Exemplification Modeling: Can You Give Me an Example, Please?

Edoardo Barba[1] , **Luigi Procopio**[1] , **Caterina Lacerra**[1] , **Tommaso Pasini**[2,*] and **Roberto Navigli**[1]

[1]Sapienza NLP Group, Department of Computer Science, Sapienza University of Rome
[2]Department of Computer Science, University of Copenhagen

{edoardo.barba, luigi.procopio, caterina.lacerra, roberto.navigli}@uniroma1.it
tommaso.pasini@di.ku.dk

Abstract

Recently, generative approaches have been used effectively to provide definitions of words in their context. However, the opposite, i.e., generating a usage example given one or more words along with their definitions, has not yet been investigated. In this work, we introduce the novel task of Exemplification Modeling (EXMOD), along with a sequence-to-sequence architecture and a training procedure for it. Starting from a set of (word, definition) pairs, our approach is capable of automatically generating high-quality sentences which express the requested semantics. As a result, we can drive the creation of sense-tagged data which cover the full range of meanings in any inventory of interest, and their interactions within sentences. Human annotators agree that the sentences generated are as fluent and semantically-coherent with the input definitions as the sentences in manually-annotated corpora. Indeed, when employed as training data for Word Sense Disambiguation, our examples enable the current state of the art to be outperformed, and higher results to be achieved than when using gold-standard datasets only. We release the pretrained model, the datasets and the software at https://github.com/SapienzaNLP/exmod.

1 Introduction

Providing the sense of a word given its context is a major topic in lexical semantics that has drawn considerable attention [Blevins and Zettlemoyer, 2020; Bevilacqua and Navigli, 2020; Barba *et al.*, 2021]. Through the years, this task has mainly been formulated in two flavors: Word Sense Disambiguation [Bevilacqua *et al.*, 2021, WSD], where the sense has to be chosen from a predefined sense inventory, and, more recently, Definition Modeling, where the sense definition for a word in context is automatically generated [Bevilacqua *et al.*, 2020]. In particular, Definition Modeling drops the need for enumerative sense inventories and leverages instead the power of auto-regressive pretrained models to generate glosses, i.e., definitions, for arbitrary words and expressions. We claim that the reverse process, i.e., providing examples for a sense starting from its definition, has advantageous applications too, both for humans and automatic systems. Indeed, on the one hand, using examples to further describe word meanings can be beneficial for second language learners [Nation, 2001]. On the other hand, generated examples can be used to enrich pre-existing knowledge bases and as a data augmentation technique for Word Sense Disambiguation systems.

In this paper, we introduce the Exemplification Modeling (EXMOD) task, which aims at generating example sentences starting from one or multiple words with their sense definitions, together with a sequence-to-sequence architecture for the task, and a procedure for training it from sense-annotated corpora. We show that our model can generalize not only over unseen words and definitions, but also across different lexical resources with diverse sense granularities. Furthermore, the proposed architecture can provide examples for up to three senses at the same time, paving the way to new scenarios on how a generative system can model the interaction among multiple senses. We evaluate the produced sentences quantitatively by employing them as additional training examples for the WSD task. The examples generated lead WSD models to perform better than when relying on manually-annotated datasets only, and to attain results higher than the current state of the art. Furthermore, we analyze the sentences generated by means of a human-annotation task, and show that annotators perceive the examples produced as fluent and semantically coherent as those in a manually-curated corpus. This work brings the following novel contributions:

- The Exemplification Modeling (EXMOD) problem, i.e., an innovative task requiring systems to generate a usage example given one or more words with their sense definitions.
- A sequence-to-sequence architecture for EXMOD that can be trained directly on already-existing lexical-semantic resources.
- An effective procedure for creating large-scale and high-quality training corpora for English WSD covering all senses of the reference inventory.
- An *in vivo* evaluation framework taking into account different sense inventories and gold training corpora, as well as a human-evaluation task for measuring the fluency and semantic coherence of the generated sentences.

[*]Work carried out while at the Sapienza University of Rome.

2 Methodology

In this Section, we introduce and formalize the task of Exemplification Modeling (Section 2.1), describe our autoregressive approach for it (Section 2.2) and present the sampling strategy over the training instances (Section 2.3).

2.1 Exemplification Modeling

Given a set of lemmas along with their definitions, we frame Exemplification Modeling as the task of generating a sentence where each input lemma is used with its intended sense.[1] For example, consider as input the lemma-definition pair `(bank, a building where financial services are offered)`, a possible correct output might be: `He went to the bank to deposit a check`. Formally, let D be the set of lemma-definition pairs $(l_1, d_1), \ldots, (l_k, d_k)$, with each definition d_i specifying the intended meaning of lemma l_i; both l_i and d_i are sequences of tokens[2] and we use l_i^j and d_i^j to denote their respective j-th token. The task can thus be formulated as follows: given the set D of lemma-definition pairs, a model has to yield a meaningful and semantically-coherent token sequence $s = s_1, \ldots, s_n$, such that, $\forall (l_i, d_i) \in D$, l_i occurs in any of its inflected forms in s with the meaning defined by its corresponding definition d_i. Furthermore, as knowing precisely where each l_i occurs in s represents a natural desideratum, we include as part of the task also yielding the set of indices $\phi_i = j_i^1, \ldots, j_i^{|\phi_i|}$ $(1 \leq j_i^1 < \cdots < j_i^{|\phi_i|} \leq n)$ where l_i is expressed in s.

