# Pre-training Universal Language Representation

**Yian Li[1,2,3], Hai Zhao[1,2,3,*]**

[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2] Key Laboratory of Shanghai Education Commission for Intelligent Interaction
and Cognitive Engineering, Shanghai Jiao Tong University, Shanghai, China
[3] MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
liya19@sjtu.edu.cn,zhaohai@cs.sjtu.edu.cn

## Abstract

Despite the well-developed cut-edge representation learning for language, most language representation models usually focus on specific levels of linguistic units. This work introduces universal language representation learning, i.e., embeddings of different levels of linguistic units or text with quite diverse lengths in a uniform vector space. We propose the training objective MiSAD that utilizes meaningful $n$-grams extracted from large unlabeled corpus by a simple but effective algorithm for pre-trained language models. Then we empirically verify that well designed pre-training scheme may effectively yield universal language representation, which will bring great convenience when handling multiple layers of linguistic objects in a unified way. Especially, our model achieves the highest accuracy on analogy tasks in different language levels and significantly improves the performance on downstream tasks in the GLUE benchmark and a question answering dataset.

## 1 Introduction

In this paper, we propose universal language representation (ULR) that uniformly embeds linguistic units in different hierarchies in the same vector space. A universal language representation model encodes linguistic units such as words, phrases or sentences into fixed-sized vectors and handles multiple layers of linguistic objects in a unified way. ULR learning may offer a great convenience when confronted with sequences of different lengths, especially in tasks such as Natural Language Understanding (NLU) and Question Answering (QA),

hence it is of great importance in both scientific research and industrial applications.

As is well known, embedding representation for a certain linguistic unit (i.e., word) enables linguistics-meaningful arithmetic calculation among different vectors, also known as word analogy (Mikolov et al., 2013). For example:

$$King - Man = Queen - Woman$$

In fact, manipulating embeddings in the vector space reveals syntactic and semantic relations between the original symbol sequences and this feature is indeed useful in true applications. For example, "*London is the capital of England*" can be formulized as:

$$England + capital \approx London$$

Then given two documents one of which contains "*England*" and "*capital*", the other contains "*London*", we consider them relevant. While a ULR model may generalize such good analogy features onto free text with all language levels involved together. For example, *Eat an onion* : *Vegetable* :: *Eat a pear* : *Fruit*.

ULR has practical values in dialogue systems, by which human-computer communication will go far beyond executing instructions. One of the main challenges of dialogue systems is dialogue state tracking (DST). It can be formulated as a semantic parsing task (Cheng et al., 2020), namely, converting natural language utterances with any length into unified representations. Thus this is essentially a problem that can be conveniently solved by mapping sequences with similar semantic meanings into similar representations in the same vector space according to a ULR model.

Another use of ULR is in the Frequently Asked Questions (FAQ) retrieval task, where the goal is to answer a user's question by retrieving question paraphrases that already have an answer from

the database. Such task can be accurately done by only manipulating vectors such as calculating and ranking vector distance (i.e., cosine similarity). The core is to embed sequences of different lengths in the same vector space. Then a ULR model retrieves the correct question-answer pair for the user query according to vector distance.

In this paper, we propose a universal language representation learning method that generates fixed-sized vectors for sequences of different lengths based on pre-trained language models (Devlin et al., 2019; Lan et al., 2019; Clark et al., 2020). We first introduce an efficient approach to extract and prune meaningful $n$-grams from unlabeled corpus. Then we present a new pre-training objective, Minimizing Symbol-vector Algorithmic Difference (MiSAD), that explicitly applies a penalty over different levels of linguistic units if their representations tend not to be in the same vector space.

To investigate our model's ability of capturing different levels of language information, we introduce an original universal analogy task derived from Google's word analogy dataset, where our model significantly improves the performance of previous pre-trained language models. Evaluation on a wide range of downstream tasks also demonstrates the effectiveness of our ULR model. Overall, our ULR-BERT obtains the highest average accuracy on our universal analogy dataset and reaches 1.1% gain over Google BERT on the GLUE benchmark. Extensive experimental results on a question answering task verifies that our model can be easily applied to real-world applications in an extremely convenient way.

## 2 Related Work

Previous language representation learning methods such as Word2Vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), LASER (Artetxe and Schwenk, 2019), InferSent (Conneau et al., 2017) and USE (Cer et al., 2018) focus on specific granular linguistic units, e.g., words or sentences. Later proposed ELMo (Peters et al., 2018), OpenAI GPT (Radford et al., 2018), BERT (Devlin et al., 2019) and XLNet (Yang et al., 2020) learns contextualized representation for each input token. Although such pre-trained language models (PrLMs) more or less are capable of offering universal language representation through their general-purpose

training objectives, all the PrLMs devote into the contextualized representations from a generic text background and pay little attention on our concerned universal language presentation.

As a typical PrLM, BERT is trained on a large amount of unlabeled data including two training targets: Masked Language Model (MLM), and Next Sentence Prediction (NSP). ALBERT (Lan et al., 2019) is trained with Sentence-Order Prediction (SOP) as a replacement of NSP. StructBERT (Wang et al., 2020) combines NSP and SOP to learn inter-sentence structural information. Nevertheless, RoBERTa (Liu et al., 2019) and SpanBERT (Joshi et al., 2020) show that single-sequence training is better than the sentence-pair scenario. Besides, BERT-wwm (Cui et al., 2019), StructBERT (Joshi et al., 2020), SpanBERT (Wang et al., 2020) perform MLM on higher linguistic levels, augmenting the MLM objective by masking whole words, trigrams or spans, respectively. ELECTRA (Clark et al., 2020) further improves pre-training through a generator and discriminator architecture. The aforementioned models may seemingly handle different sized input sequences, but all of them focus on sentence-level specific representation still for each word, which may cause unsatisfactory performance in real-world situations.

