

GENESIS: A Generative Approach to Substitutes in Context

Caterina Lacerra

Sapienza University of Rome
lacerra@di.uniroma1.it

Rocco Tripodi

University of Bologna
rocco.tripodi@unibo.it

Roberto Navigli

Sapienza University of Rome
navigli@diag.uniroma1.it

Abstract

The lexical substitution task aims at generating a list of suitable replacements for a target word in context, ideally keeping the meaning of the modified text unchanged. While its usage has increased in recent years, the paucity of annotated data prevents the finetuning of neural models on the task, hindering the full fruition of recently introduced powerful architectures such as language models. Furthermore, lexical substitution is usually evaluated in a framework that is strictly bound to a limited vocabulary, making it impossible to credit appropriate, but out-of-vocabulary, substitutes. To assess these issues, we propose GENESIS (Generating Substitutes in contexts), the first generative approach to lexical substitution. Thanks to a seq2seq model, we generate substitutes for a word according to the context it appears in, attaining state-of-the-art results on different benchmarks. Moreover, our approach allows silver data to be produced for further improving the performances of lexical substitution systems. Along with an extensive analysis of GENESIS results, we also present a human evaluation of the generated substitutes in order to assess their quality. We release the fine-tuned models, the generated datasets and the code to reproduce the experiments at <https://github.com/SapienzaNLP/genesis>.

1 Introduction

The lexical substitution task (McCarthy and Navigli, 2009) requires a system to provide adequate replacements for a target word in a given context. Through the years, two lexical substitution variants have been proposed, i.e., *candidates ranking* and *substitutes prediction* (Melamud et al., 2015). While the former aims at ranking a list of predefined candidate substitutes for a word in a given context, the latter is more challenging, requiring

a system to output a sorted list of replacements without any predefined substitutes inventory. Although it is not explicitly required by either of the two tasks, a good substitution system is expected to capture the semantics of its input and implicitly perform a *soft* disambiguation. For example, denoting *bright* as target word in the context sentence "She is a *bright* student", we expect a good substitution system to provide a set of substitutes closer to $\{intelligent, clever, smart\}$ than to $\{luminous, clear, light\}$. Thanks to this implicit disambiguation capability, lexical substitution has shown its usefulness in several scenarios, such as word sense induction (Başkaya et al., 2013; Amrami and Goldberg, 2018; Arefyev et al., 2019), data augmentation (Jia et al., 2019; Arefyev et al., 2020), word sense disambiguation (Hou et al., 2020) and semantic role labeling (Bingel et al., 2018).

However, despite having been employed in numerous downstream tasks, the lexical substitution task still presents unresolved issues that need to be addressed. First, the shortage of large-scale corpora annotated with the expected substitutes hinders the use of supervised techniques, including powerful Transformer-based language models, thus leaving the task in a possibly sub-optimal setting. Second, the evaluation metrics provided for the task are bound to the test vocabulary, hence they fail to capture the quality of substitutes outside the vocabulary; moreover, the vocabulary is usually small and often biased by the particular linguistic style and background of the annotators who developed the datasets.¹

In this paper, we focus on *substitutes prediction* and address the above problems by proposing GENESIS, a generative approach to lexical substitution. We find that not only is this novel approach effective

¹The benchmarks for lexical substitution were released more than ten years ago.

tive when tested on the lexical substitution task, but also that it can be applied to generate substitutes from raw text, enabling the effortless construction of large-scale silver data. Moreover, we conduct an annotation task to analyze the results of our model and to validate out-of-vocabulary generations.

Our contribution is threefold:

- A novel generative approach to lexical substitution that outperforms the state of the art.
- An automated method to produce high-quality silver data for lexical substitution.
- An annotation task to evaluate out-of-vocabulary generations.

2 Related Work

Through the years, several approaches have been developed to tackle the lexical substitution tasks, but, to the best of our knowledge, ours is the first attempt to apply a generative approach to it. In what follows, we first review the principal approaches and resources for the lexical substitution tasks, and then provide a brief overview of generative methods across different fields.

Lexical Substitution Approaches Since its presentation by [McCarthy and Navigli \(2007\)](#), a variety of different approaches have been explored to produce the substitutes that better fit the context. Earlier methods made use of external knowledge bases such as WordNet ([Miller, 1995](#)) to extract possible substitutes and construct delexicalized features ([Szarvas et al., 2013](#)), or they employed word embeddings to represent both the target and the substitutes in their context and rank them through *ad-hoc* metrics ([Melamud et al., 2015, 2016](#)). However, the recent spread of pre-trained language models has deeply reshaped approaches to lexical substitution, standardizing the use of contextualized word representations to provide a context-aware distribution over the output vocabulary. The first work in this direction was that of [Garí Soler et al. \(2019\)](#), where ELMo ([Peters et al., 2018](#)) embeddings are used to rank substitutes according to their cosine similarity to the target. In [Zhou et al. \(2019\)](#), instead, the input context is represented through a BERT model ([Devlin et al., 2019](#)). The authors partially mask the target word in its context, in order to obtain a representation that includes a faded target information; this representation is then used to obtain a probability distribution over the BERT

output vocabulary that is not biased towards the target. Finally, the top scoring substitutes are re-ranked with a measure of similarity that takes into account both the cosine similarity and the relative attention scores between the target and the substitute. In a similar vein, [Arefyev et al. \(2020\)](#) proposed an extensive comparison of how several pre-trained language models perform on the task, also injecting information about the target from word embeddings or rephrasing the input with dynamic patterns. Their best performing method produces an XLNet ([Yang et al., 2019](#)) contextualized embedding of the target word combined with static frequency information about proximity between target and substitute. This combined representation is then used to obtain a ranking of substitutes from the XLNet vocabulary that is further refined with postprocessing.

Despite the improvement in performances that large language models brought to the task, these methods work in a potentially sub-optimal setting, since they are used as feature extractors and are not finetuned, due to the paucity of large-scale annotated data ([Garí Soler et al., 2019](#)).

Lexical Substitution Resources The first dataset released was the Lexical Substitution Task (LST), proposed as test set for the task by [McCarthy and Navigli \(2007\)](#). It contains 2010 sentences with a single target word per sentence, including around 200 distinct targets. Each instance is associated with several substitutes that were chosen by five native English speaker annotators. The small coverage of LST led to the creation of the Turk bootstrap Word Sense Inventory ([Biemann, 2012](#), TWSI), a first attempt to collect a large-scale dataset. The author deployed Amazon Mechanical Turk to annotate 25K sentences from Wikipedia, which, however, only cover noun targets. To overcome this shortcoming, [Kremer et al. \(2014\)](#) proposed Concept In Context (CoInCo), a dataset of 2474 sentences covering 3874 distinct targets with diverse part-of-speech tags. Each sentence has one or more targets, for a total of 15k instances annotated through Amazon Mechanical Turk.