2.2 Model

To tackle the EXMOD task, we propose a two-stage approach, which first generates a usage example s where all the input lemmas occur with their expected meanings, and then computes $\phi_i \, \forall (l_i, d_i) \in D$ by means of a post-processing strategy. For the first stage, we use a sequence-to-sequence model and define its input m as follows:

$$\begin{aligned} m = & <s> l_1^1 \ \ldots \ l_1^{|l_1|} </s> d_1^1 \ \ldots \ d_1^{|d_1|} \\ & \ldots \\ & <s> l_k^1 \ \ldots \ l_k^{|l_k|} </s> d_k^1 \ \ldots d_k^{|d_k|} \end{aligned}$$

where $<s>$ and $</s>$ are two special tokens around which each input lemma is wrapped. The target example s is similarly encoded into a sequence $\hat{s}$, where, $\forall (l_i, d_i) \in D$, all contiguous spans in ϕ_i are surrounded by the special markers $<t>$ and $</t>$. For instance, given the above usage example `He went to the bank to deposit a check`, we convert it into `He went to the <t> bank </t> to deposit a check`. With this encoding, we then train the sequence-to-sequence model to learn the factorized probability:

$$p(\hat{s}|m) = \prod_{j=2}^{|\hat{s}|} p(\hat{s}_j | \hat{s}_{1:j-1}, m)$$

by minimizing the cross-entropy loss with respect to $\hat{s}$.

However, as this formulation does not compute $\phi_i \, \forall (l_i, d_i) \in D$, we perform the second stage of our approach and, denoting with o the sequence generated by our model, we apply the following four steps:

- We lemmatize each token in o;
- We pair each highlighted span, tagged with lemma l, to the pair (l_i, d_i) such that $l_i = l$;[3]
- We compute ϕ_i by considering the indices of the span aligned to each input pair (l_i, d_i).[4]
- We remove the highlighting markers from the sequence.

After these four steps, the system produces a usage example s for the input pairs in D, along with a set of indices ϕ_i indicating where the i-th input pair is expressed within s.

2.3 Sampling

We now present the method we use for sampling the training instances from a given collection $B = (D_1, s_1), \ldots, (D_{|B|}, s_{|B|})$ of inputs and expected outputs. In this work, we limit ourselves to consider only $k = |D_i| \in \{1, 2\}$ for $i = 1, \ldots, |B|$ and defer exploring training strategies with $k \geq 3$ for future work. We believe that these two values of k are significantly different from each other, as sense interactions enter the picture when dealing with $k = 2$, and we further argue that their modeling calls for different approaches. For $k = 1$, we assume all instances in B be equally adequate and define a uniform distribution over this list, sampling (D_i, s_i) with probability:

$$p(D_i, s_i) = \frac{1}{|B|}$$

For $k = 2$, instead, we focus on usage examples where the senses in D_i appear as lexical collocations[5] of one another and, thus, employ a $p(D_i, s_i)$ that fosters this choice. That is, for each input $D_i = \{(l_{i,1}, d_{i,1}), (l_{i,2}, d_{i,2})\}$ and its target sentence s_i, we set its probability $p(D_i, s_i)$ to 0 if the senses in D_i appear more than ξ tokens away from each other in s_i. For all the remaining instances $\hat{B}$, instead, we compute their probability based on the *Positive Pointwise Mutual Information*[6] (PPMI, Niwa and Nitta [1994]) measure:

$$p(D_i, s_i) = (1 - \alpha) \frac{ppmi(D_i)}{\sum_{j=1}^{|\hat{B}|} ppmi(D_j)} + \alpha \frac{1}{\sigma(s_i)|S|} \quad (1)$$

where $\sigma(s_i)$ is the number of (D_j, s_j) pairs such that $s_j = s_i$, S is the set of all the usage examples, and $ppmi$ is the function defined as:

$$ppmi(D_i) = max\left(0, log \frac{p(l_{i,1}, l_{i,2})}{p(l_{i,1})p(l_{i,2})}\right)$$

[1] Lemma-definition pairs identify a sense for the lemma, therefore, we use "lemma-definition pair" and "sense" interchangeably.

[2] We model lemmas through lists of tokens to support multi-words.

[3] If multiple pairs match l, we assign it to one at random.

[4] If no input pair is associated with any span, we discard the example.

[5] Pairs of words that co-occur more frequently than chance.

[6] We estimate the probability of the occurrence of a given word by counting its occurrences in the English Wikipedia corpus (December 2019 dump). Further details at https://github.com/SapienzaNLP/exmod (D).

The second term of Equation 1 is effectively a smoothing factor[7] that distributes a uniform probability over each usage example s_i, dividing it equally among all the training instances in which it occurs. With this strategy, we effectively foster the sampling of (D_i, s_i) pairs such that:

- $(l_{i,1}, d_{i,1})$ and $(l_{i,2}, d_{i,2})$ co-occur nearby;
- the co-occurrence is statistically significant.

Therefore, we encourage the generation of usage examples where the input lemmas actively interact between each other.