There are a series of downstream NLP tasks especially on question-answering which may be conveniently and effectively solved through ULR like solution. Actually, though in different forms, these tasks more and more tend to be solved by our suggested ULR model, including dialogue utterance regularization (Cao et al., 2020), question paraphrasing (Bonadiman et al., 2019), measuring QA similarities in FAQ tasks (Damani et al., 2020; Sakata et al., 2019).

## 3 Model

As pre-trained contextualized language models show their powerfulness in generic language representation for various downstream NLP tasks, we present a BERT-style ULR model that is especially designed to effectively learn universal, fixed-sized representations for input sequences of any granularity, i.e., words, phrases, and sentences. Our proposed pre-training method is furthermore strengthened in three-fold. First, we extract a large number of meaningful $n$-grams from monolingual corpus based on point-wise mutual information to lever-

age the multi-granular structural information. Second, inspired by word and phrase representation and their compositionality, we introduce a novel pre-training objective that directly models the extracted $n$-grams through manipulating their representations. Finally, we implement a normalized score for each $n$-gram to guide their sampling for training.

### 3.1 $n$-gram Extracting

Given a symbol sentence, Joshi et al. (2020) utilize span-level information by randomly masking and predicting contiguous segments. Different from such random sampling strategy, our method is based on point-wise mutual information (PMI) (Church and Hanks, 1989) that makes efficient use of statistics and automatically extracts meaningful $n$-grams from unlabeled corpus.

Mutual information (MI) describes the association between two tokens by comparing the probability of observing them together with the probabilities of observing them independently. Higher mutual information indicates stronger association between the tokens. To be specific, an $n$-gram is denoted as $w = (x_1, \ldots, x_{|w|})$, where $|w|$ is the number of tokens in $w$ and $n > 1$. Therefore, we present an extended PMI formula displayed as below:

$$PMI(w) = \frac{1}{|w|}\left(\log P(w) - \sum_{k=1}^{|w|}\log P(x_k)\right)$$

where the probabilities are estimated by counting the number of observations of each token and $n$-gram in the corpus, and normalizing by the corpus size. $\frac{1}{|w|}$ is an additional normalization factor which avoids extremely low scores for long $n$-grams.

We first collect all $n$-grams with lengths up to $N$ using the SRILM toolkit[1] (Stolcke, 2002), and compute PMI scores for all the $n$-grams based on their occurrences. Then, only $n$-grams with PMI scores higher than the chosen threshold are selected and input sequences are marked with the corresponding $n$-grams.

### 3.2 Training Objective

While the MLM training objective as in BERT (Devlin et al., 2019) and its extensions (Cui et al., 2019; Joshi et al., 2020; Wang et al., 2020) are

---

[1]http://www.speech.sri.com/projects/srilm/download.html

widely used for pre-trained contextualized language modeling, they do not focus on our concerned ULR, which demands an arithmetic corresponding relationship between the symbol and its represented vector. In order to directly model such demand, we propose a novel training target – Minimizing Symbol-vector Algorithmic Difference (MiSAD) – that leverages the vector space regularity of different granular linguistic units. For example, the following symbol sequence equation

$$\begin{aligned}&\text{``London is''} + \text{``the capital of England''}\\&=\text{``London is the capital of England''}\end{aligned} \quad (1)$$

indicates a vector algorithmic equation according to our ULR goal,

$$\begin{aligned}&vector(\text{``London is''}) + vector(\text{``the capital of}\\&\quad England\text{''})\\&=vector(\text{``London is the capital of England''})\end{aligned}$$
$$(2)$$

Thus, if the symbol equation (1) cannot imply the respective vector equation (2), we may set a training objective to let the ULR model forcedly learn such relationship.

Formally, we denote the input sequence by $S = \{x_1, \ldots, x_m\}$, where $m$ is the number of tokens in $S$. After $n$-gram extracting and pruning by means of PMI, each sequence is marked with several $n$-grams. During pre-training, only one of them is selected by the $n$-gram scoring function, which will be introduced in detail in Section 3.3, and the input sequence is represented as $S = \{x_1, \ldots, x_{i-1}, w, x_{j+1}, \ldots, x_m\}$, where the $n$-gram $w = \{x_i, \ldots, x_j\}$ $(1 \leq i < j \leq m)$ is a sub-sequence of $S$. Then we convert $S$ into two independent parts – the $n$-gram $w$ and the rest of the tokens $R = \{x_1, \ldots, x_{i-1}, x_{j+1}, \ldots, x_m\}$ – which are fed into the model separately along with the original complete sequence.

The Transformer encoder generates a contextualized representation for each token in the sequence. To derive fixed-sized vectors for sequences of different lengths, we use the pooled output of the [CLS] token as sequence embeddings. The model is trained to minimize the following Mean Square Error (MSE) loss:

$$\mathcal{L}_{MiSAD} = MSE(E^w + E^R, E^S)$$

where $E^w$, $E^R$ and $E^S$ are representations of $w$, $R$ and $S$, respectively, and are all normalized to unit

lengths. To enhance the robustness of the model, we jointly train MiSAD and the MLM objective $\mathcal{L}_{MLM}$ as in BERT with equal weights. Since the input sentence $S$ is split into $w + R$, we must avoid masking out the $n$-gram $w$ in the original sentence in order not to affect the semantics after vector space combination. However, tokens in $n$-grams other than $w$ have equal weights of being replaced with [MASK] as other tokens. The final loss function is as follows:

$$\mathcal{L} = \mathcal{L}_{MiSAD} + \mathcal{L}_{MLM}$$

### 3.3 $n$-gram Sampling

For a given sequence, the importance of different $n$-grams and the degree to which the model understands their semantics are different. Instead of sampling $n$-grams at random, we let the model decide which $n$-gram to choose based on the knowledge learned in the pre-training stage. Following Tamborrino et al. (2020), we employ a normalized score for each $n$-gram in the input sequence using the masked language modeling head.

