Constrained Labeled Data Generation for Low-Resource Named Entity Recognition

Ruohao Guo
University of Illinois Urbana-Champaign
ruohaog2@illinois.edu

Dan Roth
University of Pennsylvania
danroth@seas.upenn.edu

Abstract

Named Entity Recognition (NER) in low-resource languages has been a long-standing challenge in NLP. Recent work has shown great progress in two directions: developing cross-lingual features/models to transfer knowledge to low-resource languages, and translating source-language training data into low-resource target-language training data by projecting annotations with cheap resources. We focus on the second direction in this study. Existing methods suffer from the low quality of the resulting annotated data in the target language; for example, they cannot handle word order and lexical ambiguity well. To handle these limitations we propose a novel approach that uses the projected annotation to generate pseudo supervised data with a transformer language model and a constrained beam search. This allows us to generate more diverse, higher quality, as well as higher quantities of annotated data in the target language. Experiments demonstrate that, when combining our method with available cross-lingual features, it achieves state-of-the-art or competitive performance on NER in a low-resource setting, especially for languages that are distant from our source language, English. [1]

1 Introduction

Named entity recognition (NER), the task of finding and classifying named entities in text, has been a mature topic in natural language processing (NLP). However, its success is highly dependent on the amount and quality of annotated data. For most of the world's languages, the amount of supervised resource is limited. How to develop a good NER system with little to no annotated data has become a challenging problem.

[1]The code and data of the paper are available at: http://cogcomp.org/page/publication_view/945

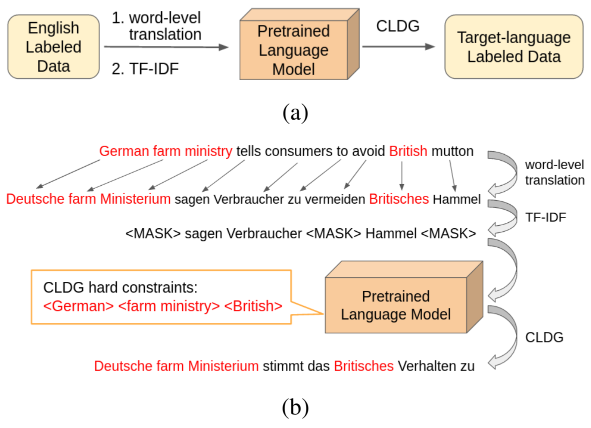
English Labeled Data
1. word-level translation
2. TF-IDF
Pretrained Language Model
CLDG
Target-language Labeled Data
(a)
German farm ministry tells consumers to avoid British mutton
word-level translation
Deutsche farm Ministerium sagen Verbraucher zu vermeiden Britisches Hammel
TF-IDF
<MASK> sagen Verbraucher <MASK> Hammel <MASK>
CLDG hard constraints:
<German> <farm ministry> <British>
Pretrained Language Model
CLDG
Deutsche farm Ministerium stimmt das Britisches Verhalten zu
(b)

Figure 1: (a) A pipeline of our data generation system; (b) An English-to-German example. An NER annotated English sentence at the top as the input produces (multiple) NER annotated German sentence(s) at the bottom. Red words are labeled named entities. Our generation method is denoted by CLDG (see §3.3 for details).

To address this challenge in low-resource NER, recent works study the benefits of weakly- or partially-annotated data (Dehghani et al., 2018; Mayhew et al., 2019), and that of transferring knowledge from the high-resource languages to the low-resource languages. Common corpora for developing cross-linguality include parallel text (Wang and Manning, 2014; Ni and Florian, 2016), Wikipedia (Nothman et al., 2013; Pan et al., 2017), and multilingual dictionaries or gazetteers (Tsai et al., 2016). However, the effectiveness of these approaches depends on the quality and quantity of data. For example, parallel text in some low-resource languages is unavailable and the dictionary size is usually smaller; there are 295 languages in Wikipedia, but most of them are too sparse to be useful. Mayhew et al. (2017) and Xie et al. (2018) employed phrase-level and word-level translation respectively to produce target-language training data by projecting annotations. Xie et al. (2018) also tried to alleviate word order divergence across languages by adding self-attention layers. However, this only makes the NER classifier insensitive to word order and the benefits of order information are still ignored.