3 In Vivo Evaluation

In this Section, we put forward an *in vivo* evaluation suite for the Exemplification Modeling task. To this end, we focus on the Word Sense Disambiguation (WSD) problem, which, given a word in context, aims at selecting its most suitable meaning, and design 5 sub-tasks to evaluate different aspects of the examples generated by an EXMOD model. Across all sub-tasks, we make use of the following lexical resources:

- **WordNet** [Miller *et al.*, 1990], an electronic dictionary where textual definitions, i.e., glosses, are assigned to *synsets*, lexical units denoting a group of synonymous senses, each associated with a different lemma.
- **SyntagNet** [Maru *et al.*, 2019], a manually-curated resource of semantic collocations, i.e., pairs of WordNet senses that appear frequently together in texts.
- **SemCor** [Miller *et al.*, 1993], a large sense-annotated corpus for WSD, comprising 33K sentences for 200K instances tagged with WordNet senses.
- **Princeton WordNet Gloss Corpus (WNG)**[8], a corpus consisting of tagged examples and definitions in WordNet. We keep only the examples and the resulting dataset features 34,275 instances.[9]
- **The Oxford Dictionary Dataset** [Chang *et al.*, 2018], containing 555,695 sentences for as many instances tagged with senses from the Oxford Dictionary of English (ODE).

3.1 Tasks

For each task, we define the data that the EXMOD system will use during the training and generation phases. We also report the cardinality k of the input set of lemma-definition pairs and the sense inventory from which the input pairs have to be drawn. In what follows, we delineate the five WSD sub-tasks:

Task 1 In this sub-task, we test the capabilities of EXMOD models to generate data in the simplest scenario, that is, producing examples for a single sense at a time from the same sense inventory that was used during the training phase.
Training: We set $k = 1$, therefore serving only (D, s) pairs with $|D| = 1$, and draw the training pairs from the concatenation of SemCor and WNG.
Generation: We query the model with all lemma-definition pairs contained in WordNet and build the WSD dataset from the resulting examples.

Task 2 This task is analogous to the previous one as far as k is concerned, but differs in the inventories used at training time. This setting, indeed, aims at assessing the capability of a model to perform zero-shot generations for senses coming from a previously unseen inventory.
Training: We set $k = 1$ and select the training instances from the Oxford Dictionary Dataset.
Generation: We query the model analogously to Task 1 and build the WSD dataset from the resulting examples.

Task 3 The third task challenges EXMOD models to generate examples that are coherent with two input senses.
Training: We vary $k \in \{1, 2\}$, therefore serving training pairs (D, s) with $|D| \leq 2$. To build the set of training instances when $k = 1$, we consider all WordNet senses appearing in the concatenation of SemCor and WNG. As for $k = 2$, we collect all the sense pairs appearing in a SemCor sentence and discard all those whose senses are further away than a window of $\xi = 10$ tokens. We recall from Section 2.3 that we sample each pair according to Equation 1.
Generation: We query the model with all the possible pairs of senses ($k = 2$) in SemCor appearing within a window of size 10, and all sense pairs of SyntagNet.

Task 4 This task evaluates the generalization ability of an EXMOD model to scale from inputs with $k \leq 2$ used at training time, to inputs with $k = 3$ used at generation time.
Training: We keep the same training setting as Task 3.
Generation: We query the model to generate examples for triplets of senses ($k = 3$) that we compute by enumerating all sense triangles in SyntagNet, i.e., sets of three senses, each in collocation with the other two.

Task 5 The last task aims at pushing an EXMOD model to its limit by considering all the possible resources at training and generation time.
Training: We vary $k \in \{1, 2\}$. For $k = 1$, we extract the training instances by considering all WordNet senses appearing in SemCor or WNG and all ODE senses in the Oxford Dictionary Dataset. For $k = 2$, we use the setting of Task 3.
Generation: We query the model varying $k \in \{1, 2, 3\}$ and draw i) senses from WordNet, ii) pairs from SemCor and SyntagNet, and iii) triplets from SyntagNet triangles.

3.2 Setup

In this Section, we detail the EXMOD and WSD models used for the experiments, as well as their training hyperparameters, the generation strategy and the evaluation data. Finally, we also propose a metric for model selection that takes into account the semantic coherence of the sentence produced with respect to the input senses.

EXMOD Model and Training. We use BART [Lewis *et al.*, 2020] as the underlying sequence-to-sequence architecture of our approach; henceforth, we refer to our model as EXMAKER. We train EXMAKER with RAdam [Liu *et al.*, 2020] for 300,000 training steps, learning rate set to $1e^{-5}$ and batches of 800 tokens, accumulating gradient for 10 steps. When dealing with multiple datasets, we perform batch sampling from each of them.

[7] We use $\alpha = 0.15$.

[8] https://wordnetcode.princeton.edu/glosstag.shtml

[9] Please refer to https://github.com/SapienzaNLP/exmod (A).

EXMOD Model Selection. Choosing the set of weights of EXMAKER that yields the best generations is not a trivial task. Indeed, as multiple usage examples may be perfectly adequate for a given set D, our generative approach is an instance of open-ended text generation where referenced metrics, such as the cross-entropy loss over a reserved sample of data, have been shown to be unsuitable [Liu *et al.*, 2016]. To overcome this issue, we propose an unreferenced metric for measuring the semantic coherence of the example generated with respect to its input set D. Specifically, we take advantage of ARES [Scarlini *et al.*, 2020], i.e., latent representations of WordNet senses that lie in a space comparable to that of BERT Large [Devlin *et al.*, 2019]. Given a generated sentence and a target word therein, we extract the target word embedding by means of BERT Large and compute its cosine similarity with the ARES embedding of the input sense. Thus, to perform model selection during training, we define two sets of validation instances for $k = 1$ and $k = 2$, with 300 and 580 samples, respectively,[10] and calculate this similarity for each pair in each input set D, aggregating them via a macro average. We compute the ARES-score every 2000 training steps and select the model with the highest performance in terms of macro average.

