

# Rare Words: A Major Problem for Contextualized Embeddings And How to Fix it by Attentive Mimicking

**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 neural network architectures with a language modeling objective has brought large improvements for many natural language processing tasks. Exemplified by BERT, a recently proposed such architecture, we demonstrate that despite being trained on huge amounts of data, deep language models still struggle to understand rare words. To fix this problem, we adapt Attentive Mimicking, a method that was designed to explicitly learn embeddings for rare words, to deep language models. In order to make this possible, we introduce *one-token approximation*, a procedure that enables us to use Attentive Mimicking even when the underlying language model uses subword-based tokenization, i.e., it does not assign embeddings to all words. To evaluate our method, we create a novel dataset that tests the ability of language models to capture semantic properties of words without any task-specific fine-tuning. Using this dataset, we show that adding our adapted version of Attentive Mimicking to BERT does substantially improve its understanding of rare words.

## 1 Introduction

Distributed representations of words are a key component of many natural language processing (NLP) systems. In particular, deep contextualized representations learned using an unsupervised language modeling objective (Peters et al., 2018) have led to large performance gains for a variety of NLP tasks. Very recently, several authors have proposed to not only use language modeling for feature extraction, but to fine-tune entire language models for specific tasks (Radford et al., 2018a; Howard and Ruder, 2018). Taking up this idea, Devlin et al. (2018) introduced BERT, a bidirectional language model based on a transformer architecture (Vaswani et al., 2017) that has achieved a new state-of-the-art for several NLP tasks.

---

<b>Q:</b>	A <i>lime</i> is a ____.
<b>A:</b>	lime, lemon, fruit
<b>Q:</b>	A <i>bicycle</i> is a ____.
<b>A:</b>	bicycle, motorcycle, bike

---

<b>Q:</b>	A <i>kumquat</i> is a ____.
<b>A:</b>	noun, horse, dog
<b>Q:</b>	A <i>unicycle</i> is a ____.
<b>A:</b>	structure, unit, chain

---

Table 1: Example queries and most probable outputs of BERT for frequent (top) and rare words (bottom)

As demonstrated by Radford et al. (2018b), it is possible for language models to solve a diverse set of tasks to some extent *without* any form of task-specific fine-tuning. This can be achieved by simply presenting the tasks in form of natural language sentences that are to be completed by the model. The very same idea can also be used to test how well a language model understands a given word: we can “ask” it for properties of that word using natural language. For example, a language model that understands the concept of “guilt” should be able to correctly complete the sentence “Guilt is the opposite of \_\_\_\_.” with the word “innocence”.

The examples in Table 1 show that, according to this measure, BERT is indeed able to understand frequent words such as “lime” and “bicycle”: it predicts, among others, that the former is a fruit and the latter is the same as a bike. However, it fails terribly for both “kumquat” and “unicycle”, two less frequent words from the same domains. This poor performance raises the question whether deep language models generally struggle to understand rare words and, if so, how this weakness can be overcome.

To answer this question, we create a novel

dataset consisting of natural language queries for properties of words with varying frequencies. This dataset consists of (i) natural language patterns such as

<W> is a \_\_\_\_.

where <W> is a placeholder for a word to be investigated, and (ii) corresponding pairs of words and targets obtained using semantic relations extracted from WordNet (Miller, 1995).

Using this dataset, we then show that BERT indeed fails to understand many rare words. To overcome this limitation, we propose to apply Attentive Mimicking (Schick and Schütze, 2019b), a method that allows us to explicitly learn high-quality representations for rare words. A prerequisite for using this method is to have high-quality embeddings for as many words as possible, because it is trained to reproduce known word embeddings. However, many deep language models including BERT make use of byte-pair encoding (Sennrich et al., 2015) or similar subword tokenization algorithms. Thus, many words are not represented by a single token but by a sequence thereof and, accordingly, do not have their own, separate embeddings.

To solve this problem, we introduce *one-token approximation* (OTA), a method that allows us to approximately infer how the embeddings of arbitrary words would look like if they were represented by a single token. While we apply this method only to BERT, it can easily be adapted for other language modeling architectures.

In summary, our contributions are as follows:

- we introduce *WordNet Language Model Probing* (WNLAMPro), a novel dataset that aims to capture the ability of language models to understand specific words;
- using this dataset, we show that the ability of BERT to understand words depends highly on their frequency;
- we define one-token approximation (OTA), an approach that allows us to obtain embeddings for multi-token words that behave similar to the sequences of their token embeddings;
- we show that OTA enables us to apply Attentive Mimicking (Schick and Schütze, 2019b) to BERT, resulting in a substantially improved understanding of rare words.

## 2 Related Work

Using language modeling as a task to obtain contextualized representations of words was first proposed by Peters et al. (2018), who train a bidirectional LSTM (Hochreiter and Schmidhuber, 1997) language model for this task and then feed the so-obtained embeddings into task-specific architectures. Several authors extend this idea by transferring not only word embeddings, but entire language modeling architectures to specific tasks (Radford et al., 2018a; Howard and Ruder, 2018; Devlin et al., 2018). Whereas the GPT model proposed by Radford et al. (2018a) is strictly unidirectional (i.e., it looks only at the left context to predict the next word) and the ULMFiT method of Howard and Ruder (2018) uses a shallow concatenation of two unidirectional models, Devlin et al. (2018) design BERT as a deep bidirectional model using a transformer architecture and a masked language modeling task.

