BERTRAM: Improved Word Embeddings Have Big Impact on Contextualized Model Performance

Timo Schick
Sulzer GmbH
Munich, Germany
timo.schick@sulzer.de

Hinrich Schütze
Center for Information and Language Processing
LMU Munich, Germany
inquiries@cislmu.org

Abstract

Pretraining deep language models has led to large performance gains in NLP. Despite this success, Schick and Schütze (2020) recently showed that these models struggle to understand rare words. For static word embeddings, this problem has been addressed by separately learning representations for rare words. In this work, we transfer this idea to pretrained language models: We introduce BERTRAM, a powerful architecture based on BERT that is capable of inferring high-quality embeddings for rare words that are suitable as input representations for deep language models. This is achieved by enabling the surface form and contexts of a word to interact with each other in a deep architecture. Integrating BERTRAM into BERT leads to large performance increases due to improved representations of rare and medium frequency words on both a rare word probing task and three downstream tasks.[1]

1 Introduction

As word embedding algorithms (e.g. Mikolov et al., 2013) are known to struggle with rare words, several techniques for improving their representations have been proposed. These approaches exploit either the contexts in which rare words occur (Lazaridou et al., 2017; Herbelot and Baroni, 2017; Khodak et al., 2018; Liu et al., 2019a), their surface-form (Luong et al., 2013; Bojanowski et al., 2017; Pinter et al., 2017), or both (Schick and Schütze, 2019a,b; Hautte et al., 2019). However, all of this prior work is designed for and evaluated on *uncontextualized* word embeddings.

Contextualized representations obtained from pretrained deep language models (e.g. Peters et al., 2018; Radford et al., 2018; Devlin et al., 2019; Liu et al., 2019b) already handle rare words implicitly using methods such as byte-pair encoding (Sennrich et al., 2016), WordPiece embeddings (Wu et al., 2016) and character-level CNNs (Baevski et al., 2019). Nevertheless, Schick and Schütze (2020) recently showed that BERT's (Devlin et al., 2019) performance on a rare word probing task can be significantly improved by explicitly learning representations of rare words using Attentive Mimicking (AM) (Schick and Schütze, 2019a). However, AM is limited in two important respects:

- For processing contexts, it uses a simple bag-of-words model, making poor use of the available information.
- It combines form and context in a shallow fashion, preventing both input signals from interacting in a complex manner.

These limitations apply not only to AM, but to all previous work on obtaining representations for rare words by leveraging form and context. While using bag-of-words models is a reasonable choice for static embeddings, which are often themselves bag-of-words (e.g. Mikolov et al., 2013; Bojanowski et al., 2017), it stands to reason that they are not the best choice to generate input representations for position-aware, deep language models.

To overcome these limitations, we introduce BERTRAM (**BERT** fo**r** **A**ttentive **M**imicking), a novel architecture for learning rare word representations that combines a pretrained BERT model with AM. As shown in Figure 1, the learned rare word representations can then be used as an improved input representation for another BERT model. By giving BERTRAM access to both surface form and contexts starting at the lowest layer, a deep integration of both input signals becomes possible.

Assessing the effectiveness of methods like BERTRAM in a contextualized setting is challenging: While most previous work on rare words was

[1]Our implementation of BERTRAM is publicly available at https://github.com/timoschick/bertram.

evaluated on datasets explicitly focusing on rare words (e.g Luong et al., 2013; Herbelot and Baroni, 2017; Khodak et al., 2018; Liu et al., 2019a), these datasets are tailored to uncontextualized embeddings and thus not suitable for evaluating our model. Furthermore, rare words are not well represented in commonly used downstream task datasets. We therefore introduce *rarification*, a procedure to automatically convert evaluation datasets into ones for which rare words are guaranteed to be important. This is achieved by replacing task-relevant frequent words with rare synonyms obtained using semantic resources such as WordNet (Miller, 1995). We rarify three common text (or text pair) classification datasets: MNLI (Williams et al., 2018), AG's News (Zhang et al., 2015) and DBPedia (Lehmann et al., 2015). BERTRAM outperforms previous work on four English datasets by a large margin: on the three rarified datasets and on WNLaMPro (Schick and Schütze, 2020).

In summary, our contributions are as follows:

- We introduce BERTRAM, a model that integrates BERT into Attentive Mimicking, enabling a deep integration of surface-form and contexts and much better representations for rare words.
- We devise rarification, a method that transforms evaluation datasets into ones for which rare words are guaranteed to be important.
- We show that adding BERTRAM to BERT achieves a new state-of-the-art on WNLaMPro (Schick and Schütze, 2020) and beats all baselines on rarified AG's News, MNLI and DBPedia, resulting in an absolute improvement of up to 25% over BERT.

2 Related Work

Surface-form information (e.g., morphemes, characters or character n-grams) is commonly used to improve word representations. For static word embeddings, this information can either be injected into a given embedding space (Luong et al., 2013; Pinter et al., 2017), or a model can directly be given access to it during training (Bojanowski et al., 2017; Salle and Villavicencio, 2018; Piktus et al., 2019). In the area of contextualized representations, many architectures employ subword segmentation methods (e.g. Radford et al., 2018; Devlin et al., 2019; Yang et al., 2019; Liu et al., 2019b). Others use convolutional neural networks to directly access character-level information (Kim et al., 2016; Peters et al., 2018; Baevski et al., 2019).

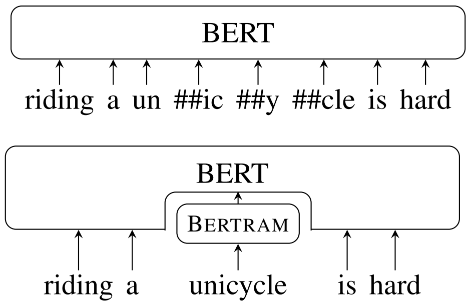

Figure 1: Top: Standard use of BERT. Bottom: Our proposal; first BERTRAM learns an embedding for "unicycle" that replaces the WordPiece sequence. BERT is then run on this improved input representation.