Generation. In each subtask, we generate from 1 to 6 examples given a set of lemma-definition pairs D by applying the following decoding strategies in parallel on EXMAKER:

- *beam-n1*: beam search returning the best beam;
- *sample-n5*: nucleus-sampling with $p = 0.9$ returning up to 5 sequences.

We group the outputs and clean them by discarding all the ill-formed generations, that is, we discard all those examples generated from D such that $\forall(l_i, d_i) \in D$, $\phi_i = \emptyset$. Finally, using ϕ_i, we tag the highlighted tokens with the sense in the reference inventory identified by the (l_i, d_i) pair.

WSD Reference Model. As reference WSD system, we choose a simple yet effective Transformer-based solution, i.e., BERT for token classification. Following Devlin *et al.* [2019], we represent each token[11] through the concatenation of the last four layers of BERT, and apply a linear classification head to map each token to a sense. We train the model for at most 50 epochs with early stopping on the validation accuracy and patience set to 3 epochs.[12] As training data, we use the concatenation of SemCor and WNG together with the silver data generated for each specific experiment.

Comparison Systems. As comparison, we report the results of the reference system trained on SemCor and WNG only, as well as on their concatenation with two other automatically-produced datasets, i.e., OneSeC [Scarlini *et al.*, 2019], which relies on information within a knowledge base to tag Wikipedia sentences, and OMSTI [Taghipour and Ng, 2015], a semi-automatic approach relying on parallel corpora. To put our results in context with the state of the art, we also report the result of the currently best-performing system, i.e., ESCHER [Barba *et al.*, 2021]. We report its results when trained on i) SemCor, ii) SemCor, WNG and the Oxford Dictionary Dataset, and iii) SemCor, WNG and EXMAKER data.

WSD Evaluation Data. As standard in WSD, we use the evaluation framework made available by Raganato *et al.* [2017]. The framework contains six test datasets, namely, Senseval-2 [Palmer *et al.*, 2001], Senseval-3 [Snyder and Palmer, 2004], SemEval-07 [Pradhan *et al.*, 2007], SemEval-13 [Navigli *et al.*, 2013] and SemEval-15 [Moro and Navigli, 2015]; and ALL, the concatenation of all the aforementioned datasets. To examine the models' ability to generalize over rare and unseen senses, we partition the ALL dataset into the three following splits:

- **MFS**, containing all instances tagged with the Most Frequent Sense (MFS) in SemCor and WNG for a specific lemma.
- **LFS**, containing all the instances that are tagged with a Least Frequent Sense, i.e., a sense that is not the MFS.
- **Unseen**, containing all the instances tagged with a sense that never appears either in SemCor or in WNG.

3.3 Results

In Table 1, we report the results of our reference model in each sub-task, as well as the various baselines.

We first compare the reference model when trained on SemCor with the WNG Examples only and when trained with additional data for the WordNet inventory coming from OMSTI, OneSeC and EXMAKER (Task 1). As one can see, the data generated by EXMAKER boosts BERT performance by 1.3 points, a result that none of the other additional datasets achieves. More interestingly, the improvement comes entirely from senses that are underrepresented in the training set (LFS and Unseen columns). These results highlight the unique ability of EXMAKER to produce examples for senses that are either rarely or never seen in the gold datasets, while confirming that a model trained for the EXMOD task can be employed effectively to generate examples for WordNet senses.

By considering the results in the second task (Task 2), we can state with confidence that EXMAKER, while being trained with definitions from a single inventory (ODE), can provide examples for senses in another inventory (WordNet) that has different sense granularity. Furthermore, training on instances from EXMAKER in Task 2 results in higher WSD performance than when training on the sentences generated by EXMAKER in Task 1. This result paves the way for new scenarios where EXMOD models are used to complete existing knowledge bases, e.g., WordNet, which lacks examples for roughly 85% of its senses.[13]

We now focus on results for Task 3 and 4, which test the ability of EXMAKER to create coherent examples for pairs or triplets of senses when trained on single definitions or pairs of definitions. Results show that the sentences generated with $k = 2$ are beneficial to the WSD model and further boost its

[10] Please refer to https://github.com/SapienzaNLP/exmod (B.1) for more details on their creation.

[11] We use the first sub-word embedding when the token is split.

[12] Please refer to https://github.com/SapienzaNLP/exmod (C) for all hyperparameters.

[13] 207,016 in total.