There are roughly two types of approaches for explicitly learning high-quality embeddings of rare words: surface-form-based approaches and context-based approaches. The former use subword information to infer a word’s meaning; this includes  $n$ -grams (Wieting et al., 2016; Bojanowski et al., 2017; Salle and Villavicencio, 2018), morphemes (Lazaridou et al., 2013; Luong et al., 2013) and characters (Pinter et al., 2017). On the other hand, context-based approaches take a look at the words surrounding a given rare word to obtain a representation for it (e.g. Herbelot and Baroni, 2017; Khodak et al., 2018). Recently, Schick and Schütze (2019a) introduced the *form-context model*, combining both approaches by jointly using surface-form and context information. The form-context model and its *Attentive Mimicking* variant (Schick and Schütze, 2019b) achieve a new state-of-the-art for high-quality representations of rare words.

Presenting tasks in the form of natural language sentences was recently proposed by McCann et al. (2018) as part of their *Natural Language Decathlon*, for which they frame ten different tasks as pairs of natural language questions and answers. They train models on triplets of questions, contexts and answers in a supervised fashion. An alternative, completely unsupervised approach proposed by Radford et al. (2018b) is to train a language model on a large corpus, present text specialized for a particular task and then let

the model complete this text. They achieve good performance on tasks such as reading comprehension, machine translation and question answering – without any form of task-specific fine-tuning.

Several existing datasets were designed to analyze the ability of word embeddings to capture semantic relations between words. For example, [Baroni and Lenci \(2011\)](#) introduce the BLESS dataset based on five different semantic relations – such as hyponymy and hypernymy –, compiled from multiple sources. [Weeds et al. \(2014\)](#) also create a dataset for semantic relations based on hypernyms and hyponyms using WordNet ([Miller, 1995](#)). These datasets differ from WNLamPro in two important respects: (i) they focus on frequent words by filtering out infrequent ones, whereas we explicitly want to analyze rare words, and (ii) they do not provide natural language patterns but either directly evaluate (uncontextualized) word embeddings using a similarity measure such as cosine distance or frame the task of identifying the relationship between two words as a supervised task.

### 3 Attentive Mimicking

*Attentive Mimicking* (AM) ([Schick and Schütze, 2019b](#)) is a method that, given a set of  $d$ -dimensional high-quality embeddings for frequent words, can be used to infer embeddings for infrequent words that are appropriate for the given embedding space. AM is an extension of the *form-context model* ([Schick and Schütze, 2019a](#)).

The key idea of the form-context model is to compute two distinct embeddings per word, where the first one exclusively uses the word’s surface-form and the other the word’s contexts, i.e., sentences in which the word was observed. Given a word  $w$  and a set of contexts  $\mathcal{C}$ , the surface-form embedding  $v_{(w,\mathcal{C})}^{\text{form}} \in \mathbb{R}^d$  is obtained by averaging over learned embeddings of all  $n$ -grams in  $w$ ; the context embedding  $v_{(w,\mathcal{C})}^{\text{context}} \in \mathbb{R}^d$  is the average over the known embeddings of all context words.

The final representation  $v_{(w,\mathcal{C})}$  of  $w$  is then a weighted sum of form embeddings and transformed context embeddings:

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

where  $A$  is a  $d \times d$  matrix and  $\alpha$  is a function of both embeddings, giving the model a chance to decide when to rely on the word’s surface form and when it is preferable to use its contexts. Specifi-

cally, [Schick and Schütze \(2019a\)](#) propose

$$\alpha = \sigma(u^\top [v_{(w,\mathcal{C})}^{\text{context}}; v_{(w,\mathcal{C})}^{\text{form}}] + b)$$

with  $u \in \mathbb{R}^{2d}$ ,  $b \in \mathbb{R}$  being learnable parameters and  $\sigma$  denoting the sigmoid function.

While the form-context model treats all contexts equally, AM extends it with a self-attention mechanism that is applied to all contexts, allowing the model to distinguish informative contexts from uninformative ones. The attention weight of each context is determined based on the idea that given a word  $w$ , two informative contexts  $C_1$  and  $C_2$  (i.e., contexts from which the meaning of  $w$  can be inferred) resemble each other more than two randomly chosen contexts in which  $w$  occurs. In other words, if many contexts for a word  $w$  are similar to each other, then it is reasonable to assume that they are more informative with respect to  $w$  than other contexts. [Schick and Schütze \(2019b\)](#) define the similarity between two contexts as

$$s(C_1, C_2) = \frac{(Mv_{C_1}) \cdot (Mv_{C_2})^\top}{\sqrt{d}}$$

with  $M \in \mathbb{R}^{d \times d}$  a learnable parameter and  $v_C$  denoting the average of embeddings for all words in a context  $C$ . The weight of a context is then defined as

$$\rho(C) \propto \sum_{C' \in \mathcal{C}} s(C, C').$$

Similar to earlier models (e.g. [Pinter et al., 2017](#)), the model is trained through *mimicking*. That is, we randomly sample words  $w$  and contexts  $\mathcal{C}$  from a large corpus and, given  $w$  and  $\mathcal{C}$ , ask the model to mimic the original embedding of  $w$ , i.e., to minimize the squared Euclidean distance between the original embedding and  $v_{(w,\mathcal{C})}$ .

