Breaking Down Word Semantics from Pre-trained Language Models through Layer-wise Dimension Selection

Anonymous ACL submission

Abstract

Contextual word embeddings obtained from pre-trained language model (PLM) have proven effective for various natural language processing tasks at the word level. However, interpreting the hidden aspects within embeddings, such as syntax and semantics, remains challenging. Disentangled representation learning has emerged as a promising approach, which separates specific aspects into distinct embeddings. Furthermore, different linguistic knowledge is believed to be stored in different layers of PLM. This paper aims to disentangle semantic sense from BERT by applying a binary mask to middle outputs across the layers, without updating pre-trained parameters. The disentangled embeddings are evaluated through binary classification to determine if the target word in two different sentences has the same meaning. Experiments with cased $\text{BERT}_{\text{base}}$ show that leveraging layer-wise information is effective and disentangling semantic sense further improve performance.

1 Introduction

Pre-trained language models (PLMs) like BERT (Devlin et al., 2019) have been successful in generating contextual word embeddings for diverse word-level NLP tasks, such as named entity recognition, part-of-speech (PoS) tagging, coreference resolution, and word sense disambiguation (WSD) (Raganato et al., 2017). The power of contextual word embeddings lies in their capacity to capture the overall traits of a word, enabling them to perform well across wide-ranging NLP tasks.

Although contextual word embeddings are capable of capturing diverse hidden aspects such as semantic sense, syntactic role, and sentiment, interpreting these factors remains a challenge. To address this issue, disentangled representation learning (DRL) (Do and Tran, 2019; Carbonneau et al., 2022a) has emerged as an approach to separate specific aspects into distinct representations. DRL has shown promise in computer vision through methods such as InfoGAN (Chen et al., 2016), but its potential for NLP tasks has yet to be fully explored (Vishnubhotla et al., 2021). In the text domain, Xu et al. (2020) proposed a constraint on the latent space of variational autoencoder (VAE) with PLM to control sentiment and topic. Colombo et al. (2021) utilized mutual information (MI) to disentangle aspects from text for fair classification and textual style transfer. The groundwork for this paper was laid by Zhang et al. (2021), who trained binary masks to find subnetworks within BERT using masking transformers (Zhao et al., 2020) to disentangle sentiment from genre, toxicity from dialect, and syntax from semantic.

w = raise
s_1 = We visited a farm where they raise chickens.
s_2 = A few important questions were raised after the attack.

Figure 1: A sample labeled as false from the WiC dataset, indicating that the target word w in sentences s_1 and s_2 do not correspond to the same meaning.

On the other hand, several studies have shown that linguistic knowledge like syntactic and semantic aspects are encoded in different layers of PLM (Jawahar et al., 2019; Tenney et al., 2019; Hewitt and Manning, 2019; Bommasani et al., 2020). Furthermore, each self-attention head seems to focus on different aspect of language information (Vig and Belinkov, 2019; Clark et al., 2019; Zhao and Bethard, 2020). However, it is common practice to represent a word simply as the sum of the last four layers' hidden states even for embedding-based WSD (Scarlini et al., 2020b; Scarlini et al., 2020a; Loureiro et al., 2021). Despite its practicality and widespread usage, it overlooks the heterogeneous information captured by different layers.

This paper studies to disentangle the semantic sense of word by masking the middle outputs from BERT in a layer-wise manner. The disentangled embeddings are evaluated by binary classification