	Task	WSD Setting: Model	Additional Dataset	k	Dev Set: SE7	Test Sets: MFS	LFS	Unseen	ALL
Baseline	—	BERT	—	—	69.6	95.1	42.5	53.43	74.4
	—	BERT	OMSTI	—	68.4	93.7	45.5	53.5	74.5
	—	BERT	OneSec	—	67.3	95.3	39.5	54.0	73.9
	—	ESCHER†	—	—	76.3	93.7	55.7	75.0	80.7
	—	ESCHER	ODE	—	77.9	94.6	56.4	76.8	81.6
Ours	Task 1)	BERT	EXMAKER	1	69.8	93.2	48.1	58.9	75.7
	Task 2)	BERT	EXMAKER	1	70.1	93.3	48.3	59.1	76.0
	Task 3)	BERT	EXMAKER	2	69.9	93.0	48.4	59.8	76.2
	Task 4)	BERT	EXMAKER	3	70.0	93.1	48.0	60.3	76.1
	Task 5)	BERT	EXMAKER	1, 2, 3	71.2	95.7	47.3	57.8	76.8
	Task 5)	ESCHER	EXMAKER	1, 2, 3	78.1	94.8	58.6	77.1	**82.3**

Table 1: F1 attained when training the reference model, BERT, on the gold training set (the concatenation of SemCor and WNG) concatenated with several silver datasets generated by different configurations of EXMAKER and its competitors, i.e., OMSTI and OneSeC. † indicates that the model is trained on SemCor instances only. Underlined scores are statistically significant with respect to ESCHER results with $p < 0.05$.

performance from 75.7 (Task 1) to 76.2 (Task 3). We hypothesize that this result is a consequence of the more complete semantics provided to EXMAKER, enabling it to generate sentences where the senses occur with a sharper connotation, thus providing, in turn, a clearer context to a WSD model. Considering the dataset generated with $k = 3$ (Task 4), the WSD model performance is comparable to that attained when trained on a dataset generated with $k = 2$ (Task 3). As EXMAKER is not trained to generate examples for triplets of word-definitions pairs, this surprising result highlights its capability to produce coherent sentences even for sets of senses larger than those seen during training.

Finally, we discuss the last task, where we aim at measuring the maximum performance that a WSD model can reach when using data from EXMAKER, i.e., when trained on all resources and used for generating sentences with $k \in \{1, 2, 3\}$. In this setting we also train the state-of-the-art model for WSD (ESCHER) on the concatenation of SemCor, WNG and EXMAKER data. Thanks to EXMAKER data, ESCHER reaches an unprecedented result of 82.3 points of F1, performing better than when using manually-annotated data only (SemCor, WNG Examples and the Oxford Dictionary Dataset), and surpassing the current state of the art (80.7) by 1.6 points. We also note that BERT performance gains 0.5 and 0.6 points on ALL with respect to Tasks 3 and 4, resulting in an overall improvement of 1.1 points over Task 1.

Through these quantitative experiments, we show that EXMAKER produces high-quality sentences even when exemplifying rare and unseen senses, fully taking advantage of different resources and sense inventories. Furthermore, the datasets generated prove to be useful as additional resources for WSD models, leading them to set a new state of the art.

4 Qualitative Analysis

We now focus on qualitatively evaluating the EXMAKER examples by means of two annotation tasks. We ask annotators to manually assign two scores to each example generated: one measuring its fluency and one measuring its semantic coherence with respect to a given lemma-definition pair.

	ALL		EXMAKER	
Task	AVG	κ	AVG	κ
Fluency	4.82	0.74	4.63	0.70
Coherence	4.83	0.69	4.82	0.68

Table 2: Results for the two tasks of qualitative analysis. We report the Likert scores average and the pairwise average of Cohen's κ.

Data to Annotate. We build two datasets: one automatically generated by EXMAKER trained as in Task 5, and one drawn from the ALL dataset. We consider a statistically significant sample[14] of 284 and 382 instances of the ALL and EXMAKER datasets, respectively, shuffle them together, and provide the anonymized examples to three English-proficient annotators.[15]

Annotation Task. We provide annotators with guidelines[16] and ask them to tag each example with two scores: the first, to evaluate its fluency and indicate whether it is logical and grammatically correct; the second, to measure to what extent the usage of the target word in the example reflects the meaning described by the given definition. Following Bevilacqua *et al.* [2020], in both cases, we ask the annotators to fill a five-level Likert scale that assigns higher scores to better sentences. We report the average computed across all instances and annotators for each measure and Cohen's κ [Cohen, 1960] as metric for the Inter-Annotator Agreement.

Results. As one can see in Table 2, the average Likert scores for EXMAKER data are 4.63 and 4.82 for fluency and semantic coherence, respectively. These results are nearly

[14] Samples are statistically significant w.r.t. the sizes of the source datasets with confidence level of 95% and a margin error of ±5.

[15] Annotators do not know the source of each sentence.

[16] Released at https://github.com/SapienzaNLP/exmod (E).

identical to those attained for the manually-annotated examples of ALL. The inter-annotator agreement is substantial [Landis and Koch, 1977] for both datasets and measures. These results underline the high quality of the sentences produced by EXMAKER, which not only have a fluency close to that of human-curated corpora, but are also capable of conveying the required input meaning into valid examples that turn out to be only slightly worse on average than those created by humans.

5 Related Work

Definition Modeling The spread of generative models is fostering the advancement of research in several topics, among which definition modeling [Noraset *et al.*, 2017] is the closest to ours. The task requires a gloss to be produced for a word, and was initially proposed as an interpretable way for analyzing the semantics of word embeddings. Early approaches to the task [Gadetsky *et al.*, 2018; Chang *et al.*, 2018] relied mainly on static word embeddings to model the input context and generate the most suitable definition for the target word. Static word embeddings were then replaced by contextualized embeddings, better modeling the semantics of a target word in context. Nevertheless, these approaches were neither able to provide definitions for multiword expressions [Mickus *et al.*, 2019], nor to take into account the word order when defining multiword expressions [Ishiwatari *et al.*, 2019]. The most recent effort in this direction is Generationary [Bevilacqua *et al.*, 2020], which exploits a sequence-to-sequence generative approach where the spans to be defined are explicitly marked. This approach closed the gap between Definition Modeling and Word Sense Disambiguation, showing that the glosses generated could easily be mapped to those in a lexical knowledge base, hence making it possible to link the target span to a sense in a dictionary.