#### 3.1 AM+CONTEXT

As we found in preliminary experiments that AM focuses heavily on the word’s surface form – an observation that is in line with results reported by [Schick and Schütze \(2019a\)](#) –, in addition to the default AM configuration of [Schick and Schütze \(2019b\)](#), we investigate another configuration AM+CONTEXT, which pushes the model to put more emphasis on a word’s contexts.<sup>1</sup> This is achieved by (i) increasing the minimum number of sampled contexts for each training instance

<sup>1</sup>Our implementation is publicly available at <https://github.com/timoschick/form-context-model>

from 1 to 8 and (ii) introducing  $n$ -gram dropout: during training, we randomly remove 10% of all surface-form  $n$ -grams for each training instance.

## 4 One-Token Approximation

As AM is trained through mimicking, it must be given high-quality embeddings of many words to learn how to make appropriate use of form and context information. Unfortunately, as many deep language models make use of subword-based tokenization, they assign embeddings to comparably few words. To overcome this limitation, we define *one-token approximation* (OTA). OTA finds an embedding for a multi-token word or phrase  $w$  that is similar to the embedding that  $w$  would have received if it had been a single token.

Let  $\Sigma$  denote the set of all characters and  $\mathcal{T} \subset \Sigma^*$  the set of all tokens used by the language model. Furthermore, let  $t : \Sigma^* \rightarrow \mathcal{T}^*$  be the tokenization function that splits each word into a sequence of tokens and  $e : \mathcal{T} \rightarrow \mathbb{R}^d$  the model’s token embedding function, which we extend to sequences of tokens in the natural way as  $e([t_1, \dots, t_n]) = [e(t_1), \dots, e(t_n)]$ .

We assume that the language model internally consists of  $l_{\max}$  hidden layers and given a sequence of token embeddings  $e = [e_1, \dots, e_n]$ , we denote by  $h_i^l(e)$  the contextualized representation of the  $i$ -th input embedding  $e_i$  at layer  $l$ . Given two additional sequences of left and right embeddings  $\ell$  and  $r$ , we define

$$\tilde{h}_i^l(\ell, e, r) = \begin{cases} h_i^l(\ell; e; r) & \text{if } i \leq |\ell| \\ h_{i+|\ell|}^l(\ell; e; r) & \text{if } i > |\ell| \end{cases}$$

That is, we “cut out” the sequence  $e$  and  $\tilde{h}_i^l(\ell, e, r)$  is then the embedding of the  $i$ -th input at layer  $l$ , either from  $\ell$  (if position  $i$  is before  $e$ ) or from  $r$  (if position  $i$  is after  $e$ ).

To obtain an embedding for an arbitrary word  $w \in \Sigma^*$ , we require a set of left and right contexts  $\mathcal{C} \subset \mathcal{T}^* \times \mathcal{T}^*$ . Given one such context  $c = (\mathbf{t}_\ell, \mathbf{t}_r)$ , the key idea of OTA is to search for the embedding  $v \in \mathbb{R}^d$  whose influence on  $\mathbf{t}_\ell$  and  $\mathbf{t}_r$  is as similar as possible to the influence of  $w$  on both sequences. That is, when we apply the language model to the sequences  $s_1 = [e(\mathbf{t}_\ell); e(t(w)); e(\mathbf{t}_r)]$  and  $s_2 = [e(\mathbf{t}_\ell); [v]; e(\mathbf{t}_r)]$ , we want the contextualized representations of  $\mathbf{t}_\ell$  and  $\mathbf{t}_r$  in  $s_1$  and  $s_2$  to be as similar as possible.

Formally, we define the *one-token approximation* of  $w$  as

$$\text{OTA}(w) = \arg \min_{v \in \mathbb{R}^d} \sum_{(\mathbf{t}_\ell, \mathbf{t}_r) \in \mathcal{C}} d(e(t(w)), [v] \mid e(\mathbf{t}_\ell), e(\mathbf{t}_r))$$

where

$$d(e, \tilde{e} \mid \ell, r) = \sum_{l=1}^{l_{\max}} \sum_{i=1}^{|\ell|+|r|} d_i^l(e, \tilde{e} \mid \ell, r)$$

$$d_i^l(e, \tilde{e} \mid \ell, r) = \|\tilde{h}_i^l(\ell, e, r) - \tilde{h}_i^l(\ell, \tilde{e}, r)\|^2.$$

$d(e, \tilde{e} \mid \ell, r)$  is differentiable with respect to  $\tilde{e}$ , so we can use gradient-based optimization to estimate  $\text{OTA}(w)$ . This idea resembles the approach of [Le and Mikolov \(2014\)](#) to infer paragraph vectors for sequences of arbitrary length.

