Distilling Relation Embeddings from Pre-trained Language Models

Asahi Ushio and **Jose Camacho-Collados** and **Steven Schockaert**
Cardiff NLP, School of Computer Science and Informatics
Cardiff University, United Kingdom
{UshioA,CamachoColladosJ,SchockaertS1}@cardiff.ac.uk

Abstract

Pre-trained language models have been found to capture a surprisingly rich amount of lexical knowledge, ranging from commonsense properties of everyday concepts to detailed factual knowledge about named entities. Among others, this makes it possible to distill high-quality word vectors from pre-trained language models. However, it is currently unclear to what extent it is possible to distill *relation embeddings*, i.e. vectors that characterize the relationship between two words. Such relation embeddings are appealing because they can, in principle, encode relational knowledge in a more fine-grained way than is possible with knowledge graphs. To obtain relation embeddings from a pre-trained language model, we encode word pairs using a (manually or automatically generated) prompt, and we fine-tune the language model such that relationally similar word pairs yield similar output vectors. We find that the resulting relation embeddings are highly competitive on analogy (unsupervised) and relation classification (supervised) benchmarks, even without any task-specific fine-tuning.[1]

1 Introduction

One of the most widely studied aspects of word embeddings is the fact that word vector differences capture lexical relations (Mikolov et al., 2013a). While not being directly connected to downstream performance on NLP tasks, this ability of word embeddings is nonetheless important. For instance, understanding lexical relations is an important prerequisite for understanding the meaning of compound nouns (Turney, 2012). Moreover, the ability of word vectors to capture semantic relations has enabled a wide range of applications beyond NLP, including flexible querying of relational databases (Bordawekar and Shmueli, 2017), schema matching (Fernandez et al., 2018), completion and retrieval of Web tables (Zhang et al., 2019), ontology completion (Bouraoui and Schockaert, 2019) and information retrieval in the medical domain (Arguello Casteleiro et al., 2020). More generally, relational similarity (or analogy) plays a central role in computational creativity (Goel, 2019), legal reasoning (Ashley, 1988; Walton, 2010), ontology alignment (Raad and Evermann, 2015) and instance-based learning (Miclet et al., 2008).

Given the recent success of pre-trained language models (Devlin et al., 2019; Liu et al., 2019; Brown et al., 2020), we may wonder whether such models are able to capture lexical relations in a more faithful or fine-grained way than traditional word embeddings. However, for language models (LMs), there is no direct equivalent to the word vector difference. In this paper, we therefore propose a strategy for extracting relation embeddings from pre-trained LMs, i.e. vectors encoding the relationship between two words. On the one hand, this will allow us to gain a better understanding of how well lexical relations are captured by these models. On the other hand, this will also provide us with a practical method for obtaining relation embeddings in applications such as the ones mentioned above.

Since it is unclear how LMs store relational knowledge, rather than directly extracting relation embeddings, we first fine-tune the LM, such that relation embeddings can be obtained from its output. To this end, we need a prompt, i.e. a template to convert a given word pair into a sentence, and some training data to fine-tune the model. To illustrate the process, consider the word pair *Paris-France*. As a possible input to the model, we could use a sentence such as "The relation between Paris and France is <mask>". Note that our aim is to find a strategy that can be applied to any pair of words, hence the way in which the input is represented needs to be sufficiently generic. We then fine-tune the LM such that its output corresponds

[1] Source code to reproduce our experimental results and the model checkpoints are available in the following repository: https://github.com/asahi417/relbert

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 9044–9062
November 7–11, 2021. ©2021 Association for Computational Linguistics

to a relation embedding. To this end, we use a crowdsourced dataset of relational similarity judgements that was collected in the context of SemEval 2012 Task 2 (Jurgens et al., 2012). Despite the relatively small size of this dataset, we show that the resulting fine-tuned LM allows us to produce high-quality relation embeddings, as confirmed in our extensive evaluation in analogy and relation classification tasks. Importantly, this also holds for relations that are of a different nature than those in the SemEval dataset, showing that this process allows us to distill relational knowledge that is encoded in the pre-trained LM, rather than merely generalising from the examples that were used for fine-tuning.

2 Related Work

Probing LMs for Relational Knowledge Since the introduction of transformer-based LMs, a large number of works have focused on analysing the capabilities of such models, covering the extent to which they capture syntax (Goldberg, 2019; Saphra and Lopez, 2019; Hewitt and Manning, 2019; van Schijndel et al., 2019; Jawahar et al., 2019; Tenney et al., 2019), lexical semantics (Ethayarajh, 2019; Bommasani et al., 2020; Vulic et al., 2020), and various forms of factual and commonsense knowledge (Petroni et al., 2019; Forbes et al., 2019; Davison et al., 2019; Zhou et al., 2020; Talmor et al., 2020; Roberts et al., 2020), among others. The idea of extracting relational knowledge from LMs, in particular, has also been studied. For instance, Petroni et al. (2019) use BERT for link prediction. To this end, they use a manually defined prompt for each relation type, in which the tail entity is replaced by a <mask> token. To complete a knowledge graph triple such as (*Dante*, *born-in*, ?) they create the input "*Dante was born in <mask>*" and then look at the predictions of BERT for the masked token to retrieve the correct answer. It is notable that BERT is thus used for extracting relational knowledge without any fine-tuning. This clearly shows that a substantial amount of factual knowledge is encoded in the parameters of pre-trained LMs. Some works have also looked at how such knowledge is stored. Geva et al. (2020) argue that the feed-forward layers of transformer-based LMs act as neural memories, which would suggest that e.g. "the place where Dante is born" is stored as a property of Florence. Dai et al. (2021) present further evidence of this view. What is less clear, then, is whether relations themselves have an explicit representation, or whether transformer models essentially store a propositionalised knowledge graph. The results we present in this paper suggest that common lexical relations (e.g. hypernymy, meronymy, has-attribute), at least, must have some kind of explicit representation, although it remains unclear how they are encoded.

Another notable work focusing on link prediction is (Bosselut et al., 2019), where GPT is fine-tuned to complete triples from commonsense knowledge graphs, in particular ConceptNet (Speer et al., 2017) and ATOMIC (Sap et al., 2019). While their model was able to generate new knowledge graph triples, it is unclear to what extent this is achieved by extracting commonsense knowledge that was already captured by the pre-trained GPT model, or whether this rather comes from the ability to generalise from the training triples. For the ConceptNet dataset, for instance, Jastrzębski et al. (2018) found that most test triples are in fact minor variations of training triples. In this paper, we also rely on fine-tuning, which makes it harder to determine to what extent the pre-trained LM already captures relational knowledge. We address this concern by including relation types in our evaluation which are different from the ones that have been used for fine-tuning.

Unsupervised Relation Discovery Modelling how different words are related is a long-standing challenge in NLP. An early approach is DIRT (Lin and Pantel, 2001), which encodes the relation between two nouns as the dependency path connecting them. Their view is that two such dependency paths are similar if the sets of word pairs with which they co-occur are similar. Hasegawa et al. (2004) cluster named entity pairs based on the bag-of-words representations of the contexts in which they appear. Along the same lines, Yao et al. (2011) proposed a generative probabilistic model, inspired by LDA (Blei et al., 2003), in which relations are viewed as latent variables (similar to topics in LDA). Turney (2005) proposed a method called Latent Relational Analysis (LRA), which uses matrix factorization to learn relation embeddings based on co-occurrences of word pairs and dependency paths. Matrix factorization is also used in the Universal Schema approach from Riedel et al. (Riedel et al., 2013), which jointly models the contexts in which words appear in a corpus with a given set of relational facts.

The aforementioned works essentially represent the relation between two words by summarising the contexts in which these words co-occur. In recent years, a number of strategies based on distributional models have been explored that rely on similar intuitions but go beyond simple vector operations of word embeddings.[2] For instance, Jameel et al. (2018) introduced a variant of the GloVe word embedding model, in which relation vectors are jointly learned with word vectors. In SeVeN (Espinosa-Anke and Schockaert, 2018) and RELATIVE (Camacho-Collados et al., 2019), relation vectors are computed by averaging the embeddings of context words, while pair2vec (Joshi et al., 2019) uses an LSTM to summarise the contexts in which two given words occur, and Washio and Kato (2018) learn embeddings of dependency paths to encode word pairs. Another line of work is based on the idea that relation embeddings should facilitate link prediction, i.e. given the first word and a relation vector, we should be able to predict the second word (Marcheggiani and Titov, 2016; Simon et al., 2019). This idea also lies at the basis of the approach from Soares et al. (2019), who train a relation encoder by fine-tuning BERT (Devlin et al., 2019) with a link prediction loss. However, it should be noted that they focus on learning relation vectors from individual sentences, as a pre-training task for applications such as few-shot relation extraction. In contrast, our focus in this paper is on characterising the overall relationship between two words.

