Coordination Generation via Synchronized Text-Infilling

Hiroki Teranishi and **Yuji Matsumoto**
RIKEN Center for Advanced Intelligence Project
{hiroki.teranishi, yuji.matsumoto}@riken.jp

Abstract

Generating synthetic data for supervised learning from large-scale pre-trained language models has enhanced performances across several NLP tasks, especially in low-resource scenarios. In particular, many studies of data augmentation employ masked language models to replace words with other words in a sentence. However, most of them are evaluated on sentence classification tasks and cannot immediately be applied to tasks related to the sentence structure. In this paper, we propose a simple yet effective approach to generating sentences with a coordinate structure in which the boundaries of its conjuncts are explicitly specified. For a given span in a sentence, our method embeds a mask with a coordinating conjunction in two ways ("X *and* <mask>", "<mask> *and* X") and forces masked language models to fill the two blanks with an identical text. To achieve this, we introduce decoding methods for BERT and T5 models with the constraint that predictions for different masks are synchronized. Furthermore, we develop a training framework that effectively selects synthetic examples for the supervised coordination disambiguation task. We demonstrate that our method produces promising coordination instances that provide gains for the task in low-resource settings.

1 Introduction

Pre-trained language models (LMs) have brought a large shift from preparing a substantial number of training examples and learning task-specific representations to collecting tons of unlabeled text and learning task-agnostic representations. For instance, fine-tuning LMs, such as BERT (Devlin et al., 2019) and BART (Lewis et al., 2020), has improved performances of many natural language processing (NLP) tasks, even when using a small amount of training data. This paradigm shift has led to many challenges to exploit knowledge of LMs rather than manually annotated data, including zero- or few-shot learning (Radford et al., 2019; Brown et al., 2020; Shin et al., 2020; Gao et al., 2021) and data augmentation (Wu et al., 2019; Gao et al., 2019; Anaby-Tavor et al., 2020; Kumar et al., 2020). However, many of these studies focus on sentence classification tasks, such as question answering and sentiment analysis, and thus are not applicable to structured prediction tasks, such as syntactic parsing. Although some studies propose data augmentation methods that manipulate the structure of a sentence (Şahin and Steedman, 2018; Shi et al., 2021), producing syntactic structures using recent pre-trained LMs has not been explored.

As an attempt to produce syntactic structures using LMs, we explore approaches to generating coordinate structures, in which two or more elements, known as *conjuncts*, are linked by a coordinating conjunction (*coordinator*), and yielding the corresponding annotation that explicitly marks which elements in the sentence are coordinated. Coordination frequently appears in natural language sentences and causes ambiguities, leading to errors in many NLP tasks (Hara et al., 2009; Chae et al., 2014; Ficler and Goldberg, 2017). We expect that performances of supervised methods for coordination disambiguation (e.g., Ficler and Goldberg, 2016b; Teranishi et al., 2017, 2019) would be improved if we can generate good-quality coordinate structures with annotation.

In this paper, we present a method that exploits large-scale pre-trained LMs to inject a coordinate structure into a given sentence with the corresponding annotation of coordination boundaries. Specifically, we use masked language models (MLMs) to generate text that seems to be in parallel with a given reference span in a sentence. To enhance MLMs to make use of the *similarity* and *replaceability* properties of conjuncts described by Ficler and Goldberg (2016b) and Teranishi et al. (2017), our method utilizes a pair of sentences, each of which embeds a mask with a conjunction differ-

Proceedings of the 29th International Conference on Computational Linguistics, pages 5914–5924
October 12–17, 2022.

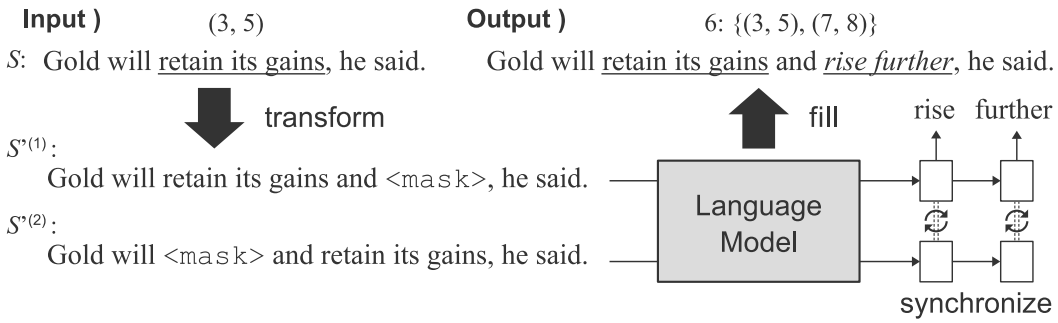

Figure 1: Overview of our approach. An input sentence with a reference span (*"retain its gains"*) is masked in two ways. Then, an MLM fills the masks with an identical text (*"rise further"*) according to the constraint that predictions for different masks are synchronized.