With regards to the choice of contexts  $\mathcal{C}$ , we define two variants, both of which do not require any additional information: STATIC and RANDOM. For the STATIC variant,  $\mathcal{C}$  consists of a single context

$$(\mathbf{t}_\ell, \mathbf{t}_r) = ([\text{CLS}], \cdot [\text{SEP}])$$

with  $[\text{CLS}]$  and  $[\text{SEP}]$  being BERT’s classification and separation token, respectively.

As the meaning of a word can often better be understood by looking at its interaction with other words, in the RANDOM variant, each pair  $(\mathbf{t}_\ell, \mathbf{t}_r) \in \mathcal{C}$  is of the form

$$(\mathbf{t}_\ell, \mathbf{t}_r) = ([\text{CLS}] t_\ell, t_r \cdot [\text{SEP}])$$

where  $t_\ell$  and  $t_r$  are uniformly sampled tokens from  $\mathcal{T}$ , with the restriction that each of them represents an actual word.

## 5 WordNet Language Model Probing

In order to assess the ability of language models to understand words as a function of their frequency, we introduce the *WordNet Language Model Probing* (WNLAMPro) dataset.<sup>2</sup> This dataset consists of two parts:

- a set of triplets  $(k, r, T)$  where  $k$  is a *key word*,  $r$  is a *relation* and  $T$  is a set of *target words*;
- a set of *patterns*  $P(r)$  for each relation  $r$ , where each pattern is a sequence of tokens that contains exactly one *key placeholder*  $\langle w \rangle$  and one *target placeholder*  $\_$ .

<sup>2</sup>WNLAMPro is publicly available at <https://github.com/timoschick/am-for-bert>

Key	Rel.	Targets
new	ANT	old
general	ANT	specific
local	ANT	global
book	HYP	product, publication, ...
basketball	HYP	game, ball, sport, ...
lingonberry	HYP	fruit, bush, berry, ...
samosa	COH	pizza, sandwich, salad, ...
harmonium	COH	brass, flute, sax, ...
immorality	COH	crime, evil, sin, fraud, ...
simulation	COR	simulation
chempistry	COR	chemistry
pinacle	COR	pinacle

Table 2: Example entries from WNLaMPro

Rel.	Patterns
ANT	<W> is the opposite of ____ . <W> is not ____ . someone who is <W> is not ____ . something that is <W> is not ____ . “<W>” is the opposite of “____” .
HYP	<W> is a ____ . a <W> is a ____ . “<W>” refers to a ____ . <W> is a kind of ____ . a <W> is a kind of ____ .
COH	<W> and ____ . “<W>” and “____” .
COR	“<W>” is a misspelling of “____” . “<W>” . did you mean “____” ?

Table 3: Patterns for all relations of WNLaMPro. The indefinite article “a” used in the HYP patterns is replaced with “an” as appropriate.

The dataset contains four different kinds of relations: ANTONYM (ANT), HYPERNYM (HYP), COHYPNOMY (COH) and CORRUPTION (COR). Examples of dataset entries for all relations are shown in Table 2; the set of patterns for each relation can be seen in Table 3.

For all but the last relation, we use WordNet (Miller, 1995) to obtain triplets  $(k, r, T)$ . To this end, we denote by  $\mathcal{V}$  the vocabulary of all words that occur at least once in the Westbury Wikipedia Corpus (WWC) (Shaoul and Westbury, 2010) and match the regular expression  $[a-z]^*$ . The set of all tokens in the BERT vocabulary is denoted by  $\mathcal{T}$ . For all triplets, we restrict the set of target words to single-token words from  $\mathcal{T}$ .

**Antonyms** For each adjective  $w \in \mathcal{V}$ , we collect all antonyms for its most frequent WordNet sense in a set  $A$  and, if  $A \cap \mathcal{T} \neq \emptyset$ , add the tuple  $(w, \text{ANTONYM}, A \cap \mathcal{T})$  to the dataset.

Rel.	Subset Size			Mean Targets		
	R	M	F	R	M	F
ANT	41	59	266	1.0	1.0	1.0
HYP	1191	1785	4750	4.0	3.9	4.2
COH	1960	2740	6126	26.0	26.0	25.0
COR	2880	–	–	1.0	–	–

Table 4: The number of entries and mean number of target words for the RARE (R), MEDIUM (M), and FREQUENT (F) subsets of WNLaMPro.

**Hypernyms** For each noun  $w \in \mathcal{V}$ , let  $H$  be the set of all hypernyms for its two most frequent senses. As direct hypernyms are sometimes highly specific (e.g., the hypernym of “dog” is “canine”), we include all hypernyms with a maximum path distance to  $w$  of 3. To avoid the inclusion of very general terms such as “object” or “unit”, we restrict  $H$  to hypernyms that have a minimum depth of 6 in the WordNet hierarchy. If  $|H \cap \mathcal{T}| \geq 3$ , we add  $(w, \text{HYPERNYM}, H \cap \mathcal{T})$  to the dataset. However, if  $|H \cap \mathcal{T}| > 20$ , we keep only the 20 most frequent target words.

