Deriving Word Vectors from Contextualized Language Models using Topic-Aware Mention Selection

Yixiao Wang, Zied Bouraoui, Luis Espinosa Anke, Steven Schockaert

▶ **To cite this version:**

Yixiao Wang, Zied Bouraoui, Luis Espinosa Anke, Steven Schockaert. Deriving Word Vectors from Contextualized Language Models using Topic-Aware Mention Selection. 6th Workshop on Representation Learning for NLP (RepL4NLP-2021), Aug 2021, Online, Thailand. pp.185-194, ⟨10.18653/v1/2021.repl4nlp-1.19⟩. ⟨hal-04404169⟩

HAL Id: hal-04404169

https://hal.science/hal-04404169v1

Submitted on 18 Jan 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

HAL Authorization

Deriving Word Vectors from Contextualized Language Models using Topic-Aware Mention Selection

Yixiao Wang[1], Zied Bouraoui[2], Luis Espinosa Anke[1] and **Steven Schockaert[1]**
[1]School of Computer Science & Informatics, Cardiff University, UK
[2] CRIL-CNRS, Université d'Artois, France
{wangy306,espinosa-ankel,schockaerts1}@cardiff.ac.uk, zied.bouraoui@cril.fr

Abstract

One of the long-standing challenges in lexical semantics consists in learning representations of words which reflect their semantic properties. The remarkable success of word embeddings for this purpose suggests that high-quality representations can be obtained by summarizing the sentence contexts of word mentions. In this paper, we propose a method for learning word representations that follows this basic strategy, but differs from standard word embeddings in two important ways. First, we take advantage of contextualized language models (CLMs) rather than bags of word vectors to encode contexts. Second, rather than learning a word vector directly, we use a topic model to partition the contexts in which words appear, and then learn different topic-specific vectors for each word. Finally, we use a task-specific supervision signal to make a soft selection of the resulting vectors. We show that this simple strategy leads to high-quality word vectors, which are more predictive of semantic properties than word embeddings and existing CLM-based strategies.

1 Introduction

In the last few years, contextualized language models (CLMs) such as BERT (Devlin et al., 2019) have largely replaced the use of static (i.e. non-contextualized) word vectors in many Natural Language Processing (NLP) tasks. However, static word vectors remain important in applications where word meaning has to be modelled in the absence of (sentence) context. For instance, static word vectors are needed for zero-shot image classification (Socher et al., 2013) and zero-shot entity typing (Ma et al., 2016), for ontology alignment (Kolyvakis et al., 2018) and completion (Li et al., 2019), taxonomy learning (Bordea et al., 2015, 2016), or for representing query terms in information retrieval systems (Nikolaev and Kotov, 2020). Moreover, Liu et al. (2020) recently found that static word vectors can complement CLMs, by serving as anchors for contextualized vectors, while Alghanmi et al. (2020) found that incorporating static word vectors could improve the performance of BERT for social media classification.

Given the impressive performance of CLMs across many NLP tasks, a natural question is whether such models can be used to learn high-quality static word vectors, and whether the resulting vectors have any advantages compared to those from standard word embedding models (Mikolov et al., 2013; Pennington et al., 2014). A number of recent works have begun to explore this question (Ethayarajh, 2019; Bommasani et al., 2020; Vulic et al., 2020). Broadly speaking, the idea is to construct a static word vector for a word w by randomly selecting sentences in which this word occurs, and then averaging the contextualized representations of w across these sentences.

Since it is not usually computationally feasible to run the CLM on all sentences mentioning w, a sample of such sentences has to be selected. This begs the question: how should these sentences be chosen? In the aforementioned works, sentences are selected at random, but this may not be optimal. If we want to use the resulting word vectors in downstream tasks such as zero-shot learning or ontology completion, we need vectors that capture the salient semantic properties of words. Intuitively, we should thus favor sentences that best reflect these properties. For instance, many of the mentions of the word *banana* on Wikipedia are about the cultivation and export of bananas, and about the specifics of particular banana cultivars. By learning a static word vector from such sentences, we may end up with a vector that does not reflect our commonsense understanding of bananas, e.g. the fact that they are curved, yellow and sweet.

The main aim of this paper is to analyze to what

extent topic models such as Latent Dirichlet Allocation (Blei et al., 2003) can be used to address this issue. Continuing the previous example, we may find that the word *banana* occurs in Wikipedia articles on the following topics: economics, biology, food or popular culture. While most mentions might be in articles on economics and biology, it is the latter two topics that are most relevant for modelling the commonsense properties of bananas. Note that the optimal selection of topics is task-dependent, e.g. in an NLP system for analyzing financial news, the economics topic would clearly be more relevant. For this reason, we propose to learn a word vector for each topic separately. Since the optimal choice of topics is task-dependent, we then rely on a task-specific supervision signal to make a soft selection of these topic-specific vectors.

Another important question is how CLMs should be used to obtain contextualized word vectors. Given a sentence mentioning w, a model such as BERT-base constructs 12 vector representations of w, i.e. one for each layer of the transformer stack. Previous work has suggested to use the average of particular subsets of these vectors. In particular, Vulic et al. (2020) found that lexical semantics is most prevalent in the representations from the early layers, and that averaging vectors from the first few layers seems to give good results on many benchmarks. On the other hand, these early layers are least affected by the sentence context (Ethayarajh, 2019), hence such strategies might not be suitable for learning topic-specific vectors. We therefore also explore a different strategy, which is to mask the target word in the given sentence, i.e. to replace the entire word by a single [MASK] token, and to use the vector representation of this token at the final layer. The resulting vector representations thus specifically encode what the given sentence reveals about the target word, making this a natural strategy for learning topic-specific vectors.