Complementary to surface form, another useful source of information for understanding rare words are the contexts in which they occur (Lazaridou et al., 2017; Herbelot and Baroni, 2017; Khodak et al., 2018). Schick and Schütze (2019a,b) show that combining form and context leads to significantly better results than using just one of the two. While all of these methods are bag-of-words models, Liu et al. (2019a) recently proposed an architecture based on *context2vec* (Melamud et al., 2016). However, in contrast to our work, they (i) do not incorporate surface-form information and (ii) do not directly access the hidden states of context2vec, but instead simply use its output distribution.

Several datasets focus on rare words, e.g., Stanford Rare Word (Luong et al., 2013), Definitional Nonce (Herbelot and Baroni, 2017), and Contextual Rare Word (Khodak et al., 2018). However, unlike our rarified datasets, they are only suitable for evaluating *uncontextualized* word representations. Rarification is related to adversarial example generation (e.g. Ebrahimi et al., 2018), which manipulates the input to change a model's prediction. We use a similar mechanism to determine which words in a given sentence are most important and replace them with rare synonyms.

3 Model

3.1 Form-Context Model

We first review the basis for our new model, the form-context model (FCM) (Schick and Schütze, 2019b). Given a set of d-dimensional high-quality embeddings for frequent words, FCM induces embeddings for rare words that are appropriate for the given embedding space. This is done as follows: Given a word w and a context C in which it occurs, a *surface-form embedding* $v_{(w,C)}^{\text{form}} \in \mathbb{R}^d$ is obtained by averaging over embeddings of all character n-grams in w; the n-gram embeddings are learned during training. Similarly, a *context embedding* $v_{(w,C)}^{\text{context}} \in \mathbb{R}^d$ is obtained by averaging over the embeddings of all words in C. Finally, both embeddings are combined using a gate

$$g(v_{(w,C)}^{\text{form}}, v_{(w,C)}^{\text{context}}) = \sigma(x^\top [v_{(w,C)}^{\text{form}}; v_{(w,C)}^{\text{context}}] + y)$$

with parameters $x \in \mathbb{R}^{2d}, y \in \mathbb{R}$ and σ denoting the sigmoid function, allowing the model to decide how to weight surface-form and context. The final representation of w is then a weighted combination of form and context embeddings:

$$v_{(w,C)} = \alpha \cdot (Av_{(w,C)}^{\text{context}} + b) + (1-\alpha) \cdot v_{(w,C)}^{\text{form}}$$

where $\alpha = g(v_{(w,C)}^{\text{form}}, v_{(w,C)}^{\text{context}})$ and $A \in \mathbb{R}^{d \times d}, b \in \mathbb{R}^d$ are parameters learned during training.

The context part of FCM is able to capture the broad topic of rare words, but since it is a bag-of-words model, it is not capable of obtaining a more concrete or detailed understanding (see Schick and Schütze, 2019b). Furthermore, the simple gating mechanism results in only a shallow combination of form and context. That is, the model is not able to combine form and context until the very last step: While it can learn to weight form and context components, the two embeddings (form and context) do not share any information and thus do not influence each other.

3.2 BERTRAM

To overcome these limitations, we introduce BERTRAM, a model that combines a pretrained BERT language model (Devlin et al., 2019) with Attentive Mimicking (Schick and Schütze, 2019a). We denote with e_t the (uncontextualized, i.e., first-layer) embedding assigned to a (wordpiece) token t by BERT. Given a sequence of such uncontextualized embeddings $\mathbf{e} = e_1, \dots, e_n$, we denote by $\mathbf{h}_j(\mathbf{e})$ the contextualized representation of the j-th token at the final layer when the model is given $\mathbf{e}$ as input.

Given a word w and a context C in which it occurs, let $\mathbf{t} = t_1, \dots, t_m$ be the sequence obtained from C by (i) replacing w with a `[MASK]` token and (ii) tokenization (matching BERT's vocabulary); furthermore, let i denote the index for which $t_i =$ `[MASK]`. We experiment with three variants of BERTRAM: BERTRAM-SHALLOW, BERTRAM-REPLACE and BERTRAM-ADD.[2]

SHALLOW. Perhaps the simplest approach for obtaining a context embedding from C using BERT is to define

$$v_{(w,C)}^{\text{context}} = \mathbf{h}_i(e_{t_1}, \dots, e_{t_m}) \,.$$

This approach aligns well with BERT's pretraining objective of predicting likely substitutes for `[MASK]` tokens from their contexts. The context embedding $v_{(w,C)}^{\text{context}}$ is then combined with its form counterpart as in FCM.

While this achieves our first goal of using a more sophisticated context model that goes beyond bag-of-words, it still only combines form and context in a shallow fashion.

REPLACE. Before computing the context embedding, we replace the uncontextualized embedding of the `[MASK]` token with the word's surface-form embedding:

$$v_{(w,C)}^{\text{context}} = \mathbf{h}_i(e_{t_1}, \dots, e_{t_{i-1}}, v_{(w,C)}^{\text{form}}, e_{t_{i+1}}, \dots, e_{t_m}) \,.$$

Our rationale for this is as follows: During regular BERT pretraining, words chosen for prediction are replaced with `[MASK]` tokens only 80% of the time and kept unchanged 10% of the time. Thus, standard pretrained BERT should be able to make use of form embeddings presented this way as they provide a strong signal with regards to how the "correct" embedding of w may look like.

ADD. Before computing the context embedding, we prepad the input with the surface-form embedding of w, followed by a colon ($e_:$):[3]

$$v_{(w,C)}^{\text{context}} = \mathbf{h}_{i+2}(v_{(w,C)}^{\text{form}}, e_:, e_{t_1}, \dots, e_{t_m}) \,.$$

The intuition behind this third variant is that lexical definitions and explanations of a word w are occasionally prefixed by "w :" (e.g., in some online dictionaries). We assume that BERT has seen many definitional sentences of this kind during pretraining and is thus able to leverage surface-form information about w presented this way.