ently, and forces the models to fill the masks with an identical text (Figure 1). For BERT and T5 (Raffel et al., 2020) as instantiations of MLMs, we introduce decoding techniques that follow the constraint that predictions for different masks are made to be synchronized, which can be potentially applied to other generation tasks, such as paraphrasing and summarization. We also describe a training framework to effectively select synthetic examples for supervised models, where a classifier is employed to filter out erroneous samples as similarly used in self-training. Experimental results of the coordination disambiguation task in low-resource settings show that synthetic coordinate structures with annotation provide gains and indicate that our method yields promising instances of coordination.

The contributions of this work are summarized as follows:

- We propose a method for generating a sentence with a coordinate structure and the corresponding annotation that indicates the boundaries of coordination.
- We demonstrate simple decoding techniques for BERT and T5 to fill different masks with an identical text, which can be potentially used for other types of text generation.
- We integrate the generation process into a training framework for supervised methods where synthetic examples are effectively examined and selected.
- Our method produces promising coordination instances that facilitate the supervised coordination disambiguation task in low-resource settings.

2 Related Work

Data augmentation using language models. Data augmentation is a widely used technique to increase the amount of training data by creating synthetic examples from existing ones. In NLP, many replacement-based methods have been explored, which typically generate new sentences by replacing words in sentences with other words. Wei and Zou (2019) demonstrated that a simple synonym replacement using WordNet (Miller, 1995) improves the performance of sentence classification tasks. As synonym-based augmentation can be applied to only a small percentage of the vocabulary covered by a hand-crafted ontology, Kobayashi (2018) instead exploited a recurrent neural network-based LM to produce replacement words that fit in the context. This LM-based method was later extended by Wu et al. (2019). They instead used BERT (Devlin et al., 2019) for replacing words, which is a Transformer-based LM trained on an objective where the model has to predict a token for a masked token in a sentence. Anaby-Tavor et al. (2020) proposed a method to generate a sentence for a category by providing the class label to GPT-2 (Radford et al., 2019). Kumar et al. (2020) further generalized data augmentation methods that exploit pre-trained LMs. Our work is similar to these studies in that we employ a pre-trained LM to generate a new sentence by filling masks with words, but differs in that it does not require a human-annotated label for an input sentence.

Self-training. Self-training is a semi-supervised approach to effectively utilize unlabeled data. Self-training employs a model (*teacher*) trained on a limited amount of labeled data to assign synthetic labels to unlabeled data, which are later used to train another model (*student*). Self-training has been successfully applied to a wide variety of NLP tasks, such as word sense disambiguation (Yarowsky, 1995), syntactic parsing (McClosky

et al., 2006), and machine translation (Zhang and Zong, 2016). The training framework used in our method is inspired by self-training, but unlike general settings of self-training, it employs a trained model only for inspecting synthetic examples that are generated by another model.

Coordination disambiguation. The task of coordination disambiguation is to identify the conjuncts for a given coordinator. Prior to the recent advancement of neural networks, many efforts have been made to explore a range of approaches to this task: rule-based methods (Agarwal and Boggess, 1992; Kurohashi and Nagao, 1992), statistical methods (Resnik, 1999; Chantree et al., 2005), and machine learning-based methods using handcrafted rules (Buyko et al., 2007; Hara et al., 2009). Recently, Ficler and Goldberg (2016b) proposed a neural network-based method that considers the similarity and replaceability between conjuncts. As opposed to the pipeline approach of Ficler and Goldberg (2016b), Teranishi et al. (2017) developed an end-to-end approach using neural networks, which was further extended by Teranishi et al. (2019).

3 Method

This section describes our approach to coordination generation exploiting LMs pre-trained with the objectives for text-infilling. Figure 1 shows the overview of our approach. Formally, for a given sentence $S_{1:N} = w_1, \ldots, w_N$, the goal of our method is to embed a coordinate structure with annotation indicating a set of conjunct spans $\{(i^{(1)}, j^{(1)}), \ldots\}$ together with a coordinator w_k. We currently restrict each generated coordination instance to have two conjuncts linked by the conjunctive coordinator *"and"*. To this end, our method first selects a *reference span* (i, j) in S and generates text homologous to the span using an LM. The produced text is then inserted with a coordinating conjunction after the reference span, which results in a coordinate structure $\{(i, j), (i', j')\}$ accompanied by the coordinator. We explain the details of the method in the following sections.