We mask one $n$-gram at a time and the model outputs probabilities of the masked tokens given their surrounding context. The score of an $n$-gram $w$ is calculated as the average probabilities of all tokens in it.

$$score_w = \frac{1}{|w|} \sum_{k=1}^{|w|} P(x_k | S^{\backslash w})$$

where $|w|$ is the length of $w$ and $S^{\backslash w}$ is the notation of an input sequence $S$ with each token within the $n$-gram $w$ replaced by the special token [MASK]. Finally, we choose the $n$-gram with the lowest score for our training target.

## 4   Implementation of ULR Pre-training

This section introduces our ULR pre-training details.

As for the pre-training corpus, we download the English Wikipedia Corpus[2] and pre-process with process_wiki.py[3], which extracts text from xml files. When processing paragraphs from Wikipedia, we find that a large number of entities are annotated with special marks, which may be useful for our task. Therefore, we identify all

the entities and treat them as high-quality $n$-grams. Then, we remove punctuation marks and characters in other languages based on regular expressions, and finally get a corpus of 2,266M words.

As for $n$-gram pruning, PMI scores of all $n$-grams with a maximum length of $N = 6$ are calculated for each document. We manually evaluate the extracted $n$-grams and find more than 50% of the top 2000 $n$-grams contain $2 \sim 3$ words, and only less than 3% $n$-grams are longer than 4. Although a larger $n$-gram vocabulary can cover longer $n$-grams, it will cause too many meaningless $n$-grams at the same time. Therefore, we empirically retain the top 3000 $n$-grams for each document. Finally, we randomly sample 10M sentences from the entire corpus to reduce training time.

During pre-training, BERT packs sentence pairs into a single sequence and use the special [CLS] token as sentence-pair representation. However, our MiSAD training objective requires single-sentence inputs. Thus in our experiments, each input is an $n$-ngram or a single sequence with a maximum length of 128. Special tokens [CLS] and [SEP] are added at the front and end of each input, respectively. Instead of training from scratch, we initialize our model with the officially released checkpoints of BERT (Devlin et al., 2019), ALBERT (Lan et al., 2019) and ELEC-TRA (Clark et al., 2020). We use Adam optimizer (Kingma and Ba, 2017) with initial learning rate of 5e-5 and linear warmup over the first 10% of the training steps. Batch size is 64 and dropout rate is 0.1. Each model is trained for one epoch over 10M training examples on four Nvidia Tesla P40 GPUs.

To derive fixed-dimensional vectors of input sequences, we apply three pooling strategies on top of the PrLM: Using the vector of the [CLS] token, mean-pooling of all token embeddings and max-pooling over time of all embeddings. The default setting is mean-pooling.

## 5   Experimental Setup

### 5.1   Tasks

We construct a universal analogy dataset in terms of words, phrases and sentences and experiment with multiple representation models to examine their ability of representing different levels of linguistic units through a task-independent evalua-

---

| girl | − | boy | + | brother | = | *daughter* | ***sister*** | *wife* | *father* | *son* |
|------|---|-----|---|---------|---|-----------|-----------|--------|----------|-------|
| worse | − | bad | + | big | = | ***bigger*** | *larger* | *smaller* | *biggest* | *better* |
| China | − | Beijing | + | Paris | = | ***France*** | *Europe* | *Germany* | *Belgium* | *London* |
| Chilean | − | Chile | + | China | = | *Japanese* | ***Chinese*** | *Russian* | *Korean* | *Ukrainian* |

Table 1: Examples from our word analogy dataset. The correct answers are in bold.

tion[4]. Furthermore, we conduct experiments on a wide range of downstream tasks from the GLUE benchmark and a question answering task.

### 5.1.1 Universal Analogy

Our universal analogy dataset is based on Google's word analogy dataset and contains three levels of tasks: words, phrases and sentences.

**Word-level** Recall that in a word analogy task (Mikolov et al., 2013), two pairs of words that share the same type of relationship, denoted as $A : B :: C : D$, are involved. The goal is to retrieve the last word from the vocabulary given the first three words. To facilitate comparison between models with different vocabularies, we construct a closed-vocabulary analogy task based on Google's word analogy dataset through negative sampling. Concretely, for each original question, we use GloVe to rank every word in the vocabulary and the top 5 results are considered to be candidate words. If GloVe fails to retrieve the correct answer, we manually add it to make sure it is included in the candidates. During evaluation, the model is expected to select the correct answer from 5 candidate words. Table 1 shows examples from our word anlogy dataset.

**Phrase-/Sentence-level** To derive higher level analogy datasets, we put word pairs from the word-level dataset into contexts so that the resulting phrase and sentence pairs also have linear relationships. Phrase and sentence templates are extrated from the English Wikipedia Corpus. Both phrase and sentence datasets have four types of semantic analogy and three kinds of syntactic analogy. Please refer to Appendix A for details about our approach of constructing the universal analogy dataset.

### 5.1.2 GLUE

The General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018) is a collection of tasks that are widely used to evaluate the performance of a model in language understand-

ing. We divide NLU tasks from the GLUE benchmark into three main categories.

**Single-Sentence Classification** Single-sentence classification tasks includes SST-2 (Socher et al., 2013), a sentiment classification task, and CoLA (Warstadt et al., 2019), a task that is to determine whether a sentence is grammatically acceptable.

