, which is actually observed previously in learning static word embeddings like word2vec. As an example, the contexts in “this is a cold spring”, “this is a hot spring”, and “this is a warm spring” are not very indicative of the meaning of “cold”, “hot”, or “warm”. Even though it is believed that longer contexts may mitigate the problem, it seems that the pre-trained language models with much longer contexts still suffer from this phenomenon. On a deeper note, common adjectives and adverbs describe things related to the physical world and purely learning their meaning based on texts may not be the most efficient way. From the results, we can also see that for a balanced improvement to pre-trained models, all word types should be taken into account and definitions of all word types contribute to the model performance.

References

Bird, S.; Loper, E.; and Klein, E. 2009. *Natural Language Processing with Python*. O’Reilly Media Inc.

Bisk, Y.; Zellers, R.; LeBras, R.; Gao, J.; and Choi, Y. 2020. PIQA: Reasoning about Physical Commonsense in Natural Language. In *AAAI*, 7432–7439.

Bowman, S. R.; Angeli, G.; Potts, C.; and Manning, C. D. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*, 632–642.

Demszky, D.; Guu, K.; and Liang, P. 2018. Transforming Question Answering Datasets Into Natural Language Inference Datasets. *CoRR* abs/1809.02922.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT (1)*, 4171–4186.

Dodge, J.; Gururangan, S.; Card, D.; Schwartz, R.; and Smith, N. A. 2019. Show Your Work: Improved Reporting of Experimental Results. In *EMNLP-IJCNLP*, 2185–2194.

Dolan, W. B.; and Brockett, C. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *IWP@IJCNLP*.

Habernal, I.; Wachsmuth, H.; Gurevych, I.; and Stein, B. 2018. The Argument Reasoning Comprehension Task: Identification and Reconstruction of Implicit Warrants. In *NAACL-HLT (1)*, 1930–1940.

Iyer, S.; Dandekar, N.; and Csernai, K. 2017. First Quora Dataset Release: Question Pairs. Online; last accessed 2020-06-01.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

Levesque, H. J.; Davis, E.; and Morgenstern, L. 2012. The Winograd Schema Challenge. In *KR*.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692.

Miller, G. A. 1995. WordNet: A Lexical Database for English. *Commun. ACM* 38(11): 39–41.

Niven, T.; and Kao, H. 2019. Probing Neural Network Comprehension of Natural Language Arguments. In *ACL (1)*, 4658–4664.

Ott, M.; Edunov, S.; Baevski, A.; Fan, A.; Gross, S.; Ng, N.; Grangier, D.; and Auli, M. 2019. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In *NAACL-HLT (Demonstrations)*, 48–53.

Pilehvar, M. T.; and Camacho-Collados, J. 2019. WiC: The Word-in-Context Dataset for Evaluating Context-Sensitive Meaning Representations. In *NAACL-HLT (1)*, 1267–1273.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language Models are Unsupervised Multitask Learners. Technical report, OpenAI.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. SQuAD: 100, 000+ Questions for Machine Comprehension of Text. In *EMNLP*, 2383–2392.

Tang, R.; Lee, J.; Xin, J.; Liu, X.; Yu, Y.; and Lin, J. 2020. Showing Your Work Doesn't Always Work. In *ACL*, 2766–2772.

Wang, A.; Pruksachatkun, Y.; Nangia, N.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2019a. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *NeurIPS*, 3261–3275.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2019b. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *ICLR*.

Wang, C.; Liang, S.; Zhang, Y.; Li, X.; and Gao, T. 2019c. Does it Make Sense? And Why? A Pilot Study for Sense Making and Explanation. In *ACL (1)*, 4020–4026.

Warstadt, A.; Singh, A.; and Bowman, S. R. 2019. Neural Network Acceptability Judgments. *Trans. Assoc. Comput. Linguistics* 7: 625–641.

White, A. S.; Rastogi, P.; Duh, K.; and Durme, B. V. 2017. Inference is Everything: Recasting Semantic Resources into a Unified Evaluation Framework. In *IJCNLP (1)*, 996–1005.

Williams, A.; Nangia, N.; and Bowman, S. R. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *NAACL-HLT (1)*, 1112–1122.

You, Y.; Li, J.; Reddi, S. J.; Hseu, J.; Kumar, S.; Bhojanapalli, S.; Song, X.; Demmel, J.; Keutzer, K.; and Hsieh, C. 2020. Large Batch Optimization for Deep Learning: Training BERT in 76 minutes. In *ICLR*.

Zellers, R.; Bisk, Y.; Schwartz, R.; and Choi, Y. 2018. SWAG: A Large-Scale Adversarial Dataset for Grounded Commonsense Inference. In *EMNLP*, 93–104.

Zellers, R.; Holtzman, A.; Bisk, Y.; Farhadi, A.; and Choi, Y. 2019. HellaSwag: Can a Machine Really Finish Your Sentence? In *ACL (1)*, 4791–4800.

Zhou, X.; Zhang, Y.; Cui, L.; and Huang, D. 2020. Evaluating Commonsense in Pre-Trained Language Models. In *AAAI*, 9733–9740.