Generative Approaches Generative pre-trained language models such as GPT ([Radford et al., 2018](#)) have shown to be highly effective in Natural Language Generation, catching the attention of the research community. Indeed, pre-trained

models such as BART (Lewis et al., 2020) suit a wide range of NLP applications. Thanks to the flexibility of seq2seq learning, these models can be easily adapted to different tasks, including sequence and token classification or sequence generation, *inter alia*. Interestingly, generative models have also been employed in tasks that are not usually formulated as sequence-to-sequence learning; for example, there have been effective applications of seq2seq architectures to definition modeling (Bevilacqua et al., 2020), cross-lingual Abstract Meaning Representation (Blloshmi et al., 2020), end-to-end Semantic Role Labeling (Blloshmi et al., 2021) and Semantic Parsing (Procopio et al., 2021; Bevilacqua et al., 2021a).

Inspired by these successful applications of generative approaches we here propose applying, for the first time, a generative seq2seq model to the lexical substitution task. Differently from previous approaches in the field, we finetune a pre-trained model to produce substitutes starting from a word in its context. Moreover, our method can be used to generate silver data for the lexical substitution task, reducing the lack of annotated data.

3 GENESIS

The task of substitutes prediction requires finding replacements for a target word in a context that ideally do not modify the overall meaning of that context. More formally, given a target word w_t occurring in a context sentence $x = w_1, \dots, w_n$, a substitution system has to assemble a ranked list s of possible replacements for w_t according to its context x . Consider as an example the context

The roses are *bright*. (1)

where the target $w_t = \textit{bright}$ appears. As output of our system, we expect a generated list of substitutes, such as $s = [\textit{vivid}, \textit{luminous}, \textit{shining}]$.

We tackle the lexical substitution task with a two-stage process: first, we use a seq2seq model that takes as input both the context and the target, and generates several possible lists of substitutes (substitutes generation, Section 3.1); second, we process the substitutes collected with the first step to obtain the final, ranked list (substitutes ranking, Section 3.2). The whole process is described in Figure 1. Throughout the paper, we will consider each target word w_t to be univocally associated with its part-of-speech (POS) tag. To improve readability we discard POS tags from the notation.

3.1 Substitutes Generation

We assume to have a seq2seq model M that, given a context x where a target word w_t occurs, is able to generate a sequence of substitutes s by modeling the probability

$$P(s|x; w_t) = \prod_{i=1}^{|s|} P(s_i | s_{0:i-1}; x; w_t)$$

where s_0 is a special start token. In order to structure both the target and the context as a single input sequence for M , we identify the target in its context by surrounding it with two special tokens. Formally, for a target word w_t in x we define the input $m_{w_t, x}$ as:

$$m_{w_t, x} = w_1 w_2 \dots \langle t \rangle w_t \langle /t \rangle \dots w_n.$$

Thus, example (1) is structured as:

The roses are $\langle t \rangle$ bright $\langle /t \rangle$.

The expected output s , instead, is a comma-separated sequence $s = s_1, \dots, s_q$ where each word s_i is a possible substitute for w_t in x . At training time, we provide the model with a sequence of gold substitutes $\hat{s} = \hat{s}_1, \dots, \hat{s}_k$ also structured as a comma-separated list. Thus, we can train M by minimizing the cross-entropy loss between the gold and the generated sequences. At inference time, for each input sequence $m_{w_t, x}$ we actually produce several substitute sequences s^1, \dots, s^b obtained with beam-search decoding (Figure 1(a)).

3.2 Substitutes Ranking

Once the model has produced a set of substitute sequences, we collect the unique substitutes and rank them according to the context.

Collection and Filtering First, we create the set W of words² that occur across the sequences s^1, \dots, s^b . W could contain inappropriate substitutes, such as the target itself or words that are closely related to the target but have a different part of speech (Figure 1(b)). To provide a cleaned list of substitutes, for each target word we define a possible output vocabulary and remove from W all the words that are not part of it, including the target itself (Figure 1(b), bold)). We denote this reduced set as W_{clean} . The building of the output vocabulary is detailed in Section 4.

²For ease of reading, we do not specify the target w_t and context x as subscripts.

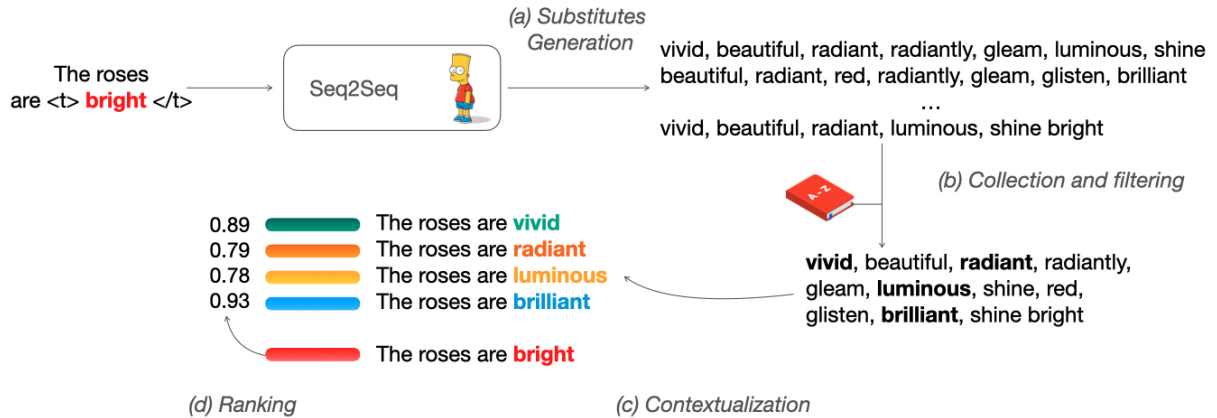


Figure 1: A schematic representation of GENESIS. The input is fed to a seq2seq model that produces several sequences of substitutes (a); the substitutes are collected and filtered according to an output vocabulary (b). We create a new sentence for each substitute by using it as replacement for the target in the original context (c); finally, we use the contextualized representations of the substitutes to rank them by similarity with the target (d).

Contextualization We denote with j the index of the target word w_t in context x , and the contextualized representation of w_t in x as:

$$\mathbf{v}_{x,j} = NLM(x)[j] \quad (2)$$

where $NLM(x)$ is the representation of x obtained through an arbitrary neural language model. Then, for each valid substitute $w_c \in W_{clean}$, we obtain a modified context x_{w_c} by replacing the target word w_t with w_c (Figure 1(c)). Now we can obtain a contextualized representation of each substitute as:

$$\mathbf{v}_{x_{w_c},j} = NLM(x_{w_c})[j]. \quad (3)$$

Ranking To produce the final ranking of the substitutes (Figure 1(d)) we compute the cosine similarity of the target word vector with respect to that of each substitute, i.e., $\cos\text{-sim}(\mathbf{v}_{x,j}, \mathbf{v}_{x_{w_c},j}) \forall w_c \in W_{clean}$, and order the substitutes by their descending cosine similarity with the target.