In this study, we propose Constrained Labeled Data Generation (CLDG), a method that generates pseudo labeled data in low-resource languages with only cheap resources: a dictionary and unannotated text in the target language. Fig. 1 illustrates the pipeline of our labeled data generation system. We first translate high-resource labeled sentences to the target language word-by-word with a dictionary. Next, we construct target-language text from the source-language named entities with a pretrained language model. We introduce a decoding strategy with declarative constraints ($i.e.$ **hard constraints**) to ensure the presence of entities in the generated text.

By constructing data artificially this way, we get sentences with the projected annotated entities, and with more natural, contextually correct, word order. Moreover, multiple annotated target language sentences can be generated with our method from a given annotated sentence in English. To the best of our knowledge, this work is the first to artificially generate labeled data via constrained text generation. Our method improves the current state-of-the-art results on NER across several low-resource languages. Since our approach generates pseudo data from the labeled source-language tokens, it can potentially generalize to other cross-lingual NLP tasks.

2 Related Work

2.1 Cross-lingual NLP

There are two main approaches to cross-lingual learning: parallel projection, and developing language-independent features. The first approach obtains pseudo labeled target-language data by projecting annotations from the source to the target using a parallel corpus. A model is then trained in the target language. It has been applied to many tasks, such as part-of-speech tagging (Fang and Cohn, 2016; Das and Petrov, 2011), NER (Wang and Manning, 2014; Mayhew et al., 2017) and parsing (McDonald et al., 2011). The second method attempts to learn language-independent features with which a model trained in the source can transfer directly to the target language. For example, Tsai et al. (2016) developed cross-lingual features from inter-language links in Wikipedia. Multilingual BERT (Devlin et al., 2019a; Pires et al., 2019) is trained on 104 languages and it can provide powerful cross-lingual contextual representations for many tasks.

2.2 Transformers for Text Generation

Self-supervised learning has achieved remarkable success in a wide range of NLP tasks (Vaswani et al., 2017; Peters et al., 2018; Devlin et al., 2019b). Pourdamghani et al. (2019) apply transformers to unsupervised machine translation, but it is hard to align named entities in the translated sentences. In terms of text generation, transformer-based models like GPT (Radford et al., 2019; Brown et al., 2020) have shown great potential. These models are pretrained on the large unsupervised corpora crawled from the web. BART (Lewis et al., 2020) proposes to learn a model by reconstructing the input corrupted by an arbitrary operation ($e.g.$ token masking, token deletion, text infilling, etc.). It is particularly effective in text generation. T5 (Raffel et al., 2020) improves transfer learning by reformulating all tasks into a unified "text-to-text" format. It achieves state-of-the-art results on benchmarks, such as summarization.

To overcome the challenge of generating coherent long text, ProGeT (Tan et al., 2020) first produces a sequence of informative words and then progressively adds tokens until completing a full passage. It evaluates word importance with TF-IDF metric. In our experiments, we use this method to select input to the language model. Unlike ProGeT which generates sequences in multiple stages, we complete the text at one time.

2.3 Constrained Text Generation

Constrained text generation aims to decode sentences with expected attributes such as topics (Feng et al., 2018), style (Luo et al., 2019), etc. In this work, we focus on hard constraints.

MaskGAN (Fedus et al., 2018) fills in missing text conditioned on context. It can be used for hard-constrained generation by masking non-constraint words, but the constraints have fixed positions in text. Insertion Transformer (Stern et al., 2019) solves this issue by inserting tokens between lexical constraints iteratively. To consider all the valid hard-constrained generation, it has to permute the constraints ordering.

Model	TF-IDF threshold			
	25%		50%	
T5-small	0.30	0.54	0.28	0.58
BART-base	0.27	0.47	0.24	0.72

Table 1: Validation loss (epoch 65) of the models on 5k English sentences. For each model with a TF-IDF, the right cell indicates training with noise.

Grid beam search (GBS) (Hokamp and Liu, 2017) offers another solution to the problem by extending beam search and applying hard constraints that allow word insertion and permutation. Fig. 3 shows a visualization of GBS. The vertical axis represents completed constraints, and the horizontal axis indicates the output sequence, including constrained and unconstrained tokens. At each step, each hypothesis produces candidates in two directions: generating a word from the model distribution, or completing a constraint. Then it selects the top k candidates as the next hypotheses to continue. Dynamic Beam Allocation was proposed to improve the speed of constrained decoding (Post and Vilar, 2018). In this paper, we extend GBS to allow the source-language constraints for text generation in the target language.