Note that there is a clear relationship between this latter strategy and CBOW (Mikolov et al., 2013): where in CBOW the vector representation of w is obtained by averaging the vector representations of the context words that co-occur with w, we similarly represent words by averaging context representations. The main advantage compared to CBOW thus comes from the higher-quality context encodings that can be obtained using CLMs. The main challenge, as already mentioned, is that we cannot consider all the mentions of w, whereas this is typically feasible for CBOW (and other standard word embedding models). Our contributions can be summarized as follows[1]:

- We analyze different strategies for deriving word vectors from CLMs, which rely on sampling mentions of the target word from a text collection.
- We propose the use of topic models to improve how these mentions are sampled. In particular, rather than learning a single vector representation for the target word, we learn one vector for each sufficiently relevant topic.
- We propose to construct the final representation of a word w as a weighted average of different vectors. This allows us to combine multiple vectors without increasing the dimensionality of the final representations. We use this approach for combining different topic-specific vectors and for combining vectors from different transformer layers.

2 Related Work

A few recent works have already proposed strategies for computing static word vectors from CLMs. While Ethayarajh (2019) relied on principal components of individual transformer layers for this purpose, most approaches rely on averaging the contextualised representations of randomly selected mentions of the target word (Bommasani et al., 2020; Vulic et al., 2020). Several authors have pointed out that the representations obtained from early layers tend to perform better in lexical semantics probing tasks. However, Bommasani et al. (2020) found that the optimal layer depends on the number of sampled mentions, with later layers performing better when a large number of mentions is used. Rather than fixing a single layer, Vulic et al. (2020) advocated averaging representations from several layers. Note that none of the aforementioned methods uses masking when computing contextualized vectors. This means that the final representations may have to be obtained by pooling different word-piece vectors, usually by averaging them.

[1] All code and data to replicate our experiments is available at https://github.com/Activeyixiao/topic-specific-vector/.

As an alternative to using topic models, Chronis and Erk (2020) cluster the contextual word vectors, obtained from mentions of the same word. The resulting multi-prototype representation is then used to compute word similarity in an adaptive way. Along similar lines, Amrami and Goldberg (2019) cluster contextual word vectors for word sense induction. Thompson and Mimno (2020) showed that clustering the contextual representations of a given set of words can produce clusters of semantically related words, which were found to be similar in spirit to LDA topics. The idea of learning topic-specific representations of words has been extensively studied in the context of standard word embeddings (Liu et al., 2015; Li et al., 2016; Shi et al., 2017; Zhu et al., 2020). To the best of our knowledge, learning topic-specific word representations using CLMs has not yet been studied. More broadly, however, some recent methods have combined CLMs with topic models. For instance, Peinelt et al. (2020) use such a combination for predicting semantic similarity. In particular they use the LDA or GSDMM topic distribution of two sentences to supplement their BERT encoding. Finally, Bianchi et al. (2020) suggested using sentence embeddings from SBERT (Reimers and Gurevych, 2019) as input to a neural topic model, with the aim of learning more coherent topics.

3 Constructing Word Vectors

In Section 3.1, we first describe different strategies for deriving static word vectors from CLMs. Section 3.2 subsequently describes how we choose the most relevant topics for each word, and how we sample topic-specific word mentions. Finally, in Section 3.3 we explain how the resulting topic-specific representations are combined to obtain task-specific word vectors.

3.1 Obtaining Contextualized Word Vectors

We first briefly recall the basics of the BERT contextualised language model. BERT represents a sentence s as a sequence of word-pieces $w_1 ..., w_n$. Frequent words will typically be represented as a single word-piece, but in general, word-pieces may correspond to sub-word tokens. Each of these word-pieces w is represented as an input vector, which is constructed from a static word-piece embedding $\mathbf{w}_0$ (together with vectors that encode at which position in the sentence the word appears, and in which sentence). The resulting sequence of word-piece vectors is then fed to a stack of 12 (for BERT-base) or 24 (for BERT-large) transformer layers. Let us write $\mathbf{w}_i^s$ for the representation of word-piece w in the i$^{\text{th}}$ transformer layer. We will refer to the representation in the last layer, i.e. $\mathbf{w}_{12}^s$ for BERT-base and $\mathbf{w}_{24}^s$ for BERT-large, as the output vector. When BERT is trained, some of the word-pieces are replaced by a special [MASK] token. The corresponding output vector then encodes a prediction of the masked word-piece.

Given a sentence s in which the word w is mentioned, there are several ways in which BERT and related models can be used to obtain a vector representation of w. If w consists of a single word-piece, a natural strategy is to feed the sentence s as input and use the output vector as the representation of w. However, several authors have found that it can be beneficial to also take into account some or all of the earlier transformer layers, where fine-grained word senses are mostly captured in the later layers (Reif et al., 2019) but word-level lexical semantic features are primarily found in the earlier layers (Vulic et al., 2020). For this reason, we will also experiment with models in which the vectors $\mathbf{w}_1^s, ..., \mathbf{w}_{12}^s$ (or $\mathbf{w}_1^s, ..., \mathbf{w}_{24}^s$ in the case of BERT-large) are all used. In particular, our model will construct a weighted average of these vectors, where the weights will be learned from training data (see Section 3.3). For words that consist of multiple word-pieces, following common practice, we compute the representation of w as the average of its word-piece vectors. For instance, this strategy was found to outperform other aggregation strategies in Bommasani et al. (2020).