3 RelBERT

In this section, we describe our proposed relation embedding model (*RelBERT* henceforth). To obtain a relation embedding for given a word pair (h, t), we first convert it into a sentence s, called the prompt. We then feed the prompt through the LM and average the contextualized embeddings (i.e. the output vectors) to get the relation embedding of (h, t). These steps are illustrated in Figure 1 and explained in more detail in the following.

3.1 Prompt Generation

Manual Prompts A basic prompt generation strategy is to rely on manually created templates,

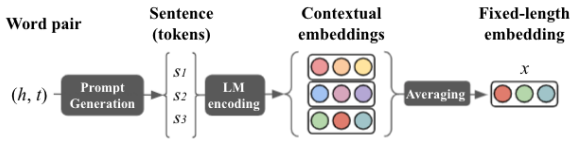

Figure 1: Pipeline to transform the word pair (h, t) to the relation embedding $\boldsymbol{x}$.

which has proven effective in factual knowledge probing (Petroni et al., 2019) and text classification (Schick and Schütze, 2021; Tam et al., 2021; Le Scao and Rush, 2021), among many others. To test whether manually generated templates can be effective for learning relation embeddings, we will consider the following five templates:

1. Today, I finally discovered the relation between **[h]** and **[t]** : **[h]** is the <mask> of **[t]**
2. Today, I finally discovered the relation between **[h]** and **[t]** : **[t]** is **[h]**'s <mask>
3. Today, I finally discovered the relation between **[h]** and **[t]** : <mask>
4. I wasn't aware of this relationship, but I just read in the encyclopedia that **[h]** is the <mask> of **[t]**
5. I wasn't aware of this relationship, but I just read in the encyclopedia that **[t]** is **[h]**'s <mask>

where <*mask*> is the LM's mask token, and **[h]** and **[t]** are slots that are filled with the head word h and tail word t from the given word pair (h, t) respectively. The main intuition is that the template should encode that we are interested in the relationship between h and t. Moreover, we avoid minimal templates such as "**[h]** is the <*mask*> of **[t]**", as LMs typically perform worse on such short inputs (Bouraoui et al., 2020; Jiang et al., 2020).

Learned Prompts The choice of prompt can have a significant impact on an LM's performance. Since it is difficult to generate manual prompts in a systematic way, several strategies for automated generation of task-specific prompts have been proposed, e.g. based on mining patterns from a corpus (Bouraoui et al., 2020), paraphrasing (Jiang et al., 2020), training an additional LM for template generation (Haviv et al., 2021; Gao et al., 2020), and prompt optimization (Shin et al., 2020; Liu et al., 2021). In our work, we focus on the latter strategy, given its conceptual simplicity and its strong reported performance on various benchmarks. Specifically, we consider AutoPrompt (Shin et al., 2020) and P-tuning (Liu et al., 2021). Note that both methods rely on training data. We will use the same training data and loss function that

[2] Interestingly, Roller and Erk (2016) showed that the direct concatenation of distributional word vectors in isolation can effectively identify Hearst Patterns (Hearst, 1992).

we use for fine-tuning the LM; see Section 3.2.

AutoPrompt initializes the prompt as a fixed-length template:

$$T = (z_1, \ldots, z_\pi, \textbf{[h]}, z_{\pi+1}, \ldots, z_{\pi+\tau}, \textbf{[t]}, z_{\pi+\tau+1}, \ldots, z_{\pi+\tau+\gamma}) \quad (1)$$

where π, τ, γ are hyper-parameters which determine the length of the template. The tokens of the form z_i are called trigger tokens. These tokens are initialized as <*mask*>. The method then iteratively finds the best token to replace each mask, based on the gradient of the task-specific loss function.[3]

P-tuning employs the same template initialization as AutoPrompt but its trigger tokens are newly introduced special tokens with trainable embeddings $\hat{\boldsymbol{e}}_{1:\pi+\tau+\gamma}$, which are learned using a task-specific loss function while the LM's weights are frozen.

3.2 Fine-tuning the LM

To fine-tune the LM, we need training data and a loss function. As training data, we assume that, for a number of different relation types r, we have access to examples of word pairs (h, t) that are instances of that relation type. The loss function is based on the following intuition: the embeddings of word pairs that belong to the same relation type should be closer together than the embeddings of pairs that belong to different relations. In particular, we use the triplet loss from Schroff et al. (2015) and the classification loss from Reimers and Gurevych (2019), both of which are based on this intuition.

Triplet Loss We draw a triplet from the relation dataset by selecting an anchor pair $a = (h_a, t_a)$, a positive example $p = (h_p, t_p)$ and a negative example $n = (h_n, t_n)$, i.e. we select word pairs a, p, n such that a and p belong to the same relation type while n belongs to a different relation type. Let us write $\boldsymbol{x}_a$, $\boldsymbol{x}_p$, $\boldsymbol{x}_n$ for the corresponding relation embeddings. Each relation embedding is produced by the same LM, which is trained to make the distance between $\boldsymbol{x}_a$ and $\boldsymbol{x}_p$ smaller than the distance between $\boldsymbol{x}_a$ and $\boldsymbol{x}_n$. Formally, this is accomplished using the following triplet loss function:

$$L_t = \max\left(0, \|\boldsymbol{x}_a - \boldsymbol{x}_p\| - \|\boldsymbol{x}_a - \boldsymbol{x}_n\| + \varepsilon\right)$$

where $\varepsilon > 0$ is the margin and $\|\cdot\|$ is the l^2 norm.

Classification Loss Following SBERT (Reimers and Gurevych, 2019), we use a classifier to predict whether two word pairs belong to the same relation. The classifier is jointly trained with the LM using the negative log likelihood loss function:

$$L_c = -\log(g(\boldsymbol{x}_a, \boldsymbol{x}_p)) - \log(1 - g(\boldsymbol{x}_a, \boldsymbol{x}_n))$$

where

$$g(\boldsymbol{u}, \boldsymbol{v}) = \text{sigmoid}(W \cdot [\boldsymbol{u} \oplus \boldsymbol{v} \oplus |\boldsymbol{v} - \boldsymbol{u}|]^T)$$

with $W \in \mathbb{R}^{3\times d}$, $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^d$, $|\cdot|$ the element-wise absolute difference, and $\oplus$ concatenation.

4 Experimental Setting

In this section, we explain our experimental setting to train and evaluate RelBERT.

4.1 RelBERT Training

Dataset We use the platinum ratings from SemEval 2012 Task 2 (Jurgens et al., 2012) as our training dataset for RelBERT. This dataset covers 79 fine-grained semantic relations, which are grouped in 10 categories. For each of the 79 relations, the dataset contains a typicality score for a number of word pairs (around 40 on average), indicating to what extent the word pair is a prototypical instance of the relation. We treat the top 10 pairs (i.e. those with the highest typicality score) as positive examples of the relation, and the bottom 10 pairs as negative examples. We use 80% of these positive and negative examples for training RelBERT (i.e. learning the prompt and fine-tuning the LM) and 20% for validation.

Constructing Training Triples We rely on three different strategies for constructing training triples. First, we obtain triples by selecting two positive examples of a given relation type (i.e. from the top-10 pairs) and one negative example (i.e. from the bottom-10 pairs). We construct 450 such triples per relation. Second, we construct triples by using two positive examples of the relation and one positive example from another relation (which is assumed to correspond to a negative example). In particular, for efficiency, we use the anchors and positive examples of the other triples from the same batch as negative examples (while ensuring that these

[3] We note that in most implementations of AutoPrompt the vocabulary to sample trigger tokens is restricted to that of the training data. However, given the nature of our training data (i.e., pairs of words and not sentences), we consider the full pre-trained LM's vocabulary.