3 Algorithm

Problem Setting. Our objective is to generate hard-constrained annotated data of higher quality and larger quantity in the target language from a source language (*e.g.* English) in an unsupervised way. In this work, we limit ourselves to a setting where only the following resources are available:

- Monolingual corpora in the target language.
- A dictionary from the source to the target.
- NER training data in the source language.

Our **data generation pipeline** consists of the following steps:

1. Word-by-word translation from the NER training data in the source language to the target languages (§3.1).
2. Taking the important translated words as input, a pretrained transformer model is used to generate the target-language NER training data. The model is pretrained from scratch with data extracted from Wikipedia (§3.2).
3. Hard constraints are applied to the generation to include the named entities with their labels (§3.3).

3.1 Word-level Translation

We adopt Cheap Translation (Mayhew et al., 2017) or Bilingual Word Embedding Translation (Xie et al., 2018) to translate training data from the source language into the target language word-by-word with a dictionary.

3.2 Pretraining Language Models

To reduce the noise introduced by wrong word-level translation, we only take the important words as input to the generation model. The vocabulary is sorted with TF-IDF scores, and only a small proportion of words with higher scores (*e.g.* 25%, is defined as the TF-IDF threshold in Table 1) are kept as the input. We extract text in the target language from Wikipedia as the training data, and train the model with the objective of reconstructing full text from important words and phrases. The selection of important words is also based on the TF-IDF scores.

In this work, we experiment with BART and T5 provided by HuggingFace (Wolf et al., 2020) for target-language model pretraining, though our method can use other off-the-shelf generative language models. Since BART and T5 are transformers with both the encoder and the decoder, text is conditionally generated from the bidirectional context.

During training, the model predicts the next token conditioned on the previous words sampled from the ground-truth data distribution. During generation, however, the model generates words conditioned on its previous imperfect prediction. Since the model has never seen such noisy input, its performance would degrade, and this training-generation discrepancy would accumulate along the generation sequence. This problem is referred to as "exposure bias" (Ranzato et al., 2016). To alleviate this issue and increase the robustness of the language model, we add noise to the gold data during training, by randomly replacing 10% of input words with others in the sentence.

We train the model on 100k English sentences and evaluate it on 5k sentences to select the best model as well as the TF-IDF threshold. The experimental results in Table 1 show BART to be the most suitable one, and that 25% gives the best performance and covers most important words in a