For both REPLACE and ADD, surface-form information is directly and deeply integrated into the

[2]We refer to these three BERTRAM configurations simply as SHALLOW, REPLACE and ADD.

[3]We experimented with other prefixes, but found that this variant is best capable of recovering w at the masked position.

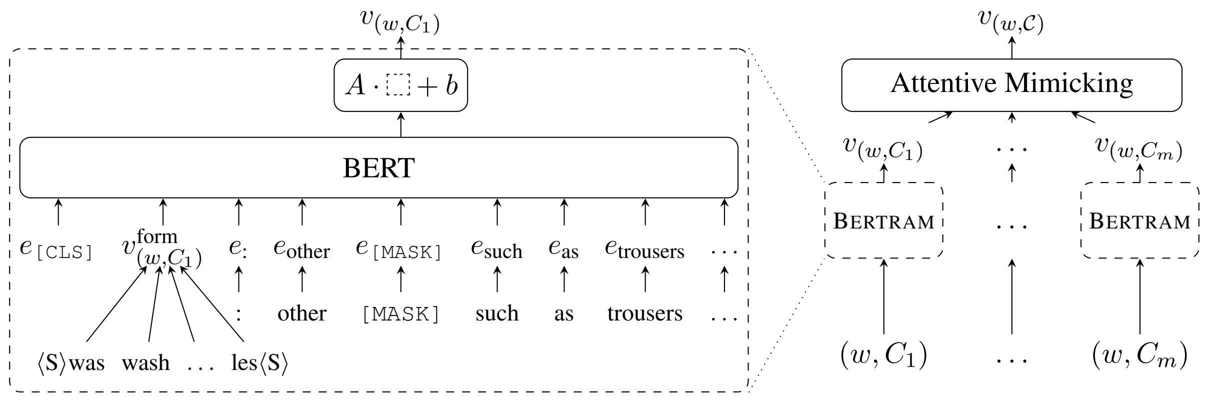

Figure 2: Schematic representation of BERTRAM-ADD processing the input word w = "washables" given a single context C_1 = "other washables such as trousers …" (left) and given multiple contexts $\mathcal{C} = \{C_1, \dots, C_m\}$ (right)

computation of the context embedding; thus, we do not require any gating mechanism and directly set $v_{(w,C)} = A \cdot v^{\text{context}}_{(w,C)} + b$. Figure 2 (left) shows how a single context is processed using ADD.

To exploit multiple contexts of a word if available, we follow the approach of Schick and Schütze (2019a) and add an AM layer on top of our model; see Figure 2 (right). Given a set of contexts $\mathcal{C} = \{C_1, \dots, C_m\}$ and the corresponding embeddings $v_{(w,C_1)}, \dots, v_{(w,C_m)}$, AM applies a self-attention mechanism to all embeddings, allowing the model to distinguish informative from uninformative contexts. The final embedding $v_{(w,\mathcal{C})}$ is then a weighted combination of all embeddings:

$$v_{(w,\mathcal{C})} = \sum\nolimits_{i=1}^{m} \rho_i \cdot v_{(w,C_i)}$$

where the self-attention layer determines the weights ρ_i subject to $\sum_{i=1}^{m} \rho_i = 1$. For further details, see Schick and Schütze (2019a).

3.3 Training

Like previous work, we use *mimicking* (Pinter et al., 2017) as a training objective. That is, given a frequent word w with known embedding e_w and a set of corresponding contexts $\mathcal{C}$, BERTRAM is trained to minimize $\|e_w - v_{(w,\mathcal{C})}\|^2$.

Training BERTRAM end-to-end is costly: the cost of processing a single training instance $(w, \mathcal{C})$ with $\mathcal{C} = \{C_1, \dots, C_m\}$ is the same as processing an entire batch of m examples in standard BERT. Therefore, we resort to the following three-stage training process:

1. We train only the context part, minimizing $\|e_w - A \cdot (\sum_{i=1}^{m} \rho_i \cdot v^{\text{context}}_{(w,C_i)}) + b\|^2$ where ρ_i is the weight assigned to each context C_i through the AM layer. Regardless of the selected BERTRAM variant, the context embedding is always obtained using SHALLOW in this stage. Furthermore, only A, b and all parameters of the AM layer are optimized.

2. We train only the form part (i.e., only the n-gram embeddings); our loss for a single example $(w, \mathcal{C})$ is $\|e_w - v^{\text{form}}_{(w,\mathcal{C})}\|^2$. Training in this stage is completely detached from the underlying BERT model.

3. In the third stage, we combine the pretrained form-only and context-only models and train all parameters. The first two stages are only run once and then used for all three BERTRAM variants because context and form are trained in isolation. The third stage must be run for each variant separately.

We freeze all of BERT's parameters during training as we – somewhat surprisingly – found that this slightly improves the model's performance while speeding up training. For ADD, we additionally found it helpful to freeze the form part in the third training stage. Importantly, for the first two stages of our training procedure, we do not have to backpropagate through BERT to obtain all required gradients, drastically increasing the training speed.

4 Dataset Rarification

The ideal dataset for measuring the quality of rare word representations would be one for which the accuracy of a model with no understanding of rare words is 0% whereas the accuracy of a model that perfectly understands rare words is 100%. Unfortunately, existing datasets do not satisfy this desideratum, not least because rare words – by their nature – occur rarely.