**Natural Language Inference** GLUE contains four NLI tasks: MNLI (Williams et al., 2018), QNIL (Rajpurkar et al., 2016), RTE (Bentivogli et al., 2009) and WNLI (Levesque et al., 2012). However, we exclude the problematic WNLI in accordance with Devlin et al. (2019).

**Semantic Similarity** MRPC (Dolan and Brockett, 2005), QQP (Chen et al., 2018) and STS-B (Cer et al., 2017) are semantic similarity tasks, where the model is required to either determine whether the two sentences are equivalent or assign a similarity score for them.

In the fine-tuning stage, pairs of sentences are concatenated into a single sequence with a special token [SEP] in between. For both single sentence and sentence pair tasks, the hidden state of the first token [CLS] is used for softmax classification. We use the same sets of hyperparameters for all the evaluated models. Experiments are ran with batch sizes in {8, 16, 32, 64} and learning rate of 3e-5 for 3 epochs.

### 5.1.3 GEOGRANNO

GEOGRANNO (Herzig and Berant, 2019) contains natural language paraphrases paired with logical forms. The dataset is manually annotated: For each natural language utterance, a correct canonical utterance paraphrase is selected. The train/dev sets have 487 and 59 paraphrase pairs, respectively. In our experiments, we focus on question paraphrase retrieval, whose task is to retrieve the correct paraphrase from all 158 different sentences when given a question. Most of the queries have only one correct answer while some have two or more matches. Evaluation metrics are Top-1/5/10 accuracy.

---

[4]Code and dataset are available at: https://github.com/Liyianan/ULR.

| Model | Word | | | Phrase | | | Sentence | | | Avg. | Gain |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | sem | syn | Avg. | sem | syn | Avg. | sem | syn | Avg. | | |
| *Word & Sentence Representation Models* | | | | | | | | | | | |
| GloVe | **82.6** | 78.0 | **80.3** | 0.0 | 40.9 | 20.5 | 0.2 | 39.8 | 20.0 | 40.3 | - |
| InferSent | 68.8 | **88.7** | 78.8 | 0.0 | 54.1 | 27.0 | 0.0 | 50.8 | 25.4 | 43.7 | - |
| GenSen | 44.5 | 84.4 | 64.5 | 0.0 | 54.4 | 27.2 | 0.0 | 44.9 | 22.4 | 38.0 | - |
| USE | 73.0 | 83.1 | 78.0 | 1.8 | 63.1 | 32.5 | 0.6 | 44.1 | 22.4 | 44.3 | - |
| LASER | 26.9 | 78.2 | 52.6 | 0.0 | 63.3 | 31.7 | 1.6 | 55.4 | 28.5 | 37.6 | - |
| *Pre-trained Contextualized Language Models* | | | | | | | | | | | |
| BERT$_{BASE}$ | 51.3 | 60.2 | 55.8 | 0.3 | **69.3** | **34.8** | 0.1 | **68.3** | **34.2** | 41.6 | - |
| BERT$_{LARGE}$ | 49.7 | 46.6 | 48.2 | 0.1 | 67.4 | 33.9 | 0.5 | 61.2 | 30.9 | 37.7 | - |
| ALBERT$_{BASE}$ | 33.7 | 38.1 | 35.9 | 0.1 | 53.6 | 26.7 | 0.1 | 60.9 | 30.5 | 31.0 | - |
| ALBERT$_{XXLARGE}$ | 38.2 | 35.6 | 36.9 | 0.8 | 52.3 | 26.6 | 0.4 | 49.4 | 24.9 | 29.5 | - |
| ELECTRA$_{BASE}$ | 22.9 | 32.4 | 27.7 | 2.2 | 57.1 | 29.7 | 0.4 | 39.5 | 20.0 | 25.8 | - |
| ELECTRA$_{LARGE}$ | 20.4 | 24.7 | 22.6 | 2.9 | 49.8 | 26.4 | 1.4 | 52.0 | 26.7 | 25.2 | - |
| *Our Universal Language Representation Models* | | | | | | | | | | | |
| ULR-BERT$_{BASE}$ | 71.7 | 70.0 | 70.8 | 1.1 | 66.8 | 34.0 | 1.5 | 63.0 | 32.3 | 45.7 | 4.1 |
| ULR-BERT$_{LARGE}$ | 80.8 | 66.2 | 73.5 | **8.4** | 60.5 | 34.5 | **4.7** | 54.3 | 29.5 | **45.8** | **8.1** |
| ULR-ALBERT$_{BASE}$ | 43.5 | 56.3 | 49.9 | 0.3 | 58.2 | 29.3 | 0.3 | 60.9 | 30.6 | 36.6 | 5.6 |
| ULR-ALBERT$_{XXLARGE}$ | 26.8 | 31.0 | 28.9 | 3.6 | 55.0 | 29.3 | 0.7 | 60.3 | 30.5 | 29.6 | 0.1 |
| ULR-ELECTRA$_{BASE}$ | 24.4 | 34.6 | 29.5 | 1.7 | 56.5 | 29.1 | 0.9 | 57.6 | 29.3 | 29.3 | 3.5 |
| ULR-ELECTRA$_{LARGE}$ | 22.0 | 31.0 | 26.5 | 2.9 | 56.7 | 29.8 | 0.8 | 52.9 | 26.9 | 27.7 | 2.5 |

Table 2: Performance of different models on the universal analogy dataset. "sem" = semantic. "syn" = syntactic.

## 5.2 Baselines

On the universal analogy task, we adopt three types of baselines including bag-of-words (BoW) model from pre-trained word embeddings: GloVe (Pennington et al., 2014), sentence embedding models: InferSent (Conneau et al., 2017), GenSen (Subramanian et al., 2018), USE (Cer et al., 2018) and LASER (Artetxe and Schwenk, 2019), and pre-trained contextualized language models: BERT, ALBERT and ELECTRA.

