LIORI at SemEval-2021 Task 2: Span Prediction and Binary Classification approaches to Word-in-Context Disambiguation

Adis Davletov
RANEPA, Moscow, Russia
Lomonosov Moscow State University, Moscow, Russia
davletov-aa@ranepa.ru
Nikolay Arefyev
Lomonosov Moscow State University, Moscow, Russia
Samsung Research Center Russia, Moscow, Russia
HSE University, Moscow, Russia
nick.arefyev@gmail.com
Denis Gordeev
RANEPA, Moscow, Russia
gordeev-di@ranepa.ru
Alexey Rey
RANEPA, Moscow, Russia
rey-ai@ranepa.ru

Abstract

This paper presents our approaches to SemEval-2021 Task 2: Multilingual and Cross-lingual Word-in-Context Disambiguation task. The first approach attempted to reformulate the task as a question answering problem, while the second one framed it as a binary classification problem. Our best system, which is an ensemble of XLM-R based binary classifiers trained with data augmentation, is among the 3 best-performing systems for Russian, French and Arabic in the multilingual subtask. In the post-evaluation period, we experimented with batch normalization, subword pooling and target word occurrence aggregation methods, resulting in further performance improvements.

1 Introduction

In the Semeval-2021 Task 2: Multilingual and Cross-lingual Word-in-Context Disambiguation task, the participants were asked to classify whether the target word, occurring in two sentences (sentence1 and sentence2), is used in the same or in a different meaning. The two sentences could be in the same language or different languages. There were two subtasks:

- **Multilingual Word-in-Context Disambiguation**, where both sentences were in the same language, either Arabic, Chinese, English, French or Russian
- **Cross-lingual Word-in-Context Disambiguation**, where the first sentence was in English and the second one was either in Arabic, Chinese, French, or Russian.

More detailed information regarding the task is provided by Martelli et al. (2021).

We participated in both tracks and experimented with two approaches[1]. The first approach fine-tunes XLM-R (Conneau et al., 2020a) as a question answering system searching in the second sentence for a word with the same meaning as the target word in the first sentence. The second approach fine-tunes XLM-R as a binary classifier, and ensembles several such classifiers. Also, we used data augmentation to double the number of training examples. This second approach took the 2nd place for the monolingual subtask in Arabic and the 3rd place for the monolingual subtask in French and Russian. In the cross-lingual subtask, the system ranked 6th for French and Arabic. The same system was applied to all subtasks and languages.

During the post-evaluation period, we performed thorough experiments with our system. We compared different subword pooling methods, including mean, max, first pooling and their combinations, and found that combinations do not help and mean pooling is overall the best choice. Unlike pooling, instead of a simple concatenation of contextualized embeddings for the target word occurrences, it is helpful to combine their difference and normalized component-wise product. Finally, we found it beneficial to add a batch normalization

[1] https://github.com/davletov-aa/mcl-wic

Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021), pages 780–786
Bangkok, Thailand (online), August 5–6, 2021.

layer before feeding those vectors into the classification head.

2 Related Work

Word Sense Disambiguation (WSD) is the task of associating the occurrence of a word in a text with its correct meaning from a predefined inventory of senses (Navigli, 2009; Scarlini et al., 2020a). Word-in-Context Disambiguation is a new declination of WSD aiming to evaluate the ability of modern language models to accurately represent context-sensitive words (Pilehvar and Camacho-Collados, 2019; Scarlini et al., 2020b). Its advantage is that it does not rely on pre-defined sense inventories. Because Word Sense Disambiguation relies on world knowledge for successful solving (Navigli, 2009), modern large pre-trained models show promising results in solving this task.

Among such works, we can mention ARES (Scarlini et al., 2020b). ARES is a semi-supervised approach for creating sense embeddings. The authors use BERT and UKB (Agirre et al., 2014) to find contexts that are similar to each other and link them to meanings in WordNet (Miller et al., 1990). Then, they enrich synset contexts with collocational information from SyntagNet (Maru et al., 2019). Finally, they enrich SemCor (Miller et al., 1993) contexts and WordNet glosses to create sense-level representations. ARES performs better than models with a comparable number of parameters such as BERT or RoBERTa (Liu et al., 2019).