**Cohyponyms** For each noun  $w \in \mathcal{V}$ , we compute its set of hypernyms  $H$  as described above (but with a maximum path distance of 2), and denote by  $C$  the union of all hyponyms for each hypernym in  $H$  with a maximum distance of 4. Let  $C' = (C \setminus \{w\}) \cap \mathcal{T}$ . If  $|C'| \geq 10$ , we add the corresponding tuple  $(w, \text{COHYPNOMY}, C')$  to the dataset. If  $|C'| > 50$ , we keep only the 50 most frequent target words.

**Corruptions** The purpose of this final relation is to investigate a model’s ability to deal with corruptions of the input that may, for example, be the result of typing errors or errors in optical character recognition. To obtain corrupted words, we take a set of frequent words from  $\mathcal{V} \cap \mathcal{T}$  and randomly apply corruptions similar to the ones used by Hill et al. (2016) and Lee et al. (2018), but we apply them on a character level instead of a word level. In concrete terms, given a word  $w = c_1 \dots c_n$ , we create a corrupted version  $\tilde{w}$  by either (i) inserting a random character  $c$  at a random position  $i \in [0, n + 1]$ , (ii) removing a character at a random position  $i \in [1, n]$  or (iii) switching the characters  $c_i$  and  $c_{i+1}$  for a random position  $i \in [0, n - 1]$ . We then add  $(\tilde{w}, \text{CORRUPTION}, w)$  to the dataset.

We split the dataset into a development and a

test set. For each relation, we randomly select 10% of all entries to be included in the development set; the remaining 90% form the test set. We purposefully do not provide a training set as WNLamPro is meant to be used *without* task-specific fine-tuning. We also define three subsets of WNLamPro based on key word counts in WWC: WNLamPro-RARE, containing all words that occur less than 10 times, WNLamPro-MEDIUM, containing all words that occur 10 or more times, but less than 100 times, and WNLamPro-FREQUENT, containing all remaining words. Statistics about the sizes of these subsets and the respective mean number of target words per relation are listed in Table 4.

## 6 Experiments

### 6.1 One-Token Approximation

Before looking at WNLamPro, we investigate the ability of OTA to come up with high-quality approximations of word embeddings. To infer embeddings for multi-token words, we initialize the OTA vector  $v$  as a zero vector (Other initialization strategies we tried did not perform better). To optimize  $v$ , we use the Adam optimizer (Kingma and Ba, 2015) with an initial learning rate of  $10^{-3}$ .

For deciding whether to use the STATIC or RANDOM set of training contexts and for finding the optimal number of iterations, we form a development set by randomly selecting 1000 one-token words from the BERT vocabulary. For each word  $w$  in this set, we measure the quality of its approximation  $\text{OTA}(w)$  by comparing it to its BERT embedding  $e(w)$ , using cosine distance. For both context variants, we search for the ideal number of iterations in the range  $\{100 \cdot i \mid 1 \leq i \leq 50\}$ . Figure 1 shows that, given a sufficient number of iterations, RANDOM consistently outperforms STATIC. For the RANDOM variant, the average cosine distance reaches its minimum at 4000 iterations. We therefore use RANDOM contexts with 4000 iterations in our experiments.

### 6.2 Evaluation of BERT on WNLamPro

We first evaluate BERT – both  $\text{BERT}_{\text{BASE}}$  and  $\text{BERT}_{\text{LARGE}}$  (Devlin et al., 2018)<sup>3</sup> – on WNLamPro test to get an impression of (i) the model’s general ability to understand the presented phrases

<sup>3</sup>We use the implementation of <https://github.com/huggingface/pytorch-pretrained-BERT>

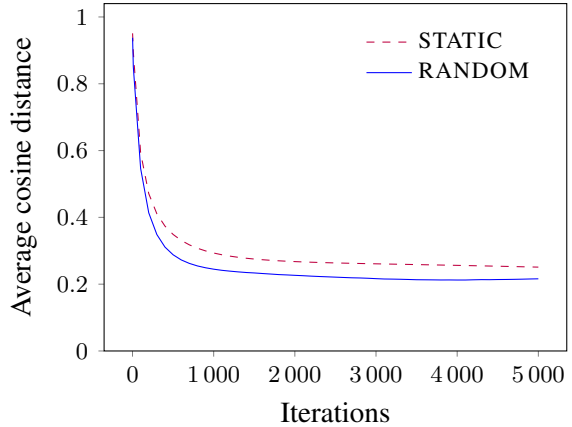


Figure 1: Performance of OTA on 1000 randomly selected one-token words for STATIC and RANDOM contexts