4 Vocabulary Definition

One of the challenges in the lexical substitution task is the lack of a predefined substitute inventory, i.e., for each target word we lack a reference list of possible replacements. Importantly, with GENESIS we can produce approximately any word in the English vocabulary as substitute, although standard evaluation benchmarks consider valid only the words in the test vocabulary. To reach a suitable trade-off between the generative power of the model and the necessity of a fair evaluation, we

define an output vocabulary that the model has to stick to, i.e., we discard any generated word that is not contained in it (Section 3.2).

To build our vocabulary, we take advantage of WordNet 3.0, a widely-used lexicographic resource structured as a graph. Each WordNet node is a *synset*, i.e., a set of different lexicalizations with the same meaning and POS, while edges represent semantic relations between synsets, such as hyponymy and hypernymy. For example, one of the synsets for the adjective *bright* is $\{bright, brilliant, vivid\}$, that is connected through the *similar-to* semantic relation to the synset $\{colorful, colourful\}$. For each target w_t we compute a set of synsets D_{w_t} that defines the output vocabulary. We initialize D_{w_t} as the set of synsets S_{w_t} where w_t appears.³ Then, for each $s_{w_t} \in S_{w_t}$ we expand D_{w_t} by collecting all the neighbors $N(s_{w_t})$ of s_{w_t} ; finally, for each neighbor n that is connected to s_{w_t} through a *hyponymy*, *hypernymy*, *similar-to* or *see-also* relation, we add all the neighbors of n to D_{w_t} . We define as possible substitutes for w_t the union of all the lexicalizations appearing in D_{w_t} .

This procedure, visualized in Figure 2, builds a vocabulary that covers all the senses enumerated in WordNet for a given target, defining a reduced range of available substitutions that is still challenging for the task. To provide a quantification of the coverage of the output vocabulary, we report that, when computed for the LST targets, it includes 25 842 distinct substitute words, while the

³We recall from Section 3 that each target is univocally tied to its POS tag.

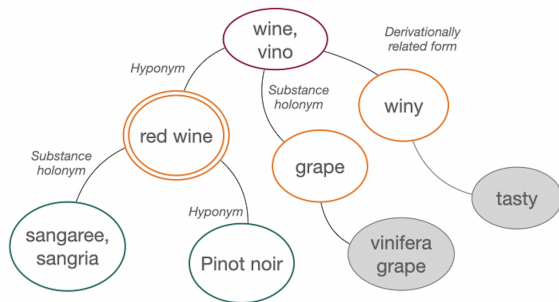


Figure 2: A visual example of how the vocabulary is constructed. We consider s_w , one of the two synsets where the noun *wine* appears (purple oval). First, we consider all its neighbors (orange ovals), then for each neighbor n of s_w connected through *hypernymy*, *hyponymy*, *similar-to* and *see-also* relations (double orange oval), we include all the neighbors of n (green ovals). The neighbors of synsets connected through different relations to s_w are discarded (grey ovals).

original test set has 3154 possible substitutes. Their intersection covers 2013 words.

5 Dataset Generation

GENESIS is able to generate substitutes starting from a word in its context. Thus, starting from a source dataset C of target words in context, we can exploit GENESIS to produce ranked lists of substitutes and, associating the generated substitutes with the targets, obtain silver datasets for the lexical substitution task. To this end, first, we finetune GENESIS on a gold dataset for the lexical substitution task; then, we give as input to the finetuned model the corpus C , generating as output a list of replacements for each input instance. The input instances, associated with the generated substitutes, constitute the silver corpus. To ensure the quality of the generated substitutes, we apply a similarity threshold λ on the ranking step of GENESIS (cf. Section 3), keeping only the substitutes whose similarity to the target is higher than λ . As source dataset C we exploit SemCor (Miller et al., 1993), a manually annotated corpus where instances are sense-tagged according to the WordNet sense inventory⁴. While it is typically used as a training corpus for English Word Sense Disambiguation (WSD), as we show, its manually-curated sense distribution is also beneficial for lexical substitution. Indeed, having a frequency of target words that

⁴We use the version released at <http://nlp.uniroma1.it/wsdeval/training-data>, with sense annotations that leverage WordNet 3.0.

follows a sense distribution involves the generation of substitutes for different senses of the same word, helping lexical substitution systems finetuned on the silver dataset to generalize more effectively.

6 Experimental Setup

In this section, we specify the setting used to tackle the lexical substitution task.

Model We use BART (Lewis et al., 2020) as seq2seq model, trained through the RAdam optimiser ($1r 10^{-5}$); we train it for a maximum of 100 epochs, with early stopping and patience set to 2 epochs. The input is fed to the model in batches of up to 600 tokens. To obtain the contextualized representations in Equations (2) and (3) we use the average of the last four⁵ hidden layers of BERT large cased. Both BART and BERT are used through the HuggingFace (Wolf et al., 2020) implementations.

Datasets We finetune BART on the concatenation of CoInCo and TWSI. Indeed, the former is originally distributed without training split, with only test and dev sets released; the latter, instead, contains only nouns, so it is not suitable for training alone. Thus, we concatenate the two datasets and produce new train and dev splits by randomly reserving 30% of the target contexts for the dev set \mathcal{CT}_D and the remaining 70% as training split \mathcal{CT}_T . As test set we use LST, the dataset originally released for the SemEval-2007 task. As regards the generated datasets, we denote with GENSEM COR_n the dataset obtained by generating substitutes for a sample of n contexts randomly drawn from SemCor. Starting from the size of \mathcal{CT}_T , i.e., 37k sentences, we generate four different samples by doubling the dataset size each time, with each sample including all the sentences in the previous one, i.e. $\text{GENSEM COR}_{37k} \subset \text{GENSEM COR}_{74k}$ and so on. The final dataset, that includes all the SemCor sentences for which at least one substitute has been generated, is identified by GENSEM COR. We highlight that, when training on GENSEM COR datasets, we use only silver data, without concatenating gold corpora. The properties of the gold and generated datasets are summarised in Table 1.

Evaluation Metrics We evaluate the performance of our model using the metrics originally proposed for the task (McCarthy and Navigli,

⁵These layers better capture the semantics than the last layer only, as shown in Devlin et al. (2019) and Loureiro and Jorge (2019), *inter alia*.