However, there has been substantial progress in the field of language modelling since BERT first appeared. Many researchers have noticed that BERT is undertrained and that training it longer and on more data, increases the model performance. Among such new models, we may name XLM-RoBERTa (XLM-R) (Conneau et al., 2020a). XLM-R, as well as BERT, is based on a Transformer model (Vaswani et al., 2017). XLM-R uses masked language modelling objective (Devlin et al., 2018; Lample and Conneau, 2019) for model training, where some tokens are replaced with a special ”[MASK]” token and the model is to restore the masked tokens. XLM-R was trained on a cleaned two-terabyte CommonCrawl Corpus in 100 languages.

A new promising approach to language task modelling is treating any natural language task as a question answering problem. Among such works, we can mention (Cohen et al., 2020) where the authors restructured relation classification as a Question Answering (QA) like span prediction problem. It allowed them to get state-of-the-art results for TACRED and SemEval 2010 task 8 datasets. We decided to adopt a similar approach to the task of word sense disambiguation.

Set	Pos	Neg
train-en-en	4000	4000
dev-en-en	500 (0)	500 (0)
dev-ar-ar	500 (349)	500 (351)
dev-ru-ru	500 (337)	500 (363)
dev-fr-fr	500 (366)	500 (334)
dev-zh-zh	500 (323)	500 (377)
trial-xx-xx	x (x)	y (y)

Table 1: Statistics of the data provided by organizers. The numbers in brackets show the portion used as training examples. In trial set there were up to 8 examples for each of 9 multilingual and cross-lingual sets.

3 Submitted Systems Description

Our systems are based on XLM-RoBERTa (XLM-R) model Conneau et al. (2020b). We used XLM-R large model as a backbone in all our submissions but switched to XLM-R base for some of the post-evaluation experiments. Two model training scenarios have been tested. In the first case (AG), due to the symmetric nature of the dataset, we decided to augment the dataset and flip the first and the second sentences. In the second case (MTL), multi-task learning was applied. More detailed descriptions are provided in the following sections. We used transformers library (Wolf et al., 2019).

3.1 Data

In all our experiments we used only the datasets provided in the shared task. For training, we employed the whole English train set, 70% of the development sets for other languages and all the trial data. The remaining data were used to select hyperparameters and do early stopping. We employed a lexical split resulting in different target words for training and validation. Table 1 presents detailed statistics. Optionally, in systems with **AG** suffix, train and test time data augmentation was performed by swapping sentences in each example to double the amount of data. If the predictions for symmetric examples were conflicting with each other we assumed the prediction is negative.

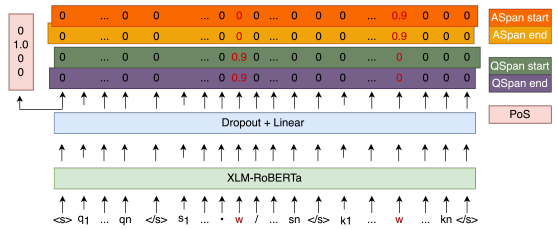

Figure 1: QA-based model architecture

3.2 QA Systems

Inspired by the work of Cohen et al. (2020), in our preliminary experiments we tried to solve MCL-WIC in Question Answering (QA) task manner, where we predict the start and end positions (the span) of the answer in a given text.

Given the target word w and a pair of tokenized sentences $[s_1 \ldots \boldsymbol{w} \ldots s_n]$ and $[k_1 \ldots \boldsymbol{w} \ldots k_n]$, we form the following input to XLM-R model with marked by • and / symbols target word in the first sentence:
$[CLS]$ *Find the same sense of the marked word* $[EOS]$ $s_1 \ldots$ • $\boldsymbol{w}$ / $\ldots$ s_n $[EOS]$ $k_1 \ldots \boldsymbol{w} \ldots k_n$ $[EOS]$. We tokenize target word in context and its left and right contexts for both sentences separately.