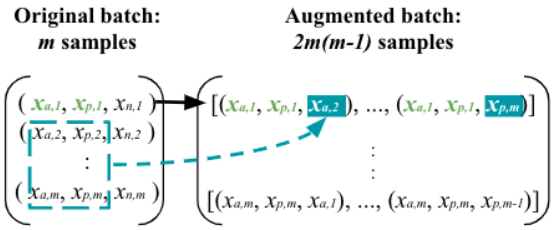

Figure 2: Batch augmentation where the original batch with m samples is augmented with $2m(m-1)$ samples.

triples are from different relations). Figure 2 illustrates this idea. Note how the effective batch size thus increases quadratically, while the number of vectors that needs to be encoded by the LM remains unchanged. In our setting, this leads to an additional 13500 triples per relation. Similar in-batch negative sampling has been shown to be effective in information retrieval (Karpukhin et al., 2020; Gillick et al., 2019). Third, we also construct training triples by considering the 10 high-level categories as relation types. In this case, we choose two positive examples from different relations that belong to the same category, along with a positive example from a relation from a different category. We add 5040 triples of this kind for each of the 10 categories.

Training RelBERT training consists of two phases: prompt optimization (unless a manually defined prompt is used) and language model fine-tuning. First we optimize the prompt over the training set with the triplet loss L_t while the parameters of the LM are frozen. Subsequently, we fine-tune the LM with the resulting prompt, using the sum of the triplet loss L_t and the classification loss L_c over the same training set. We do not use the classification loss during the prompt optimisation, as that would involve training the classifier while optimizing the prompt. We select the best hyper-parameters of the prompting methods based on the final loss over the validation set. In particular, when manual prompts are used, we choose the best template among the five candidates described in Section 3.1. For AutoPrompt and P-tuning, we consider all combinations of $\pi \in \{8, 9\}$, $\tau \in \{1, 2\}$, $\gamma \in \{1, 2\}$. We use RoBERTa (Liu et al., 2019) as our main LM, where the initial weights were taken from the `roberta-large` model checkpoint shared by the Huggingface transformers model hub (Wolf et al., 2020). We use the Adam optimizer (Kingma and Ba, 2014) with learning rate 0.00002, batch size 64 and we fine-tune the model for 1 epoch. For AutoPrompt, the top-50 tokens are considered and the number of iterations is set to 50. In each iteration, one of the input tokens is re-sampled and the loss is re-computed across the entire training set.[4] For P-tuning, we train the weights that define the trigger embeddings (i.e. the weights of the input vectors and the parameters of the LSTM) for 2 epochs. Note that we do not tune RelBERT on any task-specific training or validation set. We thus use the same relation embeddings across all the considered evaluation tasks.

4.2 Evaluation Tasks

We evaluate RelBERT on two relation-centric tasks: solving analogy questions (unsupervised) and lexical relation classification (supervised).

Analogy Questions We consider the task of solving word analogy questions. Given a query word pair, the model is required to select the relationally most similar word pair from a list of candidates. To solve this task, we simply choose the candidate whose RelBERT embedding has the highest cosine similarity with the RelBERT embedding of the query pair. Note that this task is completely unsupervised, without the need for any training or tuning. We use the five analogy datasets that were considered by Ushio et al. (2021): the SAT analogies dataset (Turney et al., 2003), the U2 and U4 analogy datasets, which were collected from an educational website[5], and datasets that were derived[6] from BATS (Gladkova et al., 2016) and the Google analogy dataset (Mikolov et al., 2013b). These five datasets consist of tuning and testing fragments. In particular, they contain 37/337 (SAT), 24/228 (U2), 48/432 (U4), 50/500 (Google), and 199/1799 (BATS) questions for validation/testing. As there is no need to tune RelBERT on task-specific data, we only use the test fragments. For SAT, we will also report results on the full dataset (i.e. the testing fragment and tuning fragment combined), as this allows us to compare the performance with published results. We will refer to this full version of the SAT dataset as SAT†.

[4] We should note that AutoPrompt takes considerably longer than any other components of RelBERT. More details on experimental training times are included in the appendix.

[5] https://englishforeveryone.org/Topics/Analogies.html

[6] In particular, they were converted into the same format of multiple-choice questions as the other datasets.

	BLESS	CogALex	EVALution	K&H+N	ROOT09
Random	8,529/609/3,008	2,228/3,059	-	18,319/1,313/6,746	4,479/327/1,566
Meronym	2,051/146/746	163/224	218/13/86	755/48/240	-
Event	2,657/212/955	-	-	-	-
Hypernym	924/63/350	255/382	1,327/94/459	3,048/202/1,042	2,232/149/809
Co-hyponym	2,529/154/882	-	-	18,134/1,313/6,349	2,222/162/816
Attribute	1,892/143/696	-	903/72/322	-	-
Possession	-	-	377/25/142	-	-
Antonym	-	241/360	1,095/90/415	-	-
Synonym	-	167/235	759/50/277	-	-

Table 1: Number of instances for each relation type across training/validation/test sets of all lexical relation classification datasets.

Lexical Relation Classification We consider the task of predicting which relation a given word pair belongs to. To solve this task, we train a multi-layer perceptron (MLP) which takes the (frozen) RelBERT embedding of the word pair as input. We consider the following widely-used multi-class relation classification benchmarks: K&H+N (Necşulescu et al., 2015), BLESS (Baroni and Lenci, 2011), ROOT09 (Santus et al., 2016b), EVALution (Santus et al., 2015), and CogALex-V Subtask 2 (Santus et al., 2016a). Table 1 shows the size of the training, validation and test sets for each of the relation classification dataset. The hyperparameters of the MLP classifier are tuned on the validation set of each dataset. Concretely, we tune the learning rate from $[0.001, 0.0001, 0.00001]$ and the hidden layer size from $[100, 150, 200]$. CogALex-V only has testing fragments so for this dataset we employ the default configuration of Scikit-Learn (Pedregosa et al., 2011), which uses a 100-dimensional hidden layer and is optimized using Adam with a learning rate of 0.001. These datasets focus on the following lexical relations: co-hyponymy (cohyp), hypernymy (hyp), meronymy (mero), possession (poss), synonymy (syn), antonymy (ant), attribute (attr), event, and random (rand).

4.3 Baselines

As baselines, we consider two standard word embedding models: GloVe (Pennington et al., 2014) and FastText (Bojanowski et al., 2017), where word pairs are represented by the vector difference of their word embeddings (*diff*).[7] For the classification experiments, we also consider the concatenation of the two word embeddings (*cat*) and their element-wise multiplication[8] (*dot*). We furthermore experiment with two pre-trained word pair embedding models: pair2vec (Joshi et al., 2019) (*pair*) and RELATIVE (Camacho-Collados et al., 2019) (*rel*). For these word pair embeddings, as well as for RelBERT, we concatenate the embeddings from both directions, i.e. (h, t) and (t, h). For the analogy questions, two simple statistical baselines are included: the expected random performance and a strategy based on point-wise mutual information (PMI) Church and Hanks (1990). In particular, the PMI score of a word pair is computed using the English Wikipedia, with a fixed window size of 10. We then choose the candidate pair with the highest PMI as the prediction. Note that this PMI-based method completely ignores the query pair. We also compare with the published results from Ushio et al. (2021), where a strategy is proposed to solve analogy questions by using LMs to compute an *analogical proportion score*. In particular, a four-word tuple (a, b, c, d) is encoded using a custom prompt and perplexity based scoring strategies are used to determine whether the word pair (a, b) has the same relation as the word pair (c, d). Finally, for the SAT† dataset, we compare with the published results from GPT-3 (Brown et al., 2020), LRA (Turney, 2005) and SuperSim (Turney, 2013); for relation classification we report the published results of the LexNet (Shwartz et al., 2016) and SphereRE (Wang et al., 2019) relation classification models, taking the results from the latter publication. We did not reproduce these latter methods in similar conditions as our work, and hence they are not fully comparable. More-

[7] Vector difference is the most common method for encoding relations, and has been shown to be the most reliable in the context of word analogies (Hakami and Bollegala, 2017).

[8] Multiplicative features have been shown to provide consistent improvements for word embeddings in supervised relation classification tasks (Vu and Shwartz, 2018).

over, these approaches are a different nature, as the aim of our work is to provide universal relation embeddings instead of task-specific models.

5 Results

In this section, we present our main experimental results, testing the relation embeddings learned by RelBERT on analogy questions (Section 5.1) and relation classification (Section 5.2).

5.1 Analogy Questions

Table 2 shows the accuracy on the analogy benchmarks. The RelBERT models substantially outperform the baselines on all datasets, except for the Google analogy dataset.[9] Comparing the different prompt generation approaches, we can see that, surprisingly, the manual prompt consistently outperforms the automatically-learned prompt strategies.