We will also experiment with a strategy that relies on masking. In this case, the word w is replaced by a single [MASK] token (even if w would normally be tokenized into more than one word-piece). Let us write $\mathbf{m}_w^s$ for the output vector corresponding to this [MASK] token. Since this vector corresponds to BERT's prediction of what word is missing, this vector should intuitively capture the properties of w that are asserted in the given sentence. We can thus expect that these vectors $\mathbf{m}_w^s$ will be more sensitive to how the sentences mentioning w are chosen. Note that in this case, we only use the output layer, as the earlier layers are less likely to be informative.

To obtain a static representation of w, we first select a set of sentences $s_1, ..., s_n$ in which w is mentioned. Then we compute vector representations

$\mathbf{w}^{s_1}, ..., \mathbf{w}^{s_n}$ of w from each of these sentences, using any of the aforementioned strategies. Our final representation $\mathbf{w}$ is then obtained by averaging these sentence-specific representations, i.e.:

$$\mathbf{w} = \frac{\sum_{i=1}^{n} \mathbf{w}^{s_i}}{\| \sum_{i=1}^{n} \mathbf{w}^{s_i} \|}$$

3.2 Selecting Topic-Specific Mentions

To construct a vector representation of $\mathbf{w}$, we need to select some sentences $s_1, ..., s_n$ mentioning w. While these sentences are normally selected randomly, our hypothesis in this paper is that purely random strategies may not be optimal. Intuitively, this is because the contexts in which a given word w is most frequently mentioned might not be the most informative ones, i.e. they may not be the contexts which best characterize the properties of w that matter for a given task. To test this hypothesis, we experiment with a strategy based on topic models. Our strategy relies on the following steps:

1. Identify the topics which are most relevant for the target word w;
2. For each of the selected topics t, select sentences $s_1^t, ..., s_n^t$ mentioning w from documents that are closely related to this topic.

For each of the selected topics t, we can then use the sentences $s_1^t, ..., s_n^t$ to construct a topic-specific vector $\mathbf{w}^t$, using any of the strategies from Section 3.1. The final representation of w will be computed as a weighted average of these topic-specific vectors, as will be explained in Section 3.3.

We now explain these two steps in more detail. First, we use Latent Dirichlet Allocation (LDA) (Blei et al., 2003) to obtain a representation of each document d in the considered corpus as a multinomial distribution over m topics. Let us write $\tau_i(d)$ for the weight of topic i in the representation of document d, where $\sum_{i=1}^{m} \tau_i(d) = 1$. Suppose that the word w is mentioned N_w times in the corpus, and let d_j^w be the document in which the j^{th} mention of w occurs. Then we define the importance of topic i for word w as follows:

$$\tau_i(w) = \frac{1}{N_w} \sum_{j=1}^{N_w} \tau_i(d_j^w) \quad (1)$$

In other words, the importance of topic i for word w is defined as the average importance of topic i for the documents in which w occurs. To select the set of topics $\mathcal{T}_w$ that are relevant to w, we rank the topics from most to least important and then select the smallest set of topics whose cumulative importance is at least 60%, i.e. $\mathcal{T}_w$ is the smallest set of topics such that $\sum_{t_i \in \mathcal{T}_w} \tau_i(w) \geq 0.6$.

For each of the topics t_i in $\mathcal{T}_w$ we select the corresponding sentences $s_1^t, ..., s_n^t$ as follows. We rank all the documents in which w is mentioned according to $\tau_i(d)$. Then, starting with the document with the highest score (i.e. the document for which topic i is most important), we iterate over the ranked list of documents, selecting all sentences from these documents in which w is mentioned, until we have obtained a total of n sentences.

3.3 Combining Word Representations

Section 3.1 highlighted a number of strategies that could be used to construct a vector representation of a target word w. As mentioned before, it can be beneficial to combine vector representations from different transformer layers. To this end, we propose to learn a weighted average of the different input vectors, using a task specific supervision signal. In particular, let $\mathbf{w}_1, ..., \mathbf{w}_k$ be the different vector representations we have available for word w (e.g. the vectors from different transformer layers). To combine these vectors, we compute a weighted average as follows:

$$\lambda_i = \frac{\exp(a_i)}{\sum_{j=1}^{k} \exp(a_i)} \quad (2)$$

$$\mathbf{w} = \frac{\sum_i \lambda_i \mathbf{w}_i}{\| \sum_i \lambda_i \mathbf{w}_i \|} \quad (3)$$

where the scalar parameters $a_1, ...a_k \in \mathbb{R}$ are jointly learned with the model in which $\mathbf{w}$ is used. Another possibility would be to concatenate the input vectors $\mathbf{w}_1, ..., \mathbf{w}_k$. However, this significantly increases the dimensionality of the word representations, which can be challenging in downstream applications. In initial experiments, we also confirmed that this concatenation strategy indeed under-performs the use of weighted averages.