3.1 Text generation for a span

To generate an alternative candidate for a reference span, a simple approach is to replace the reference span with a mask and use an MLM to fill it, regarding the infilling text as paired with the reference. We initially tested this approach, but found that it is not suitable for coordination generation because the model cannot see the reference span, and thus a produced span could be considerably different from the reference even if the resulting sentence is grammatically correct. Alternatively, we transform a sentence S into a twin of masked sentences $S'^{(1)}$ and $S'^{(2)}$, each of which embeds a mask with a conjunction differently, but the masks must be filled with an identical text. Concretely, we insert a conjunction followed by a mask (e.g. "*and* `<mask>`") after a reference span to obtain $S'^{(1)}$ and insert the same conjunction preceded by the same mask before the reference for $S'^{(2)}$, as depicted in Figure 1. This treatment enables the model to exploit the reference as well as the context for producing conjuncts that are similar and exchangeable (Ficler and Goldberg, 2016b; Teranishi et al., 2017).

3.2 Text infilling using MLM

This section describes how we can force an MLM to fill masks in different sentences with an identical text.

3.2.1 Non-autoregressive LM

We explain a synchronized text-infilling method using BERT (Devlin et al., 2019) as an instance of non-autoregressive LMs. To prepare an input sequence for BERT, we replace each masked span in $S'^{(1)}$ and $S'^{(2)}$ with a sequence of `[MASK]` tokens whose length is equal to the number of the tokens in the reference span, and concatenate $S'^{(1)}$ and $S'^{(2)}$ with a `[SEP]` token, as shown in Figure 2. Feeding the resulting sequence to BERT, the model emits the score vector (i.e., logit) $\mathbf{u}_t \in \mathbb{R}^V$ for each mask token, where V is the vocabulary size. When two masked spans start at l-th and m-th tokens in an input sequence respectively, infilling sequences $T'^{(1)}$ and $T'^{(2)}$ for the two spans are predicted as follows:

$$\begin{aligned} \mathbf{u}'_{l+t} = \mathbf{u}'_{m+t} &= f(\mathbf{u}_{l+t}, \mathbf{u}_{m+t}) \\ T'^{(1)}_{t+1} &= \arg\max(\mathbf{u}'_{l+t}) \\ T'^{(2)}_{t+1} &= \arg\max(\mathbf{u}'_{m+t}) \end{aligned} \quad (1)$$

where t (≥ 0) is an offset from the beginning of the masked spans. Consequently, predicted tokens $T'^{(1)}_{t+1}$ and $T'^{(2)}_{t+1}$ become the same under the restriction of the function f. As a synchronization function f, we choose the element-wise min function rather than the element-wise mean function to avoid picking a token that overfits in either $S'^{(1)}$ or $S'^{(2)}$ context, which happens when the value for

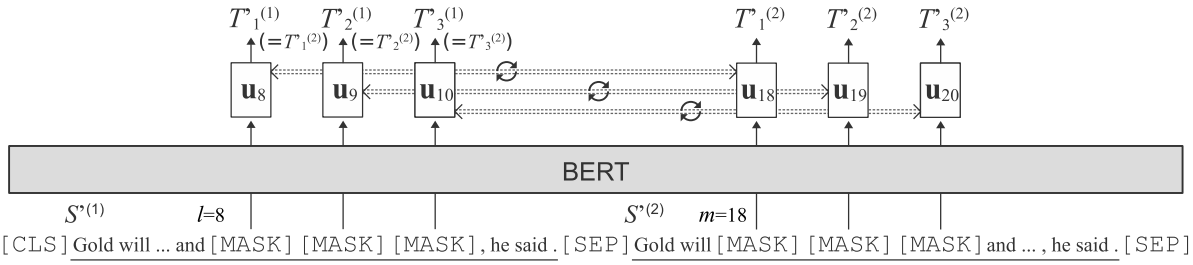

Figure 2: Synchronized decoding on BERT.

a token is extremely high in one logit but is not very high in the other logit. Intuitively, for the two masked sentences *"<X> primarily designed the Apple II."* and *"<X> is a founder of Apple Inc."*, synchronized decoding with the mean function would predict *"Steve Jobs"* for <X> by mistake when it overestimates the value of *"Steve Jobs"* in the second sentence even though the model knows *"Steve Wozniak"* fits adequately in both sentences.