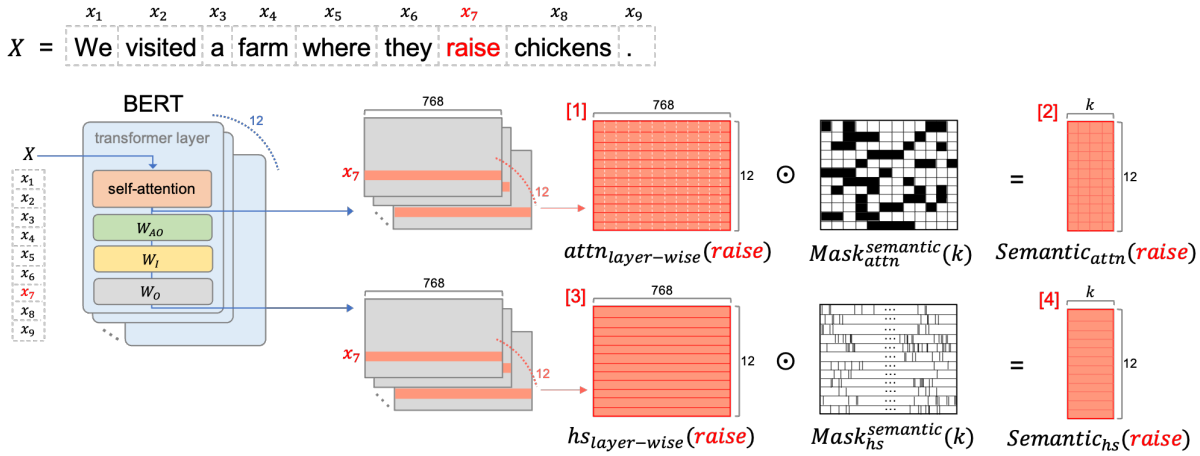

Figure 2: Taking BERT$_{\text{base}}$, it illustrates the way to disentangle semantic sense of the word *raise* from middle outputs (layer-wise hidden states or self-attention outputs) of input sentence X. Binary trained $Mask_{*}^{semantic}(k)$ disentangles sense and reduces the output dimensionality from 768 to k for each layer. $attn_{layer-wise}(x)$ is divided into 12 attention heads and $Mask_{attn}^{semantic}(k)$ masks it head-wise, while $Mask_{hs}^{semantic}(k)$ masks $hs_{layer-wise}(x)$ dimension-wise. If a word consists of more than one token, mean of tokens' outputs is used to represent a word.

task, specifically utilizing the WiC format (shown in Figure 1). The task has practical implications in real-world scenarios, such as re-ranking search engine results with relevance for ambiguous queries. The contributions of this work are as follows:

- The study presents a method to disentangle semantic sense from BERT, masking embedding dimensions without updating pre-trained model parameters.
- Layer-wise information is successfully used to determine the semantic similarity between the target word in two different sentences.
- It includes a comprehensive analysis about the behavior of the trained mask across the layers.

2 Approach

The aim is to disentangle semantic sense from layer-wise middle outputs about word obtained from pre-trained BERT (specifically, Cased BERT$_{\text{base}}$ is used). A binary mask is trained to select k dimensions of hidden states or self-attention outputs (not the attention weights) for each layer, as illustrated in Figure 2. Using the subsets of them, the semantic similarities between the target word w in sentences s_1 and s_2 are computed. Subsequently, for evaluation, a binary classifier is trained as a predictor-based metric (Carbonneau et al., 2022b) to determine whether the two w share the same meaning or not.

2.1 Disentanglement by Masking

Considering a transformer layer of BERT (Zhao et al., 2020), the input $\mathbf{X} \in \mathbb{R}^{N \times h}$ (N: maximum token length, h: hidden size) undergoes multi-head self-attention. The outputs are then transformed by W_{AO} and further passed to the next layer through W_I and W_O. In this paper, the attention outputs before W_{AO} are used to preserve head-wise information, while the hidden states after W_O are used, to represent a word. While it is common to use hidden states, in line with previous studies emphasizing the interpretability of attention weights, the outputs right after self-attention are explored as an alternative. This is because hidden states tend to lose such information through W_{AO}, W_I and W_O.

Together with semantic sense, the study attempts to train mask for PoS or semantic role (SVO: Subject, Verb, Object), taking inspiration from Zhang et al. (2021) who tried to disentangle two distinct aspects by minimizing the overlap between masks using L_{ovl} [1]. Unlike the prior study that masks a part of architecture and preserve the original embedding size, the masking approach in this work directly masks embeddings and reduces the dimensionality, akin to feature selection.