On SAT†, RelBERT outperforms LRA, which represents the state-of-the-art in the zero-shot setting, i.e. in the setting where no training data from the SAT dataset is used. RelBERT moreover outperforms GPT-3 in the few-shot setting, despite not using any training examples. In contrast, GPT-3 encodes a number of training examples as part of the prompt.

It can furthermore be noted that the other two relation embedding methods (i.e. pair2vec and RELATIVE) perform poorly in this unsupervised task. The analogical proportion score from Ushio et al. (2021) also underperforms RelBERT, even when tuned on dataset-specific tuning data.

5.2 Lexical Relation Classification

Table 3 summarizes the results of the lexical relation classification experiments, in terms of macro and micro averaged F1 score. The RelBERT models achieve the best results on all datasets except for BLESS and K&H+N, where the performance of all models is rather close. We observe a particularly large improvement over the word embedding and SotA models on the EVALution dataset. When comparing the different prompting strategies, we again find that the manual prompts perform surprisingly well, although the best results are now obtained with learned prompts in a few cases.

[9]The Google analogy dataset has been shown to be biased toward word similarity and therefore to be well suited to word embeddings (Linzen, 2016; Rogers et al., 2017).

Model	SAT†	SAT	U2	U4	Google	BATS
Random	20.0	20.0	23.6	24.2	25.0	25.0
PMI	23.3	23.1	32.9	39.1	57.4	42.7
LRA	*56.4*	-	-	-	-	-
SuperSim	*54.8*	-	-	-	-	-
GPT-3 (zero)	*53.7*	-	-	-	-	-
GPT-3 (few)	*65.2**	-	-	-	-	-
RELATIVE	24.9	24.6	32.5	27.1	62.0	39.0
pair2vec	33.7	34.1	25.4	28.2	66.6	53.8
GloVe	48.9	47.8	46.5	39.8	96.0	68.7
FastText	49.7	47.8	43.0	40.7	**96.6**	72.0
Analogical Proportion Score						
· GPT-2	*41.4*	*35.9*	*41.2*	*44.9*	*80.4*	*63.5*
· BERT	*32.6*	*32.9*	*32.9*	*34.0*	*80.8*	*61.5*
· RoBERTa	*49.6*	*42.4*	*49.1*	*49.1*	*90.8*	*69.7*
Analogical Proportion Score (tuned)						
· GPT-2	*57.8**	*56.7**	*50.9**	*49.5**	*95.2**	*81.2**
· BERT	*42.8**	*41.8**	*44.7**	*41.2**	*88.8**	*67.9**
· RoBERTa	*55.8**	*53.4**	*58.3**	*57.4**	*93.6**	*78.4**
RelBERT						
· Manual	**69.5**	**70.6**	**66.2**	**65.3**	92.4	**78.8**
· AutoPrompt	61.0	62.3	61.4	63.0	88.2	74.6
· P-tuning	54.0	55.5	58.3	55.8	83.4	72.1

Table 2: Test accuracy (%) on analogy datasets. Results marked with * are not directly comparable, as they use a subset or the entire dataset to tune the model. Results in bold represent the best accuracy excluding those marked with *. Underlined results show the best accuracy over all the models. Results in italics were taken from the original papers.

6 Analysis

To better understand how relation embeddings are learned, in this section we analyze the model's performance in more detail.

6.1 Training Data Overlap

In our main experiments, RelBERT is trained using the SemEval 2012 Task 2 dataset. This dataset contains a broad range of semantic relations, including hypernymy and meronymy relations. This raises an important question: Does RelBERT provide us with a way to extract relational knowledge from the parameters of the pre-trained LM, or is it learning to construct relation embeddings from the triples in the training set? What is of particular interest is whether RelBERT is able to model types of relations that it has not seen during training. To answer this question, we conduct an additional experiment to evaluate RelBERT on lexical relation classification, using a version that was trained without the relations from the *Class Inclusion* category, which is the high-level category in the SemEval dataset that

Model		BLESS		CogALexV		EVALution		K&H+N		ROOT09	
		macro	micro	macro	micro	macro	micro	macro	micro	macro	micro
GloVe	*cat*	92.9	93.3	42.8	73.5	56.9	58.3	88.8	94.9	86.3	86.5
	cat+dot	**93.1**	93.7	51.9	79.2	55.9	57.3	89.6	95.1	88.8	89.0
	cat+dot+pair	91.8	92.6	56.4	81.1	58.1	59.6	89.4	95.7	89.2	89.4
	cat+dot+rel	91.1	92.0	53.2	79.2	58.4	58.6	89.3	94.9	89.3	89.4
	diff	91.0	91.5	39.2	70.8	55.6	56.9	87.0	94.4	85.9	86.3
	diff+dot	92.3	92.9	50.6	78.5	56.5	57.9	88.3	94.8	88.6	88.9
	diff+dot+pair	91.3	92.2	55.5	80.2	56.0	57.4	88.0	95.5	89.1	89.4
	diff+dot+rel	91.1	91.8	52.8	78.6	56.9	57.9	87.4	94.6	87.7	88.1
FastText	*cat*	92.4	92.9	40.7	72.4	56.4	57.9	88.1	93.8	85.7	85.5
	cat+dot	92.7	93.2	48.5	77.4	56.7	57.8	89.1	94.0	88.2	88.5
	cat+dot+pair	90.9	91.5	53.0	79.3	56.1	58.2	88.3	94.3	87.7	87.8
	cat+dot+rel	91.4	91.9	50.6	76.8	57.9	59.1	86.9	93.5	87.1	87.4
	diff	90.7	91.2	39.7	70.2	53.2	55.5	85.8	93.3	85.5	86.0
	diff+dot	92.3	92.9	49.1	77.8	55.2	57.4	86.5	93.6	88.5	88.9
	diff+dot+pair	90.0	90.8	53.9	79.0	55.8	57.8	86.6	94.2	87.7	88.1
	diff+dot+rel	90.6	91.3	53.6	78.2	57.1	58.0	86.3	93.4	86.9	87.4
RelBERT	Manual	91.7	92.1	**71.2**	**87.0**	68.4	69.6	88.0	96.2	**90.9**	**91.0**
	AutoPrompt	91.9	92.4	68.5	85.1	**69.5**	**70.5**	**91.3**	97.1	90.0	90.3
	P-tuning	91.3	91.8	67.8	84.9	69.1	70.2	88.5	96.3	89.8	89.9
SotA	LexNET	-	89.3	-	-	-	60.0	-	98.5	-	81.3
	SphereRE	-	**93.8**	-	-	-	62.0	-	**99.0**	-	86.1

Table 3: Macro/micro F1 score (%) for lexical relation classification.

	BLESS	CogALex	EVAL	K&H+N	ROOT09
rand	93.7 (+0.3)	94.3 (-0.2)	-	97.9 (+0.2)	91.2 (-0.1)
mero	89.8 (+1.4)	72.9 (+2.7)	69.2 (+1.9)	74.5 (+5.4)	-
event	86.5 (-0.3)	-	-	-	-
hyp	94.1 (+0.8)	60.9 (-0.7)	61.7 (-1.5)	93.5 (+5.0)	83.0 (-0.4)
cohyp	96.4 (+0.3)	-	-	97.8 (+1.2)	97.4 (-0.5)
attr	92.6 (+0.3)	-	84.7 (+1.6)	-	-
poss	-	-	67.1 (-0.2)	-	-
ant	-	76.8 (-2.6)	81.3 (-0.9)	-	-
syn	-	49.9 (-0.6)	53.6 (+2.7)	-	-
macro	92.2 (+0.5)	71.0 (-0.2)	69.3 (+0.9)	90.9 (+2.9)	90.5 (-0.4)
micro	92.5 (+0.4)	86.9 (-0.1)	70.2 (+0.6)	97.2 (+1.0)	90.7 (-0.3)

Table 4: Per-class F1 score of RelBERT trained without hypernymy relations and the absolute difference with respect to the original model (parentheses), along with the macro and micro averaged F1 for each dataset (%).

includes the *hypernymy* relation. Hypernymy is of particular interest, as it can be found across all the considered lexical relation classification datasets, which is itself a reflection of its central importance in lexical semantics. In Table 4, we report the difference in performance compared to the original RelBERT model (i.e. the model that was fine-tuned on the full SemEval training set). As can be seen, the overall changes in performance are small, and the new version actually outperforms the original RelBERT model on a few datasets. In particular, hypernymy is still modelled well, confirming that RelBERT is able to generalize to unseen relations.