Dataset	targets	contexts	avg substitutes
\mathcal{CT}_T	4,301	37,172	6.6
\mathcal{CT}_D	297	3,095	6.8
LST	205	2,003	4.0
GENSEMCOR _{37k}	9,423	37,000	4.2
GENSEMCOR _{74k}	12,726	67,231	3.4
GENSEMCOR _{148k}	17,714	137,497	3.4
GENSEMCOR	22,435	220,237	4.2

Table 1: The number of targets, context sentences and average substitutes per instance in the gold (first block) and in the generated (second block) datasets.

2009), i.e., *best* and *out-of-ten* (*oot*), together with their *mode* variations. The *best* metric allows a system to produce as many substitutes as are considered useful, by dividing the credit for each correct guess by the number of produced guesses. The best substitute should ideally be provided first. For each test instance, we provide the scorer only with the first substitute from the ranking detailed in Section 3.2. The *oot* metric evaluates up to ten candidates that have all the same relevance for the target, without dividing the credit of each correct guess by the number of produced guesses. In this case, we provide the scorer with the first ten substitutes as ranked by GENESIS. The *mode* variations of the *best* and *oot* metrics evaluate only the subset of the test set where a *mode* exist, i.e., where a majority of the annotators selected a single substitute as the best replacement. The formalization of the metrics employed is detailed in the supplementary materials, Section A. In addition to the standard metrics, we follow Arefyev et al. (2019) and also report p@1, p@3 and r@10.

Fallback Strategy When evaluating a system on the *oot* metrics, there is no advantage in providing less than ten substitutes. For this reason, whenever the procedure described in Section 3 results in less than ten substitutes, we apply a two-stage fallback strategy. First, we include in the substitutes all those words generated that were discarded when cutting on the output vocabulary. Second, if the list still has less than ten candidates, we extract the substitutes from the vocabulary that are not produced by the model, rank them according to their cosine similarity with the target (cf. Section 3.2) and extend the sequence produced until it reaches ten substitutes.

6.1 Parameter Selection

GENESIS has several features that can be personalized, from the model configuration to the generation parameters. With the aim of obtaining the best-performing setting for the lexical substitution task, we conduct an extensive tuning of GENESIS configuration, testing how each parameter affects the results on the dev dataset (\mathcal{CT}_D). Here we briefly describe the best-performing settings, while we report the results for each variation of the parameters in the supplementary material (Sections B, C).

Model Parameters We experiment with different values of dropout, encoder layer dropout and decoder layer dropout. We investigate how the variation of each parameter influences the performances by exploring values in the range [0, 0.6]. When training on the \mathcal{CT}_T dataset, the best performing setup uses dropout = 0.5, encoder layer dropout = 0.2 and decoder layer dropout = 0.6. This setting is used to perform all the experiments on the \mathcal{CT}_T dataset and to generate the datasets from SemCor. Then, a new selection of parameters is made on the GENSEMCOR_{37k} dataset, resulting in a new configuration with dropout = 0.1, encoder layer dropout = 0.6 and decoder layer dropout = 0.2. This configuration is used for the experiments on the GENSEMCOR datasets.

Generation Parameters Several decoding strategies are available for seq2seq models. We experiment with beam sizes and check whether the use of sampling is beneficial for the task. The optimal configuration has beam size 50 and no sampling.

Dataset Generation Parameters The generated substitutes are filtered through the similarity threshold λ . We tune it experimenting with the values in [0.5, 0.7, 0.8], with the best performing dataset obtained with 0.7.

7 Experiments

Once all the parameters have been tuned, we train GENESIS on \mathcal{CT}_T , cut the substitutes generated according to the output vocabulary, apply the fallback strategy and test on the LST dataset.

Baseline Using a predefined vocabulary limits the possible outputs of our model; therefore, to assess whether the performances of GENESIS are mainly influenced by the restricted vocabulary, we






Dataset	Approach	best	best-mode	oot	oot-mode	p@1	p@3	r@10
-	Zhou et al. (2019)	20.3	34.2	55.4	68.4	51.1	-	-
-	Arefyev et al. (2020)	-	-	-	-	49.5	34.9	47.2
-	vocabulary baseline	0.5	0.6	45.0	60.5	1.2	23.9	39.2
\mathcal{CT}_T	GENESIS –  	19.2 ± 0.6	31.1 ± 1.5	45.7 ± 3.7	60.0 ± 4.6	47.9 ± 1.1	33.7 ± 1.2	40.7 ± 3.4
\mathcal{CT}_T	GENESIS – 	20.6 ± 0.8	33.2 ± 1.7	39.5 ± 2.9	52.3 ± 4.0	49.2 ± 1.9	38.6 ± 1.3	32.4 ± 2.9
\mathcal{CT}_T	GENESIS	20.6 ± 0.8	33.2 ± 1.7	50.0 ± 2.4	65.1 ± 2.4	49.2 ± 1.9	38.6 ± 1.3	44.7 ± 2.2
GENSEM COR _{37k}	GENESIS	21.3 ± 0.3	34.5 ± 0.7	50.8 ± 1.1	65.7 ± 1.6	50.5 ± 0.8	39.0 ± 0.6	45.6 ± 1.0
GENSEM COR _{74k}	GENESIS	21.6 ± 0.3	35.0 ± 0.7	52.4 ± 0.4	67.5 ± 0.6	51.9 ± 0.5	39.8 ± 0.3	47.1 ± 0.5
GENSEM COR _{148k}	GENESIS	21.3 ± 0.2	34.1 ± 0.5	52.7 ± 0.4	67.0 ± 0.5	52.1 ± 0.7	39.6 ± 0.5	47.6 ± 0.5
GENSEM COR	GENESIS	21.2 ± 0.3	34.1 ± 0.3	52.2 ± 0.4	66.4 ± 0.5	51.2 ± 0.7	39.7 ± 0.5	47.2 ± 0.5




Table 2: Results on the lexical substitution task of GENESIS trained on the \mathcal{CT}_T dataset (third block) and on GENSEM COR (fourth block). We compare GENESIS with the two latest approaches to the task (first block) and to a baseline (second block). –  and –  indicate that the output vocabulary cut and fallback strategy are discarded, respectively. For all the metrics, the higher the better.

devise a baseline that for each target word ranks by cosine similarity all the substitutes contained in the vocabulary built for the target, deploying the contextualization detailed in Section 3.

Comparison Systems We choose as comparison systems the two most recent approaches to the task, i.e., the BERT-based system proposed by Zhou et al. (2019) and the best-performing solution presented by Arefyev et al. (2020), i.e., an XLNet-based model enhanced with the injection of specific embedding information about the target word. These two models achieve the currently highest reported results on the task. In these approaches the language models are used in a feature-based approach, i.e., they are not finetuned for the task. As already noted by Arefyev et al. (2020), both models output a probability distribution over a BPE-based vocabulary, making it tricky to reconstruct words at inference time. GENESIS, instead, overcomes this limit by relying on the decoding strategy of the generative model.