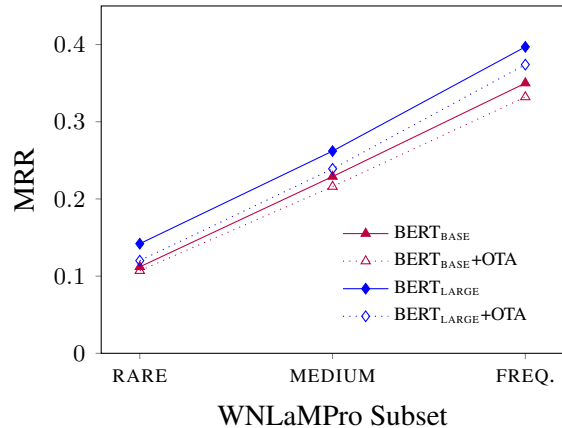


Figure 2: Mean reciprocal rank on the entire WNLamPro for  $\text{BERT}_{\text{BASE}}$ ,  $\text{BERT}_{\text{LARGE}}$  and their OTA variants

and (ii) the difference in performance for rare and frequent words.

To measure the performance of a language model on WNLamPro, we proceed as follows. Let  $x = (k, r, T)$  be a dataset entry,  $t \in T$  a target word,  $p \in P(r)$  a pattern and  $p[k]$  the same pattern where the key placeholder  $\langle w \rangle$  is replaced by  $k$ . Furthermore, let  $(a_1, \dots, a_n)$  be the model’s responses (sorted in descending order by their probability) when it is asked to predict a replacement word for the target placeholder in  $p[k]$ . Then there is some  $j$  such that  $a_j = t$ . We denote with

$$\text{rank}(p[k], t) = j$$

$$\text{precision}_i(p[k], T) = \frac{|\{a_1, \dots, a_i\} \cap T|}{i}$$

the *rank* of  $t$  and *precision* at  $i$  when the model is

queried with  $p[k]$ .<sup>4</sup> We may then define:

$$\text{rank}(x) = \min_{p \in P(r)} \min_{t \in T} \text{rank}(p[k], t)$$

$$\text{precision}_i(x) = \max_{p \in P(r)} \text{precision}_i(p[k], T)$$

That is, for each triplet  $x$ , we compute the lowest rank and highest precision of  $x$  that can be achieved using any pattern. We do so because our interest is not in testing the model’s ability to understand a given pattern, but its ability to understand a given word: by letting the model choose the best pattern for each word, we minimize the probability that its response is of poor quality simply because it did not understand a given pattern.

For each of BERT<sub>BASE</sub> and BERT<sub>LARGE</sub>, we also try a variant where all key words are replaced with their corresponding one-token approximations. We do so to get an understanding of how OTA performs for words consisting of multiple tokens. The mean reciprocal rank (MRR) over the entire WNLaMPro can be seen in Figure 2 for both model configurations with and without OTA. While BERT<sub>LARGE</sub> consistently outperforms BERT<sub>BASE</sub>, for both models the score depends heavily on the word frequency: the MRR on WNLaMPro-RARE is 32% and 36% of the MRR on WNLaMPro-FREQUENT for the base and large configurations, respectively. The OTA variants perform slightly worse than the original models, but the difference is only marginal, allowing us to conclude that OTA is indeed able to infer single embeddings of decent quality for multi-token words.

The general trend that the understanding of a word increases with its frequency becomes even more obvious when looking at Figure 3, where the distribution of ranks for the COHY-PONYM subset of WNLaMPro is shown as a function of WWC word counts. As can be seen, for words that occur  $\leq 256$  ( $2^8$ ) times in WWC, the most probable rank interval is  $[64, 128)$ . With more observations, BERT’s understanding of words drastically improves: more than 50% of all words with more than 256 ( $2^8$ ) observations achieve a rank of  $\leq 16$ .

### 6.3 Attentive Mimicking

We train two variants of Attentive Mimicking: the default configuration of Schick and Schütze (2019b) and the AM+CONTEXT configuration

<sup>4</sup>We only look at the first 100 system responses and set  $\text{rank}(p[k], t) = \infty$  if  $t \notin \{a_1, \dots, a_{100}\}$ .

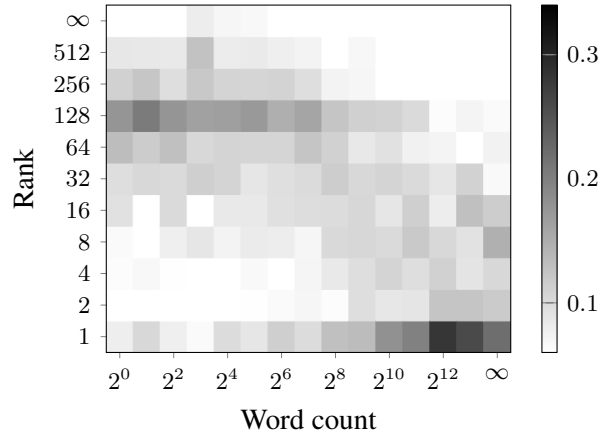


Figure 3: Performance of BERT<sub>BASE</sub> for the COHY-PONYM subset of WNLaMPro. Each cell  $(i, j)$  of the heat map is colored based on the percentage of all dataset entries with key word counts in the range  $(2^{j-1}, 2^j]$  whose rank is in the range  $(2^{i-1}, 2^i]$ .