Word Sense Disambiguation Differently from Definition Modeling, Word Sense Disambiguation is a long-standing task in NLP [Navigli, 2009], which aims at assigning each content word in a text to its most suitable meaning drawn from a sense inventory. Approaches to this task are either knowledge-based, such as graph and heuristic-driven algorithms [Maru *et al.*, 2019], or supervised, such as neural networks [Bevilacqua and Navigli, 2020; Blevins and Zettlemoyer, 2020]. Knowledge-based methods rely solely on lexical-semantic knowledge bases, and, in general, perform worse than their supervised counterparts. These latter, indeed, consistently attain state-of-the-art results thanks to their ability to learn from data. Nevertheless, their results are still limited by the lack of large-scale sense-annotated corpora: currently-used datasets cover less than 20% of the senses in WordNet, i.e., the *de facto* standard sense inventory of English.

Automatically-generated Data for WSD Since supervised approaches show better performance in general, several efforts have been put into creating sense-annotated corpora automatically. OMSTI [Taghipour and Ng, 2015] exploited human annotations of Chinese senses within a parallel corpus to automatically disambiguate their corresponding English sentences. MuLaN [Barba *et al.*, 2020] also employed human annotations, but focused on producing data in languages other than English by relying on a cross-lingual sentence retrieval step to project English annotations potentially to hundreds of languages. Conversely, OneSeC [Scarlini *et al.*, 2019] does not require manually-annotated data, but instead leverages the information within a knowledge base and in Wikipedia to produce sense-tagged data.

To the best of our knowledge, this work is the first to formulate the Exemplification Modeling (EXMOD) task, i.e., the task of generating example sentences given one or more sense definitions. EXMOD is similar in its generative nature to Definition Modeling, however, its goal is to provide usage examples of selected word meanings, rather than defining words in contexts. As a by product, the proposed architecture for the task (EXMAKER) can generate data to train models for the Word Sense Disambiguation problem. Despite not being the main goal of this work, our approach is novel in comparison to other methods for creating silver data. Indeed, EXMAKER generates examples *ex novo* and *on demand*, while all the other systems tag already-existing sentences.

6 Conclusions

In this work, we introduced the new task of Exemplification Modeling (EXMOD), aimed at generating a usage example for a given set of words with their definitions. We showed that the task can be tackled by means of an encoder-decoder architecture (EXMAKER) trained on already-available data for Word Sense Disambiguation (WSD). Human evaluation showed that the examples produced can be confused with those drawn from a manually-curated corpus as they are fluent and semantically-coherent with the input. Finally, we proposed an *in vivo* evaluation to measure the performance of systems for the EXMOD task automatically. This was based on the supervised WSD task, where the examples generated for the EXMOD task could be used to train a WSD model. Results show that the examples provided by EXMAKER lead WSD models to attain better performance than when using manually-tagged data only, while, at the same time, paving the way towards a full-fledged generative approach for data augmentation in Word Sense Disambiguation.

As future work we plan to expand the generation of examples on languages other than English and to enlarge the number of input senses that the architecture can handle. We release the software, all data and the annotation guidelines for the human-evaluation task at https://github.com/SapienzaNLP/exmod.

Acknowledgments

The authors gratefully acknowledge the support of the ERC Consolidator Grant MOUSSE No. 726487 under the European Union's Horizon 2020 research and innovation programme.

This work was partially supported by the MIUR under the grant "Dipartimenti di eccellenza 2018-2022" of the Department of Computer Science of the Sapienza University of Rome.

References

[Barba *et al.*, 2020] Edoardo Barba, Luigi Procopio, Niccolò Campolungo, Tommaso Pasini, and Roberto Navigli. MuLaN: Multilingual Label propagatioN for Word Sense Disambiguation. In *Proc. of IJCAI*, 2020.

[Barba *et al.*, 2021] Edoardo Barba, Tommaso Pasini, and Roberto Navigli. ESC: Redesigning WSD with extractive sense comprehension. In *Proc. of NAACL*, pages 4661–4672, 2021.

[Bevilacqua and Navigli, 2020] Michele Bevilacqua and Roberto Navigli. Breaking through the 80% glass ceiling: Raising the state of the art in word sense disambiguation by incorporating knowledge graph information. In *Proc. of ACL*, pages 2854–2864, 2020.

[Bevilacqua *et al.*, 2020] Michele Bevilacqua, Marco Maru, and Roberto Navigli. Generationary or: "how we went beyond word sense inventories and learned to gloss". In *Proc. of EMNLP*, pages 7207–7221, 2020.

[Bevilacqua *et al.*, 2021] Michele Bevilacqua, Tommaso Pasini, Alessandro Raganato, and Roberto Navigli. Recent trends in Word Sense Disambiguation: A survey. In *Proc. of IJCAI*, 2021.

[Blevins and Zettlemoyer, 2020] Terra Blevins and Luke Zettlemoyer. Moving down the long tail of word sense disambiguation with gloss informed bi-encoders. In *Proc. of ACL*, pages 1006–1017, 2020.