Model	Google		BATS		
	Mor	Sem	Mor	Sem	Lex
FastText	95.4	98.1	90.4	71.1	33.8
Manual	89.8	95.8	87.0	66.2	75.1
AutoPrompt	90.5	85.1	85.3	59.8	68.0
P-tuning	87.4	78.1	82.9	60.9	61.8

Table 5: Test accuracy for the high-level categories of BATS and Google, comparing FastText and RelBERT.

As a further analysis, Table 5 shows a breakdown of the Google and BATS analogy results, showing the average performance on each of the top-level categories from these datasets.[10] While RelBERT is outperformed by FastText on the morphological relations, it should be noted that the differences are small, while such relations are of a very different nature than those from the SemEval dataset. This confirms that RelBERT is able to model a broad range of relations, although it can be expected that better results would be possible by including task-specific training data into the fine-tuning process (e.g. including morphological relations for tasks where such relations matter).

[10] A full break-down showing the results for individual relations is provided in the appendix.

Target	Nearest Neighbors
barista:coffee	baker:bread, brewer:beer, bartender:cocktail, winemaker:wine, bartender:drink, baker:cake
bag:plastic	bottle:plastic, bag:leather, container:plastic, box:plastic, jug:glass, bottle:glass
duck:duckling	chicken:chick, pig:piglet, cat:kitten, ox:calf, butterfly:larvae, bear:cub
cooked:raw	raw:cooked, regulated:unregulated, sober:drunk, loaded:unloaded, armed:unarmed, published:unpublished
chihuahua:dog	dachshund:dog, poodle:dog, terrier:dog, chinchilla:rodent, macaque:monkey, dalmatian:dog
dog:dogs	cat:cats, horse:horses, pig:pigs, rat:rats, wolf:wolves, monkey:monkeys
spy:espionage	pirate:piracy, robber:robbery, lobbyist:lobbying, scout:scouting, terrorist:terrorism, witch:witchcraft

Table 6: Nearest neighbors of selected word pairs, in terms of cosine similarity between RelBERT embeddings. Candidate word pairs are taken from the RELATIVE pair vocabulary.

6.2 Language Model Comparison

Figure 3 compares the performance of RelBERT with that of the vanilla pre-trained RoBERTa model (i.e. when only the prompt is optimized). As can be seen, the fine-tuning process is critical for achieving good results. In Figure 3, we also compare the performance of our main RelBERT model, which is based on RoBERTa, with versions that were instead initialized with BERT (Devlin et al., 2019) and ALBERT (Lan et al., 2019).[11] RoBERTa clearly outperforms the other two LMs, which is in accordance with findings from the literature suggesting that RoBERTa captures more semantic knowledge (Li et al., 2020; Warstadt et al., 2020).

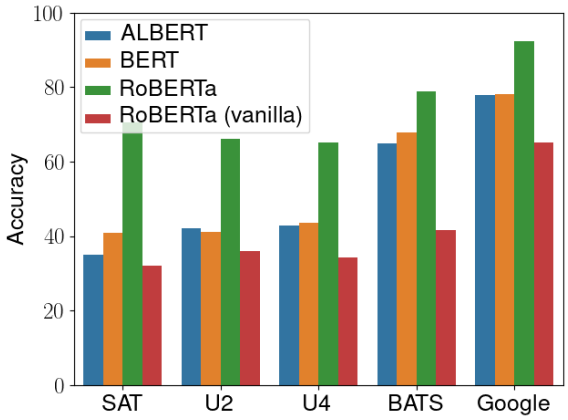

Figure 3: Test accuracy (%) on analogy dataset of the vanilla RoBERTa model (i.e. without fine-tuning) and variants of RelBERT with different language models. Each variant uses the best manual prompt based on the SemEval tuning data.

6.3 Qualitative Analysis

To give further insight into the nature of RelBERT embeddings, Table 6 shows the nearest neighbors of some selected word pairs from the evaluation datasets. To this end, we computed RelBERT relation vectors for all pairs in the Wikipedia pre-trained RELATIVE vocabulary (over 1M pairs).[12] The neighbors are those word pairs whose RelBERT embedding has the highest cosine similarity within the full pair vocabulary. As can be seen, the neighbors mostly represent word pairs that are relationally similar, even for morphological relations (e.g. *dog:dogs*), which are not present in the SemEval dataset. A more extensive qualitative analysis, including a comparison with RELATIVE, is provided in the appendix.

7 Conclusion

We have proposed a strategy for learning relation embeddings, i.e. vector representations of pairs of words which capture their relationship. The main idea is to fine-tune a pre-trained language model using the relational similarity dataset from SemEval 2012 Task 2, which covers a broad range of semantic relations. In our experimental results, we found the resulting relation embeddings to be of high quality, outperforming state-of-the-art methods on several analogy and relation classification benchmarks. Among the models tested, we obtained the best results with RoBERTa, when using manually defined templates for encoding word pairs. Importantly, we found that high-quality relation embeddings can be obtained even for relations that are unlike those from the SemEval dataset, such as morphological and encyclopedic relations. This suggests that the knowledge captured by our relation embeddings is largely distilled from the pre-trained language model, rather than being acquired during training.

Acknowledgements

Jose Camacho-Collados acknowledges support from the UKRI Future Leaders Fellowship scheme.

[11] We used `bert-large-cased` and `albert-xlarge-v1` from the HuggingFace model hub.

[12] https://github.com/pedrada88/relative

References

Mercedes Arguello Casteleiro, Julio Des Diz, Nava Maroto, Maria Jesus Fernandez Prieto, Simon Peters, Chris Wroe, Carlos Sevillano Torrado, Diego Maseda Fernandez, and Robert Stevens. 2020. Semantic deep learning: Prior knowledge and a type of four-term embedding analogy to acquire treatments for well-known diseases. *JMIR Med Inform*, 8(8).

Kevin D Ashley. 1988. Arguing by analogy in law: A case-based model. In *Analogical Reasoning*, pages 205–224. Springer.

Marco Baroni and Alessandro Lenci. 2011. How we BLESSed distributional semantic evaluation. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 1–10, Edinburgh, UK. Association for Computational Linguistics.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting pretrained contextualized representations via reductions to static embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781.

Rajesh Bordawekar and Oded Shmueli. 2017. Using word embedding to enable semantic queries in relational databases. In *Proceedings of the 1st Workshop on Data Management for End-to-End Machine Learning*, pages 1–4.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779.

Zied Bouraoui, Jose Camacho-Collados, and Steven Schockaert. 2020. Inducing relational knowledge from bert. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7456–7463.

Zied Bouraoui and Steven Schockaert. 2019. Automated rule base completion as bayesian concept induction. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, pages 6228–6235.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the Annual Conference on Neural Information Processing Systems*.

Jose Camacho-Collados, Luis Espinosa-Anke, Jameel Shoaib, and Steven Schockaert. 2019. A latent variable model for learning distributional relation vectors. In *Proceedings of IJCAI*.

Kenneth Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, and Furu Wei. 2021. Knowledge neurons in pretrained transformers. *arXiv preprint arXiv:2104.08696*.

Joe Davison, Joshua Feldman, and Alexander M Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 1173–1178.

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

Luis Espinosa-Anke and Steven Schockaert. 2018. SeVeN: Augmenting word embeddings with unsupervised relation vectors. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2653–2665, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 55–65.

Raul Castro Fernandez, Essam Mansour, Abdulhakim A Qahtan, Ahmed Elmagarmid, Ihab Ilyas, Samuel Madden, Mourad Ouzzani, Michael Stonebraker, and Nan Tang. 2018. Seeping semantics: Linking datasets using word embeddings for data discovery. In *2018 IEEE 34th International Conference on Data Engineering*, pages 989–1000.

Maxwell Forbes, Ari Holtzman, and Yejin Choi. 2019. Do neural language representations learn physical commonsense? In *Proceedings of the 41th Annual Meeting of the Cognitive Science Society*, pages 1753–1759.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2020. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 528–537, Hong Kong, China. Association for Computational Linguistics.

Anna Gladkova, Aleksandr Drozd, and Satoshi Matsuoka. 2016. Analogy-based detection of morphological and semantic relations with word embeddings: what works and what doesn't. In *Proceedings of the NAACL Student Research Workshop*, pages 8–15, San Diego, California. Association for Computational Linguistics.

Ashok Goel. 2019. Computational design, analogy, and creativity. In *Computational Creativity*, pages 141–158. Springer.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *arXiv preprint arXiv:1901.05287*.