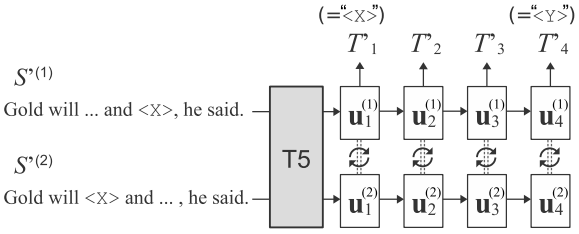

Figure 3: Synchronized decoding on T5.

3.2.2 Sequence-to-sequence LM

We adopt T5 (Raffel et al., 2020) as a sequence-to-sequence LM for our method. In the pre-training scheme of T5, the model consisting of a Transformer encoder and decoder (Vaswani et al., 2017) is trained to generate infilling texts for masked spans. For example, "*Thank you* <X> *me to your party* <Y> *week.*" is recovered by generating "<X> *for inviting* <Y> *last* <Z>", where the last extra token <Z> indicates the end of the sequence. Following this, we feed $S'^{(1)}$ and $S'^{(2)}$ into the encoder independently, but, for each decoding step, the decoder is constrained to output identical tokens for the two inputs (as illustrated in Figure 3):

$$\begin{aligned} \mathbf{u}'^{(1)}_t = \mathbf{u}'^{(2)}_t &= f(\mathbf{u}^{(1)}_t, \mathbf{u}^{(2)}_t) \\ T'^{(1)}_t &= \arg\max(\mathbf{u}'^{(1)}_t) \\ T'^{(2)}_t &= \arg\max(\mathbf{u}'^{(2)}_t) \end{aligned} \quad (2)$$

where f is the element-wise min function and $\mathbf{u}^{(1)}_t$ and $\mathbf{u}^{(2)}_t$ are logits for $S'^{(1)}$ and $S'^{(2)}$ at a decoding step t, respectively. Unlike text infilling by BERT, the decoder in T5 can produce a longer or shorter sequence than the reference span; however, we restrict the number of produced tokens between one-third and three times of the length of the reference span.

3.3 Span selection

Choosing a promising reference span is crucial for our coordination generation method because it becomes one of the conjuncts in a resulting coordinate structure. Ideally, choosing reference spans manually is preferable, which is a similar setting to active learning, but it is laborious. As an alternative, one strategy for automatic span selection is to randomly choose a sequence of tokens, as done in SpanBERT (Joshi et al., 2020). However, such a sequence is readily disqualified as a conjunct when it appears across two constituents. Alternatively, we can apply a constituency parser to a sentence and randomly pick one constituent from the parse tree. The constituency-based span selection is particularly suitable for our method because we can also restrict reference spans with their constituent labels[1].

4 Training Framework for Coordination Disambiguation

A synthetic example generated by the method described in Section 3 can be immediately seen as an annotated sentence, but the resulting sentence can be inconsistent with its coordination boundary. For example, we expect a sentential conjunct for the reference *"He said Mary had dinner with John."*, but an MLM can produce *"Bill"* rather than a sentence because both *"John and Bill"* as objects in $S'^{(1)}$ and *"Bill and Mary"* as subjects in $S'^{(2)}$ are

[1]Other than constituency, coordination involving lexical compounds can be created using hand-crafted rules based on part-of-speech. We tried this for the supervised task described in Section 5 for preliminary experiments, but it did not lead to a significant performance gain.

Algorithm 1 Training with generation and filtering

Input: labeled sentences L; unlabeled sentences U; predictor M; generator G

Constants: batch size N_{batch}; number of generated examples in a batch K ($\leq N_{batch}$); maximum number of generation trials K_{max}; threshold for filtering δ

Output: trained model M

train M on L
repeat
  $B \leftarrow \{\}$
  $k \leftarrow 0$
  while $|B| < K$ **and** $k < K_{max}$ **do**
    $x \leftarrow \textsc{Sample}(U, 1)$
    $(x', y') \leftarrow G(x)$
    if $\textsc{CalcScore}((x', y'), M) \geq \delta$ **then**
      $B \leftarrow B \cup \{(x', y')\}$
    end if
    $k \leftarrow k + 1$
  end while
  $B \leftarrow B \cup \textsc{Sample}(L, N_{batch} - K)$
  update M on B
until convergence
return M

coincidentally valid coordination. However, the resulting boundary annotation *"He said [Mary had dinner with John] and [Bill]."* is not considered to be correct. Although we can remove such examples by employing a filtering method that ensures the consistency between conjuncts, for example, using part-of-speech tags, developing such rule-based methods requires much effort.