On GLUE and GEOGRANNO, we especially evaluate our model and two baseline models:

**BERT** The officially released pre-trained BERT models (Devlin et al., 2019).

**MLM-BERT** BERT models trained with the same additional steps with our model on Wikipedia using only the MLM objective.

**ULR-BERT** Our universal language representation model trained on Wikipedia with MLM and MiSAD.

## 6 Results

### 6.1 Universal Analogy

Results on our universal analogy dataset are reported in Table 2. Generally, semantic analogies are more challenging than the syntactic ones and

higher-level relationships between sequences are more difficult to capture, which is observed in almost all the evaluated models. On the word analogy task, GloVe achieves the highest accuracy (80.3%) while its performance drops sharply on higher-level tasks. All well trained PrLMs like BERT, ALBERT and ELECTRA hardly exhibit arithmetic characteristics and increasing the model size usually leads to a decrease in accuracy.

However, training models with our properly designed MiSAD objective greatly improves the performance, especially in word-level analogy. Especially, ULR-BERT obtains 15% ∼ 25% absolute gains, such results are so strong to be comparable to GloVe, which especially focuses on the linear word analogy feature from its training scheme. Meanwhile GloVe performs far worse than our model on higher-level analogies. Overall, ULR-BERT achieves the highest average accuracy (45.8%), an absolute gain of 8.1% over BERT, indicating that it has indeed more effectively learned universal language representations across different linguistic units. It demonstrates that our pre-training method is effective and can be adapted to different PrLMs.

### 6.2 GLUE

Table 3 shows the performance on the GLUE benchmark. Our model improves the BERT$_{BASE}$ and BERT$_{LARGE}$ by 1.1% and 0.7% on average,

| Model | Single Sentence | | Natural Language Inference | | | Semantic Similarity | | | Avg. | Gain |
|---|---|---|---|---|---|---|---|---|---|---|
| | CoLA (mc) | SST-2 (acc) | MNLI m/mm(acc) | QNLI (acc) | RTE (acc) | MRPC (F1) | QQP (F1) | STS-B (pc) | | |
| *In literature* | | | | | | | | | | |
| BERT$_{BASE}$ | 52.1 | 93.5 | 84.6/83.4 | 90.5 | 66.4 | 88.9 | 71.2 | 87.1 | 79.7 | - |
| BERT$_{LARGE}$ | 60.5 | 94.9 | **86.7**/85.9 | 92.7 | 70.1 | 89.3 | 72.1 | 87.6 | 82.2 | - |
| *Our implementation* | | | | | | | | | | |
| BERT$_{BASE}$ | 53.5 | 92.1 | 84.5/83.7 | 90.6 | 67.1 | 87.5 | 71.6 | 85.3 | 79.5 | - |
| MLM-BERT$_{BASE}$ | 51.9 | 94.0 | 84.5/83.9 | 90.4 | 66.6 | 88.1 | 71.6 | 86.2 | 79.7 | 0.2 |
| ULR-BERT$_{BASE}$ | 56.5 | 94.3 | 84.6/84.0 | 91.0 | 68.0 | 89.0 | 71.6 | 86.6 | 80.6 | 1.1 |
| BERT$_{LARGE}$ | 60.5 | 94.9 | 86.1/85.6 | 92.8 | 68.8 | 89.6 | 72.1 | 87.3 | 82.0 | - |
| MLM-BERT$_{LARGE}$ | **62.6** | 94.5 | 86.6/85.6 | 92.8 | 67.1 | 88.9 | **72.3** | 87.2 | 82.0 | 0 |
| ULR-BERT$_{LARGE}$ | 61.8 | **95.0** | **86.7/86.0** | **93.0** | **71.0** | **90.2** | **72.3** | **88.2** | **82.7** | 0.7 |

Table 3: Test results on the GLUE benchmark scored by the evaluation server[5]. We exclude the problematic WNLI dataset and recalculate the "Avg." score. Results for BERT$_{BASE}$ and BERT$_{LARGE}$ are obtained from Devlin et al. (2019). "mc" and "pc" are Matthews correlation coefficient (Matthews, 1975) and Pearson correlation coefficient, respectively.

| Model | Top-1 | Top-5 | Top-10 |
|---|---|---|---|
| GloVe | 0.3 | 2.7 | 7.4 |
| LASER | 6.3 | 9.5 | 12.7 |
| BM25 | 27.1 | 62.5 | 76.4 |
| BERT$_{BASE}$ | 29.6 | 58.9 | 67.1 |
| MLM-BERT$_{BASE}$ | 37.0 | 66.8 | 72.6 |
| ULR-BERT$_{BASE}$ | **39.7** | 66.0 | **77.3** |
| BERT$_{LARGE}$ | 15.9 | 42.7 | 54.2 |
| MLM-BERT$_{LARGE}$ | 24.5 | 57.8 | 70.7 |
| ULR-BERT$_{LARGE}$ | 35.1 | **68.8** | **77.3** |

Table 4: Question paraphrase retrieval accuracy of different models on the train-dev set of GEOGRANNO.

respectively. Since our model is established on the released checkpoints of Google BERT, we make additional comparison with MLM-BERT that is trained under the same procedure as our model except for the pre-training objective. While the model trained with more MLM updates may improve the performance on some tasks, it underperforms BERT on datasets such as MRPC, RTE and SST-2. Our model exceeds MLM-BERT$_{BASE}$ and MLM-BERT$_{LARGE}$ by 0.9% and 0.7% on average respectively. The main gains from the base model are in CoLA (+4.6%) and RTE (+1.4%), which are entirely contributed by our MiSAD training objective. Overall, our model improves the performance of its baseline on every dataset in the GLUE benchmark, demonstrating its effectiveness in real applications of natural language understanding.