Huda Hakami and Danushka Bollegala. 2017. Compositional approaches for representing relations between words: A comparative study. *Knowledge-Based Systems*, 136:172–182.

Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. 2004. Discovering relations among named entities from large corpora. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 415–422.

Adi Haviv, Jonathan Berant, and Amir Globerson. 2021. BERTese: Learning to speak to BERT. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3618–3623, Online. Association for Computational Linguistics.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics*.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4129–4138.

Shoaib Jameel, Zied Bouraoui, and Steven Schockaert. 2018. Unsupervised learning of distributional relation vectors. In *Annual Meeting of the Association for Computational Linguistics*, pages 23–33.

Stanislaw Jastrzębski, Dzmitry Bahdanau, Seyedarian Hosseini, Michael Noukhovitch, Yoshua Bengio, and Jackie Chi Kit Cheung. 2018. Commonsense mining as knowledge base completion? a study on the impact of novelty. In *Proceedings of the Workshop on Generalization in the Age of Deep Learning*, pages 8–16.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Mandar Joshi, Eunsol Choi, Omer Levy, Daniel Weld, and Luke Zettlemoyer. 2019. pair2vec: Compositional word-pair embeddings for cross-sentence inference. 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 3597–3608, Minneapolis, Minnesota. Association for Computational Linguistics.

David Jurgens, Saif Mohammad, Peter Turney, and Keith Holyoak. 2012. SemEval-2012 task 2: Measuring degrees of relational similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 356–364, Montréal, Canada. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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*.

Teven Le Scao and Alexander M Rush. 2021. How many data points is a prompt worth? *arXiv e-prints*, pages arXiv–2103.

Chuanrong Li, Lin Shengshuo, Zeyu Liu, Xinyi Wu, Xuhui Zhou, and Shane Steinert-Threlkeld. 2020. Linguistically-informed transformations (LIT): A method for automatically generating contrast sets. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 126–135, Online. Association for Computational Linguistics.

Dekang Lin and Patrick Pantel. 2001. DIRT - discovery of inference rules from text. In *Proceedings of the 7th Iternational Conference on Knowledge Discovery and Data Mining*, pages 323–328.

Tal Linzen. 2016. Issues in evaluating semantic spaces using word analogies. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 13–18, Berlin, Germany. Association for Computational Linguistics.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. GPT understands, too. *arXiv preprint arXiv:2103.10385*.

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. *CoRR*, abs/1907.11692.

Diego Marcheggiani and Ivan Titov. 2016. Discrete-state variational autoencoders for joint discovery and factorization of relations. *Transactions of the Association for Computational Linguistics*, 4:231–244.

Laurent Miclet, Sabri Bayoudh, and Arnaud Delhay. 2008. Analogical dissimilarity: definition, algorithms and two experiments in machine learning. *Journal of Artificial Intelligence Research*, 32:793–824.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013a. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia. Association for Computational Linguistics.

Silvia Necşulescu, Sara Mendes, David Jurgens, Núria Bel, and Roberto Navigli. 2015. Reading between the lines: Overcoming data sparsity for accurate classification of lexical relationships. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*, pages 182–192, Denver, Colorado. Association for Computational Linguistics.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830.

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, Doha, Qatar. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? 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 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Elie Raad and Joerg Evermann. 2015. The role of analogy in ontology alignment: A study on LISA. *Cognitive Systems Research*, 33:1–16.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 74–84.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 5418–5426.

Anna Rogers, Aleksandr Drozd, and Bofang Li. 2017. The (too many) problems of analogical reasoning with word vectors. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 135–148, Vancouver, Canada. Association for Computational Linguistics.

Stephen Roller and Katrin Erk. 2016. Relations such as hypernymy: Identifying and exploiting hearst patterns in distributional vectors for lexical entailment. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2163–2172, Austin, Texas. Association for Computational Linguistics.

Enrico Santus, Anna Gladkova, Stefan Evert, and Alessandro Lenci. 2016a. The CogALex-V shared task on the corpus-based identification of semantic relations. In *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon (CogALex - V)*, pages 69–79, Osaka, Japan. The COLING 2016 Organizing Committee.

Enrico Santus, Alessandro Lenci, Tin-Shing Chiu, Qin Lu, and Chu-Ren Huang. 2016b. Nine features in a random forest to learn taxonomical semantic relations. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4557–4564, Portorož, Slovenia. European Language Resources Association (ELRA).

Enrico Santus, Frances Yung, Alessandro Lenci, and Chu-Ren Huang. 2015. EVALution 1.0: an evolving semantic dataset for training and evaluation of distributional semantic models. In *Proceedings of the 4th Workshop on Linked Data in Linguistics: Resources and Applications*, pages 64–69, Beijing, China. Association for Computational Linguistics.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3027–3035.

Naomi Saphra and Adam Lopez. 2019. Understanding learning dynamics of language models with SVCCA. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3257–3267.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving hypernymy detection with an integrated path-based and distributional method. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2389–2398, Berlin, Germany. Association for Computational Linguistics.

Étienne Simon, Vincent Guigue, and Benjamin Piwowarski. 2019. Unsupervised information extraction: Regularizing discriminative approaches with relation distribution losses. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 1378–1387.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905.

Robert Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. oLMpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.

Derek Tam, Rakesh R Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. 2021. Improving and simplifying pattern exploiting training. *arXiv preprint arXiv:2103.11955*.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *Proceeding of the 7th International Conference on Learning Representations (ICLR)*.

Peter D. Turney. 2005. Measuring semantic similarity by latent relational analysis. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, pages 1136–1141.

Peter D Turney. 2012. Domain and function: A dual-space model of semantic relations and compositions. *Journal of Artificial Intelligence Research*, 44:533–585.

Peter D. Turney. 2013. Distributional semantics beyond words: Supervised learning of analogy and paraphrase. *Transactions of the Association for Computational Linguistics*, 1:353–366.

Peter D. Turney, Michael L. Littman, Jeffrey Bigham, and Victor Shnayder. 2003. Combining independent modules in lexical multiple-choice problems. In *Recent Advances in Natural Language Processing III*, pages 101–110.

Asahi Ushio, Luis Espinosa-Anke, Steven Schockaert, and Jose Camacho-Collados. 2021. BERT is to NLP what AlexNet is to CV: Can Pre-Trained Language Models Identify Analogies? In *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Marten van Schijndel, Aaron Mueller, and Tal Linzen. 2019. Quantity doesn't buy quality syntax with neural language models. 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 5831–5837, Hong Kong, China. Association for Computational Linguistics.

Tu Vu and Vered Shwartz. 2018. Integrating multiplicative features into supervised distributional methods for lexical entailment. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 160–166, New Orleans, Louisiana. Association for Computational Linguistics.

Ivan Vulic, Edoardo Maria Ponti, Robert Litschko, Goran Glavas, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 7222–7240.

Douglas Walton. 2010. Similarity, precedent and argument from analogy. *Artificial Intelligence and Law*, 18(3):217–246.

Chengyu Wang, Xiaofeng He, and Aoying Zhou. 2019. SphereRE: Distinguishing lexical relations with hyperspherical relation embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1727–1737, Florence, Italy. Association for Computational Linguistics.

Alex Warstadt, Yian Zhang, Xiaocheng Li, Haokun Liu, and Samuel R. Bowman. 2020. Learning which features matter: RoBERTa acquires a preference for linguistic generalizations (eventually). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 217–235, Online. Association for Computational Linguistics.

Koki Washio and Tsuneaki Kato. 2018. Filling missing paths: Modeling co-occurrences of word pairs and dependency paths for recognizing lexical semantic relations. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1123–1133.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. 2011. Structured relation discovery using generative models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1456–1466.

Li Zhang, Shuo Zhang, and Krisztian Balog. 2019. Table2vec: Neural word and entity embeddings for table population and retrieval. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1029–1032.

Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. 2020. Evaluating commonsense in pretrained language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9733–9740.

A Additional Experimental Results

In this section, we show additional experimental results that complement the main results of the paper.

A.1 Vanilla LM Comparison

We show comparisons of versions of RelBERT with optimized prompt with/without finetuning. Figure 4 shows the absolute accuracy drop from RelBERT (i.e. the model with fine-tuning) to the vanilla RoBERTa model (i.e. without fine-tuning) with the same prompt. In all cases, the accuracy drop for the models without fine-tuning is substantial.