This does not mean that rare words are not important: As we shift our focus in NLP from words and sentences as the main unit of processing to larger units like paragraphs and documents, rare words will occur in a high proportion of such larger "evaluation units". Rare words are also clearly a hallmark of human language competence, which should be the ultimate goal of NLP. Our work is part of a trend that sees a need for evaluation tasks in NLP that are more ambitious than what we have now.[4]

To create more challenging datasets, we use *rarification*, a procedure that automatically transforms existing text classification datasets in such a way that rare words become important. We require a pretrained language model M as a baseline, an arbitrary text classification dataset $\mathcal{D}$ containing labeled instances $(\mathbf{x}, y)$ and a *substitution dictionary* S, mapping each word w to a set of rare synonyms $S(w)$. Given these ingredients, our procedure consists of three steps: (i) splitting the dataset into a train set and a set of test candidates, (ii) training the baseline model on the train set and (iii) modifying a subset of the test candidates to generate the final test set.

Dataset Splitting. We partition $\mathcal{D}$ into a training set $\mathcal{D}_{\text{train}}$ and a set of *test candidates*, $\mathcal{D}_{\text{cand}}$. $\mathcal{D}_{\text{cand}}$ contains all instances $(\mathbf{x}, y) \in \mathcal{D}$ such that for at least one word w in $\mathbf{x}$, $S(w) \neq \emptyset$ – subject to the constraint that the training set contains at least one third of the entire data.

Baseline Training. We finetune M on $\mathcal{D}_{\text{train}}$. Let $(\mathbf{x}, y) \in \mathcal{D}_{\text{train}}$ where $\mathbf{x} = w_1, \ldots, w_n$ is a sequence of words. We deviate from the finetuning procedure of Devlin et al. (2019) in three respects:

- We randomly replace 5% of all words in $\mathbf{x}$ with a `[MASK]` token. This allows the model to cope with missing or unknown words, a prerequisite for our final test set generation.

- As an alternative to overwriting the language model's uncontextualized embeddings for rare words, we also want to allow models to *add* an alternative representation during test time, in which case we simply separate both representations by a slash (cf. §5.3). To accustom the language model to this duplication of words, we replace each word w_i with "w_i / w_i" with a probability of 10%. To make sure that the model does not simply learn to always focus on the first instance during training, we randomly mask each of the two repetitions with probability 25%.

- We do not finetune the model's embedding layer. We found that this does not hurt performance, an observation in line with recent findings of Lee et al. (2019).

Test Set Generation. Let $p(y \mid \mathbf{x})$ be the probability that the finetuned model M assigns to class y given input $\mathbf{x}$, and $M(\mathbf{x}) = \arg\max_{y \in \mathcal{Y}} p(y \mid \mathbf{x})$ be the model's prediction for input $\mathbf{x}$ where $\mathcal{Y}$ denotes the set of all labels. For generating our test set, we only consider candidates that are classified *correctly* by the baseline model, i.e., candidates $(\mathbf{x}, y) \in \mathcal{D}_{\text{cand}}$ with $M(\mathbf{x}) = y$. For each such entry, let $\mathbf{x} = w_1, \ldots, w_n$ and let $\mathbf{x}_{w_i = t}$ be the sequence obtained from $\mathbf{x}$ by replacing w_i with t. We compute

$$w_i = \underset{w_j : S(w_j) \neq \emptyset}{\arg\min}\; p(y \mid \mathbf{x}_{w_j = \texttt{[MASK]}}),$$

i.e., we select the word w_i whose masking pushes the model's prediction the farthest away from the correct label. If removing this word already changes the model's prediction – that is, $M(\mathbf{x}_{w_i = \texttt{[MASK]}}) \neq y$ –, we select a random rare synonym $\hat{w}_i \in S(w_i)$ and add $(\mathbf{x}_{w_i = \hat{w}_i}, y)$ to the test set. Otherwise, we repeat the above procedure; if the label still has not changed after masking up to 5 words, we discard the candidate. Each instance $(\mathbf{x}_{w_{i_1} = \hat{w}_{i_1}, \ldots, w_{i_k} = \hat{w}_{i_k}}, y)$ of the resulting test set has the following properties:

- If each w_{i_j} is replaced by `[MASK]`, the entry is classified incorrectly by M. In other words, understanding the words w_{i_j} is necessary for M to determine the correct label.

- If the model's internal representation of each $\hat{w}_{i_j}$ is sufficiently similar to its representation of w_{i_j}, the entry is classified correctly by M. That is, if the model is able to understand the rare words $\hat{w}_{i_j}$ and to identify them as synonyms of w_{i_j}, it will predict the correct label.

[4] Cf. (Bowman, 2019): "If we want to be able to establish fair benchmarks that encourage future progress toward robust, human-like language understanding, we'll need to get better at creating clean, challenging, and realistic test datasets."

Model	RARE	MEDIUM
BERT (base)	0.112	0.234
+ AM (Schick and Schütze, 2020)	0.251	0.267
+ BERTRAM-SHALLOW	0.250	0.246
+ BERTRAM-REPLACE	0.155	0.216
+ BERTRAM-ADD	**<u>0.269</u>**	**<u>0.367</u>**
BERT (large)	0.143	0.264
RoBERTa (large)	0.270	0.275
+ BERTRAM-ADD	**<u>0.306</u>**	**<u>0.323</u>**

Table 1: MRR on WNLaMPro test for baseline models and various BERTRAM configurations. Best results per base model are underlined, results that do not differ significantly from the best results in a paired t-test ($p < 0.05$) are bold.

Task	**Entry**
MNLI	i think i will go finish up my ~~laundry~~ **washables**.
AG's	[. . .] stake will ~~improve~~ **meliorate** symantec's consulting contacts [. . .]
DBPedia	yukijiro hotaru [. . .] is a ~~japanese~~ **nipponese** ~~actor~~ **histrion**.
MNLI	a smart person is ~~often~~ **ofttimes** correct in their ~~answers~~ **ansers**.
MNLI	the southwest has a lot of farming and ~~vineyards~~ **vineries** that make ~~excellent~~ **fantabulous** merlot.

Table 2: Examples from rarified datasets. Crossed out: replaced words. Bold: replacements.