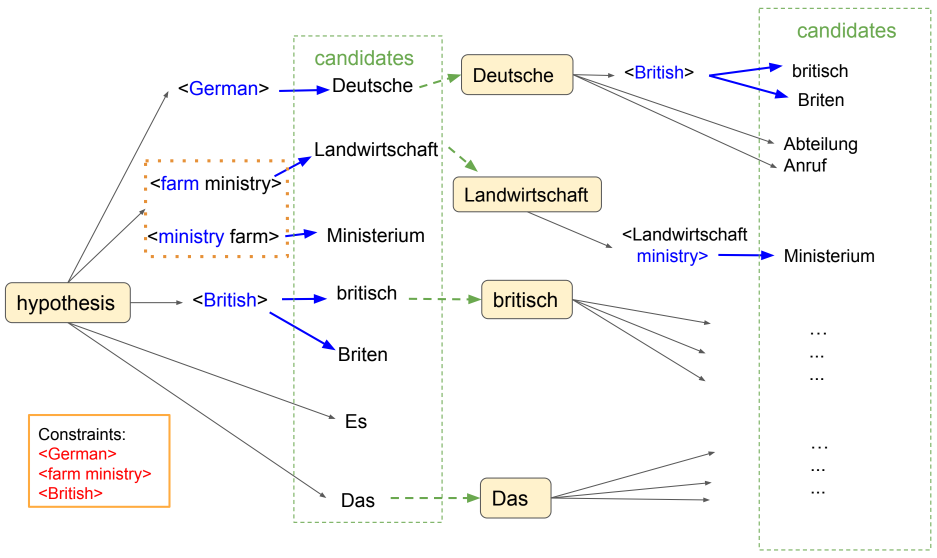

Figure 2: A visualization of CLDG with the example from Fig. 1. Yellow rectangles represent hypotheses, blue arrows are translating blue words from the source language to the target language, the green blocks represent candidates, and green arrows show the selected candidates (beam size = 4) for the next hypotheses. "<farm ministry>" is an example of a phrase-level constraint. "Landwirtschaft" is translated from "farm". Since it is selected as one candidate, it closes the hypothesis, and therefore its next token must finish the current constraint, *i.e.* the next candidate must be the translation from "ministry". "<British>" shows an example of multiple translations.

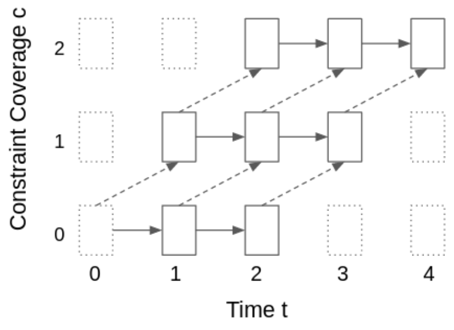

Figure 3: Illustration of GBS (Hokamp and Liu, 2017). The vertical axis c indicates coverage of hard constraints. Each rectangle represents a beam containing k hypotheses. Dashed arrows start or continue a constraint based on whether the current constraint is finished or not, while solid arrows generate new words. Beams on the top layer contain candidates covering all the constraints.

sentence. Therefore in §4, we pretrain BART-base with a threshold 25%.

3.3 Hard-constrained Generation

Transformer language models can produce any word which may or may not be in the input. To ensure the presence of source-entities in the generated text, we extend GBS to **CLDG** (**C**onstrained **L**abeled **D**ata **G**eneration). Fig. 2 illustrates our constrained decoding process. See the Appendix for its **pseudo-code**. The constraints are the named entities in the source language; they are first translated in the target language (with their labels) and CLDG *constrains* the output sequence to include them in the output sequence.

We use the coordinate system in Fig. 3. In each grid (t, c), candidates of new tokens are produced by generating all the possible tokens from previous hypotheses in grid $(t-1, c)$, and choosing one token for each constraint of each hypothesis in grid $(t-1, c-1)$. Once we start a phrase-level constraint, we close hypotheses and only choose the next token for the current incomplete constraint. Then we select candidates with the top-k (k is beam size) scores as hypotheses for the current grid. Since constraints are named entities in the source language, we use a dictionary to translate them into the target language in GBS decoding.

Unlike the original GBS with constraints in the target language, CLDG produces hypotheses from multi-translations with different token lengths for each constraint in the source language. *i.e.* when decoding one sample, the number of beam nodes along the vertical direction in Fig. 3 varies for different hypothesis paths. This is due to multiple translation choices.

With open-ended GBS generation that uses top-k candidates selecting strategy, the model tends to generate similar text when constraints and input are the same. Besides, it also suffers from issues like repetitive generation, etc. The sampling method (Holtzman et al., 2019) can be used to address the above mentioned problems. However, when candidates from constraints and non-constraints are put together for sampling, the tokens in the constraints have little chance to be selected. This tends to construct sentences with most entities appearing in the end. We modify GBS to select hypotheses from both constraints candidates and new generation candidates, separately and evenly. For the constraints we use top-k beam search to pick candidates, while for produced tokens we sample the top-k hypotheses among candidates of beam search, which better solves this problem and also gives more diverse data when decoding multiple times.

Another potential problem in our method is the unintended introduction of new named entities into the generation process, as only translated named entities have labels. This would degrade the quality of pseudo labeled data, thus leading to a low NER recall. To cope with this issue, we adopt the following methods: (1) We restrict the number of new unconstrained tokens to be less than a parameter $max\_unconstrained$. Once the number of unconstrained tokens hits the maximum bound, only constraints are considered in subsequent decoding. (2) We use a naive NER predictor trained on previously-produced data to detect and relabel the added entities. Experiments show this can effectively improve the model performance.