8 Results

We report our results in Table 2. The baseline (second block) performs poorly when it comes to predicting the most appropriate substitute (*best*, p@1, p@3), while it is quite strong in evaluating the top ten substitutes (*oot*, r@10). This is somehow to be expected: the average number of substitutes in the test set is four (cf. Table 1), hence, there is a good chance that the ten substitutes in WordNet that are closest to the target include the gold ones. As regards the results obtained with GENESIS, in each configuration we report the average of five runs with their standard deviation.

First, we inspect the performances of GENESIS without output vocabulary and without fallback strategy (GENESIS –  ). The generative approach alone is noisy, showing performances that are lower than the state of the art in any metric. Indeed, when adding the cut on the output vocabulary (GENESIS – ), the scores increase on *best*, *best-mode*, p@1 and p@3, reaching performances that are higher than the previous state of the art on *best* and p@3. At the same time, though, reducing the substitutes to the output vocabulary leads to the production of less than ten substitutes, thus decreasing the recall scores, as shown by the drop on r@10 and *oot*. This is further confirmed by the improvement on the *oot* and r@10 metrics given by the use of the fallback strategy (GENESIS), i.e., when using the complete system and always providing ten substitutes. In the fourth block of the table we present the results obtained when finetuning BART over the generated datasets. We start with GENSEM COR_{37k}, that is comparable in size with \mathcal{CT}_T . In this case, the silver dataset performs better than the gold one, with results that are better than the state of the art on p@3, *best* and *best-mode*, besides being way more stable, as shown by the reduced variance across the metrics. We believe this improved behavior is due to the wider variety of targets (and consequently substitutes) to be found in the generated datasets (cf. Table 1), which helps the model to generalize more effectively. Increasing the size of the sample considered to 148k helps improve the results, achieving state-of-the-art performances on five metrics out of seven⁶. With 148k sentences the system seems to reach a stable point,

⁶Both Arefyev et al. (2020) and we ourselves failed in reproducing the results reported in Zhou et al. (2019).

and keeping on adding sentences does not bring any additional useful information to the model.

9 Qualitative Evaluation

Our quantitative analysis shows that the substitutes produced by GENESIS are good enough to outperform the previous approaches to the task. The evaluation setting, though, is inherently limited to a fixed vocabulary⁷. Hence, we devise an annotation task to assess the quality of generated substitutes, investigating whether GENESIS is able to generate substitutes that, even when not appearing in the gold standard, are judged good replacements by human annotators.

Annotation Task We set up a test where an annotator is provided with a target word in context and a set of substitutes that are equally distributed among the gold and the generated ones. The annotator is required to select, if there are any, all the substitutes that are not suitable replacements for the target in the given context. We select three annotators with certified proficiency in English and previous experience in linguistic annotation tasks and present them with a sample of 322 test instances drawn from the LST dataset⁸. The annotators are asked to select the inappropriate substitutes from an anonymized shuffled set of three gold and three generated substitutes, obtained with GENESIS trained on \mathcal{CT}_T . For all the instances, the gold substitutes do not appear in the generated ones and vice versa. The annotation guidelines are reported in the supplementary material (Section D).

Inter-Annotator Agreement Since each annotator may select more than one substitute, we measure the inter-annotator agreement (IAA) using Kraemer’s κ coefficient (Kraemer, 1980), an extension of the better known Cohen’s κ (Cohen, 1960) that allows multiple answers to be provided by annotators. We follow Landis and Koch (1977) to interpret κ values in the range $(0.4, 0.6]$ as *moderate agreement*, values in $(0.6, 0.8]$ as *substantial agreement* and those in $(0.8, 1.0)$ as *almost perfect agreement*. Annotations are usually considered reliable if their IAA agreement is equal or higher than

⁷We recall from Section 4 that the vocabulary of LST has slightly more than 3000 words.

⁸The sample size is significant with respect to the source dataset with confidence level of 95% and a margin error of ± 5 . The annotation interface was developed through Label Studio <https://github.com/heartexlabs/label-studio#try-out-label-studio>.

0.67 (Eugenio and Glass, 2004).

Results As expected, the percentage of bad substitutes is higher in the generated dataset than in the manually produced gold, with 21% of the generated replacements considered inappropriate, versus the 13% discarded from the gold dataset, with an inter-annotator agreement of 0.71. The high percentage of accepted substitutes among the generated ones reflects the good quality of the replacements provided by GENESIS, confirming the validity of the approach. The results on the gold set, instead, raise some questions on its completeness and on the validity of an evaluation setting entirely dependent on such a restricted vocabulary. Indeed, more than 10% of gold substitutes are considered inappropriate by the annotators, and 40% of the discarded substitutes are gold. Moreover, we recall that, for each instance given to the annotators, the gold and the generated substitutes are disjoint, thus meaning that all the generated substitutes accepted by the annotators (79% of the ones proposed) are missing from the gold but still considered as suitable replacements.

To give a deeper insight into the incompleteness of the gold dataset, in Table 3 we provide an example of substitutes generated by GENESIS compared with the gold ones. GENESIS is able to provide a richer variety of appropriate substitutes compared to the gold, which lacks several valid substitutes. GENESIS shows its effectiveness in particular when the target is an adjective (e.g. *tremendous*, *bright*); with nouns and verbs, it still manages to provide additional good substitutes in comparison to the gold (e.g. *rest*, *skip*), but it shows shorter generations, leading to less substitutes. On the adverbs, instead, it sometimes fails to capture the semantics of the target, producing replacements that are not appropriate for the context (e.g. *late*) or that do not fit syntactically in the sentence (e.g. *earlier*).

10 Conclusions

In this paper we presented GENESIS, the first generative approach to lexical substitution. The method is simple but versatile: by finetuning a seq2seq model and post-processing its output we are able to generate appropriate substitutes for target words in contexts. Testing GENESIS on the lexical substitution task, we show performances that surpass the state of the art on several measures. At the same time, our approach can be used to produce large-scale silver data, which, when used as train-

Context	Gold Substitutes	Generated Substitutes
I think this idea has tremendous promise.	great, wonderful, enormous, terrific, huge, vast	huge, great, big, abundant, major, enormous, terrific, wonderful, excellent, ample, plentiful
A little bit of rest and relaxation is waiting for Michelle after the college national finals rodeo.	respite, repose, quiet, leisure, inactivity, sleep	recuperation, repose, reprieve, respite, relaxation, break
Well worth buying, well worth reading, but skip the bits you find irrelevant - another type of blended learning!	miss out, disregard, jump over, miss, avoid, jump, ignore, leave	ditch, disregard, dismiss, omit, neglect, leave
Use market tools to address environmental issues, such as eliminating subsidies for industries that severely harm the environment, like coal.	badly, gravely, seriously, dramatically	extremely, highly, significantly, hugely, greatly, really, extensively, tremendously, exceptionally
Very late in the movie, he finally does marry Scarlett.	far, on, near, end, near the end of, well into, latterly, last minute	very, soon, further, too, extra
If you wish to collect your robes earlier you should contact the above number to arrange collection.	beforehand, sooner, prior to that, by then, before	previously, before
He was bright and independent and proud.	intelligent, clever	sharp, enthusiastic, intelligent, talented
Let your child pick one bug to glue on the lid.	insect	fly, insectoid, critter, insect, creature