For the purpose of improving supervised methods for the coordination disambiguation task, we instead propose a training framework inspired by self-training. The algorithm of our training method is described in Algorithm 1. At the first step of the procedure, a supervised model M for the task is trained on a limited amount of labeled examples L. In the following steps, the model is further trained on labeled examples L and synthetic examples generated from a large number of unlabeled sentences U. Concretely, in each training iteration, we pick one unlabeled sentence x that contains no coordination from U and feed it to an LM G (that encapsulates a span selection mechanism) to obtain a sentence x' that has a coordinate structure with the corresponding annotation y' using the method explained in Section 3. After generation, we employ the model M to assign a score to the example (x', y') for inspection. If the score is greater than or equal to a predefined threshold δ, the example is verified as legitimate and then added to a mini-batch. We repeat this generation–inspection process until we obtain K examples or until the process is repeated K_{max} times, which could happen when the generator G does not give promising examples. For the rest of $N_{batch} - K$ examples, we sample labeled examples from L. Note that we sample an unlabeled sentence x from U with replacement to reuse it because different coordinate structures can be created in it by choosing different reference spans.

	PTB	GENIA
# train sentences	39,832 (15,481)	14,326 (8,101)
# train instances	19,890	11,596
# dev sentences	1,700 (673)	1,361 (777)
# dev instances	848	1,146
# test sentences	2,416 (873)	1,360 (773)
# test instances	1,099	1,166

Table 1: Statistics of the Penn Treebank and the GENIA Treebank. Numbers in parentheses represent the number of sentences that have coordination instances[2].

5 Experiments

We evaluate our method using generated examples to train supervised models for the coordination disambiguation task.

5.1 Experimental settings

Data construction. We perform experiments on the coordination-annotated Penn Treebank (PTB) (Ficler and Goldberg, 2016a) and the GENIA Treebank (GENIA) (Tateisi et al., 2005). The statistics of the corpora are shown in Table 1. To simulate low-resource situations, where additional training examples could improve performance, we randomly sample 300 (250 for training and 50 for validation) and 600 (500 for training and 100 for validation) sentences containing coordination with annotation from the training set; the rest of sentences in the training set can be used as unlabeled seed data for generation. Specifically, the PTB training set consists of 39,832 sentences, 15,481 of which have coordinate structures. A random subset of them is used as a set of labeled sentences L, while 24,351 sentences with no coordination

[2]Following the previous studies (Ficler and Goldberg, 2016b; Teranishi et al., 2019), we focus on coordinate structures formed by "and", "or", "and/or", or "but".

#	Model	NP, NX	ADJP, ADVP	VP	PP	S, SBAR	Overall
			Penn Treebank				
250	baseline (no generation)	66.21	64.72	65.42	52.30	51.69	62.83 ± 2.36
250	+ generation by BERT	68.56	71.63	64.22	52.30	51.94	64.49 ± 2.42
250	+ generation by T5	**71.17**	**74.54**	**68.54**	**80.00**	**59.37**	**68.68** ± 1.51
500	baseline (no generation)	73.34	80.72	71.35	69.23	62.51	71.04 ± 1.59
500	+ generation by BERT	73.24	76.72	72.96	76.92	62.13	71.44 ± 1.57
500	+ generation by T5	**75.09**	**81.81**	**73.56**	**84.61**	**63.64**	**73.17** ± 1.56
			GENIA Treebank				
250	baseline (no generation)	48.39	69.50	62.96	57.46	32.65	50.49 ± 2.37
250	+ generation by BERT	54.81	71.00	62.45	65.35	41.77	56.07 ± 1.95
250	+ generation by T5	**57.26**	**73.50**	**65.41**	**66.47**	**42.53**	**58.39** ± 2.07
500	baseline (no generation)	54.71	69.50	65.93	64.50	53.16	57.11 ± 0.86
500	+ generation by BERT	57.60	75.00	69.03	65.63	54.93	59.87 ± 1.08
500	+ generation by T5	**58.10**	**78.50**	**69.29**	**67.32**	**57.46**	**60.70** ± 0.78

Table 2: Fine-tuning results with synthetic data on the PTB and GENIA test sets[5]. The first column indicates the number of annotated examples taken from the training set. We report accuracy averaged across five runs (with standard deviation) throughout sampling, generation, and fine-tuning.

are used to construct a set of unlabeled sentences U[3]. The same operation is also performed for the GENIA training set. We use the validation sets constructed from the training sets for model selection and the test sets for evaluation. The original development sets in the corpora are reserved for extensive analyses.