3.4 Implementation Details of CLDG

Lexical Ambiguity. We tackle the problem of multiple translations by allowing multiple candidate tokens, each for one translation. The language model will choose the better candidates among all candidates to continue its generation. In many cases, one entry has too many (>35) translations, which would lead to poor generation quality if we consider all of them. To handle this problem, we consider a subset of frequently-occurred translations (Mayhew et al., 2017).

Word Order. We address the problem of word order in two levels in the decoding: (1) The global phrase order in a sentence; (2) The local word order within phrases. When there is no phrase-level translation in the dictionary, we first translate word by word. Then we reorder and select the most appropriate one based on the language model. For example, when translating the organization "University of XXX" from English to Chinese, a word-to-word translation would be "大学 XXX", which does not fit the Chinese grammar. With our method, however, we can get text with the correct order "XXX 大学". Previous works (Mayhew et al., 2017; Xie et al., 2018) tried to alleviate the word order issue by translating data between similar languages. However, there is no such limitation for CLDG. We can always start from English as there is much more labeled data in English.

Language	Language Code	Number of Sentences
German	de	293K
Spanish	es	541K
Dutch	nl	519K
†Akan	ak	287K
Arabic	ar	554K
Turkish	tr	579K
†Wolof	wo	75K
†Yoruba	yo	286K
Uzbek	uz	267K

Table 2: Wikipedia statistics for pretraining (§3.2). † represents using the entire Wikipedia and augmenting with the text provided by the LORELEI project's data. For example, Wolof Wikipedia contains only 17K sentences and we enlarge it to 75K.

Generating More Data. We aim to generate multiple labeled sentences in the target language for each source sentence. Our experiments show that by training on a combination of generated data, the model performs better.

4 Experiments

We generate target-language annotated data via the pipeline in §3. Then we train an NER model on the generated data. We use the standard BiLSTM-CRF architecture (Ma and Hovy, 2016) with an AllenNLP implementation (Gardner et al., 2018).

4.1 Datasets

We evaluate our method on the benchmark CoNLL 2002 and 2003 NER datasets (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003) which contain 4 languages: English, Spanish, German, Dutch. Previous study shows that English is closely related to the above European languages in terms of word order (Mayhew et al., 2017). Hence, in order to demonstrate the advantages of our method, we add several languages that are dissimilar to English: Akan, Arabic, Turkish, Uzbek, Wolof, Yoruba. We evaluate their performances on the LORELEI project's data (Strassel and Tracey, 2016). Among the 9 languages we evaluate, Wolof

Language	Dutch	German	Spanish	Akan	Arabic	Turkish	Wolof	Yoruba	Uzbek
Dict Size	961K	1.36M	1.25M	15K	410K	630K	5K	465K	65K

Table 3: Dictionary size for each language in CT method.

Method	Extra Resources
CT	MASTERLEXES dictionary
BWET	fastText embeddings, 1.5K word pairs in MASTERLEXES dictionary

Table 4: Resources used in each method.

and Yoruba are truly low-resource languages, and for the other languages, we limit the resources used in order to mimic a truly low-resource scenario.

In all the experiments, we choose English CoNLL train set as the source and generate training data in the target language. CoNLL has 4 named entities labels PER, LOC, ORG, MISC, while LORELEI contains PER, LOC, ORG, GPE. To address this mismatch, we manually changed some MISC and LOC labels in CoNLL to GPE.

4.2 Compared Methods

We experiment with different methods as described below. Resources used for each approach are reported in Table 4. All the methods are evaluated on the same NER model with multilingual BERT (Devlin et al., 2019a), hereafter mBERT, as word embeddings. For each experiment, we run 5 times using different seeds and report the mean and standard deviation (Reimers and Gurevych, 2017).

1. **Zero-shot Learning** We train an NER model on English CoNLL data, and directly evaluate it on the target language.

2. **Cheap Translation (CT)** Cheap translation (Mayhew et al., 2017) translates labeled data with MASTERLEXES dictionaries (Rolston and Kirchhoff, 2016). Prior study shows that a larger dictionary has a better chance of covering valuable entries for NER, such as context words of named entities (Mayhew et al., 2017). Since dictionaries for LORELEI languages are much smaller, we augment them with the lexicons provided in the LORELEI project. Dictionary sizes are presented in Table 3.