Table 3: An excerpt of the GENESIS output for LST sentences when training on GENSEM_{COR74k}, compared with gold substitutes. The generated substitutes reported do not include those added through the fallback strategy.

ing corpora for GENESIS, lead to outperformance over the state of the art on five out of seven metrics. Moreover, large-scale datasets could possibly be deployed to finetune lighter models for the task. Finally, we conduct an annotation task to evaluate the quality of generated substitutes, which results in recognizing 79% of the proposed replacements as good substitutes and also highlights some weaknesses of the current evaluation setting, in that it is strictly bound to an incomplete output vocabulary. As future work, we plan to extend GENESIS to other languages for which the lexical substitution task has been proposed, such as Italian (Torral, 2009) and German (Miller et al., 2015). Moreover, we will investigate how the substitutes produced can be deployed in lexical-semantic tasks such as

WSD (Bevilacqua et al., 2021b) or Lexical Simplification (Paetzold and Specia, 2017).

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 supported in part by the MIUR under grant “Dipartimenti di eccellenza 2018-2022” of the Department of Computer Science of the Sapienza University of Rome.

References

- Asaf Amrami and Yoav Goldberg. 2018. [Word Sense Induction with Neural biLM and Symmetric Patterns](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4860–4867, Brussels, Belgium.
- Nikolay Arefyev, Boris Sheludko, and Alexander Panchenko. 2019. [Combining Lexical Substitutes in Neural Word Sense Induction](#). In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, pages 62–70, Varna, Bulgaria.
- Nikolay Arefyev, Boris Sheludko, Alexander Podolskiy, and Alexander Panchenko. 2020. [Always Keep your Target in Mind: Studying Semantics and Improving Performance of Neural Lexical Substitution](#). In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1242–1255, Online.
- Osman Başkaya, Enis Sert, Volkan Cirik, and Deniz Yuret. 2013. [AI-KU: Using Substitute Vectors and Co-Occurrence Modeling For Word Sense Induction and Disambiguation](#). In *Second Joint Conference on Lexical and Computational Semantics, Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation*, pages 300–306, Atlanta, Georgia, USA.
- Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021a. [One SPRING to Rule Them Both: Symmetric AMR Semantic Parsing and Generation without a Complex pipeline](#). In *Proceedings of the AAAI Conference on Artificial Intelligence*, 14, pages 12564–12573, Online.
- Michele Bevilacqua, Marco Maru, and Roberto Navigli. 2020. [Generatory or “How We Went beyond Word Sense Inventories and Learned to Gloss”](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 7207–7221, Online.
- Michele Bevilacqua, Tommaso Pasini, Alessandro Raganato, Roberto Navigli, et al. 2021b. [Recent Trends in Word Sense Disambiguation: A Survey](#). In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 4330–4338, Online.
- Chris Biemann. 2012. [Turk Bootstrap Word Sense Inventory 2.0: A Large-Scale Resource for Lexical Substitution](#). In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, pages 4038–4042, Istanbul, Turkey.
- Joachim Bingel, Gustavo Paetzold, and Anders Søgaard. 2018. [Lexi: A tool for adaptive, personalized text simplification](#). In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 245–258, Santa Fe, New Mexico, USA.
- Rexhina Blloshmi, Simone Conia, Rocco Tripodi, and Roberto Navigli. 2021. [Generating Senses and RoLes: An End-to-End Model for Dependency- and Span-based Semantic Role Labeling](#). In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, pages 4559–4567, Online.
- Rexhina Blloshmi, Rocco Tripodi, and Roberto Navigli. 2020. [XL-AMR: Enabling cross-lingual AMR parsing with transfer learning techniques](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 2487–2500, Online.
- Jacob Cohen. 1960. [A coefficient of agreement for nominal scales](#). *Educational and psychological measurement*, 20(1):37–46.
- 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*, pages 4171–4186, Minneapolis, Minnesota.
- Barbara Di Eugenio and Michael Glass. 2004. [The kappa statistic: A second look](#). *Computational Linguistics*, 30(1):95–101.
- Aina Garí Soler, Anne Cocos, Marianna Apidianaki, and Chris Callison-Burch. 2019. [A Comparison of Context-sensitive Models for Lexical Substitution](#). In *Proceedings of the 13th International Conference on Computational Semantics - Long Papers*, pages 271–282, Gothenburg, Sweden.
- Bairu Hou, Fanchao Qi, Yuan Zang, Xurui Zhang, Zhiyuan Liu, and Maosong Sun. 2020. [Try to substitute: An unsupervised Chinese Word Sense Disambiguation Method Based on HowNet](#). In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1752–1757, Online.
- Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. [Certified Robustness to Adversarial Word Substitutions](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 4129–4142, Hong Kong, China.
- HC Kraemer. 1980. [Extension of the kappa coefficient](#). *Biometrics*, 36(2):207–216.
- Gerhard Kremer, Katrin Erk, Sebastian Padó, and Stefan Thater. 2014. [What Substitutes Tell Us - Analysis of an “All-Words” Lexical Substitution Corpus](#). In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 540–549, Gothenburg, Sweden.
- JR Landis and GG Koch. 1977. [The measurement of observer agreement for categorical data](#). *Biometrics*, 33(1):159–174.

- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. **BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension**. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online.
- Daniel Loureiro and Alípio Jorge. 2019. **Language Modelling Makes Sense: Propagating Representations through WordNet for Full-Coverage Word Sense Disambiguation**. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5682–5691, Florence, Italy.
- Diana McCarthy and Roberto Navigli. 2007. **SemEval-2007 Task 10: English Lexical Substitution Task**. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 48–53, Prague, Czech Republic.
- Diana McCarthy and Roberto Navigli. 2009. **The English lexical substitution task**. *Lang. Resour. Evaluation*, 43(2):139–159.
- Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. **context2vec: Learning Generic Context Embedding with Bidirectional LSTM**. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 51–61, Berlin, Germany.
- Oren Melamud, Omer Levy, and Ido Dagan. 2015. **A Simple Word Embedding Model for Lexical Substitution**. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 1–7, Denver, Colorado.
- George A. Miller. 1995. **WordNet: A Lexical Database for English**. *Communications of the ACM*, 38(11):39–41.
- George A. Miller, Claudia Leacock, Randee Teng, 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*.
- Tristan Miller, Darina Benikova, and Sallam Abualhaija. 2015. **GermEval 2015: LexSub – A shared task for German-language lexical substitution**. In *Proceedings of GermEval*, pages 9–17, Duisburg, Germany.
- Gustavo H. Paetzold and Lucia Specia. 2017. **A survey on lexical simplification**. *Journal of Artificial Intelligence Research*, 60(1):549–593.
- Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. **Deep Contextualized Word Representations**. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2227–2237, New Orleans, Louisiana.
- Luigi Procopio, Rocco Tripodi, and Roberto Navigli. 2021. **SGL: Speaking the graph languages of semantic parsing via multilingual translation**. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 325–337, Online.
- Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. **Improving language understanding by generative pre-training**. *OpenAI Blog*.
- György Szarvas, Chris Biemann, and Iryna Gurevych. 2013. **Supervised All-Words Lexical Substitution using Delexicalized Features**. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1131–1141, Atlanta, Georgia.
- Antonio Toral. 2009. **The Lexical Substitution task at EVALITA 2009**. In *Proceedings of EVALITA Workshop, 11th Congress of Italian Association for Artificial Intelligence*, Reggio Emilia, Italy.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. **Transformers: State-of-the-Art Natural Language Processing**. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online.
- Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. **XLNet: Generalized Autoregressive Pretraining for Language Understanding**. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019*, pages 5754–5764, Vancouver, Canada.
- Wangchunshu Zhou, Tao Ge, Ke Xu, Furu Wei, and Ming Zhou. 2019. **BERT-based Lexical Substitution**. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3368–3373, Florence, Italy.

A Metrics

We measured the performances on the task with the standard metrics for lexical substitution, i.e., *best* and *oot*.

Preliminaries We define T as the set of test instances and H as the set of annotators for the test set. Then, A is the set of instances in T for which the system provides at least one substitute. For the item $i \in A$ we identify the set of substitutes provided by the system as a_i , while h_i represents the set of responses for the item i provided by annotator $h \in H$. Finally, for each i we compute the multiset union H_i for all h_i for each annotator; each unique type res in H_i will have an associated frequency $freq_{res}$ for the number of times it appears in H_i . For example, let us consider an item for *happy* and assume the annotators had supplied answers as follows:

annotator id	responses
1	glad, merry
2	glad
3	cheerful, glad
4	merry
5	joyial

then H_i would be [*glad glad glad merry merry cheerful joyial*]. The *res* with associated frequencies would be *glad* 3, *merry* 2, *cheerful* 1, *joyial* 1.

As regards the *mode* variations, we define as the mode m_i the most frequent response for instance $i \in T$, if it exists. The sets where this mode exists are TM and AM , respectively, for the gold substitutes and the system ones.

best and best-mode Defining P_b and R_b as *best* precision and *best* recall respectively, we formulate *best* as

$$best = 2 * \frac{P_b R_b}{P_b + R_b} \quad (4)$$

where

$$P_b = \frac{1}{|A|} \sum_{a_i: i \in A} \frac{\sum_{res \in a_i} freq_{res}}{|H_i| * |a_i|} \quad (5)$$

$$R_b = \frac{1}{|T|} \sum_{a_i: i \in T} \frac{\sum_{res \in a_i} freq_{res}}{|H_i| * |a_i|} \quad (6)$$

As regards the *mode* variation, we modify precision and recall as

$$P_b = \frac{1}{|AM|} \sum_{bg_i \in AM} 1 \text{ if } bg = m_i \quad (7)$$

$$R_b = \frac{1}{|TM|} \sum_{bg_i \in TM} 1 \text{ if } bg = m_i \quad (8)$$

respectively, where bg is the best guess in the list of substitutes provided by the system. Then, *best-mode* is computed as in Equation 4.

oot and oot-mode In this case, we define P_o and R_o as *oot* precision and *oot* recall, respectively. Then we compute *oot* as

$$oot = 2 * \frac{P_o R_o}{P_o + R_o} \quad (9)$$

where

$$P_o = \frac{1}{|A|} \sum_{a_i: i \in A} \frac{\sum_{res \in a_i} freq_{res}}{|H_i|} \quad (10)$$

$$R_o = \frac{1}{|T|} \sum_{a_i: i \in T} \frac{\sum_{res \in a_i} freq_{res}}{|H_i|} \quad (11)$$

while in the *mode* variation precision and recall are slightly modified as

$$P_o = \frac{1}{|AM|} \sum_{a_i: i \in AM} 1 \text{ if any guess } \in a_i = m_i \quad (12)$$

$$R_o = \frac{1}{|TM|} \sum_{a_i: i \in TM} 1 \text{ if any guess } \in a_i = m_i \quad (13)$$

respectively. Once again, *oot-mode* can be computed by following Equation 9.

B Parameter Selection

As regards the model parameters, we explored how dropout, encoder layer dropout and decoder layer dropout affect the performances of the model. We found two different groups of better performing parameters when training on \mathcal{CT}_T and on GENSEM-COR_{37k}. In both cases, the variation of the results was measured on the \mathcal{CT}_D set. To maintain a feasible number of experiments, we set one parameter at a time, setting the dropout first, then the encoder layer dropout and finally the decoder layer

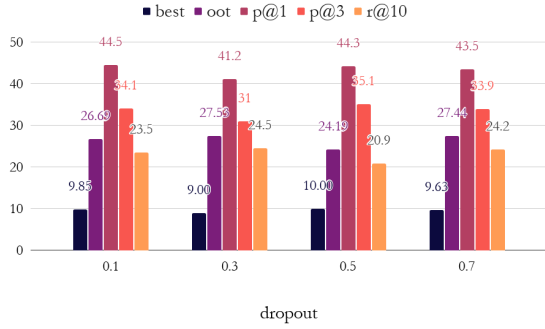


Figure 4: The results on the dev set for each tested value of dropout. The chosen value is 0.5.

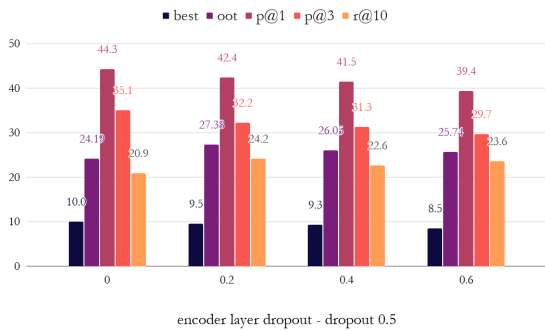


Figure 5: The results on the dev set for the encoder layer dropout, when keeping dropout fixed at 0.5. The chosen value is 0.2.

dropout. In Figures 4, 5 and 6 there are the results of each experiment for the tuning of parameters for the models trained on \mathcal{CT}_T , while Figures 7, 8 and 9 report the performances when training on $\text{GENSEM}\text{COR}_{37k}$. Often there is no single value for which the model performs uniformly better on all the metrics. In these cases, we tried to select the values that maximised more than one metric, or those that provided a higher improvement.