If topic-specific vectors are used, we also want to compute a weighted average of the available vectors. However, (2)–(3) cannot be used in this case, because the set of topics for which topic-specific vectors are available differs from word to word. Let us write $\mathbf{w}_{topic}^i$ for the representation of word w that was obtained for topic t_i, where we

assume $\mathbf{w}^i_{topic} = \mathbf{0}$ if $t_i \notin \mathcal{T}_w$. We then define:

$$\mu^w_i = \frac{\exp(b_i) \cdot \mathbb{1}[t_i \in \mathcal{T}_w]}{\sum_{j=1}^{k} \exp(b_i) \cdot \mathbb{1}[t_j \in \mathcal{T}_w]} \quad (4)$$

$$\mathbf{w}_{topic} = \frac{\sum_i \mu^w_i \mathbf{w}^i_{topic}}{\| \sum_i \mu^w_i \mathbf{w}^i_{topic} \|} \quad (5)$$

where $\mathbb{1}[t_i \in \mathcal{T}_w] = 1$ if topic t_i is considered to be relevant for word w (i.e. $t_i \in \mathcal{T}_w$), and $\mathbb{1}[t_i \in \mathcal{T}_w] = 0$ otherwise. Note that the softmax function in (4) relies on the scalar parameters $b_1, ..., b_k \in \mathbb{R}$, which are independent of w. However, the softmax is selectively applied to those topics that are relevant to w, which is why the resulting weight μ^w_i is dependent on w, or more precisely, on the set of topics $\mathcal{T}_w$.

4 Evaluation

We compare the proposed strategy with standard word embeddings and existing CLM-based strategies. In Section 4.1 we first describe our experimental setup. Section 4.2 then provides an overview of the datasets we used for the experiments, where we focus on lexical classification benchmarks. These benchmarks in particular allow us to assess how well various semantic properties can be predicted from the word vectors. The experimental results are discussed in Section 4.3 and a qualitative analysis is presented in Section 4.4.

4.1 Experimental Setup

We experiment with a number of different strategies for obtaining word vectors:

$\mathbf{C}_{\mathbf{last}}$ We take the vector representation of w from the last transformer layer (i.e. $\mathbf{w}^s_{12}$ or $\mathbf{w}^s_{24}$).

$\mathbf{C}_{\mathbf{input}}$ We take the input embedding of w (i.e. $\mathbf{w}_0$).

$\mathbf{C}_{\mathbf{avg}}$ We take the average of $\mathbf{w}_0, \mathbf{w}^s_1, ..., \mathbf{w}^s_{12}$ for the *base* models and $\mathbf{w}_0, \mathbf{w}^s_1, ..., \mathbf{w}^s_{24}$ for the *large* models.

$\mathbf{C}_{\mathbf{all}}$ We use all of $\mathbf{w}_0, \mathbf{w}^s_1, ..., \mathbf{w}^s_{12}$ as input for the *base* models, and all of $\mathbf{w}_0, \mathbf{w}^s_1, ..., \mathbf{w}^s_{24}$ for the *large* models. These vectors are then aggregated using (2)–(3), i.e. we use a learned soft selection of the transformer layers.

$\mathbf{C}_{\mathbf{mask}}$ We replace the target word by [MASK] and use the corresponding output vector.

For words consisting of more than one word-piece, we average the corresponding vectors in all cases, except for $\mathbf{C}_{\mathbf{mask}}$ where we always end up with a single vector (i.e. we replace the entire word by a single [MASK] token). We also consider three variants that rely on topic-specific vectors:

$\mathbf{T}_{\mathbf{last}}$ We learn topic-specific vectors using the last transformer layers. These vectors are then used as input to (4)–(5).

$\mathbf{T}_{\mathbf{avg}}$ Similar to the previous case but using the average of all transformer layers.

$\mathbf{T}_{\mathbf{mask}}$ Similar to the previous cases but using the output vector of the masked word mention.

Furthermore, we consider variants of $\mathbf{T}_{\mathbf{last}}$, $\mathbf{T}_{\mathbf{avg}}$ and $\mathbf{T}_{\mathbf{mask}}$ in which a standard (i.e. unweighted) average of the available topic-specific vectors is computed, instead of relying on (4)–(5). We will refer to these averaging-based variants as $\mathbf{A}_{\mathbf{last}}$, $\mathbf{A}_{\mathbf{avg}}$ and $\mathbf{A}_{\mathbf{mask}}$. As baselines, we also consider the two Word2vec models (Mikolov et al., 2013):

SG 300-dimensional Skip-gram vectors trained on a May 2016 dump of the English Wikipedia, using a window size of 5 tokens, and minimum frequency threshold of 10.

CBOW 300-dimensional Continuous Bag-of-Words vectors trained on the same corpus and with the same hyperparameters as **SG**.