Note that the test set is closely coupled to the baseline model M because we select the words to be replaced based on M's predictions. Importantly, however, the model is never queried with any rare synonym during test set generation, so its representations of rare words are *not* taken into account for creating the test set. Thus, while the test set is not suitable for comparing M with an entirely different model M', it allows us to compare various strategies for representing rare words in the embedding space of M. Definitional Nonce (Herbelot and Baroni, 2017) is subject to a similar constraint: it is tied to a specific (uncontextualized) embedding space based on Word2Vec (Mikolov et al., 2013).

5 Evaluation

5.1 Setup

For our evaluation of BERTRAM, we follow the experimental setup of Schick and Schütze (2020). We experiment with integrating BERTRAM both into $\text{BERT}_{\text{base}}$ and $\text{RoBERTa}_{\text{large}}$ (Liu et al., 2019b). Throughout our experiments, when BERTRAM is used to provide input representations for one of the two models, we use the same model as BERTRAM's underlying language model. Further training specifications can be found in Appendix A.

While BERT was trained on BookCorpus (Zhu et al., 2015) and a large Wikipedia dump, we follow previous work and train BERTRAM only on the much smaller Westbury Wikipedia Corpus (WWC) (Shaoul and Westbury, 2010); this of course gives BERT a clear advantage over BERTRAM. This advantage is even more pronounced when comparing BERTRAM with RoBERTa, which is trained on a corpus that is an order of magnitude larger than the original BERT corpus. We try to at least partially compensate for this as follows: In our downstream task experiments, we gather the set of contexts $\mathcal{C}$ for each word from WWC+BookCorpus during inference.[5]

5.2 WNLaMPro

We evaluate BERTRAM on the WNLaMPro dataset (Schick and Schütze, 2020). This dataset consists of cloze-style phrases like "A *lingonberry* is a ____." and the task is to correctly fill the slot (____) with one of several acceptable target words (e.g., "fruit", "bush" or "berry"), which requires understanding of the meaning of the phrase's *keyword* ("lingonberry" in the example). As the goal of this dataset is to probe a language model's ability to understand rare words *without* any task-specific finetuning, Schick and Schütze (2020) do not provide a training set. The dataset is partitioned into three subsets based on the keyword's frequency in WWC: RARE (occurring fewer than 10 times) MEDIUM (occurring between 10 and 100 times), and FREQUENT (all remaining words).

For our evaluation, we compare the performance of a standalone BERT (or RoBERTa) model with one that uses BERTRAM as shown in Figure 1 (bottom). As our focus is to improve representations for rare words, we evaluate our model only on WNLaMPro RARE and MEDIUM. Table 1 gives results; our measure is mean reciprocal rank (MRR). We see that supplementing BERT with any of the proposed methods results in noticeable improvements for the RARE subset, with ADD clearly outperforming SHALLOW and REPLACE. Moreover, ADD performs surprisingly well for more frequent words, improving the score for WNLaMPro-MEDIUM by

[5]We recreate BookCorpus with the script at `github.com/soskek/bookcorpus`. We refer to the joined corpus of WWC and BookCorpus as WWC+BookCorpus.

Model	MNLI			AG's News			DBPedia		
	All	Msp	WN	All	Msp	WN	All	Msp	WN
BERT (base)	50.5	49.1	53.4	56.5	54.8	61.9	49.3	46.0	57.6
+ Mimick (Pinter et al., 2017)	37.2	38.2	38.7	45.3	43.9	50.5	36.5	35.8	41.1
+ A La Carte (Khodak et al., 2018)	44.6	45.7	46.1	52.4	53.7	56.1	51.1	48.7	59.3
+ AM (Schick and Schütze, 2020)	50.9	50.7	53.6	58.9	59.8	62.6	60.7	63.1	62.8
+ BERTRAM	53.3	52.5	55.6	**62.1**	**63.1**	**65.3**	64.2	67.9	64.1
+ BERTRAM-SLASH	56.4	55.3	58.6	**62.9**	**63.3**	**65.3**	65.7	67.3	67.2
+ BERTRAM-SLASH + INDOMAIN	**59.8**	**57.3**	**62.7**	**62.5**	**62.1**	**66.6**	**74.2**	**74.8**	**76.7**
RoBERTa (large)	67.3	68.7	68.4	63.7	68.1	65.7	65.5	67.3	66.6
+ BERTRAM-SLASH	70.1	**71.5**	70.9	64.6	68.4	64.9	71.9	73.8	73.9
+ BERTRAM-SLASH + INDOMAIN	**71.7**	**71.9**	**73.2**	**68.1**	**71.9**	**69.0**	**76.0**	**78.8**	**77.3**

Table 3: Accuracy of standalone BERT and RoBERTa, various baselines and BERTRAM on rarified MNLI, AG's News and DBPedia. The five BERTRAM instances are BERTRAM-ADD. Best results per baseline model are underlined, results that do not differ significantly from the best results in a two-sided binomial test ($p < 0.05$) are bold. Msp/WN: subset of instances containing at least one misspelling/synonym. All: all instances.

58% compared to $\text{BERT}_{\text{base}}$ and 37% compared to Attentive Mimicking. This makes sense considering that the key enhancement of BERTRAM over AM lies in improving context representations and interconnection of form and context; the more contexts are given, the more this comes into play. Noticeably, despite being both based on and integrated into a $\text{BERT}_{\text{base}}$ model, our architecture even outperforms $\text{BERT}_{\text{large}}$ by a large margin. While RoBERTa performs much better than BERT on WNLaMPro, BERTRAM still significantly improves results for both rare and medium frequency words. As it performs best for both the RARE and MEDIUM subset, we always use the ADD configuration of BERTRAM in the following experiments.