2.1.1 Masking Self-attention Outputs

$$Semantic_{attn}(x) = Mask_{attn}^{semantic}(k) \odot attn_{layer-wise}(x) \quad (1)$$

A binary mask $Mask_{attn}^{semantic}(k)$ selects k/a (a: attention head size) [2] heads per layer to get $Semantic_{attn}(x)$, which is the disentangled se-

[1] To train distinct two masks $M^{(a)}$, $M^{(b)}$ for each layer l, $L_{ovl} = \frac{1}{|L|} \sum_{l \in L} \sum_{i,j} \mathbb{1}(M_{i,j}^{(a)} + M_{i,j}^{(b)} > 1)$

[2] If $a = 64$ and $k = 256$, the mask selects 4 heads.

mantic sense representation of x. The mask has the same size as $attn_{layer-wise}(x) \in \mathbb{R}^{h \times l}$ (h: hidden size, l: number of layers, e.g., h=768 and l=12 in BERT$_{\text{base}}$), and reduces the size of $attn_{layer-wise}(x)$ from $\mathbb{R}^{h \times l}$ to $\mathbb{R}^{k \times l}$.

2.1.2 Masking Hidden States

$$Semantic_{hs}(x) = Mask_{hs}^{semantic}(k) \odot hs_{layer-wise}(x) \quad (2)$$

About hidden states, $Mask_{hs}^{semantic}(k)$ chooses k dimensions from $hs_{layer-wise}(x)$ per layer to get $Semantic_{hs}(x)$.

2.1.3 Loss

The loss builds upon the *Triplet Loss* framework proposed by Zhang et al. (2021), who aimed to disentangle two aspects a and b from a text by training binary masks utilizing a *straight-through* estimator (Bengio et al., 2013) over the architecture of PLM. It defines triplets (x_0, x_1, x_2) that satisfy two conditions: (1) $y_0^{(a)} = y_1^{(a)} \neq y_2^{(a)}$ [3], and (2) $y_0^{(b)} \neq y_1^{(b)} = y_2^{(b)}$. The customized losses $L^{(a)}$ and $L^{(b)}$ are defined as follows. ($z^{(a)}$ and $z^{(b)}$ are the disentangled embeddings of a and b respectively.)

$$L^{(a)} = max(-cos(z_0^{(a)}, z_1^{(a)}) + cos(z_0^{(a)}, z_2^{(a)}), 0)_4$$
$$L^{(b)} = max(-cos(z_0^{(b)}, z_2^{(b)}) + cos(z_0^{(b)}, z_1^{(b)}), 0) \quad (3)$$

The final loss depends on the number of aspects being disentangled, $L_{final} = L^{(a)}$ for a single aspect and $L_{final} = \frac{1}{2}\lambda(L^{(a)} + L^{(b)}) + (1 - \lambda)L_{ovl}$ for two aspects. Here the hyperparameter λ serves as a weighting factor, balancing the loss with L_{ovl}.

2.2 Layer-wise Similarity Calculator

Given two sentences s_1 and s_2 containing a target word w, the similarity calculator $sim_{x,y} = f(x, y)$ takes representations of w_{s_1} and w_{s_2} (w_{s_i} is w in s_i) as inputs. It computes cosine similarity layer-wise, resulting in a vector $\in \mathbb{R}^{1 \times l}$ (l=12 in BERT$_{\text{base}}$).

3 Experiment

3.1 Dataset

This study uses three datasets: WiC (Pilehvar and Camacho-Collados, 2019), CoarseWSD-20 (Loureiro et al., 2021), and SemCor (Miller et al., 1994). WiC is formatted as binary classification without semantic sense labels (Figure 1), with train/dev/test splits and unlabeled test data. CoarseWSD-20 consists of 20 words with an average of 2.65 distinct semantic senses per word, with train/test splits of 23,370 and 10,196 sentences, respectively. SemCor consists of documents where some words in certain sentences are tagged with semantic senses. It includes a total of 276,878 sense-tagged sentences, 20,398 unique words, and an average of 1.64 distinct senses per word.

			train	dev	test
Binary Classifer		WiC	4,885	543	638
		CoarseWSD-20	180,000	20,000	100,000
		SemCor	285,088	31,672	89,760
Masker	semantic	CoarseWSD-20	168,264	18,696	-
		SemCor	402,480	44,712	-
	(+) PoS	CoarseWSD-20	163,320	18,144	-
		SemCor	235,368	26,144	-
	(+) SVO	CoarseWSD-20	16,704	1,856	-
		SemCor	38,768	4,304	-

Table 1: Processed data statistics of each dataset.