### 6.3 GEOGRANNO

Table 4 shows the performance on GEOGRANNO. As we can see, 4 out of 6 evaluated pre-

trained language models significantly outperform BM25 for Top-1 accuracy, indicating the superiority of embedding-based models over the statistical method. Among all the evaluated models, our ULR-BERT yields the highest accuracies (39.7%/68.8%/77.3%). To be specific, our ULR model exceeds BERT by 10.1% and 19.2% Top-1 accuracy and obtains 2.7% and 10.6% improvements compared with MLM-BERT, which are consistent with the results on the GLUE benchmark. Since $n$-grams and sentences of different lengths are involved in the pre-training of our model, it is especially better at understanding the semantics of input sequences and mapping queries to their paraphrases according to the learned sense of semantic equality.

## 7 Ablation Study

In this section, we explore to what extent does our model benefit from the MiSAD objective and sampling strategy, and further confirm that our pre-training procedure improves the model's ability of encoding variable-length sequences.

### 7.1 Effect of Training Objectives

To make a fair comparison, we train BERT with the same additional updates using different combinations of training tasks:

**NSP-BERT** is trained with MLM and NSP, whose goal is to distinguish whether two input sentences are consecutive. For each sentence, we choose its following sentence 50% of the time and randomly sample a sentence 50% of the time.

| Model | Single Sentence | | Natural Language Inference | | | Semantic Similarity | | | Avg. | Gain |
|-------|-----------------|-|---------------------------|-|-|---------------------|-|-|------|------|
| | CoLA (mc) | SST-2 (acc) | MNLI m/mm(acc) | QNLI (acc) | RTE (acc) | MRPC (F1) | QQP (F1) | STS-B (pc) | | |
| BERT | 53.5 | 92.1 | 84.5/83.7 | 90.6 | 67.1 | 87.5 | 71.6 | 85.3 | 79.5 | - |
| MLM-BERT | 51.9 | 94.0 | 84.5/83.9 | 90.4 | 66.6 | 88.1 | 71.6 | 86.2 | 79.7 | 0.2 |
| NSP-BERT | 53.5 | 93.2 | 84.1/83.5 | 90.5 | 66.1 | 87.7 | **72.1** | 84.5 | 79.5 | 0 |
| SOP-BERT | 50.9 | 92.7 | 84.0/83.1 | 90.7 | 66.5 | 85.0 | 70.9 | 83.9 | 78.6 | -0.9 |
| ULR-BERT | **56.5** | **94.3** | **84.6/84.0** | **91.0** | **68.0** | **89.0** | 71.6 | **86.6** | **80.6** | 1.1 |

Table 5: Comparison of base models using different training objectives on the GLUE test set.

| Model | CoLA | | | RTE | | | MRPC | | |
|-------|------|-|-|-----|-|-|------|-|-|
| | std. | mean | max | std. | mean | max | std. | mean | max |
| BERT | 1.51 | 57.0 | 58.3 | 1.92 | 68.1 | 70.4 | 0.52 | 90.4 | 90.9 |
| ULR-BERT | 1.31 | 59.3 | 60.2 | 1.83 | 69.3 | 72.6 | 0.61 | 90.8 | 91.5 |

Table 6: Standard deviation, mean, and maximum performance on the GLUE dev set when fintuing BERT and ULR-BERT with 5 random seeds.

**SOP-BERT** is trained with MLM and SOP, a substitute of the NSP task that aims at better modeling the coherence between sentences. Consistent with Lan et al. (2019), we sample two consecutive sentences in the same document as a positive sample, and reverse their order 50% of the time to create a negative sample.

For both baselines and ULR, we use the same set of parameters for 5 runs, and average scores on the GLUE test set are reported in Table 5. Although we expect NSP and SOP to help the model better understand the relationship between sentences and benefit tasks like natural language inference, they hardly improve the performance on GLUE according to our strict implementation. Specifically, NSP-BERT outperforms MLM-BERT on datasets such as CoLA, QNLI and QQP while less satisfactory on other tasks. SOP-BERT is on a par with MLM-BERT on three NLI tasks but it sharply decreases the score on other datasets. In general, single-sentence training with only the MLM objective accounts for better performance as described by Lan et al. (2019); Liu et al. (2019). Besides, our training strategy which combines MLM and MiSAD yields the most considerable gains compared with other training objectives.

Table 6 shows standard deviation, mean and maximum performance on CoLA/RTE/MRPC dev set when fine-tuning BERT and ULR-BERT over 5 random seeds, which clearly shows that our model is generally more stable and yields better results compared with BERT.

## 7.2 Effect of Sampling Strategies

We compare our PMI-based $n$-gram sampling scheme with two alternatives. Specifically, we train the following two baseline models under the same model settings except for the sampling strategy.

**Random Spans** We replace our $n$-gram module with the masking strategy as proposed by Joshi et al. (2020), where the sampling probability of span length $l$ is based on a geometric distribution $l \sim Geo(p)$. The parameter $p$ is set to 0.2 and maximum span length $l_{max} = 6$.

**Named Entities** We only retain named entities that are annotated in the Wikipedia Corpus.

Table 7 shows the effect of different sampling schemes on the GLUE dev set. As we can see, our PMI-based $n$-gram sampling is preferable to other strategies on 6 out of 8 tasks. CoLA and RTE are more sensible to sampling strategies than other tasks. On average, using named entities and meaningful $n$-grams is better than randomly sampled spans. We attribute the source to the reason is that random span sampling ignores important semantic and syntactic structure of a sequence, resulting in a large number of meaningless segments. Compared with using only named entities, our PMI-based approach automatically discovers structures within any sequence and is not limited to any granularity, which is critical to pre-training universal language representation.