[Chang *et al.*, 2018] Ting-Yun Chang, Ta-Chung Chi, Shang-Chi Tsai, and Yun-Nung Chen. xSense: Learning sense-separated sparse representations and textual definitions for explainable word sense networks. *arXiv preprint arXiv:1809.03348*, 2018.

[Cohen, 1960] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

[Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL*, pages 4171–4186, 2019.

[Gadetsky *et al.*, 2018] Artyom Gadetsky, Ilya Yakubovskiy, and Dmitry Vetrov. Conditional generators of words definitions. In *Proc. of ACL*, pages 266–271, 2018.

[Ishiwatari *et al.*, 2019] Shonosuke Ishiwatari, Hiroaki Hayashi, Naoki Yoshinaga, Graham Neubig, Shoetsu Sato, Masashi Toyoda, and Masaru Kitsuregawa. Learning to describe unknown phrases with local and global contexts. In *Proc. of NAACL*, pages 3467–3476, 2019.

[Landis and Koch, 1977] J Richard Landis and Gary G Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.

[Lewis *et al.*, 2020] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proc. of ACL*, pages 7871–7880, 2020.

[Liu *et al.*, 2016] Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proc. of EMNLP*, pages 2122–2132, 2016.

[Liu *et al.*, 2020] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. In *ICLR*, 2020.

[Maru *et al.*, 2019] Marco Maru, Federico Scozzafava, Federico Martelli, and Roberto Navigli. SyntagNet: Challenging supervised word sense disambiguation with lexical-semantic combinations. In *Proc. of EMNLP*, pages 3534–3540, 2019.

[Mickus *et al.*, 2019] Timothee Mickus, Denis Paperno, and Matthieu Constant. Mark my word: A sequence-to-sequence approach to definition modeling. In *Proc. of DeepNLP*, 2019.

[Miller *et al.*, 1990] George A. Miller, R.T. Beckwith, Christiane D. Fellbaum, D. Gross, and K. Miller. Introduction to WordNet: an online lexical database. *International Journal of Lexicography*, 3(4):235–244, 1990.

[Miller *et al.*, 1993] George A Miller, Claudia Leacock, Randee Tengi, and Ross T Bunker. A semantic concordance. In *Proc. of Human Language Technology*, pages 303–308, 1993.

[Moro and Navigli, 2015] Andrea Moro and Roberto Navigli. SemEval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. In *Proc. of SemEval*, pages 288–297, 2015.

[Nation, 2001] Ian SP Nation. *Learning vocabulary in another language*. Cambridge University Press, 2001.

[Navigli *et al.*, 2013] Roberto Navigli, David Jurgens, and Daniele Vannella. SemEval-2013 task 12: Multilingual word sense disambiguation. In *Proc. of SemEval*, pages 222–231, 2013.

[Navigli, 2009] Roberto Navigli. Word Sense Disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):1–69, 2009.

[Niwa and Nitta, 1994] Yoshiki Niwa and Yoshihiko Nitta. Co-occurrence vectors from corpora vs. distance vectors from dictionaries. In *Proc. of COLING*, 1994.

[Noraset *et al.*, 2017] Thanapon Noraset, Chen Liang, Larry Birnbaum, and Doug Downey. Definition modeling: Learning to define word embeddings in natural language. In *Proc. of AAAI*, 2017.

[Palmer *et al.*, 2001] Martha Palmer, Christiane Fellbaum, Scott Cotton, Lauren Delfs, and Hoa Trang Dang. English tasks: All-words and verb lexical sample. In *Proc. of Senseval-2*, pages 21–24, 2001.

[Pradhan *et al.*, 2007] Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. SemEval-2007 task-17: English lexical sample, SRL and all words. In *Proc. of SemEval*, pages 87–92, 2007.

[Raganato *et al.*, 2017] Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. Word Sense Disambiguation: A unified evaluation framework and empirical comparison. In *Proc. of EACL*, pages 99–110, 2017.

[Scarlini *et al.*, 2019] Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. Just "OneSeC" for producing multilingual sense-annotated data. In *Proc. of ACL*, pages 699–709, 2019.

[Scarlini *et al.*, 2020] Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. With more contexts comes better performance: Contextualized sense embeddings for all-round word sense disambiguation. In *Proc. of EMNLP*, pages 3528–3539, 2020.

[Snyder and Palmer, 2004] Benjamin Snyder and Martha Palmer. The English all-words task. In *Proc. of Senseval-3*, pages 41–43, 2004.

[Taghipour and Ng, 2015] Kaveh Taghipour and Hwee Tou Ng. One million sense-tagged instances for word sense disambiguation and induction. In *Proc. of CoNLL*, pages 338–344, 2015.

A WNG Filtering

We perform a mild filtering over the examples in WordNet, enforcing reasonable minimum length and structure. In particular, we discard examples:

- that are shorter than 4 tokens;
- that do not contain at least 1 noun and 1 verb.

Furthermore, as the exact span where the sense occurs in a given example is not specified, we search for it looking for a span whose lemma is equal to that of the sense; should this search operation fail, we discard the whole example.