Model	MRR	
	5 Epochs	10 Epochs
AM	0.258	0.253
AM+CONTEXT	<b>0.262</b>	<b>0.276</b>
AM – OTA	0.219	0.220
AM – form	0.138	0.133
AM – context	0.227	0.225

Table 5: Results on WNLaMPro dev for various configurations of AM

(§3.1) that puts more emphasis on contexts. To decide which method to apply and to determine the optimal number of training epochs, we use WNLaMPro dev. As evaluating AM on WNLaMPro is a time-consuming operation, the only values we try are 5 and 10 epochs; furthermore, we perform hyperparameter optimization only for BERT<sub>BASE</sub> and take the same parameters for BERT<sub>LARGE</sub>. As proposed by Schick and Schütze (2019a), we train AM on all words that occur at least 100 times in WWC; for each such word that is represented by multiple tokens in the BERT vocabulary, we simply use its OTA as a target vector to be mimicked. Importantly, we train AM on contexts from WWC (containing slightly less than  $10^9$  words), whereas the original BERT model was trained on the concatenation of BooksCorpus (Zhu et al., 2015) (containing  $0.8 \cdot 10^9$  words) and a much larger version of Wikipedia (containing  $2.5 \cdot 10^9$  words). As especially for rare words, each occurrence may be crucial for obtaining a high-quality representation, this of course gives BERT a clear

Set	Model	RARE			MEDIUM			FREQUENT		
		MRR	P@3	P@10	MRR	P@3	P@10	MRR	P@3	P@10
ANT	BERT <sub>BASE</sub>	0.149	0.065	0.025	0.089	0.044	0.021	0.390	0.170	0.061
	BERT <sub>BASE</sub> + AM	<b>0.449</b>	<b>0.167</b>	<b>0.075</b>	<b>0.511</b>	<b>0.176</b>	<b>0.064</b>	<b>0.482</b>	<b>0.195</b>	<b>0.074</b>
	BERT <sub>LARGE</sub>	0.234	0.083	0.044	0.218	0.088	0.036	0.541	0.209	0.081
	BERT <sub>LARGE</sub> + AM	<b>0.529</b>	<b>0.194</b>	<b>0.075</b>	<b>0.558</b>	<b>0.195</b>	<b>0.068</b>	<b>0.570</b>	<b>0.228</b>	<b>0.088</b>
HYP	BERT <sub>BASE</sub>	0.276	0.122	0.066	0.327	0.151	0.077	<b>0.416</b>	<b>0.204</b>	<b>0.109</b>
	BERT <sub>BASE</sub> + AM	<b>0.300</b>	<b>0.135</b>	<b>0.074</b>	<b>0.343</b>	<b>0.158</b>	<b>0.081</b>	0.377	0.181	0.096
	BERT <sub>LARGE</sub>	0.284	0.128	0.065	<b>0.350</b>	<b>0.169</b>	<b>0.086</b>	<b>0.462</b>	<b>0.226</b>	<b>0.117</b>
	BERT <sub>LARGE</sub> + AM	<b>0.299</b>	<b>0.137</b>	<b>0.074</b>	0.323	0.149	0.079	0.401	0.193	0.101
COH	BERT <sub>BASE</sub>	0.147	0.065	0.054	0.177	0.089	0.070	<b>0.294</b>	<b>0.150</b>	<b>0.116</b>
	BERT <sub>BASE</sub> + AM	<b>0.213</b>	<b>0.106</b>	<b>0.082</b>	<b>0.213</b>	<b>0.110</b>	<b>0.090</b>	0.262	0.136	0.108
	BERT <sub>LARGE</sub>	0.174	0.085	0.067	0.210	<b>0.109</b>	<b>0.091</b>	<b>0.337</b>	<b>0.183</b>	<b>0.143</b>
	BERT <sub>LARGE</sub> + AM	<b>0.227</b>	<b>0.110</b>	<b>0.087</b>	<b>0.216</b>	0.106	0.089	0.292	0.153	0.121
COR	BERT <sub>BASE</sub>	0.020	0.007	0.004	–	–	–	–	–	–
	BERT <sub>BASE</sub> + AM	<b>0.254</b>	<b>0.095</b>	<b>0.038</b>	–	–	–	–	–	–
	BERT <sub>LARGE</sub>	0.062	0.022	0.012	–	–	–	–	–	–
	BERT <sub>LARGE</sub> + AM	<b>0.261</b>	<b>0.095</b>	<b>0.038</b>	–	–	–	–	–	–

Table 6: Performance of both BERT models with and without AM for WNLaMPro test, subdivided by relation and key word count

advantage over our proposed method.

To understand the influence of OTA on the performance of AM, in addition to the two configurations described above we also try a variant without OTA. Of course, the training set for this variant contains only one-token words. To see whether we actually need both form and context information, we additionally investigate the influence of dropping the context and form parts of AM, respectively.