We show results for four pre-trained CLMs (Devlin et al., 2019; Liu et al., 2019): BERT-base-uncased, BERT-large-uncased, RoBERTa-base-uncased, RoBERTa-large-uncased[2]. As the corpus for sampling word mentions, we used the same Wikipedia dump as for training the word embeddings models. For $\mathbf{C}_{\mathbf{mask}}$, $\mathbf{C}_{\mathbf{last}}$, $\mathbf{C}_{\mathbf{avg}}$ and $\mathbf{C}_{\mathbf{all}}$ we selected 500 mentions. For the topic-specific strategies ($\mathbf{T}_{\mathbf{last}}$, $\mathbf{T}_{\mathbf{avg}}$ and $\mathbf{T}_{\mathbf{mask}}$) we selected 100 mentions per topic. To obtain the topic assignments, we used Latent Dirichlet Allocation (Blei et al., 2003) with 25 topics. We set $\alpha = 0.0001$ to restrict the total number of topics attributed to a document, and use default values for the other hyper-parameters[3]. To select the relevant topics for a given word w, we find the smallest set of topics whose cumulative importance score $\tau_i(w)$ is at least 60%, with

[2] We used the implementations from https://github.com/huggingface/transformers.

[3] We used the implementation from https://radimrehurek.com/gensim/wiki.html.

a maximum of 6 topics. In the experiments, we restrict the vocabulary to those words with at least 100 occurrences in Wikipedia.

4.2 Datasets

For the experiments, we focus on a number of lexical classification tasks, where categories of individual words need to be predicted. In particular, we used two datasets which are focused on commonsense properties (e.g. *dangerous*): the extension of the McRae feature norms dataset (McRae et al., 2005) that was introduced by Forbes et al. (2019)[4] and the CSLB Concept Property Norms[5]. We furthermore used the WordNet supersenses dataset[6], which groups nouns into broad categories (e.g. *human*). Finally, we also used the BabelNet domains dataset[7] (Camacho-Collados and Navigli, 2017), which assigns lexical entities to thematic domains (e.g. *music*).

In our experiments, we have only considered properties/classes for which sufficient positive examples are available, i.e. at least 10 for McRae, 30 for CSLB, and 100 for WordNet supersenses and BabelNet domains. For the McRae dataset, we used the standard training-validation-test split. For the other datasets, we used random splits of 60% for training, 20% for tuning and 20% for testing. An overview of the datasets is shown in Table 2.

For all datasets, we consider a separate binary classification problem for each property and we report the (unweighted) average of the F1 scores for the different properties. To classify words, we feed their word vector directly to a sigmoid classification layer. We optimise the network using AdamW with a cross-entropy loss. The batch size and learning rate were tuned, with possible values chosen from 4,8,16 and 0.01, 0.005, 0.001, 0.0001 respectively. Note that for $\mathbf{C_{all}}$ and the topic-specific variants, the classification network jointly learns the parameters of the classification layer and the attention weights in (2) and (4) for combining the input vectors.

4.3 Results

The results are shown in Table 1. We consistently see that the topic-specific variants outperform the different **C**-variants by a small margin. This confirms our main hypothesis, namely that using topic models to determine how context sentences are selected has a material effect on the quality of the resulting word representations. Among the **C**-variants, the best results are obtained by $\mathbf{C_{mask}}$. None of the three **T**-variants consistently outperforms the others. Surprisingly, the **A**-variants outperform the corresponding **T**-variants in several cases. This suggests that the outperformance of the topic-specific vectors primarily comes from the fact that the context sentences for each word were sampled in a more balanced way (i.e. from documents covering a broader range of topics), rather than from the ability to adapt the topic weights based on the task. This is a clear benefit for applications, as the **A**-variants allow us to simply represent each word as a static word vector.

The performance of SG and CBOW is still strong. In particular, these traditional word embedding models outperform all of the **C**-variants, as well as the **T** and **A** variants in BabelNet Domains. This seems to be related, at least in part, to the lower dimensionality of these vectors. The classification network has to be learned from a rather small number of examples, especially for McRae and CSLB. Having 768 or 1024 dimensional input vectors can be problematic in such cases. To analyse this effect, we used Principal Component Analysis (PCA) to reduce the dimensionality of the CLM-derived vectors to 300. For this experiment, we focused in particular on $\mathbf{C_{mask}}$ and $\mathbf{T_{mask}}$. The results are also shown in Table 1 as $\mathbf{C_{mask}}$**-PCA** and $\mathbf{T_{mask}}$**-PCA**. As can be seen, this dimensionality reduction step has a clearly beneficial effect, with $\mathbf{T_{mask}}$**-PCA** outperforming all baselines, except for the BabelNet domains benchmark. The latter benchmark is focused on thematic similarity rather than semantic properties, which the CLM-based representations seem to struggle with.

4.4 Qualitative analysis

Topic-specific vectors can be expected to focus on different properties, depending on the chosen topic. In this section, we present a qualitative analysis in support of this view. In Table 3 we list, for a sample of words from the WordNet supersenses dataset, the top 5 nearest neighbours per topic in terms of cosine similarity. For this analysis, we used the BERT-base masked embeddings. We can see that for the word *'partner'*, its topic-specific em-