The architecture of our QA system could be seen in Figure 1. We predict the span A (answer) of the target word in the second sentence if it is used in the same meaning as in the first sentence and the span of the $[CLS]$ token otherwise. Also, we additionally predict the span of Q (question) of the target word in the first sentence. We use a dropout layer followed by a linear layer over XLM-R output o_i from the last layer at timesteps i to predict the probability that o_i is the start or the end of the spans Q and A.

As for each target word we had its part of speech label (PoS), we decided to predict it using a linear layer over the output corresponding to $[CLS]$ token from the last layer of XLM-R.

During the training process, we optimize the weighted sum of cross-entropy losses of A span, Q span, and PoS predictions. And as the corresponding weights, we take the softmax over the learnable weights' vector $V \in R^3$.

We fine-tuned the models in the settings from Table 2. Four times per training epoch we were validating our models and saving the best one. During the inference we assumed the positive answer if the model predicted possible span A that satisfied conditions $A_{start} < A_{end}$ and $A_{start} > PrefixQuestion$. We did not try to train QA systems with symmetric data augmentation.

Hyperparameter	Value
weight decay	0.1
warmup_proportion	0.1
dropout	0.1
learning rate	1e-4
learning_rate_scheduler	linear_warmup
optimizer	Adam
epochs	50
batch samples	64
max_sequence_length	256
max_gradient_norm	1.0

Table 2: Training hyperparameters of **MTL-EN** and **MTL-XX** systems, submitted to the competition

Further, we will be referring to the model validated on the English development set as the **MTL-EN** model. And as **MTL-XX** we will be referring to the models validated on one of the remaining development sets for the Russian, Arabic, French and Chinese languages.

3.3 BC Systems

Along with QA models, we tried a more traditional and straightforward approach of fine-tuning XLM-R as a binary classifier (BC).

So, given the target word w and a pair of tokenized sentences $[s_1 \ldots \boldsymbol{w} \ldots s_n]$ and $[k_1 \ldots \boldsymbol{w} \ldots k_n]$ we formed the following input example to XLM-R model:
$[CLS]$ $s_1 \ldots \boldsymbol{w} \ldots s_n$ $[EOS]$ $k_1 \ldots \boldsymbol{w} \ldots k_n$ $[EOS]$. The sentences were tokenized the same way as in QA models.

We feed it to XLM-R and pool outputs o_s and o_k from the last layer from the subwords corresponding to the target word in two sentences. In our submissions we either took the output from the **first** subword, or used **max** pooling. In the post-evaluation, we also tried **mean** pooling and found, that it consistently provides the best results. Then we tried concatenating it with first (**mf**) or max (**mm**) pooled vectors, as well as both of them (**mmf**). Finally, we tried concatenating min, max and mean pooled outputs (**mmm**).

After obtaining fixed-sized representations of the first and the second target word occurrence, we concatenate them and feed them to the binary classifier, which is the sequence of dropout, linear, tanh, dropout and linear layers. The architecture of the model is depicted in Figure 2.