For experimentation, three datasets are processed and train/dev/test subsets are extracted (Table 1). See Section A.1 for the data processing method.

3.2 Experimental Details

The masker and binary classifier are trained using a batch size of 8, Adam optimizer (Kingma and Ba, 2017), and a learning rate of 0.01. Early stopping is utilized based on the dev set performance. The masker training explores different hidden states dimensions ($n_dim = [128, 384, 512]$) and the number of attention heads ($n_head = [4, 6, 8]$). The binary classifier consists of a linear layer followed by a sigmoid function. It takes cosine similarities between a pair and outputs a Boolean value indicating whether w_{s_1} and w_{s_2} share the same meaning.

3.3 Result

Table 2 compares the performance of the proposed approaches with the Baseline (highlighted in blue), which sums the last four layers' hidden states to represent a word. For Figure 2, the Baseline uses the last four rows of [1](attention) or [3](hidden), the (+) Layer-wise Calculator uses the entire [1] or [3], and the (+) Disentanglement by Masking approach utilizes [2] or [4]. WiC lacks semantic sense labels, so the evaluation is limited to the layer-wise calculator as it cannot be converted into triplets for masker training. The best score per dataset is in **bold**. Regardless of the dataset, layer-wise information improves performance by 1-2%

[3] It means x_0 and x_1 are same in terms of aspect a, while x_0 and x_2 are differ.

[4] The function $cos(x, y)$ computes the cosine similarity between two vectors x and y.

Figure 3: The visualization of $Mask_{hs}^{Semantic}(128) \in \mathbb{R}^{768\times12}$ which is the trained binary mask. Black denotes the selected(=1) dimensions, and white is unselected(=0) dimensions.

WiC		attention	hidden
Baseline (Cased BERT$_{\text{base}}$)		0.682 ($\pm$ 0.002)	0.651 ($\pm$ 0.003)
(+) Layer-wise Calculator		**0.705** ($\pm$ 0.002)	0.667 ($\pm$ 0.002)
CoarseWSD-20		attention	hidden
Baseline (Cased BERT$_{\text{base}}$)		0.754 ($\pm$ 0.001)	0.743 ($\pm$ 0.000)
(+) Layer-wise Calculator		0.781 ($\pm$ 0.000)	0.768 ($\pm$ 0.001)
(+) Disentanglement by Masking	semantic	0.788 ($\pm$ 0.007)	**0.802** ($\pm$ 0.002)
	w/ PoS	0.790 ($\pm$ 0.003)	**0.802** ($\pm$ 0.004)
	w/ SVO	0.796 ($\pm$ 0.006)	0.793 ($\pm$ 0.006)
SemCor		attention	hidden
Baseline (Cased BERT$_{\text{base}}$)		0.777 ($\pm$ 0.001)	0.758 ($\pm$ 0.007)
(+) Layer-wise Calculator		0.780 ($\pm$ 0.001)	0.764 ($\pm$ 0.003)
(+) Disentanglement by Masking	semantic	0.774 ($\pm$ 0.001)	**0.781** ($\pm$ 0.001)
	w/ PoS	0.781 ($\pm$ 0.003)	0.776 ($\pm$ 0.008)
	w/ SVO	0.775 ($\pm$ 0.004)	0.776 ($\pm$ 0.005)

Table 2: Mean accuracy and standard deviation in binary classification across three test sets for each dataset. Masker configuration: $n_dim = 128$, $n_head = 6$

compared to the baseline. Moreover, incorporating disentangled embeddings further enhances hidden states (hidden) performance by around 2%.

Despite being uncommon as word representations, self-attention outputs that preserve head-wise information seem to be inherently superior to hidden states for semantic representation. However, the masking approach proves more effective for hidden states, and disentangled hidden states perform the best among the experimental units. It is notable that there is improvement even with a 1/6 reduction in the embedding size from 768 to 128. In addition, adjacent layers tend to exhibit similar dimensions selected by the mask, particularly in the later layers for semantic sense (Figure 4) and in the earlier layers for PoS (Figure 5), regardless of each dataset or implementation. While not generating the identical masks every time, the trained masks for hidden states shared these similar attributes. However, no such tendency is observed with attention outputs.