[4] https://github.com/mbforbes/physical-commonsense

[5] https://cslb.psychol.cam.ac.uk/propnorms

[6] https://wordnet.princeton.edu/download

[7] http://lcl.uniroma1.it/babeldomains/

	BERT-base				BERT-large				RoBERTa-base				RoBERTa-large			
	MC	**CS**	**SS**	**BD**	**MC**	**CS**	**SS**	**BD**	**MC**	**CS**	**SS**	**BD**	**MC**	**CS**	**SS**	**BD**
SG	59.6	54.5	55.6	**49.1**	59.6	54.5	55.6	49.1	59.6	54.5	55.6	**49.1**	59.6	54.5	55.6	**49.1**
CBOW	61.1	50.6	48.4	45.0	61.1	50.6	48.4	45.0	61.1	50.6	48.4	45.0	61.1	50.6	48.4	45.0
$\mathbf{C}_{\mathbf{mask}}$	60.8	51.7	59.7	42.6	61.8	53.4	59.5	42	62.5	51.8	58.5	40	61.3	53.2	59.2	42.8
$\mathbf{C}_{\mathbf{last}}$	60.0	51.4	59	46.1	56.2	53.4	59	42.3	56.5	43.4	58.2	42.1	57.9	47.7	58.8	41.9
$\mathbf{C}_{\mathbf{input}}$	58.8	40.1	50.2	40.3	57.2	42	51.7	40.2	45.8	24.1	44.4	37.9	41.2	20.6	52.6	40.0
$\mathbf{C}_{\mathbf{avg}}$	59.9	49.6	59.1	44.2	60.0	47.1	58.9	42.5	53.7	40.7	50.2	40.3	59.5	47.4	58.8	42.9
$\mathbf{C}_{\mathbf{all}}$	59.9	51.2	59.5	46.4	61.7	50.7	58.4	42.6	45.3	39.3	52.6	36.7	48.2	40.2	56.6	40.4
$\mathbf{T}_{\mathbf{mask}}$	60.9	54.1	60.5	45.8	62.8	55	61.4	46.2	58.6	49.4	56.7	42.1	59.2	50.4	57.2	43.3
$\mathbf{T}_{\mathbf{last}}$	63.0	51.8	59.7	47.3	62.1	55.8	61.6	**49.2**	52.3	43.9	54.6	41.3	62.1	48.8	59.5	45.1
$\mathbf{T}_{\mathbf{avg}}$	61.0	52.7	59.6	43.4	**65.2**	54.8	60.7	48.4	54.5	39.9	55.9	41.5	59.5	47.4	60.0	45.2
$\mathbf{A}_{\mathbf{mask}}$	63.1	53.9	59.2	41.4	63.2	56.8	60.6	41.5	**64.0**	55.3	60.6	40.8	63.4	**57.3**	62	42.3
$\mathbf{A}_{\mathbf{last}}$	62.8	52.4	59.6	44.4	61.4	55.5	60.6	46.7	55.7	36.8	56.5	39. 7	59.6	47.8	59.7	42.5
$\mathbf{A}_{\mathbf{avg}}$	61.3	49.7	57.9	44.4	63.3	52.2	59.4	43.8	57.6	40.6	56.4	39.8	59.4	47.3	58.5	42.4
$\mathbf{C}_{\mathbf{mask}}$**-PCA**	61.8	52.6	58.8	41.2	62.3	53.2	60.1	41.6	61.5	52.6	59.2	40.3	62.2	51.5	59.1	40.5
$\mathbf{T}_{\mathbf{mask}}$**-PCA**	**63.3**	**56.2**	**62.6**	46.9	64.4	**57.3**	**62.6**	48.0	61.6	**55.8**	**62.5**	46.0	**65.4**	56.3	**64.1**	46.4

Table 1: Results of lexical feature classification experiments for the extended McRae feature norms (MC), CSLB norms (CS), WordNet Supersenses (SS) and BabelNet domains (BD). Results are reported in terms of F1 (%).

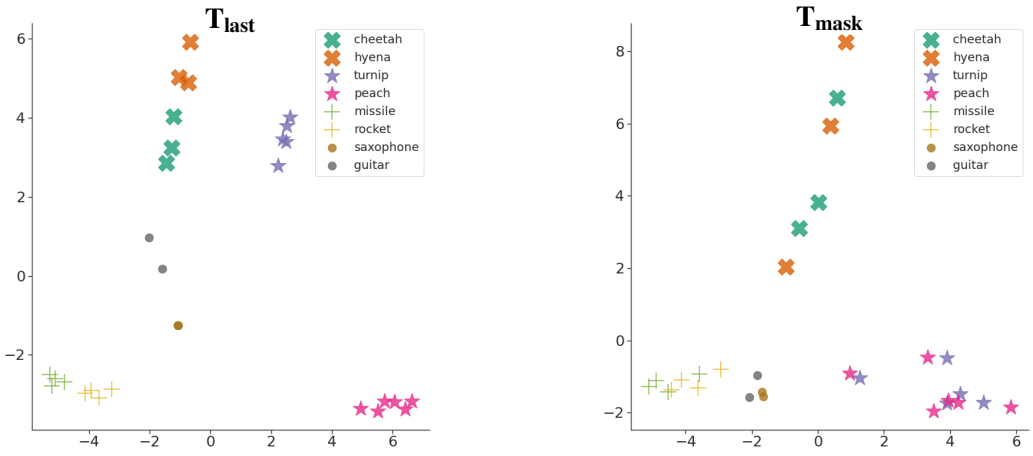

Figure 1: BERT-base topic-specific vectors when using the output vectors without using masking (left) and with masking (right). Words have been selected from the McRae dataset.