## 7.3 Application to Different Models

Experiments on the universal analogy task reveal that our proposed training scheme can be adapted

| Model | Single Sentence | | Natural Language Inference | | | Semantic Similarity | | | Avg. |
|---|---|---|---|---|---|---|---|---|---|
| | CoLA (mc) | SST-2 (acc) | MNLI m/mm(acc) | QNLI (acc) | RTE (acc) | MRPC (F1) | QQP (F1) | STS-B (pc) | |
| Random Spans | 56.1 | 93.1 | 84.5/**84.9** | 91.5 | 66.1 | **91.5** | **87.9** | 89.8 | 82.8 |
| Named Entities | 57.1 | 93.1 | 84.4/84.7 | 91.6 | 67.5 | 90.8 | **87.9** | **89.9** | 83.0 |
| PMI $n$-grams | **59.3** | **93.6** | **84.7**/84.9 | **91.8** | **69.3** | 90.8 | 87.8 | **89.9** | **83.6** |

Table 7: Comparison of base models using different sampling strategies on the GLUE dev set.

| Model | GEOGRANNO | GLUE dev |
|---|---|---|
| BERT | 29.6/58.9/67.1 | 82.6 |
| ALBERT | 18.4/41.1/52.6 | 83.0 |
| ELECTRA | 11.2/21.1/26.6 | 86.5 |
| ULR-BERT | 39.7/66.0/77.3 | 83.6 |
| ULR-ALBERT | 24.9/44.7/55.9 | 83.4 |
| ULR-ELECTRA | 26.8/51.8/65.5 | 86.9 |

Table 8: Comparison of different base models on GEOGRANNO and GLUE. We report the Top-1/5/10 accuracy on GEOGRANNO.

| *Group by query length* $|q|$ | | | |
|---|---|---|---|
| **Model** | 1~6 (32.6%) | 7~8 (36.7%) | 9~15 (30.7%) |
| BERT | 73.9 | 64.9 | 62.5 |
| ULR-BERT | 79.8 | 76.1 | 75.9 |
| | +5.9 | +11.2 | +13.4 |
| *Group by* $abs(|q| - |Q|)$ | | | |
| **Model** | ≥0 (100%) | ≥2 (62.2%) | ≥3 (43.3%) |
| BERT | 67.1 | 63.0 | 57.0 |
| ULR-BERT | 77.3 | 76.2 | 70.3 |
| | +10.2 | +13.2 | +13.3 |

Table 9: Comparison of Top-10 accuracy of BERT and ULR-BERT on different subsets of GEOGRANNO.

to various pre-trained langauge models. In this subsection, we compare our model with BERT, ALBERT and ELECTRA on GEOGRANNO and the GLUE benchmark.

Table 8 shows the results on GEOGRANNO and the GLUE dev set, where our approach can enhance the performance of all three pre-trained models. Among all the evaluated models, ULR-BERT achieves the largest gains on GLUE while ULR-ELECTRA obtains the most significant improvement on GEOGRANNO. It further verifies the effectiveness and universality of our model.

### 7.4 Model Universality and Representation Consistency

In previous evaluations on GEOGRANNO, our model has shown considerable improvement (10.2% Top-10 Acc.) over BERT$_{BASE}$. The task

involves text matching between linguistic units at different levels where queries are sentences and labels are often phrases. Thus the performance on such task highly depends on the model's ability to uniformly deal with linguistic units of different granularities.

In the following, we explore deeper details and interpretability of how our proposed objective act at different levels of linguistic units. Specifically, we group the dataset according to query length $|q|$ and the absolute difference between query length and Question length $abs(|q| - |Q|)$, respectively, to evaluate the universality of ULR-BERT and the consistency of the learned representations.

Results are shown in Table 9. As the length of the query increases, the performance of BERT drops sharply. Similarly, BERT is more sensible to the difference between query length and question length. In contrast, ULR-BERT is more stable when dealing with sequences of different lengths, which we speculate is due to the interaction between different levels of linguistic units in the pretraining procedure.

## 8 Conclusion

This work formally introduces universal language representation learning to enable unified vector operations among different language hierarchies. For such a purpose, we propose three highlighted ULR learning enhancement, including the newly designed training objective, Minimizing Symbolvector Algorithmic Difference (MiSAD). In detailed model implementation, we extend BERT's pre-training objective to a more general level, which leverages information from sequences of different lengths in a comprehensive way. In addition, we provide a universal analogy dataset as a task-independent evaluation benchmark. Overall experimental results show that our proposed ULR model is generally effective in a broad range of NLP tasks including natural language question answering and so on.

# References

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Luisa Bentivogli, Peter Clark, Dagan Ido, and Giampiccolo Danilo. 2009. The fifth pascal recognizing textual entailment challenge. In *Proceedings of TAC*.

Daniele Bonadiman, Anjishnu Kumar, and Arpit Mittal. 2019. Large scale question paraphrase retrieval with smoothed deep metric learning. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 68–75.

Ruisheng Cao, Su Zhu, Chenyu Yang, Chen Liu, Rao Ma, Yanbin Zhao, Lu Chen, and Kai Yu. 2020. Unsupervised dual paraphrasing for two-stage semantic parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6806–6817.

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.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174.

Zihan Chen, Hongbo Zhang, Xiaoji Zhang, and Leqi Zhao. 2018. Quora question pairs.

Jianpeng Cheng, Devang Agrawal, Héctor Martínez Alonso, Shruti Bhargava, Joris Driesen, Federico Flego, Dain Kaplan, Dimitri Kartsaklis, Lin Li, Dhivya Piraviperumal, Jason D. Williams, Hong Yu, Diarmuid Ó Séaghdha, and Anders Johannsen. 2020. Conversational semantic parsing for dialog state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8107–8117.