A.2 Comparison with ALBERT & BERT

We use RoBERTa in our main experiments and here we train RelBERT with ALBERT and BERT instead, and evaluate them on both of the analogy and relation classification tasks. Table 7 shows the accuracy on the analogy questions, while Table 8 shows the accuracy on the relation classification task. In both tasks, we can confirm that RoBERTa

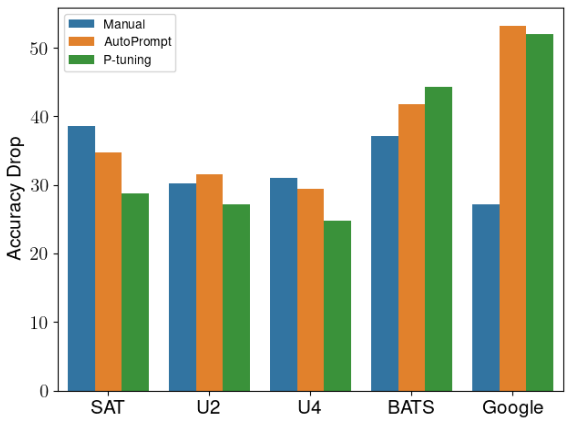

Figure 4: Test accuracy drop of the vanilla models without fine-tuning (measured in terms of absolute percentage points in comparison with RelBERT) on analogy datasets.

achieves the best performance within the LMs, by a relatively large margin in most cases.

A.3 Word Embeddings

Table 9 shows additional results of word embeddings on analogy test together with RelBERT results. We concatenate the RELATIVE and pair2vec vectors with the word vector difference. However, this does not lead to better results.

B Experimental Details and Model Configurations

In this section, we explain models' configuration in the experiments, and details on RelBERT's training time.

B.1 Prompting Configuration

Table 10 shows the best prompt configuration based on the validation loss for the SemEval 2012 Task 2 dataset in our main experiments using RoBERTa.

B.2 MLP Configuration in Relation Classification

Table 11 shows the best hyperparameters in the validation set of the MLPs for relation classification.

B.3 Training Time

Training a single RelBERT model with a custom prompt takes about half a day on two V100 GPUs. Additionally, to achieve prompt by AutoPrompt technique takes about a week on a single V100, while P-tuning takes 3 to 4 hours, also on a single V100.

Model	SAT†	SAT	U2	U4	Google	BATS
ALBERT						
· Manual	34.2	35.0	42.1	42.8	78.0	64.9
· AutoPrompt	34.0	35.0	36.4	33.8	25.0	30.5
· P-tuning	32.4	32.6	33.8	33.6	35.0	37.8
BERT						
· Manual	40.6	40.9	41.2	43.5	78.2	67.9
· AutoPrompt	36.4	36.5	36.8	35.4	51.6	43.5
· P-tuning	38.0	38.0	38.2	37.0	56.6	45.3
RoBERTa						
· Manual	**69.5**	**70.6**	**66.2**	**65.3**	**92.4**	**78.8**
· AutoPrompt	61.0	62.3	61.4	63.0	88.2	74.6
· P-tuning	54.0	55.5	58.3	55.8	83.4	72.1

Table 7: Test accuracy (%) of ALBERT, BERT, and RoBERTa on analogy datasets.

C Implementation Details of AutoPrompt

All the trigger tokens are initialized by mask tokens and updated based on the gradient of a loss function L_t. Concretely, let us denote the loss value with template T as $L_t(T)$. The candidate set for the jth trigger is derived by

$$\tilde{\mathcal{W}}_j = \underset{w \in \mathcal{W}}{\text{top-}k}\left[\boldsymbol{e}_w^T \nabla_j L_t(T)\right] \quad (2)$$

where the gradient is taken with respect to j^{th} trigger token and $\boldsymbol{e}_w$ is the input embedding for the word w. Then we evaluate each token based on the loss function as

$$z_j = \underset{w \in \tilde{\mathcal{W}}_j}{\text{argmin}}\left[L_t(rep(T, j, w))\right] \quad (3)$$

where $rep(T, j, w)$ replaces the jth token in T by w and j is randomly chosen. We ignore any candidates that do not improve current loss value to further enhance the prompt quality.

D Additional Analysis

In this section, we analyze our experimental results based on prediction breakdown and provide an extended qualitative analysis.

D.1 Prediction Breakdown

Table 12 shows a detailed break-down of the BATS results.

Model		BLESS		CogALexV		EVALution		K&H+N		ROOT09	
		macro	micro	macro	micro	macro	micro	macro	micro	macro	micro
ALBERT	Manual	86.2	87.1	54.9	81.1	62.6	62.2	82.6	91.7	86.4	86.8
	AutoPrompt	88.4	88.9	42.2	75.6	56.0	56.4	87.1	94.8	84.4	85.1
	P-tuning	90.1	90.6	44.9	73.1	58.2	59.7	90.2	95.9	85.9	85.9
BERT	Manual	90.9	91.2	65.2	83.4	67.8	68.3	91.6	97.6	90.1	90.4
	AutoPrompt	90.3	90.7	40.6	75.8	60.4	59.5	90.2	97.2	86.6	86.1
	P-tuning	87.6	88.0	52.7	79.2	61.9	63.3	86.2	95.1	85.2	85.3
RoBERTa	Manual	91.7	92.1	**71.2**	**87.0**	68.4	69.6	88.0	96.2	**90.9**	**91.0**
	AutoPrompt	**91.9**	**92.4**	68.5	85.1	**69.5**	**70.5**	**91.3**	**97.1**	90.0	90.3
	P-tuning	91.3	91.8	67.8	84.9	69.1	70.2	88.5	96.3	89.8	89.9

Table 8: Macro/micro F1 score (%) for lexical relation classification of ALBERT, BERT, and RoBERTa.

Model		SAT†	SAT	U2	U4	Google	BATS
GloVe	*diff*	48.9	47.8	46.5	47.8	96.0	68.7
	diff+rel	45.9	40.4	46.9	35.4	87.6	67.3
	diff+pair	35.1	33.8	29.4	30.6	78.0	56.3
FastText	*diff*	49.7	47.8	43.0	40.7	**96.6**	72.0
	diff+rel	37.3	35.9	39.5	35.6	85.8	67.5
	diff+pair	33.4	33.5	27.2	28.7	75.4	52.1
RelBERT	Manual	**69.5**	**70.6**	**66.2**	**65.3**	92.4	**78.8**
	AutoPrompt	61.0	62.3	61.4	63.0	88.2	74.6
	P-tuning	54.0	55.5	58.3	55.8	83.4	72.1

Table 9: Test accuracy (%) on analogy datasets (SAT† refers to the full SAT dataset).

Model	Prompt	π	τ	γ	template type
BERT	Manual	-	-	-	3
	AutoPrompt	8	2	3	-
	P-tuning	8	2	2	-
ALBERT	Manual	-	-	-	4
	AutoPrompt	8	3	3	-
	P-tuning	8	2	3	-
RoBERTa	Manual	-	-	-	4
	AutoPrompt	9	2	2	-
	P-tuning	9	3	2	-

Table 10: Best prompting configuration.

D.2 Qualitative Analysis

Tables 13 shows the nearest neighbors of a number of selected word pairs, in terms of their RelBERT and RELATIVE embeddings. In both cases cosine similarity is used to compare the embeddings and the pair vocabulary of the RELATIVE model is used to determine the universe of candidate neighbors.

The results for the RelBERT embeddings show their ability to capture a wide range of relations. In most cases the neighbors make sense, despite the fact that many of these relations are quite different from those in the SemEval dataset that was used for training RelBERT. The results for RELATIVE are in general much noisier, suggesting that RELATIVE embeddings fail to capture many types of relations. This is in particular the case for the morphological examples, although various issues can be observed for the other relations as well.

Model	**Data**	Manual	AutoPrompt	P-tuning
ALBERT	BLESS	(1e-4, 150)	(1e-3, 200)	(1e-4, 150)
	CogA	(1e-3, 100)	(1e-3, 100)	(1e-3, 100)
	EVAL	(1e-4, 100)	(1e-3, 200)	(1e-4, 100)
	K&H	(1e-4, 150)	(1e-4, 150)	(1e-4, 200)
	ROOT	(1e-5, 200)	(1e-4, 100)	(1e-4, 150)
BERT	BLESS	(1e-4, 200)	(1e-3, 100)	(1e-3, 100)
	CogA	(1e-3, 100)	(1e-3, 100)	(1e-3, 100)
	EVAL	(1e-5, 150)	(1e-3, 200)	(1e-3, 100)
	K&H	(1e-4, 200)	(1e-4, 200)	(1e-3, 150)
	ROOT	(1e-5, 100)	(1e-3, 150)	(1e-4, 150)
RoBERTa	BLESS	(1e-5, 200)	(1e-3, 100)	(1e-4, 200)
	CogA	(1e-3, 100)	(1e-3, 100)	(1e-3, 100)
	EVAL	(1e-4, 150)	(1e-5, 100)	(1e-5, 200)
	K&H	(1e-3, 200)	(1e-3, 150)	(1e-5, 200)
	ROOT	(1e-5, 100)	(1e-5, 200)	(1e-3, 200)

Table 11: Best MLP configuration for the relation classification experiment. Each entry shows the learning rate and hidden layer size. Note that CogALex uses the default configuration due to the lack of validation set.