Using layer-wise information instead of collapsing it led to improved performance. The trained weights of binary classifier, which utilizes the mask shown in Figure 3, were: [-2.914, -0.724, 1.403, 2.640, 3.660, 2.565, -3.712, -1.123, 0.081, 1.649, 0.518, 5.169]. Considering that the binary classifier determines the priority of information from each layer, it is observed that similarities between disentangled embeddings of layers 1, 2 and 7, 8 are rejected, while the last layer's similarity is most heavily reflected. Training PoS or SVO masks alongside the semantic sense mask, in anticipation of achieving additional improvement, did not yield the expected results. On the other hand, layer-wise information proves its own effectiveness, as our methodology achieved a 70.5% accuracy on WiC testset, comparable to SenseBERT$_{\text{BASE}}$ (Levine et al., 2020), a model explicitly fine-tuned for sense disambiguation, which achieved a score of 70.3%.

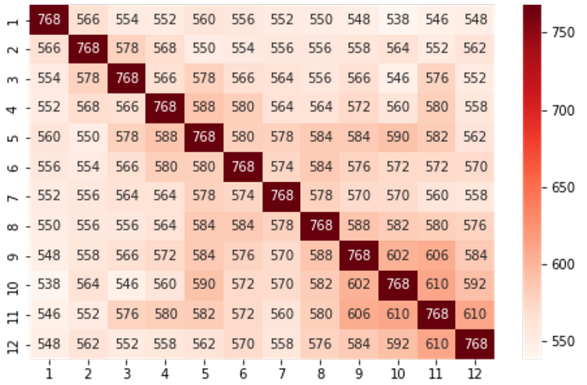

Figure 4: Number of equivalent values out of 768 dimensions across layers of trained mask depicted in Figure 3

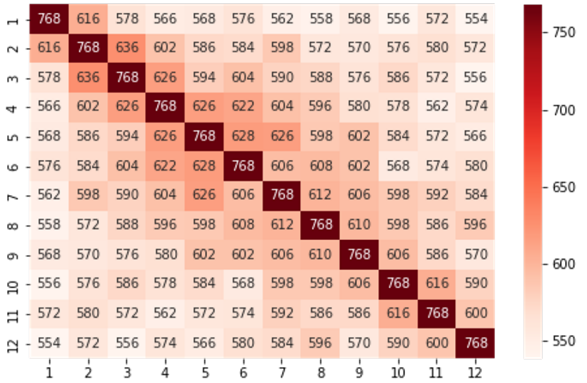

Figure 5: Number of dimensions with equivalent values across layers in $Mask_{hs}^{PoS}(128)$

4 Conclusion

The study investigated extracting the intrinsic semantic sense of words from pre-trained BERT$_{\text{base}}$ without modifying it, and uncovered the presence of noisy dimensions within embeddings to represent semantic sense. The merit of the masking approach in this paper can be extended to the ability to obtain aspect-specific representations by applying diverse learned masks to pre-existing offline saved embeddings, such as those utilized in a search engine system where embeddings per document are already indexed. Limitations and future directions are discussed in Section 5.

5 Limitation

The main limitation of this study is its focus on Cased $\text{BERT}_{\texttt{base}}$ exclusively. In experiments using Uncased $\text{BERT}_{\texttt{base}}$, the Baseline outperformed Cased $\text{BERT}_{\texttt{base}}$, which may explain why Uncased $\text{BERT}_{\texttt{base}}$ is commonly preferred as the backbone model for embedding-based WSD. However, the performance improvement of Uncased $\text{BERT}_{\texttt{base}}$ through disentanglement was marginal.