Table 5 shows results for all model variants on WNLaMPro dev. We can see that OTA is indeed helpful for training the model, substantially improving its score. Results for the model variants using only form or context are in line with the findings of Schick and Schütze (2019a): it is essential for good performance to use both form and context. Furthermore, the AM+CONTEXT variant improves upon the default configuration of AM and training it for 10 epochs performs better than 5 epochs. Based on these findings, we use only the AM+CONTEXT variant trained for 10 epochs in the following experiments.

A detailed comparison of BERT’s performance with and without Attentive Mimicking can be seen in Table 6, where the MRR as well as the precision at 3 and 10 are shown for each relation and frequency. As can be seen, AM substantially improves the score for rare words, regardless of whether we use it in combination with BERT<sub>BASE</sub>

or BERT<sub>LARGE</sub>. This allows us to conclude that indeed, AM does help BERT to get a better understanding of rare words. For the ANTONYM subset, the embeddings obtained using AM even improve scores for frequent words. The benefit of applying AM for medium frequency words depends largely on the model being used: for BERT<sub>LARGE</sub>, using AM only brings a consistent improvement for the ANTONYM relation, whereas for BERT<sub>BASE</sub>, using AM is always helpful.

## 7 Conclusion

We have introduced WNLaMPro, a new dataset that allows us to explicitly investigate the ability of deep contextualized language models to understand rare words. Using this dataset, we have shown that BERT struggles with words if they are too rare. To address this, we apply Attentive Mimicking (AM) to deep contextualized word embeddings, even if they do not use embeddings on the word level. To be able to do this, we introduced OTA, an effective method to obtain “single-token” embeddings for multi-token words. Using this method, we showed that AM is able to substantially improve BERT’s understanding of rare words.

Future work might investigate whether more complex architectures than AM can bring further benefit to deep language models; it would also be interesting to see whether training AM on a larger



corpus – such as the one used for training BERT by Devlin et al. (2018) – is beneficial. Furthermore, it would be interesting to see the impact of integrating AM on downstream tasks and whether our results can also be transferred to other contextualized language models.

## Acknowledgments

We are grateful for the support of the European Research Council for this work (ERC #740516).

## References

- Marco Baroni and Alessandro Lenci. How we blessed distributional semantic evaluation. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 1–10. Association for Computational Linguistics, 2011.
- Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5: 135–146, 2017. URL <http://aclweb.org/anthology/Q17-1010>.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- Aurélie Herbelot and Marco Baroni. 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, 2017. URL <http://aclweb.org/anthology/D17-1030>.
- Felix Hill, Kyunghyun Cho, and Anna Korhonen. Learning distributed representations of sentences from unlabelled data. *arXiv preprint arXiv:1602.03483*, 2016.
- Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
- Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.
- Mikhail Khodak, Nikunj Saunshi, Yingyu Liang, Tengyu Ma, Brandon Stewart, and Sanjeev Arora. 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, 2018. URL <http://aclweb.org/anthology/P18-1002>.
- Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *The International Conference on Learning Representations (ICLR)*, 2015.
- Angeliki Lazaridou, Marco Marelli, Roberto Zamparelli, and Marco Baroni. Compositionally derived representations of morphologically complex words in distributional semantics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1517–1526. Association for Computational Linguistics, 2013. URL <http://www.aclweb.org/anthology/P13-1149>.
- Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196, 2014.
- Jason Lee, Elman Mansimov, and Kyunghyun Cho. Deterministic non-autoregressive neural sequence modeling by iterative refinement. *arXiv preprint arXiv:1802.06901*, 2018.
- Thang Luong, Richard Socher, and Christopher Manning. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113, 2013.
- Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*, 2018.
- George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.
- Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.
- Yuval Pinter, Robert Guthrie, and Jacob Eisenstein. 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, 2017. URL <http://aclweb.org/anthology/D17-1010>.
- Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. URL [https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/languageunsupervised/language\\_understanding\\_paper.pdf](https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/languageunsupervised/language_understanding_paper.pdf), 2018a.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, Technical report, OpenAI, 2018b.

- Alexandre Salle and Aline Villavicencio. Incorporating subword information into matrix factorization word embeddings. In *Proceedings of the Second Workshop on Subword/Character Level Models*, pages 66–71. Association for Computational Linguistics, 2018.
- Timo Schick and Hinrich Schütze. Learning semantic representations for novel words: Leveraging both form and context. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, 2019a. URL <https://arxiv.org/abs/1811.03866>.
- Timo Schick and Hinrich Schütze. Attentive mimicking: Better word embeddings by attending to informative contexts. In *Proceedings of the Seventeenth Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2019b. To appear.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909, 2015. URL <http://arxiv.org/abs/1508.07909>.
- Cyrus Shaoul and Chris Westbury. The westbury lab wikipedia corpus, 2010.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017. URL <http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf>.
- Julie Weeds, Daoud Clarke, Jeremy Reffin, David Weir, and Bill Keller. Learning to distinguish hypernyms and co-hyponyms. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2249–2259. Dublin City University and Association for Computational Linguistics, 2014.
- John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Charagram: Embedding words and sentences via character n-grams. *CoRR*, abs/1607.02789, 2016.
- Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 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, 2015.