5.3 Downstream Task Datasets

To measure the effect of adding BERTRAM to a pretrained deep language model on downstream tasks, we rarify (cf. §4) the following three datasets:

- MNLI (Williams et al., 2018), a natural language inference dataset where given two sentences a and b, the task is to decide whether a entails b, a and b contradict each other or neither;
- AG's News (Zhang et al., 2015), a news classification dataset with four different categories (*world*, *sports*, *business* and *science/tech*);
- DBPedia (Lehmann et al., 2015), an ontology dataset with 14 classes (e.g., *company*, *artist*) that have to be identified from text snippets.

For all three datasets, we create rarified instances both using $\text{BERT}_{\text{base}}$ and $\text{RoBERTa}_{\text{large}}$ as a baseline model and build the substitution dictionary S using the synonym relation of WordNet (Miller, 1995) and the *pattern* library (Smedt and Daelemans, 2012) to make sure that all synonyms have consistent parts of speech. Furthermore, we only consider synonyms for each word's most frequent sense; this filters out much noise and improves the quality of the created sentences. In addition to WordNet, we use the misspelling dataset of Piktus et al. (2019). To prevent misspellings from dominating the resulting datasets, we only assign misspelling-based substitutes to randomly selected 10% of the words contained in each sentence. Motivated by the results on WNLaMPro-MEDIUM, we consider every word that occurs less than 100 times in WWC+BookCorpus as being rare. Example entries from the rarified datasets obtained using $\text{BERT}_{\text{base}}$ as a baseline model can be seen in Table 2. The average number of words replaced with synonyms or misspellings is 1.38, 1.82 and 2.34 for MNLI, AG's News and DBPedia, respectively.

Our default way of injecting BERTRAM embeddings into the baseline model is to replace the sequence of uncontextualized subword token embeddings for a given rare word with its BERTRAM-based embedding (Figure 1, bottom). That is, given a sequence of uncontextualized token embeddings $\mathbf{e} = e_1, \ldots, e_n$ where $e_i, \ldots, e_j$ with $1 \leq i \leq j \leq n$ is the sequence of embeddings for a single rare word w with BERTRAM-based embedding $v_{(w,\mathcal{C})}$, we replace $\mathbf{e}$ with

$$\mathbf{e}' = e_1, \ldots, e_{i-1}, v_{(w,\mathcal{C})}, e_{j+1}, \ldots, e_n\,.$$

As an alternative to replacing the original sequence of subword embeddings for a given rare word, we also consider BERTRAM-SLASH, a configuration where the BERTRAM-based embedding is simply added and both representations are separated using a single slash:

$$\mathbf{e}_{\text{SLASH}} = e_1, \ldots, e_j, e_{/}, v_{(w,\mathcal{C})}, e_{j+1}, \ldots, e_n\,.$$

The intuition behind this variant is that in BERT's pretraining corpus, a slash is often used to separate two variants of the same word (e.g., "useable / usable") or two closely related concepts (e.g., "company / organization", "web-based / cloud") and thus, BERT should be able to understand that both $e_i, \ldots, e_j$ and $v_{(w,\mathcal{C})}$ refer to the same entity. We therefore surmise that whenever some information is encoded in one representation but not in the other, giving BERT both representations is helpful.

By default, the set of contexts $\mathcal{C}$ for each word is obtained by collecting all sentences from WWC+BookCorpus in which it occurs. We also try a variant where we add in-domain contexts by giving BERTRAM access to all texts (but not labels) found in the test set; we refer to this variant as INDOMAIN.[6] Our motivation for including this variant is as follows: Moving from the training stage of a model to its production use often causes a slight domain shift. This is turn leads to an increased number of input sentences containing words that did not – or only very rarely – appear in the training data. However, such input sentences can easily be collected as additional unlabeled examples during production use. While there is no straightforward way to leverage these unlabeled examples with an already finetuned BERT model, BERTRAM can easily make use of them without requiring any labels or any further training: They can simply be included as additional contexts during inference. As this gives BERTRAM a slight advantage, we also report results for all configurations without using indomain data. Importantly, adding indomain data increases the number of contexts for more than 90% of all rare words by at most 3, meaning that they can still be considered rare despite the additional indomain contexts.

Table 3 reports, for each task, the accuracy on the entire dataset (All) as well as scores obtained considering only instances where at least one word was replaced by a misspelling (Msp) or a WordNet synonym (WN), respectively.[7] Consistent with results

[6]For the MNLI dataset, which consists of text pairs (a, b), we treat a and b as separate contexts.

[7]Note that results for BERT and RoBERTa are only loosely comparable because the datasets generated from both baseline models through rarification are different.

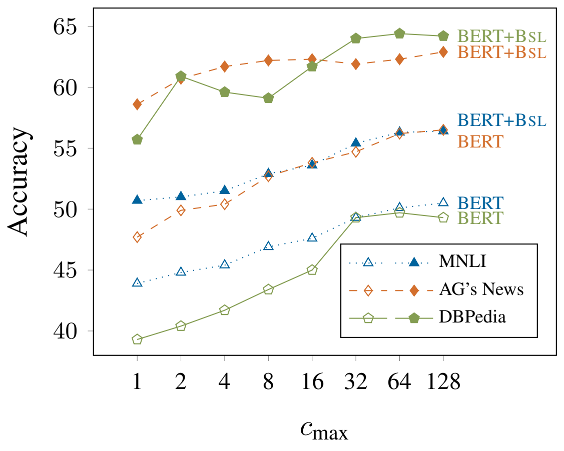

Figure 3: BERT vs. BERT combined with BERTRAM-SLASH (BERT+BSL) on three downstream tasks for varying maximum numbers of contexts $c_{\max}$

on WNLaMPro, combining BERT with BERTRAM consistently outperforms both a standalone BERT model and one combined with various baseline models. Using the SLASH variant brings improvements across all datasets as does adding INDOMAIN contexts (exception: BERT/AG's News). This makes sense considering that for a rare word, every single additional context can be crucial for gaining a deeper understanding. Correspondingly, it is not surprising that the benefit of adding BERTRAM to RoBERTa is less pronounced, because BERTRAM uses only a fraction of the contexts available to RoBERTa during pretraining. Nonetheless, adding BERTRAM significantly improves RoBERTa's accuracy for all three datasets both with and without adding INDOMAIN contexts.