B ARES-score

B.1 Validation Split

To build the validation sets that we use for the ARES-score, we exploit the lexical resource SyntagNet [Maru *et al.*, 2019] for both $k = 1$ and $k = 2$. As for $k = 1$, we generate 3 groups of instances, each amounting to 100 elements, by randomly sampling from SyntagNet senses that are MFS, LFS and Unseen, respectively. As for $k = 2$, we build the following groups of senses:

- Senses that are jointly seen in the gold training set and that are both MFS;
- MFS senses that are seen in the gold training set, but never together;
- senses that are jointly seen in the gold training set and with either of them LFS;
- senses that are seen in the gold training set, but never together, with either of them LFS;
- senses that are jointly seen in the gold training set and that are both LFS;
- senses that are seen in the gold training set, but never together, that are both LFS;
- senses with either one of them unseen in the training set and the other MFS;
- senses with either one of them unseen in the training set and the other LFS;
- senses that are unseen in the training set.

We extract, by random sampling from SyntagNet, 60 pairs for each group, resulting in a validation set for $k = 2$ of 540 instances.

B.2 Comparison with Cross-Entropy Loss

Our choice of the ARES-score as the model selection strategy, was mostly motivated by theoretical concerns and we defer a more formal and thorough analysis of its appropriateness in the context of EXMOD for future work. Nonetheless, we compare, in Table 3, how EXMAKER, trained as depicted in task 5, fares i) when considering ARES as the development strategy, and ii) when considering, instead, the more common cross-entropy loss over a held-out sample of data. Hence, to create an appropriate setting, we reserve 800 sentences from SemCor in both trainings ($k = 1$ and $k = 2$); we further split them into 2 parts of identical size, *dev* and *test*. We use *dev* as the held-out sample of data we provide to the cross-entropy criterion at training time. Conversely we use *test* as the reference test set we will consider in this simulated version of task 5. We report results in Table 3.

The outcome of this experiment seems to suggest that ARES-score is indeed an adequate development metric for EXMOD, leading to the choice of a model that generates better examples for the input senses.

Selection Strategy	Performance
ARES	73.8
Cross-Entropy Loss	73.1

Table 3: Comparison between ARES score and cross-entropy over a held-out sample of data as model selection strategies. Reported scores are on SemEval-2007

C WSD Reference Model Training

In Table 4 we describe the hyperparameters we used to train the Word Sense Disambiguation model.

Hyperparameter	Value
Optimizer	Adam
Learning Rate	2×10^{-5}
Batch Size	16
Max Epochs	50
Patience	3
Validation Metric	ARES-score

Table 4: Hyperparameters utilized for the training of the WSD reference model.

D Qualitative Analysis Guideline

We employed a five-levels Likert scale for both the annotation tasks. We report an excerpt of the two tasks, together with the guidelines released to the annotators, in Table 5 and Table 6 for the fluency and coherence evaluation, respectively.

To break the uncomfortable silence, Haney began to talk.	
1	Wrong. The sentence has serious grammatical and structural issues. It doesn't make any sense. *Example: The runs cat before.*
2	Wrong. The sentence has no meaning, although the grammar is somehow correct. *Example: The book runs fast.*
3	Neutral. The sentence is meaningful, but incomplete. The grammar is mostly correct. *Example: and the cat was going there.*
4	Correct. The sentence has a meaning and it's complete. There are minor grammatical errors. *Example: The cat run really fast.*
5	Correct. The sentence is meaningful and grammatical. *Example: The cat runs really fast.*

Table 5: An excerpt of the fluency annotation task. For each annotation item (first row), we gave annotators a list of Likert scores (left column) with a brief description and an *example* (right column).

E EXMAKER Exemplification Samples

In Table 7 we show some samples that were annotated in the qualitative analysis produced by the EXMAKER system described in Task 5.

To break the *uncomfortable* silence, Haney began to talk. uncomfortable: *conducive to or feeling mental discomfort*.	
1	Wrong. The sentence does not use the given lemma.
2	Wrong. The sentence uses the given lemma in one of its senses, but not the one given as input.
3	Neutral. The sentence uses the given lemma in an ambiguous context. It is not possible to determine whether the sense is the one specified in the gloss or not.
4	Correct. The sentence uses the given lemma in a sense that is close to the one given in input.
5	Correct. The sentence uses the lemma with the given sense.

Table 6: An excerpt of the semantic coherence annotation task. The sentence is provided with a target *lemma* and its *gloss* from WordNet (first row). Then, for each Likert score (left column) there is a brief explanation on how to assign it (right column).

$k = 1$	- Input: {protester} someone who participates in a public display of group feeling. - Output: The {protesters} were dispersed by the police.
$k = 1$	- Input: {interest} the power of attracting or holding one's attention. - Output: The place was of great {interest} to him because of its location on the river.
$k = 2$	- Input: {give} cause to have, in the abstract sense or physical sense. {member} one of the persons who compose a social group. - Output: The {members} of the audience {gave} a standing ovation.
$k = 2$	- Input: {storage} an electronic memory device. {system} a procedure or process for obtaining an objective. - Output: This {storage} {system} is designed to provide the same level of protection for the data stored in the hard disk.
$k = 3$	- Input: {baby} a very young child (birth to 1 year) who has not yet begun to walk or talk. {milk} produced by mammary glands of female mammals for feeding their young . {feed} give food to someone . - Output: The {milk} was {fed} to the {baby} by his mother.
$k = 3$	- Input: {boy} a young male human. {eat} ingesting food. {hamburger} a type of meat, usually round. - Output: " I 've never {eaten} a {hamburger} before " , said the {boy} .

Table 7: Example outputs that EXMAKER generates for $k \in \{1, 2, 3\}$. For improved readability, we use curly brackets as the highlighting markers.