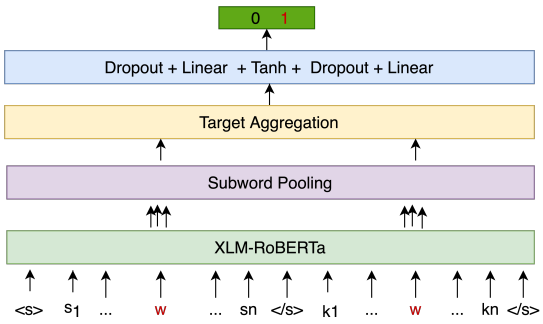

Figure 2: BC-based model architecture

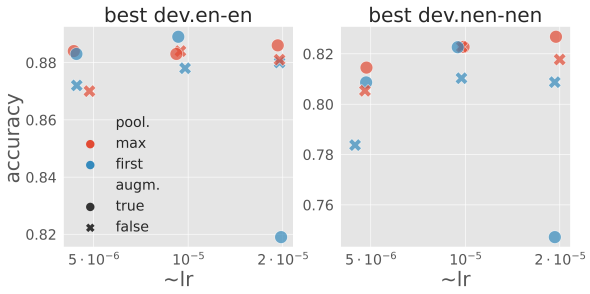

Figure 3: Data augmentation effect on xlmr.large performance

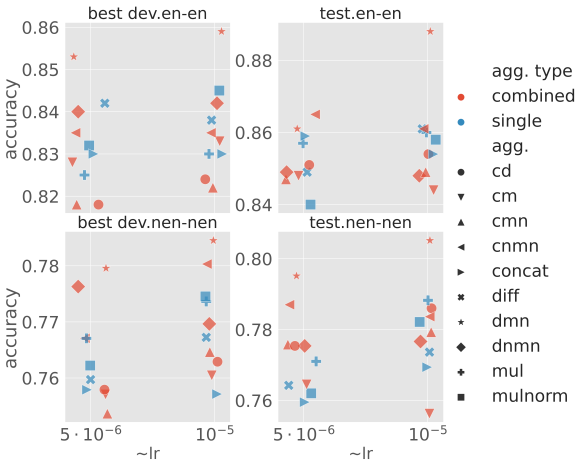

Figure 4: Comparison of target aggregation methods for xlmr.base (mean pooling, no batchnorm)

Figure 5: Comparison of subword pooling methods for xlmr.base (dmn agg. with batchnorm)

In the post-evaluation period we tried replacing concatenation (**concat**) with alternative aggregation techniques. First, we tried using component-wise difference (**diff**) or multiplication (**mul**) with an optional normalization of word occurrence vectors (**mulnorm**). Then we tried combining representations obtained with different aggregation techniques by concatenating them. We denote those combinations by letter sequences, where **c** stands for the concatenation of the first and the second vector, **d** for their difference and **m** for their component-wise product. The inputs to each of those operations can be optionally normalized, which is denoted by **n** after the corresponding operation. For instance, **dmn** means that we concatenate the difference of non-normalized and the product of normalized vectors. Also in the post-evaluation, we found it beneficial to apply batch normalization before feeding aggregated representations into the classification head.

During training, we applied 2-class softmax and optimized the cross-entropy loss. We fine-tuned BC models using almost the same settings as for QA models. Here, we used the constant learning rate of 1e-5 with linear warmup during the first 10% of training steps. During post-evaluation, we added linear learning rate decay.

Our submitted BC systems are ensembles of these three models: first, first-AG and max-AG differing by subwords pooling strategy and by use of data augmentation. We would be referring to the ensemble of these models validated on the English dev set as ENS-EN and as ENS-XX for an ensemble of models validated on corresponding dev sets.

4 Experiments and Results

As our submissions showed us that BC models perform much better than QA models, in our post-evaluation experiments we focused on them. In the following experiments, we report the best accuracy obtained during training on the English development set (*best dev.en-en*) and the best average accuracy on all non-English development sets (*best dev.nen-nen*) from our own split. For those epochs where the best dev set accuracy is achieved,

we report accuracy on the official English test set (*test.en-en*) and averaged over official non-English test sets (*test.nen-nen*). Due to space limitations, we report the results on all test sets only for our best models in Table 3.

In the first experiment, we trained the submitted models with and without data augmentation. Figure 3 shows that the augmentation never decreases performance, at least if the learning rate is selected properly. Thus, we always performed data augmentation in the following experiments.

Figure 4 compares target aggregation techniques. The concatenation of the difference of non-normalized vectors and component-wise product of normalized vectors (**dmn**) proved to outperform all other methods by a large margin, especially when the learning rate is properly selected. Thus, we used this technique for the following experiments. A simple concatenation of target embeddings, which was used in our submissions, is more than 3% worse and is often outperformed by the difference or component-wise product.

Since concatenating normalized and non-normalized vectors can make training difficult, we decided to apply batch normalization before feeding those vectors into the classification head. Figure 6 shows that batch normalization does not improve results for English much, but considerably improves the average performance on other languages. This is probably due to the fact, that the overwhelming majority of training examples are in English.

Also, from figure 6 we notice that for English different poolings give similar performance. For other languages, first pooling is a bad option. We hypothesise that this results from non-English words being split into sub-word tokens more frequently. Mean pooling consistently outperforms other poolings. Figure 5 additionally compares combinations of different subword poolings. However, those combinations did not improve results compared to single mean pooling.