Dataset	**Type**	**Words**	**Properties**
McRae	Commonsense	475	49
CSLB	Commonsense	570	54
WN supersenses	Taxonomic	24,324	24
BN domains	Topical	43,319	34

Table 2: Overview of the considered datasets.

beddings correspond to its usage in the context of 'finance', 'stock market' and 'fiction'. These three embeddings roughly correspond to three different senses of the word[8]. This de-conflation or implicit disambiguation is also found for words such as *'cell'*, *'port'*, *'bulb'* or *'mail'*, which shows a striking relevance of the role of mail in the election topic, being semantically similar in the corresponding vector space to words such as 'telemarketing', 'spam' or 'wiretap'. In the case of *'fingerprint'*, we can also see some implicit disambiguation (distinguishing between fingerprinting in computer science, as a form of hashing, and the more traditional sense). However, we also see a more topical distinction, revealing differences between the role played by fingerprints in fictional works and forensic research. This tendency of capturing different contexts is more evidently shown in the last four examples. First, for *'sky'* and *'strength'*, the topic-wise embeddings do *not represent different senses*

[8]In fact, we can directly pinpoint these vectors to the following WordNet (Miller, 1995) senses: `partner.n.03`, `collaborator.n.03` and `spouse.n.01`.

WORD	TOPIC	NEAREST NEIGHBOURS
partner	{research, professor, science, education, institute}	beneficiary, creditor, investor, employer, stockholder
	{football, republican, coach, senate, representatives}	lobbyist, bookkeeper, cashier, stockbroker, clerk
	{game, book, novel, story, reception}	nanny, spouse, lover, friend, secretary
cell	{protein, disease, medical, cancer, cells}	lymphocyte, macrophage, axon, astrocyte, organelle
	{food, plant, water, gas, power, oil}	electrode, electrolyte, cathode, anode, substrate
	{physics, mathematics, space, ngc, theory}	surface, torus, mesh, grid, cone
port	{station, building, railway, historic, church}	harbor, seaport, dock, waterfront, city
	{radio, station, fm, software, data, forewings}	link, gateway, router, line, socket
	{game, book, novel, story, reception}	version, remake, compilation, patch, modification
bulb	{station, building, railway, historic, church}	lamp, transformer, dynamo, projector, lighting
	{protein, disease, medical, cancer, cells}	epithelium, ganglion, nucleus, gland, cortex
	{species, genus, described, description, flowers}	rootstock, fern, vine, tuber, clover
mail	{station, building, railway, historic, church}	cargo, grain, baggage, coal, livestock
	{game, book, novel, story, reception}	paper, jewelry, telephone, telegraph, typewriter
	{party, election, minister, elected, elections}	telemarketing, spam, wiretap, internet, money
fingerprint	{radio, station, fm, software, data, forewings}	signature, checksum, bitmap, texture, text
	{game, book, novel, story, reception}	cadaver, skull, wiretap, body, tooth
	{party, election, minister, elected, elections}	wiretap, forensics, postmortem, polygraph, check
sky	{greek, ancient, castle, king, roman}	underworld, sun, afterlife, zodiac, moon
	{river, lake, mountain, island, village}	horizon, ocean, earth, sun, globe
	{physics, mathematics, space, ngc, theory}	ionosphere, sun, globe, earth, heliosphere
strength	{food, plant, water, gas, power}	stiffness, ductility, hardness, permeability, viscosity
	{game, book, novel, story, reception}	intelligence, agility, charisma, power, telepathy
	{army, regiment, navy, ship, air}	morale, firepower, resistance, force, garrison
noon	{physics, mathematics, space, ngc, theory}	declination, night, equinox, perihelion, latitude
	{army, regiment, navy, ship, air}	dawn, sunset, night, morning, shore
galaxy	{physics, mathematics, space, ngc, theory}	nebula, quasar, pulsar, nova, star
	{game, book, novel, story, reception}	globe, future, world, planet, nation

Table 3: Nearest neighbours of topic-specific embeddings for a sample of words from the WordNet SuperSenses dataset, using BERT-base embeddings. The top 6 selected samples illustrate clear topic distributions per word sense, and the bottom 4 also show topical properties within the same sense. The most relevant words for each topic are shown under the **TOPIC** column.

of these words, but rather indicate different types of usage (possibly related to cultural or commonsense properties). Specifically, we see that the same sense of *'sky'* is used in mythological, landscaping and geological contexts. Likewise, *'strength'* is clustered into different mentions, but while this word also preserves the same sense, it is clearly used in different contexts: physical, as a human feature, and in military contexts. Finally, *'noon'* and *'galaxy'* (which only occur in two topics), also show this topicality. In both cases, we have representations that reflect their physics and everyday usages, for the same senses of these words.

As a final analysis, In Figure 1 we plot a two-dimensional PCA-reduced visualization of selected words from the McRae dataset, using two versions of the topic-specific vectors: $\mathbf{T_{mask}}$ and $\mathbf{T_{last}}$. In both cases, BERT-base was used to obtain the vectors. We select four pairs of concepts which are topically related, which we plot with the same datapoint marker (animals, plants, weapons and musical instruments). For $\mathbf{T_{last}}$, we can see that the different topic-specific representations of the same word are clustered together, which is in accordance with the findings from Ethayarajh (2019). For $\mathbf{T_{mask}}$, we can see that the representations of words with similar properties (e.g. *cheetah* and *hyena*) become more similar, suggesting that $\mathbf{T_{mask}}$ is more tailored towards modelling the semantic properties of words, perhaps at the expense of a reduced ability to differentiate between closely related words. The case of *turnip* and *peach* is particularly striking, as the vectors are clearly separated in the $\mathbf{T_{last}}$ plot, while being clustered together in the $\mathbf{T_{mask}}$ plot.