C Generation Parameters

We compared the results obtained on \mathcal{CT}_D after finetuning GENESIS on \mathcal{CT}_T , without any kind of filtering on the output vocabulary and without fallback strategy, in order to evaluate how each decoding parameter directly affects the generation quality. As evaluation metrics, we considered only the metrics that affect the whole dataset, i.e., we excluded the *mode* variations from our analysis. As generation parameters we experimented with beam size and top-k sampling. We compared the results for $k = 5, 10$ with those obtained without sampling, i.e., always picking the most probable

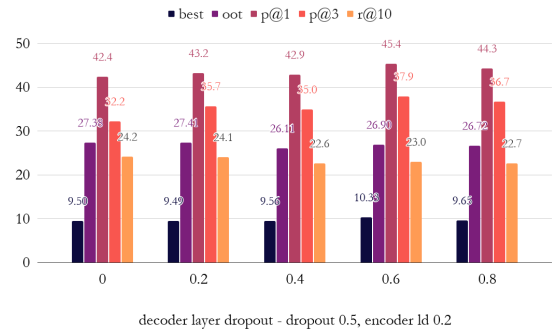


Figure 6: The results on the dev set for the decoder layer dropout, when keeping dropout and encoder layer dropout fixed at 0.5 and 0.2 respectively. The chosen value is 0.6.

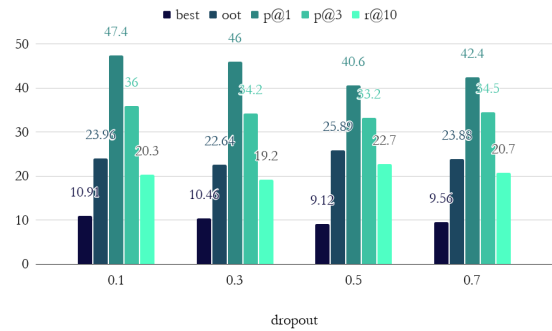


Figure 7: The results on the dev set for each tested value of dropout. The chosen value is 0.1.

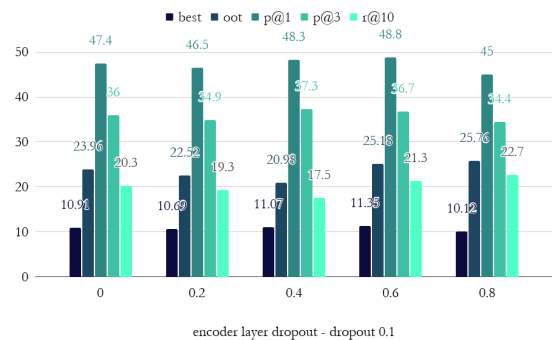


Figure 8: The results on the dev set for the encoder layer dropout, when keeping dropout fixed at 0.1. The chosen value is 0.6.

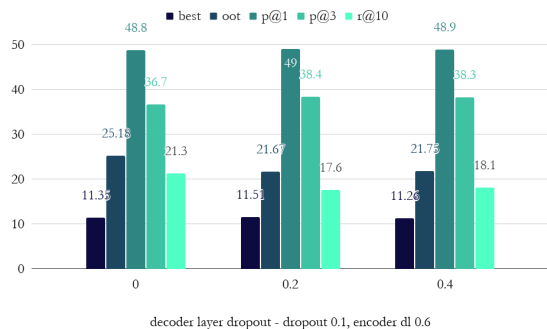


Figure 9: The results on the dev set for the decoder layer dropout, when keeping dropout and encoder layer dropout fixed at 0.1 and 0.6 respectively. The chosen value is 0.2.

one. In all the three cases, we used beam size 5 and postprocessed the generations of each beam as described in Section 3. We can see in Figure 10 that sampling increases the variety of the generated sequences, and consequently the precision’s scores, while sticking to the most probable candidate results in a higher recall and *oot* score, as we can see in the graph. Considering that there is no configuration that achieves the best results on all the metrics, and the increase in memory and time requirements to keep track of the *k* higher ranked words, we decided not to use sampling at decoding time. As regards beam size, we compared the results obtained when using 5, 15, 25 or 50 beams, described in Figure 11. As expected, generating more sequences results in a higher variety of words generated, thus leading to higher *oot* and *r@10*. At the same time, a broader generation may imply "dirtier" substitutes, with words that are close to the target but are not appropriate replacements, slightly decreasing *best* and *p@k* scores.

D Annotators Guidelines

For the annotation task, we provided each annotator with a set of instances comprising a context with a single target word (in bold) and six possible substitutes.

Consider as a running example the instance:

I **bought** this dress last year.

1) buy; 2) sold; 3) acquire; 4) get; 5) purchasing; 6) grease one’s palm.

The annotator had to select all the inappropriate substitutes, sticking to the following guidelines:

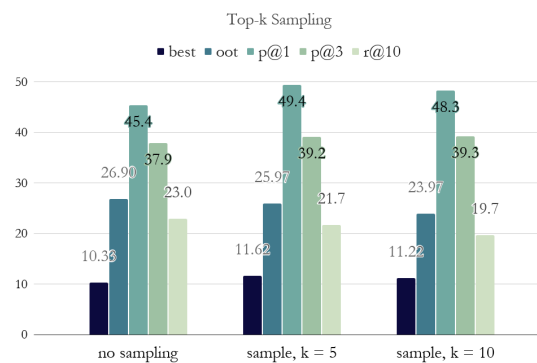


Figure 10: The results on the dev set when not using sampling (left) and when using it with top-5 (center) and top-10 (right) most probable elements in the output distribution.

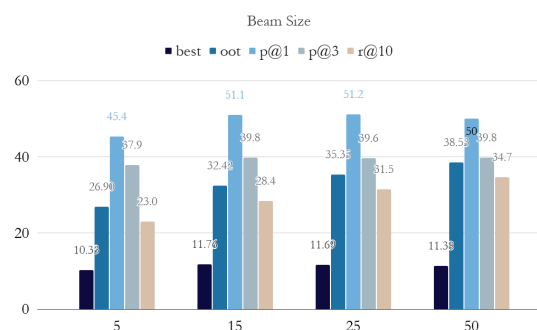


Figure 11: The variation of performances according to each beam size.

1. A substitute is wrong if it is an inflection of the target (1).
2. A substitute is wrong if its replacement modifies the meaning of the sentence (2, 6).
3. A substitute is wrong if its replacement in the context results in a wrong structure of the sentence (6).
4. A substitute is wrong if it has an inflected form that is different from that of the target (5).
5. A substitute is correct if it is in its base form and not in the same inflection as that of the target (3, 4).