Finally, we estimated how much the additional multilingual training data help compared to using only English training examples and counting on cross-lingual zero-shot transfer. In table 3 we denote xlmr.base models fine-tuned only on English train and trial data as **enonly**. We see that including non-English examples into the training set improves the results by 1.5-3% for multilingual and even more for cross-lingual scenarios. Surprisingly, it also gives some improvement for English.

Table 3 summarizes the results of our submitted systems and the following post-evaluation experiments. During the evaluation period, we made a total of 4 submission attempts, two for the Question Answering based approach and two for the binary classifier approach. During training the best checkpoint was selected either individually for each language using corresponding dev set accuracy (XX), or by English dev set accuracy (EN). We see that the first approach to the task (MTL-EN, MTL-XX) shows much worse results compared to the second one (ENS-EN, ENS-XX). For the second approach, we submitted two ensembles consisting of three models shown in the same Table 3. As expected, ensembling the models helped to improve the results greatly.

As we figured out that **dmn** target aggregation and **mean** subword pooling performs significantly better compared to other variants for XLMR.base model, we trained XLMR.large version with hyperparameters from the best XLMR.base model. The results of the models validated either by score on English dev set (EN), or by the average score for non-English dev sets (nEN), or by scores on each dev set individually (XX), are shown in the third group of results in the Table3. We see that these models outperform any single model from the evaluation phase for all multilingual subtask's test sets and test.en-ar set from cross-lingual subtask.

And lastly, we report the results for an ensemble of three XLMR.large models: two mean-dmn models trained with learning rates $1e-5$ and $2e-5$ and one mean-cnmn model trained with learning rate equal to $1e-5$. We see that using ensemble of models with new subword pooling and target aggregation techniques helps us to improve our official results from competition. We improved our results for test.ru-ru ($3 \rightarrow 2$), test.fr-fr ($2 \rightarrow 1$), test.en-en ($15 \rightarrow 12$), test.zh-zh ($21 \rightarrow 17$), test.en-ar ($6 \rightarrow 4$) and test.en-zh ($17 \rightarrow 12$) sets.

5 Conclusion

In this paper, we have described our approach to SemEval-2021 Task 2. We tried treating Word-in-Context Disambiguation as question answering and binary classification problems. In our case, binary classification turned out to be a more promising approach. Also, we found that mean pooling over subwords is the best option, batch normalization helps when added before classification head, and concate-

Model	ar-ar	ru-ru	fr-fr	en-en	zh-zh	en-ar	en-ru	en-fr	en-zh
our submissions: question answering based									
MTL-EN	73.9	75.0	77.9	84.7	77.7	63.1	66.8	69.8	69.3
MTL-XX	77.0	76.2	73.9	84.7	76.5	–	–	–	–
our submissions: binary classifier based									
ENS-EN	84.6(2)	85.3(10)	86.4(3)	91.1(15)	83.9(21)	86.5(6)	87.0(8)	87.2(6)	86.0(17)
ENS-XX	83.8(8)	86.6(3)	86.3(4)	91.1(15)	83.5(22)	–	–	–	–
first-concat-EN	83.5	84.2	84.8	90.0	82.2	85.4	86.4	86.4	84.4
first-concat-XX	82.0	84.6	84.8	90.0	82.1	–	–	–	–
first-concat-noAG-EN	82.3	82.5	85.4	90.8	81.4	84.9	85.7	86.1	85.8
first-concat-noAG-XX	82.9	84.0	84.9	90.8	80.9	–	–	–	–
max-concat-EN	83.2	85.8	84.1	89.8	84.5	85.7	82.6	84.2	81.9
max-concat-XX	83.6	83.3	84.7	89.8	84.9	–	–	–	–
post-evaluation results: xlmr.large, mean-dmn,lr=1e-5									
XX	84.0	86.1	84.5	90.7	83.5	–	–	–	–
EN	83.6	86.4	85.8	90.7	84.7	86.3	85.4	85.5	85.5
nEN	84.2	84.7	85.2	89.9	84.3	84.7	84.2	83.9	83.7
post-evaluation results: xlmr.large, mean-dmn,lr=2e-5 + mean-cnmn,lr=2e-5 + mean-dmn,lr=1e-5									
ENS-EN	84.6(2)	87.0(2)	87.5(1)	91.4(12)	84.8(17)	87.6(4)	86.2(12)	86.2(7)	87.1(12)
post-evaluation results: xlmr.base, mean-dmn									
XX	83.3	80.7	82.8	88.8	81.3	–	–	–	–
enonly-XX	80.5	79.4	80.7	87.1	80.1	–	–	–	–
EN	78.1	80.9	82.8	88.8	81.9	78.8	82.2	82.1	83.6
enonly-EN	81.9	78.6	80.7	87.1	79.6	75.5	79.5	76.7	73.4
nEN	82.1	80.7	83.4	89.1	81.3	80.7	82.4	82.7	79.9
enonly-nEN	81.3	79.4	81.4	87.8	81.0	74.3	77.9	75.9	72.5