Evaluation and model. We evaluate models on the basis of their abilities to predict the correct beginning and ending positions of the coordinate structure for a given coordinating conjunction. As a supervised model for the task, we employ the pre-trained BERT-base-cased model combined with a coordination prediction module used in Teranishi et al. (2019), which gives scores to all possible endpoint pairs for a conjunction. The score of an endpoint pair (i, j) for the conjunction at the k-th position in a sentence is computed as follows:

$$\begin{aligned} \textsc{Score}_\theta(i,j,k) &= \text{MLP}(F_\theta(i,j,k)) \\ F_\theta(i,j,k) &= [\mathbf{h}_i - \mathbf{h}_{k+1}; \mathbf{h}_j - \mathbf{h}_{k-1}] \end{aligned} \quad (3)$$

where MLP is a network consisting of two linear transformations with a ReLU activation, and $\mathbf{h}_t$ is a representation vector for the t-th token produced from the last layer of the Transformer in the BERT model. All scores for possible pairs are normalized by the softmax function and thus each score takes a value between 0 and 1.

Training methodology. The entire network is updated for 10,000 steps and the best-performing model on the constructed validation set is selected for evaluation on the test set. We regard the model trained only on L as a baseline, whereas the proposed model is trained first on L for 1,000 steps (line 1 in Algorithm 1) and later on L together with generated examples using U for the rest of 9,000 steps (line 2–15 in Algorithm 1). We set $N_{batch} = 16$, $K = 8$, $K_{max} = 16$ and $\delta = 0.7$ for the proposed training framework in all experiments. We use BERT-large-cased and T5-large models for coordination generation. For span selection, we use the Berkeley neural parser (Kitaev et al., 2019) as the constituency-based span selection instead of conducting the manual selection for an active learning setting. Targeted spans are limited to non-terminals of a parse tree labeled with NP, VP, ADJP, ADVP, PP, SBAR, or S as they are typical types of conjuncts[4]. We use the AdamW optimizer with a learning rate of 2e-5, which is linearly decreased with no warm-up, gradient clip with the threshold of 1.0, and weight decay of 0.01. We evaluate models every 100 steps and apply early stopping when the performance is not improved for 1,000 steps after the initial 1,000 steps. An MLP in Eq. 3 has one hidden layer of 256 units with dropout ($p = 0.5$). We use the last token of each word when computing scores for word-level boundaries of coordination.

[3]In practice, to give sufficient context information for generation, we exclude sentences from U that consist of less than 10 words.

[4]We exclude the root constituent that covers an entire sentence.

Setting	Overall
baseline	64.74
+ generation by BERT	66.98
– filtering	64.43
– sync + self-train	65.77
+ generation by T5	69.38
– filtering	66.13
– sync + self-train	65.37

Table 3: Results of the ablation study conducted on the PTB development set. We report accuracy averaged across five runs.

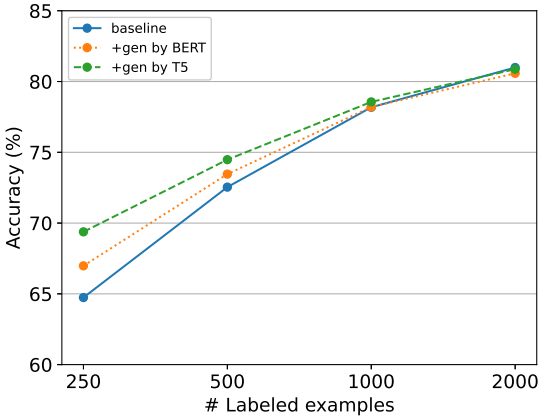

Figure 4: Influence of training dataset size evaluated on the PTB development set.

5.2 Results and analysis

5.2.1 Performance

Table 2 shows the experimental results on the PTB and GENIA test sets. Trained with coordination generation by our proposed method, the coordination disambiguation model generally achieves better performances than the baseline, which is trained only on a limited amount of manually annotated data. This indicates that both of the BERT and T5 models used with the proposed method yield promising examples that provide performance gains for the task on the PTB as well as on the GENIA, although the underlying parser and LMs for generation are not particularly adapted to the biomedical domain. A comparison between different LMs for generation shows that the T5 models produce better-quality examples than those produced by the BERT models for all types of coordination. This is because BERT is pre-trained to predict a single token for each mask independently, whereas T5 is pre-trained through sequence-to-sequence learning to generate consecutive tokens for masked spans. Thus, the BERT models only work when generating conjuncts each consisting of a few tokens; however, this would be mitigated by fine-tuning for span prediction, as used in the training of SpanBERT (Joshi et al., 2020)[6]. Surprisingly, the performances on clause-level coordination (labeled as S or SBAR) are improved by generation with T5, although T5 models are pre-trained to fill relatively small masks and thus the out-of-the-box models should have poor ability to generate text for a long masked span.