To further understand for which words using BERTRAM is helpful, Figure 3 looks at the accuracy of BERT$_{\text{base}}$ both with and without BERTRAM as a function of word frequency. That is, we compute the accuracy scores for both models when considering only entries $(\mathbf{x}_{w_{i_1}=\hat{w}_{i_1},\ldots,w_{i_k}=\hat{w}_{i_k}}, y)$ where each substituted word $\hat{w}_{i_j}$ occurs less than $c_{\max}$ times in WWC+BookCorpus, for different values of $c_{\max}$. As one would expect, $c_{\max}$ is positively correlated with the accuracies of both models, showing that the rarer a word is, the harder it is to understand. Interestingly, the gap between standalone BERT and BERT with BERTRAM remains more or less constant regardless of $c_{\max}$. This suggests that using BERTRAM may even be helpful for more frequent words.

To investigate this hypothesis, we perform another rarification of MNLI that differs from the

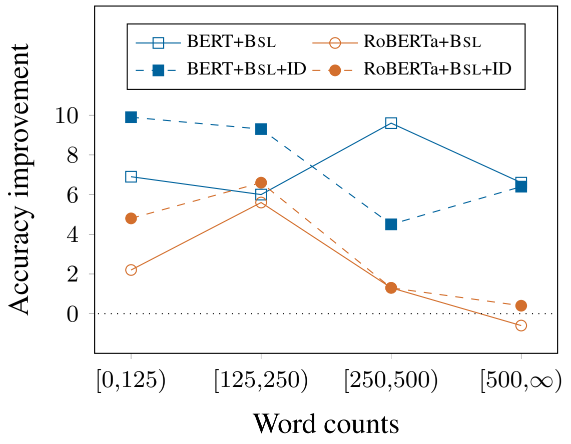

Figure 4: Improvements for BERT (base) and RoBERTa (large) when adding BERTRAM-SLASH (+BSL) or BERTRAM-SLASH + INDOMAIN (+BSL+ID) on MNLI-1000

previous rarification in two respects. First, we increase the threshold for a word to count as rare from 100 to 1000. Second, as this means that we have more WordNet synonyms available, we do not use the misspelling dictionary (Piktus et al., 2019) for substitution. We refer to the resulting datasets for BERT$_{\text{base}}$ and RoBERTa$_{\text{large}}$ as *MNLI-1000*.

Figure 4 shows results on MNLI-1000 for various rare word frequency ranges. For each value $[c_0, c_1)$ on the x-axis, the y-axis shows improvement in accuracy compared to standalone BERT or RoBERTa when only dataset entries are considered for which each rarified word occurs between c_0 (inclusively) and c_1 (exclusively) times in WWC+BooksCorpus. We see that for words with frequency less than 125, the improvement in accuracy remains similar even without using misspellings as another source of substitutions. Interestingly, for every single interval of rare word counts considered, adding BERTRAM-SLASH to BERT considerably improves its accuracy. For RoBERTa, adding BERTRAM brings improvements only for words occurring less than 500 times. While using INDOMAIN data is beneficial for rare words – simply because it gives us additional contexts for these words –, when considering only words that occur at least 250 times in WWC+BookCorpus, adding INDOMAIN contexts does not help.

6 Conclusion

We have introduced BERTRAM, a novel architecture for inducing high-quality representations for rare words in BERT's and RoBERTa's embedding spaces. This is achieved by employing a powerful pretrained language model and deeply integrating surface-form and context information. By replacing important words with rare synonyms, we created downstream task datasets that are more challenging and support the evaluation of NLP models on the task of understanding rare words, a capability that human speakers have. On all of these datasets, BERTRAM improves over standard BERT and RoBERTa, demonstrating the usefulness of our method.

Our analysis showed that BERTRAM is beneficial not only for rare words (our main target in this paper), but also for frequent words. In future work, we want to investigate BERTRAM's potential benefits for such frequent words. Furthermore, it would be interesting to explore more complex ways of incorporating surface-form information – e.g., by using a character-level CNN similar to the one of Kim et al. (2016) – to balance out the potency of BERTRAM's form and context parts.

Acknowledgments

This work was funded by the European Research Council (ERC #740516). We would like to thank the anonymous reviewers for their helpful comments.

References

Alexei Baevski, Sergey Edunov, Yinhan Liu, Luke Zettlemoyer, and Michael Auli. 2019. Cloze-driven pretraining of self-attention 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 5359–5368, Hong Kong, China. Association for Computational Linguistics.

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.

Sam Bowman. 2019. Google T5 explores the limits of transfer learning.

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.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

Jeroen Van Hautte, Guy Emerson, and Marek Rei. 2019. Bad form: Comparing context-based and form-based few-shot learning in distributional semantic models. *Computing Research Repository*, arXiv:1910.00275.

Aurélie Herbelot and Marco Baroni. 2017. High-risk learning: acquiring new word vectors from tiny data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 304–309. Association for Computational Linguistics.

Mikhail Khodak, Nikunj Saunshi, Yingyu Liang, Tengyu Ma, Brandon Stewart, and Sanjeev Arora. 2018. A la carte embedding: Cheap but effective induction of semantic feature vectors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12–22. Association for Computational Linguistics.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. 2016. Character-aware neural language models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2741–2749. AAAI Press.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations (ICLR)*.

Angeliki Lazaridou, Marco Marelli, and Marco Baroni. 2017. Multimodal word meaning induction from minimal exposure to natural text. *Cognitive Science*, 41(S4):677–705.

Jaejun Lee, Raphael Tang, and Jimmy Lin. 2019. What would Elsa do? Freezing layers during transformer fine-tuning. *Computing Research Repository*, arXiv:1911.03090.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 6(2):167–195.