Table 3: Results on all cross-lingual and monolingual test sets. XX denotes models validated on corresponding dev sets. For instance, XX model's result for ru-ru set was obtained by model validated on dev.ru-ru set. nEN denotes models validated by averaged scores for non-English dev sets and EN denotes the ones validated on the English dev set. During the evaluation period we submitted MTL-EN, MTL-XX, ENS-EN and ENS-XX models' predictions.

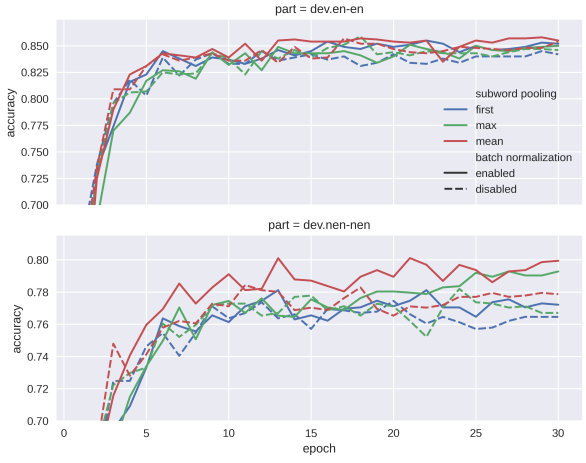

Figure 6: Effect of batch normalization on xlmr.base

nation of target embeddings is outperformed by the combination of the difference and the product of normalized embeddings.

References

Eneko Agirre, Oier Lopez de Lacalle, and Aitor Soroa. 2014. Random walks for knowledge-based word sense disambiguation. *Computational Linguistics*, 40(1):57–84.

Amir DN Cohen, Shachar Rosenman, and Yoav Goldberg. 2020. Relation Extraction as Two-way Span-Prediction. *arXiv preprint arXiv:2010.04829*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020a. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau, Shijie Wu, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. 2020b. Emerging cross-lingual structure in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6022–6034, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual Language Model Pretraining. *Advances in Neural Information Processing Systems (NeurIPS)*.

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. *arXiv preprint arXiv:1907.11692*.

Federico Martelli, Najla Kalach, G. Tola, and Roberto Navigli. 2021. SemEval 2021 Task 2: Multilingual and Cross-lingual Word-in-Context Disambiguation (MCL-WiC). In *Proceedings of the Fifteenth Workshop on Semantic Evaluation (SemEval-2021)*, Bangkok, Thailand (online). Association for Computational Linguistics.

Marco Maru, Federico Scozzafava, Federico Martelli, and Roberto Navigli. 2019. SyntagNet: Challenging supervised word sense disambiguation with lexical-semantic combinations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3534–3540, Hong Kong, China. Association for Computational Linguistics.

George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. 1990. Introduction to WordNet: An on-line lexical database. *International journal of lexicography*, 3(4):235–244.

George A Miller, Claudia Leacock, Randee Tengi, and Ross T Bunker. 1993. A semantic concordance. In *HUMAN LANGUAGE TECHNOLOGY: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993*.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):1–69.

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.

Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. 2020a. SensEmBERT: Context-enhanced sense embeddings for multilingual word sense disambiguation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8758–8765.

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*. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. HuggingFace's Transformers: State-of-the-Art Natural Language Processing. *CoRR*, abs/1910.03771.