[5] *Overall* includes other minor categories, such as UCP and QP.

[6] We intended to use a SpanBERT model as a variant of BERT for generation, but the weights of the LM heads for masked span prediction are not officially provided.

5.2.2 Ablation study

We conduct additional experiments for ablation study on the PTB development set. We report performances of models trained using 250 labeled examples and synthetic examples without filtering by setting $\delta = 0$ or without synchronized decoding for generation. Specifically in the latter setting, we use only one masked sentence where a mask token and conjunction are inserted only after a reference (equivalent to $S'(1)$) and employ an LM to fill a mask without synchronized decoding. However, because the LM has no clue about the boundaries of a reference span in the masked sentence, it likely produces a text that does not seem to be coordinated with the reference (e.g., *"I had lunch and* `<mask>` *with Mary."* $\rightarrow$ *"I [had lunch] and [dinner] with Mary."*). Such an example is easily rejected by filtering during training, which results in that training without synchronization is no longer different from training only on L. Therefore, when not in use of the synchronized decoding, we discard the corresponding boundary annotation y' produced by G for the resulting sentence x' and instead use a predictor model M to assign the most probable endpoint pair y'' to the sentence x'. The example (x', y'') is then added if its score is greater than or equal to the threshold δ. This setting can be seen as typical self-training.

Table 3 shows the results of the ablation study. Without filtering synthetic examples in the training, the performances are substantially degraded, which indicates that the proposed generation method produces erroneous examples, but also that the influence of noisy examples for supervised learning is effectively mitigated by filtering. However, the proposed generation method combined with T5

1.	Accepted	We're sorry to report that on Monday President Bush accepted the resignation of William Allen as chairman [of the U.S. Civil Rights Commission] and [of the U.S. Department of Justice].
2.	Accepted	The opposition charged that the money was used [to bribe Indian government officials] and [to support the Congress], an allegation denied by Mr. Gandhi's administration.
3.	Rejected	"[I wouldn't say it's quite a veto] and [it's not a veto]," Mr. Boren demurs.
4.	Rejected	Most analysts had expected a sharper decline [after the steep rise in August] and [earlier in this year].

Table 4: Synthetic labeled examples generated using the proposed method with T5. The underlined text indicates the reference span used in generation. The first column indicates whether the instance is considered valid or not by the predictor model.

without filtering still leads to performance gain, whereas generation with BERT without filtering slightly impairs the performance. This result supports the superiority of the T5 model over the BERT model for the proposed method. The models self-trained on sentences generated by LMs without the synchronized decoding outperform the baseline, demonstrating the effectiveness of self-training. However, they underperform the models trained with the proposed method that employs the synchronized decoding. This indicates that the boundary annotations assigned by the predictor itself become noises for training and would be amplified as training progresses, whereas the annotations produced by the synchronized decoding are more consistent regardless of the training progress.

5.2.3 Influence of the size of labeled examples

We further investigate the performance improvement for the cases in which the size of human-annotated labeled examples is increased. We sample 250, 500, 1,000, and 2,000 labeled examples from the PTB training set to construct L and report performances on the PTB development set. Figure 4 shows results. The models trained with the proposed generation method with BERT and T5 enhance performances for low-resource settings, as demonstrated in Table 2. However, the improvement is no longer observed when preparing sufficient labeled examples. Note that when using the full training set (15,481 sentences) for L, the baseline model and the model with generation using T5 achieve 86.36% and 86.24% accuracy on the PTB development set, respectively. Thus, employing the proposed method with many hand-annotated examples does not degrade the performance significantly.

5.2.4 Qualitative analysis

Table 4 shows synthetic examples generated by the proposed method using the T5 model. For Examples 1 and 2, the LM successfully captures the boundary of the prepositional phrase and the verb phrase, respectively, and generates the texts that are coordinated with the reference spans with the help of the proposed masking strategy and the synchronized decoding. For Example 3, the model produces a clause that corresponds to the reference, but the resulting boundary annotation does not appear to be a valid conjunct for the reference span due to the generator model's misunderstanding of the coordination boundary. This malformed example is effectively rejected for training by the predictor model. The adverbial coordinate structure in Example 4 generated by the LM seems to have the correct boundary, but is mistakenly rejected because the predictor model is not sure whether it is correct or not.

6 Limitations

The main weakness of the proposed approach is its dependence on a pre-trained large LM, which may not be available in an extremely low-resource setting in terms of the amount of unlabeled text or computational resources. Furthermore, the effectiveness of our method has been examined for gen-

erating a limited form of coordination in English (i.e., two conjuncts joined by *"and"*). Thus, additional treatments for other forms of coordination in English or for coordination in other languages using multilingual LMs would need to be developed and tested with more thorough qualitative analyses through human evaluation.