Qianchu Liu, Diana McCarthy, and Anna Korhonen. 2019a. Second-order contexts from lexical substitutes for few-shot learning of word representations. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 61–67, Minneapolis, Minnesota. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. RoBERTa: A robustly optimized BERT pretraining approach. *Computing Research Repository*, arXiv:1907.11692.

Thang Luong, Richard Socher, and Christopher Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113, Sofia, Bulgaria. Association for Computational Linguistics.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional LSTM. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 51–61, Berlin, Germany. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Computing Research Repository*, arXiv:1301.3781.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*.

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, New Orleans, Louisiana. Association for Computational Linguistics.

Aleksandra Piktus, Necati Bora Edizel, Piotr Bojanowski, Edouard Grave, Rui Ferreira, and Fabrizio Silvestri. 2019. Misspelling oblivious word embeddings. 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 3226–3234, Minneapolis, Minnesota. Association for Computational Linguistics.

Yuval Pinter, Robert Guthrie, and Jacob Eisenstein. 2017. Mimicking word embeddings using subword RNNs. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 102–112. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alexandre Salle and Aline Villavicencio. 2018. Incorporating subword information into matrix factorization word embeddings. In *Proceedings of the Second Workshop on Subword/Character LEvel Models*, pages 66–71, New Orleans. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2019a. Attentive mimicking: Better word embeddings by attending to informative contexts. 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 489–494, Minneapolis, Minnesota. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2019b. Learning semantic representations for novel words: Leveraging both form and context. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*.

Timo Schick and Hinrich Schütze. 2020. Rare words: A major problem for contextualized embeddings and how to fix it by attentive mimicking. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Cyrus Shaoul and Chris Westbury. 2010. The westbury lab wikipedia corpus.

Tom De Smedt and Walter Daelemans. 2012. Pattern for python. *Journal of Machine Learning Research*, 13(Jun):2063–2067.

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. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Transformers: State-of-the-art natural language processing.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *Computing Research Repository*, arXiv:1609.08144.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. *Computing Research Repository*, arXiv:1906.08237.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 649–657. Curran Associates, Inc.

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

A Training Details

Our implementation of BERTRAM is based on PyTorch (Paszke et al., 2017) and the Transformers library (Wolf et al., 2019). To obtain target embeddings for frequent multi-token words (i.e., words that occur at least 100 times in WWC+BookCorpus) during training, we use one-token approximation (OTA) (Schick and Schütze, 2020). For RoBERTa$_{\text{large}}$, we found increasing the number of iterations per word from $4{,}000$ to $8{,}000$ to produce better OTA embeddings using the same evaluation setup as Schick and Schütze (2020). For all stages of training, we use Adam (Kingma and Ba, 2015) as optimizer.

Context-Only Training. During the first stage of our training process, we train BERTRAM with a maximum sequence length of 96 and a batch size of 48 contexts for BERT$_{\text{base}}$ and 24 contexts for RoBERTa$_{\text{large}}$. These parameters are chosen such that a batch fits on a single Nvidia GeForce GTX 1080Ti. Each context in a batch is mapped to a word w from the set of training words, and each batch contains at least 4 and at most 32 contexts per word. For BERT$_{\text{base}}$ and RoBERTa$_{\text{large}}$, we pretrain the context part for 5 and 3 epochs, respectively. We use a maximum learning rate of $5 \cdot 10^{-5}$ and perform linear warmup for the first 10% of training examples, after which the learning rate is linearly decayed.

Form-Only Training. In the second stage of our training process, we use the same parameters as Schick and Schütze (2020), as our form-only model is the very same as theirs. That is, we use a learning rate of 0.01, a batch size of 64 words and we apply n-gram dropout with a probability of 10%. We pretrain the form-only part for 20 epochs.

Combined Training. For the final stage, we use the same training configuration as for context-only training, but we keep n-gram dropout from the form-only stage. We perform combined training for 3 epochs. For ADD, when using RoBERTa as an underlying language model, we do not just prepad the input with the surface-form embedding followed by a colon, but additionally wrap the surface-form embedding in double quotes. That is, we prepad the input with $e_{"}, v^{\text{form}}_{(w,C)}, e_{"}, e_{:}$. We found this to perform slightly better in preliminary experiments with some toy examples.

B Evaluation Details

WNLaMPro In order to ensure comparability with results of Schick and Schütze (2020), we use only WWC to obtain contexts for WNLaMPro keywords.

Rarified Datasets To obtain rarified instances of MNLI, AG's News and DBPedia, we train BERT$_{\text{base}}$ and RoBERTa$_{\text{large}}$ on each task's training set for 3 epochs. We use a batch size of 32, a maximum sequence length of 128 and a weight decay factor of 0.01. For BERT, we perform linear warmup for the first 10% of training examples and use a maximum learning rate of $5 \cdot 10^{-5}$. After reaching its peak value, the learning rate is linearly decayed. For RoBERTa, we found training to be unstable with these parameters, so we chose a lower learning rate of $1 \cdot 10^{-5}$ and performed linear warmup for the first $10{,}000$ training steps.

To obtain results for our baselines on the rarified datasets, we use the original Mimick implementation of Pinter et al. (2017), the A La Carte implementation of Khodak et al. (2018) and the Attentive Mimicking implementation of Schick and Schütze (2019a) with their default hyperparameter settings. As A La Carte can only be used for words with at least one context, we keep the original BERT embeddings whenever no such context is available.

While using BERTRAM allows us to completely remove the original BERT embeddings for all rare words and still obtain improvements in accuracy on all three rarified downstream tasks, the same is not true for RoBERTa, where removing the original sequence of subword token embeddings for a given rare word (i.e., not using the SLASH variant) hurts performance with accuracy dropping by 5.6, 7.4 and 2.1 points for MNLI, AG's News and DBPedia, respectively. We believe this to be due to the vast amount of additional contexts for rare words in RoBERTa's training set that are not available to BERTRAM.