The difference between Cased and Uncased models lies in how the tokenizer constructs its own vocabulary, specifically regarding whether or not the casing of tokens is preserved. The vocabulary of Cased PLM can be considered more fragmented (less normalized) compared to that of Uncased PLM. Cased tokens likely possess specific traits, such as their frequent usage for proper nouns like names of places or individuals. As the tokenizer logic itself influences word representation, contextual embeddings derived from cased tokens may contain unique information absent in those generated from uncased tokens. This hypothesis could be expanded to propose that there might be additional disentangled aspects within the embeddings. Nevertheless, this hypothesis remains speculative, and further research is required to investigate the relationship between vocabulary composition, differences in embeddings, and disentanglement methodologies.

References

Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR*, abs/1308.3432.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781, Online. Association for Computational Linguistics.

Marc-André Carbonneau, Julian Zaïdi, Jonathan Boilard, and Ghyslain Gagnon. 2022a. Measuring disentanglement: A review of metrics. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15.

Marc-André Carbonneau, Julian Zaïdi, Jonathan Boilard, and Ghyslain Gagnon. 2022b. Measuring disentanglement: A review of metrics. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *Advances in neural information processing systems*, 29.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*.

Pierre Colombo, Chloé Clavel, and Pablo Piantanida. 2021. A novel estimator of mutual information for learning to disentangle textual representations. *CoRR*, abs/2105.02685.

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.

Kien Do and Truyen Tran. 2019. Theory and evaluation metrics for learning disentangled representations. *arXiv preprint arXiv:1908.09961*.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

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

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

Yoav Levine, Barak Lenz, Or Dagan, Ori Ram, Dan Padnos, Or Sharir, Shai Shalev-Shwartz, Amnon Shashua, and Yoav Shoham. 2020. SenseBERT: Driving some sense into BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4656–4667, Online. Association for Computational Linguistics.

Daniel Loureiro, Kiamehr Rezaee, Mohammad Taher Pilehvar, and Jose Camacho-Collados. 2021. Analysis and evaluation of language models for word sense disambiguation. *Computational Linguistics*, 47(2):387–443.

George A. Miller, Martin Chodorow, Shari Landes, Claudia Leacock, and Robert G. Thomas. 1994. Using a semantic concordance for sense identification.

In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. 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 1267–1273, Minneapolis, Minnesota. Association for Computational Linguistics.

Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017. Word sense disambiguation: A unified evaluation framework and empirical comparison. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 99–110, Valencia, Spain. Association for Computational Linguistics.

Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. 2020a. Sensembert: Context-enhanced sense embeddings for multilingual word sense disambiguation. volume 34.

Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. 2020b. With more contexts comes better performance: Contextualized sense embeddings for all-round word sense disambiguation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3528–3539, Online. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Jesse Vig and Yonatan Belinkov. 2019. Analyzing the structure of attention in a transformer language model. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 63–76, Florence, Italy. Association for Computational Linguistics.

Krishnapriya Vishnubhotla, Graeme Hirst, and Frank Rudzicz. 2021. An evaluation of disentangled representation learning for texts. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1939–1951.

Peng Xu, Jackie Chi Kit Cheung, and Yanshuai Cao. 2020. On variational learning of controllable representations for text without supervision. In *International Conference on Machine Learning*, pages 10534–10543. PMLR.

Xiongyi Zhang, Jan-Willem Meent, and Byron Wallace. 2021. Disentangling representations of text by masking transformers. pages 778–791.

Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. 2020. Masking as an efficient alternative to finetuning for pretrained language models. pages 2226–2241.

Yiyun Zhao and Steven Bethard. 2020. How does BERT's attention change when you fine-tune? an analysis methodology and a case study in negation scope. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4729–4747, Online. Association for Computational Linguistics.

A Appendix

A.1 Data Processing

Raw train data of WiC and CoarseWSD-20 is split into train/dev (9:1 ratio), using CoarseWSD-20 test data and WiC dev data for test respectively. SemCor is divided into train/test (8:2 ratio), with the train set further split into train/dev (9:1 ratio). For CoarseWSD-20 and SemCor, triplets are sampled to train masker (Section 2.1.3). PoS and SVO labels for words are tagged by Spacy [5] and Textacy [6]. Furthermore, True- and false-labeled pairs are sampled for binary classification, following the format of WiC. To ensure robust evaluation, three distinct processed datasets are prepared, each of equal size.

[5] https://github.com/explosion/spaCy
[6] https://github.com/chartbeat-labs/textacy