7 Conclusion

We present a method that exploits pre-trained LMs for generating coordination with boundary annotation. Synchronized decoding by BERT and T5 for two masked sentences with conjunctions leads to coordinate structures that are consistent with synthetic annotation. The experimental results for coordination disambiguation show that examples generated by our method provide gains to supervised methods. The extensive study also supports the effectiveness of the proposed training framework for removing erroneous examples. As future work, we intend to develop a fine-tuning method on LMs specialized for coordination generation and apply synchronized decoding to other LMs and to other generation tasks.

Acknowledgments

This study was partly supported by JSPS KAKENHI Grant Number 22K17957. We are grateful to the anonymous reviewers for their helpful insights and comments.

References

Rajeev Agarwal and Lois Boggess. 1992. A simple but useful approach to conjunct identification. In *30th Annual Meeting of the Association for Computational Linguistics*, pages 15–21, Newark, Delaware, USA. Association for Computational Linguistics.

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do not have enough data? deep learning to the rescue! *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7383–7390.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Ekaterina Buyko, Katrin Tomanek, and Udo Hahn. 2007. Resolution of coordination ellipses in biological named entities using conditional random fields. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, pages 163–171.

Jeongmin Chae, Younghee Jung, Taemin Lee, Soonyoung Jung, Chan Huh, Gilhan Kim, Hyeoncheol Kim, and Heungbum Oh. 2014. Identifying non-elliptical entity mentions in a coordinated np with ellipses. *Journal of Biomedical Informatics*, 47:139–152.

Francis Chantree, Adam Kilgarriff, Anne. de Roeck, and Alistair Willis. 2005. Disambiguating coordinations using word distribution information. In *Proceedings of Recent Advances in Natural Language Processing*, pages 287–294.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jessica Ficler and Yoav Goldberg. 2016a. Coordination annotation extension in the Penn Tree Bank. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 834–842, Berlin, Germany. Association for Computational Linguistics.

Jessica Ficler and Yoav Goldberg. 2016b. A neural network for coordination boundary prediction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 23–32, Austin, Texas. Association for Computational Linguistics.

Jessica Ficler and Yoav Goldberg. 2017. Improving a strong neural parser with conjunction-specific features. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 343–348, Valencia, Spain. Association for Computational Linguistics.

Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. 2019. Soft contextual data augmentation for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5539–5544, Florence, Italy. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Kazuo Hara, Masashi Shimbo, Hideharu Okuma, and Yuji Matsumoto. 2009. Coordinate structure analysis with global structural constraints and alignment-based local features. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 967–975, Suntec, Singapore. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual constituency parsing with self-attention and pre-training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3499–3505, Florence, Italy. Association for Computational Linguistics.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457, New Orleans, Louisiana. Association for Computational Linguistics.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26, Suzhou, China. Association for Computational Linguistics.

Sadao Kurohashi and Makoto Nagao. 1992. Dynamic programming method for analyzing conjunctive structures in Japanese. In *COLING 1992 Volume 1: The 14th International Conference on Computational Linguistics*.

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. Association for Computational Linguistics.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, New York City, USA. Association for Computational Linguistics.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Philip Resnik. 1999. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11(1):95–130.

Gözde Gül Şahin and Mark Steedman. 2018. Data augmentation via dependency tree morphing for low-resource languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5004–5009, Brussels, Belgium. Association for Computational Linguistics.

Haoyue Shi, Karen Livescu, and Kevin Gimpel. 2021. Substructure substitution: Structured data augmentation for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3494–3508, Online. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Yuka Tateisi, Akane Yakushiji, Tomoko Ohta, and Jun'ichi Tsujii. 2005. Syntax annotation for the GENIA corpus. In *Companion Volume to the Proceedings of Conference including Posters/Demos and tutorial abstracts*.

Hiroki Teranishi, Hiroyuki Shindo, and Yuji Matsumoto. 2017. Coordination boundary identification with similarity and replaceability. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 264–272, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Hiroki Teranishi, Hiroyuki Shindo, and Yuji Matsumoto. 2019. Decomposed local models for coordinate structure parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for*

Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 3394–3403, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In *Computational Science – ICCS 2019*, pages 84–95. Springer International Publishing.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Jiajun Zhang and Chengqing Zong. 2016. Exploiting source-side monolingual data in neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545, Austin, Texas. Association for Computational Linguistics.