Kenneth Ward Church and Patrick Hanks. 1989. Word association norms, mutual information, and lexicography. In *27th Annual Meeting of the Association for Computational Linguistics*, pages 76–83.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for Chinese BERT.

Sonam Damani, Kedhar Nath Narahari, Ankush Chatterjee, Manish Gupta, and Puneet Agrawal. 2020. Optimized transformer models for FAQ answering. In *PAKDD 2020*, Lecture Notes in Computer Science.

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.

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

Jonathan Herzig and Jonathan Berant. 2019. Don't paraphrase, detect! rapid and effective data collection for semantic parsing. 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 3810–3820.

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.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

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.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. KR'12, page 552–561.

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.

B.W. Matthews. 1975. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442 – 451.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/languageunsupervised/language understanding paper.pdf*.

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.

Wataru Sakata, Tomohide Shibata, Ribeka Tanaka, and Sadao Kurohashi. 2019. FAQ retrieval using query-question similarity and BERT-based query-answer relevance. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1113–1116.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew 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.

Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *Seventh international conference on spoken language processing*.

Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher J Pal. 2018. Learning general purpose distributed sentence representations via large scale multi-task learning. In *International Conference on Learning Representations*.

Alexandre Tamborrino, Nicola Pellicanò, Baptiste Pannier, Pascal Voitot, and Louise Naudin. 2020. Pre-training is (almost) all you need: An application to commonsense reasoning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3878–3887.

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. 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.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2020. XLNet: Generalized autoregressive pretraining for language understanding.

# A  Universal Analogy

As a new task, universal representation has to be evaluated in a multiple-granular analogy dataset. In this section, we introduce the procedure of constructing different levels of analogy datasets based on Google's word analogy dataset.

## A.1  Word-level analogy

The goal of a word analogy task is to solve questions like "$A$ is to $B$ as $C$ is to ?", which is to retrieve the last word from the vocabulary given the first three words. The objective can be formulated as maximizing the cosine similarity between the target word embedding and the linear combination of the given vectors:

$$d^* = \arg\max_{d^*} cosine(c + b - a, d)$$
$$cosine(u, v) = \frac{u \cdot v}{\|u\|\|v\|}$$

where $a$, $b$, $c$, $d$ represent embeddings of the corresponding words and are all normalized to unit lengths.

We construct a closed-vocabulary analogy task based on Google's word analogy dataset through negative sampling. During evaluation, the model

| Dataset | #p | #q | #c | #l (p/s) |
|---|---|---|---|---|
| capital-common | 23 | 506 | 5 | 6.0/12.0 |
| capital-world | 116 | 4524 | 5 | 6.0/12.0 |
| city-state | 67 | 2467 | 5 | 6.0/12.0 |
| male-female | 23 | 506 | 5 | 4.1/10.1 |
| present-participle | 33 | 1056 | 2 | 4.8/8.8 |
| positive-comparative | 37 | 1322 | 2 | 3.4/6.1 |
| positive-negative | 29 | 812 | 2 | 4.4/9.2 |
| All | 328 | 11193 | - | 5.4/10.7 |

Table 10: Statistics of our analogy datasets. #p and #q are the number of pairs and questions for each category. #c is the number of candidates for each dataset. #l (p/s) is the average sequence length in phrase/sentence-level analogy datasets.

is expected to select the correct answer from 5 candidate words.

## A.2 Phrase/Sentence-level analogy

To investigate the arithmetic properties of vectors for higher levels of linguistic units, we present phrase and sentence analogy tasks based on the proposed word analogy dataset. Statistics are shown in Table 10.

### A.2.1 Semantic

Semantic analogies can be divided into four subsets: "capital-common", "capital-world", "city-state" and "male-female". The first two sets can be merged into a larger dataset: "capital-country", which contains pairs of countries and their capital cities; the third involves states and their cities; the last one contains pairs with gender relations. Considering GloVe's poor performance on word-level "country-currency" questions ($<$32%), we discard this subset in phrase and sentence-level analogies. Then we put words into contexts so that the resulting phrases and sentences also have linear relationships. For example, based on relationship

*Athens* : *Greece* :: *Baghdad* : *Iraq*,

we select phrases and sentences that contain the word "*Athens*" from the English Wikipedia Corpus. We manually modify some words to ensure text coherence: "*He was hired as being professor of physics by the university of Athens.*" and create examples:

*hired by ... Athens* : *hired by ... Greece* :: *hired by ... Baghdad* : *hired by ... Iraq*.

However, we found that such a question is identical to word-level analogy for BOW methods like averaging GloVe vectors, because they treat embeddings independently despite the content and word order. To avoid lexical overlapping between sequences, we replace certain words and phrases with their synonyms and paraphrases, e.g.,

*hired by ... Athens* : *employed by ... Greece* :: *employed by ... Baghdad* : *hired by ... Iraq*.

### A.2.2 Syntactic

We consider three typical syntactic analogies: Tense, Comparative and Negation, corresponding to three subsets: "present-participle", "positive-comparative", "positive-negative", where the model needs to distinguish the correct answer from "past tense", "superlative" and "positive", respectively. For example, given phrases

*Pigs are bright* : *Pigs are brighter than goats* :: *The train is slow*,

the model need to give higher similarity score to the sentence that contains "*slower*" than the one that contains "*slowest*". Similarly, we add synonyms and synonymous phrases for each question to evaluate the model ability of learning context-aware embeddings rather than interpreting each word in the question independently. For instance, "*pleasant*" $\approx$ "*not unpleasant*" and "*unpleasant*" $\approx$ "*not pleasant*".