	Relation	FastText	Manual	AutoPrompt	P-tuning
Encyclopedic	UK city:county	33.3	28.9	28.9	40.0
	animal:shelter	44.4	88.9	77.8	84.4
	animal:sound	80.0	86.7	82.2	75.6
	animal:young	53.3	62.2	64.4	51.1
	country:capital	82.2	37.8	17.8	35.6
	country:language	93.3	51.1	55.6	51.1
	male:female	88.9	60.0	55.6	62.2
	name:nationality	60.0	73.3	51.1	40.0
	name:occupation	86.7	75.6	75.6	77.8
	things:color	88.9	97.8	88.9	91.1
Lexical	antonyms:binary	26.7	64.4	68.9	77.8
	antonyms:gradable	44.4	88.9	93.3	88.9
	hypernyms:animals	44.4	91.1	80.0	55.6
	hypernyms:misc	42.2	71.1	60.0	64.4
	hyponyms:misc	31.1	55.6	55.6	48.9
	meronyms:member	44.4	68.9	48.9	53.3
	meronyms:part	31.1	77.8	71.1	55.6
	meronyms:substance	26.7	75.6	66.7	53.3
	synonyms:exact	17.8	80.0	71.1	66.7
	synonyms:intensity	28.9	77.8	64.4	53.3
Morphological	adj+ly	95.6	84.4	88.9	82.2
	adj+ness	100.0	97.8	93.3	97.8
	adj:comparative	100.0	97.8	100.0	91.1
	adj:superlative	97.8	100.0	93.3	100.0
	noun+less	77.8	100.0	97.8	100.0
	over+adj	75.6	84.4	80.0	82.2
	un+adj	60.0	97.8	91.1	97.8
	verb 3pSg:v+ed	100.0	75.6	84.4	68.9
	verb inf:3pSg	100.0	93.3	91.1	84.4
	verb inf:v+ed	100.0	91.1	91.1	88.9
	verb inf:v+ing	100.0	97.8	97.8	95.6
	verb v+ing:3pSg	97.8	82.2	68.9	68.9
	verb v+ing:v+ed	97.8	86.7	82.2	84.4
	verb+able	97.8	93.3	80.0	84.4
	verb+er	95.6	100.0	95.6	95.6
	verb+ment	95.6	77.8	77.8	62.2
	verb+tion	84.4	77.8	66.7	68.9
	noun:plural	78.7	87.6	88.8	69.7
	re+verb	75.6	62.2	86.7	66.7

Table 12: Break-down of BATS results per relation type.

Category	Target	Nearest Neighbors RelBERT	Nearest Neighbors RELATIVE
Commonsense	barista:coffee	baker:bread, brewer:beer, bartender:cocktail, winemaker:wine, bartender:drink, baker:cake	venue:bar, restaurant:kitchen, restaurant:grill, nightclub:open, pub:bar, night:concert
	restaurant:waitress	restaurant:waiter, diner:waitress, bar:bartender, hospital:nurse, courthouse:clerk, office:clerk	coincidentally:first, ironically:first, ironically:name, notably:three, however:new, instance:character
	car:garage	car:pit, plane:hangar, auto:garage, baby:crib, yacht:harbour, aircraft:hangar	shelter:house, elevator:building, worker:mine, worker:factory, plane:hangar, horse:stable
	ice:melt	snow:melt, glacier:melt, ice:drift, crust:melt, polar ice:melt, ice:thaw	glacier:melt, snow:melt, water:freeze, crack:form, ice:surface, ice:freeze
	dolphin:swim	squid:swim, salmon:swim, shark:swim, fish:swim, horse:run, frog:leap	fisherman:fish, fisherman:catch, must:protect, diver:underwater, dog:human, scheme:make
	flower:fragrant	orchid:fragrant, cluster:fragrant, jewel:precious, jewel:valuable, soil:permeable, vegetation:abundant	flower:greenish, flower:white, flower:yellowish, flower:creamy, flower:pale yellow, flower:arrange
	coconut:milk	coconut:oil, goat:milk, grape:juice, palm:oil, olive:oil, camel:milk	dry:powder, mix:sugar, candy:chocolate, cook:fry, butter:oil, milk:coffee
	bag:plastic	bottle:plastic, bag:leather, container:plastic, box:plastic, jug:glass, bottle:glass	tube:glass, bottle:plastic, typically:plastic, frame:steel, shoe:leather, wire:metal
	duck:duckling	chicken:chick, pig:piglet, cat:kitten, ox:calf, butterfly:larvae, bear:cub	adult:young, worker:queen, queen:worker, bird:fly, chick:adult, female:mat
Gender	man:woman	men:women, male:female, father:mother, boy:girl, hero:heroine, king:queen	man:boy, woman:child, child:youth, officer:crew, bride:groom, child:teen
Antonymy	cooked:raw	raw:cooked, regulated:unregulated, sober:drunk, loaded:unloaded, armed:unarmed, published:unpublished	annual:biennial, raw:cook, herb:subshrub, aquatic:semi, shrub:small, fry:grill
	normal:abnormal	ordinary:unusual, usual:unusual, acceptable:unacceptable, stable:unstable, rational:irrational, legal:illegal	acute:chronic, mouse:human, negative:positive, fat:muscle, cell:tissue, motor:sensory
Meronymy	helicopter:rotor	helicopter:rotor blades, helicopter:wing, bicycle:wheel, motorcycle:wheel, airplane:engine, plane:engine	aircraft:engine, engine:crankshaft, landing gear:wheel, engine:camshaft, rotor:blade, aircraft:wing
	bat:wing	butterfly:wing, eagle:wing, angel:wing, cat:paw, lion:wings, fly:wing	mouse:tail, dog:like, dragon:like, human:robot, leopard:spot, cat:like
	beer:alcohol	wine:alcohol, cider:alcohol, soda:sugar, beer:liquor, beer:malt, lager:alcoho	steel:carbon, cider:alcohol, humidity:average, rate:average, household:non, consume:beer
	oxygen:atmosphere	helium:atmosphere, hydrogen:atmosphere, nitrogen:atmosphere, methane:atmosphere, carbon:atmosphere	carbon dioxide:atmosphere, cloud:atmosphere, methane:atmosphere, nitrogen:soil, gas:atmosphere
Hypernymy	chihuahua:dog	dachshund:dog, poodle:dog, terrier:dog, chinchilla:rodent, macaque:monkey, dalmatian:dog	julie:katy, tench:pike, catfish:pike, sunfish:perch, salmonid:salmon, raw:marinate
	pelican:bird	toucan:bird, puffin:bird, egret:bird, peacock:bird, grouse:bird, pigeon:bird	drinking:contaminate, drinking:source, pelican:distinctive, boiling:pour, aquifer:table, fresh:source
	tennis:sport	hockey:sport, soccer:sport, volleyball:sport, cricket:sport, golf:sport, football:sport	hockey:sport, golf:sport, badminton:sport, boxing:sport, rowing:sport, volleyball:sport
Morphology	dog:dogs	cat:cats, horse:horses, pig:pigs, rat:rats, wolf:wolves, monkey:monkeys	shepherd:dog, landrace:breed, like:dog, farm:breed, breed:animal, captive:release
	tall:tallest	strong:strongest, short:shortest, smart:smartest, weak:weakest, big:biggest, small:smallest	rank:world, summit:nato, redistricting:district, delegation:congress, debate:congress
	spy:espionage	pirate:piracy, robber:robbery, lobbyist:lobbying, scout:scouting, terrorist:terrorism, witch:witchcraft	group:call, crime:criminal, action:involve, cop:police, action:one, group:make

Table 13: Nearest neighbors of selected word pairs, in terms of cosine similarity between RelBERT embeddings. Candidate word pairs are taken from the RELATIVE pair vocabulary.