5 Conclusions

We have proposed a strategy for learning static word vectors, in which topic models are used to help select diverse mentions of a given target word and a contextualized language model is subsequently used to infer vector representations from the selected mentions. We found that selecting an equal number of mentions per topic substantially outperforms purely random selection strategies. We also considered the possibility of learning a weighted average of topic-specific vector representations, which in principle should allow us to "tune" word representations to different tasks, by learning task-specific topic importance weights. However, in practice we found that a standard average of the topic specific vectors leads to a comparable performance, suggesting that the outperformance of our vectors comes from the fact that they are obtained from a more diverse set of contexts.

Acknowledgments

This work was performed using the computational facilities of the Advanced Research Computing @ Cardiff (ARCCA) Division, Cardiff University and using HPC resources from GENCI-IDRIS (Grant 2021-[AD011012273]).

References

Israa Alghanmi, Luis Espinosa Anke, and Steven Schockaert. 2020. Combining bert with static word embeddings for categorizing social media. In *Proceedings of the Sixth Workshop on Noisy User-generated Text*, pages 28–33.

Asaf Amrami and Yoav Goldberg. 2019. Towards better substitution-based word sense induction. *arXiv:1905.12598*.

Federico Bianchi, Silvia Terragni, and Dirk Hovy. 2020. Pre-training is a hot topic: Contextualized document embeddings improve topic coherence. *CoRR*, abs/2004.03974.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting pretrained contextualized representations via reductions to static embeddings. In *Proceedings ACL*, pages 4758–4781.

Georgeta Bordea, Paul Buitelaar, Stefano Faralli, and Roberto Navigli. 2015. SemEval-2015 task 17: Taxonomy extraction evaluation (TExEval). In *Proceedings SemEval*, pages 902–910.

Georgeta Bordea, Els Lefever, and Paul Buitelaar. 2016. Semeval-2016 task 13: Taxonomy extraction evaluation (texeval-2). In *Proceedings SemEval*, pages 1081–1091.

Jose Camacho-Collados and Roberto Navigli. 2017. BabelDomains: Large-scale domain labeling of lexical resources. In *Proceedings EACL*, pages 223–228.

Gabriella Chronis and Katrin Erk. 2020. When is a bishop not like a rook? when it's like a rabbi! multi-prototype BERT embeddings for estimating semantic relationships. In *Proceedings CoNLL*, pages 227–244.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings NAACL-HLT*.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings EMNLP*, pages 55–65.

Maxwell Forbes, Ari Holtzman, and Yejin Choi. 2019. Do neural language representations learn physical commonsense? *Proceedings CogSci*.

Prodromos Kolyvakis, Alexandros Kalousis, and Dimitris Kiritsis. 2018. Deepalignment: Unsupervised ontology matching with refined word vectors. In *Proceedings NAACL-HLT*, pages 787–798.

Na Li, Zied Bouraoui, and Steven Schockaert. 2019. Ontology completion using graph convolutional networks. In *Proceedings ISWC*, pages 435–452.

Shaohua Li, Tat-Seng Chua, Jun Zhu, and Chunyan Miao. 2016. Generative topic embedding: a continuous representation of documents. In *Proceedings ACL*.

Qianchu Liu, Diana McCarthy, and Anna Korhonen. 2020. Towards better context-aware lexical semantics: Adjusting contextualized representations through static anchors. In *Proceedings EMNLP*, pages 4066–4075.

Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2015. Topical word embeddings. In *Proceedings AAAI*, pages 2418–2424.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Yukun Ma, Erik Cambria, and Sa Gao. 2016. Label embedding for zero-shot fine-grained named entity typing. In *Proceedings COLING*, pages 171–180.

Ken McRae et al. 2005. Semantic feature production norms for a large set of living and nonliving things. *Behavior research methods*, 37:547–559.

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

George A Miller. 1995. Wordnet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

Fedor Nikolaev and Alexander Kotov. 2020. Joint word and entity embeddings for entity retrieval from a knowledge graph. In *Proceedings ECIR*, pages 141–155.

Nicole Peinelt, Dong Nguyen, and Maria Liakata. 2020. tBERT: Topic models and BERT joining forces for semantic similarity detection. In *Proceedings ACL*, pages 7047–7055.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings EMNLP*, pages 1532–1543.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B. Viégas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of BERT. In *Proceedings NeurIPS*, pages 8592–8600.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings EMNLP*, pages 3982–3992.

Bei Shi, Wai Lam, Shoaib Jameel, Steven Schockaert, and Kwun Ping Lai. 2017. Jointly learning word embeddings and latent topics. In *Proceedings SIGIR*, pages 375–384.

Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. 2013. Zero-shot learning through cross-modal transfer. In *Proceedings NIPS*, pages 935–943.

Laure Thompson and David Mimno. 2020. Topic modeling with contextualized word representation clusters. *CoRR*, abs/2010.12626.

Ivan Vulic, Edoardo Maria Ponti, Robert Litschko, Goran Glavas, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. In *Proceedings EMNLP*, pages 7222–7240.

Lixing Zhu, Deyu Zhou, and Yulan He. 2020. A neural generative model for joint learning topics and topic-specific word embeddings. *Trans. Assoc. Comput. Linguistics*, 8:471–485.