3. **Bilingual Word Embeddings Translation (BWET)** This approach (Xie et al., 2018) translates annotated source-language data into the target language by inducing a cross-lingual word-level mapping with the fastText embeddings trained on Wikipedia and the MASTERLEXES dictionaries.

4. **Our Method (CLDG)** We follow the procedure described in §3 to produce training data. Table 2 presents statistics of monolingual corpora used for language model pretraining. See §4.3 for detailed description.

5. **Google Translate** Google Translate is used to translate English CoNLL train set into the target language sentence by sentence. We project labels across translations using fast align (Dyer et al., 2013). For languages supported by Google Translate, this serves as an upper bound for the translation quality.

6. **Supervised Learning** We train on the target-language gold data and consider it as an upper bound for the cross-lingual learning.

4.3 Experimental Setup for CLDG

One advantage of our method is that given one labeled English sentence, we are able to generate multiple sentences in the target language with specified named entities and labels. Moreover, we can adjust the extent and the range of reordering in generation according to the characteristics of each language, which means a more coherent ordering in the target context. Aside from named entities, we can also adjust how many additional phrases in the source are regarded as constraints. In one extreme setting, we only include source-language entities to generate open-ended text.

To demonstrate the universality of our method, we first apply one general setting to the generation of all the European languages and all the non-European languages, respectively. Then we fine-tune the generation setting on each language as well as generate more data with different settings to obtain a better result. Results are presented and analyzed in §4.4.

In the general setting for LORELEI languages, we concatenate two sets of data. One is generated without reordering, with translation based on the most frequent source-word pair in the dictionary.

	ar	ak	tr	wo	yo	uz	Avg
Mayhew et al. (2017)	—	—	51.79	—	38.52	—	—
Zero-shot	36.99	32.53	63.25	31.68	43.16	53.97	43.60
CT	41.95±2.87	46.25±1.74	62.26±2.91	37.42±2.21	49.99±4.44	54.80±3.86	48.78
BWET	34.02±3.49	37.03±2.02	62.19±3.07	38.04±1.37	47.51±5.52	53.39±2.02	45.36
CLDG	**44.54**±2.82	**47.32**±1.80	**66.16**±2.77	**40.32**±1.62	**56.02**±1.95	**56.80**±2.14	51.86
Google Translate	47.79±3.00	—	65.56±3.17	—	51.02±5.80	58.61±2.25	—
Supervised	60.88	75.06	81.32	76.69	73.75	80.62	74.72

Table 5: NER F1 scores on LORELEI (low-resource) languages. Akan (ak) and Wolof (wo) are not present in the Wikipedia data used to pretrain mBERT. They are not supported by Google Translate either. All the methods listed in the second and third groups (from "Zero-shot" to "Supervised") are evaluated on the same model described in §4.2. CLDG is significantly better than the other methods on LORELEI.

	de	es	nl	Avg
Mayhew et al. (2017)	57.23	64.10	63.37	61.57
Xie et al. (2018)	57.76	72.37	79.49	69.87
Zero-shot	62.26	75.82	75.61	71.23
CT	67.11	71.25	76.72	71.69
BWET	68.57	76.95	77.09	74.20
CLDG	66.47	**79.27**	78.03	74.59
†Wu and Dredze (2019)	71.10	74.50	79.50	75.03
†Wu et al. (2020)	**73.65**	78.14	**80.98**	77.59
†CLDG	71.44*	77.92	80.58	76.65
Google Translate	68.40	65.7	73.39	69.16
Supervised	80.98	88.19	89.65	86.27

Table 6: NER F1 on CoNLL (high-resource) languages. Methods in the third and fifth groups are evaluated on the model described in §4.2. † denotes freezing the bottom 3 layers of mBERT by following the implementation in Wu et al. (2020). * means adding English data to training. CLDG is competitive with the other methods (see analyses in 4.4.1).

	de	es	nl
Zero-shot	28.72	42.13	39.35
CLDG	50.59	61.18	64.70

Table 7: NER F1 on European languages using BERT.

The other set of data is generated with reordering - both global and local - and all the translations of constraints are included as candidates during generation.

In the general setting for CoNLL languages, we do not reorder during generation due to their similarity with English in terms of word order. Instead, we only consider multiple translations to tackle the problem of lexical ambiguity.

4.4 Results

We compare all the methods for different languages in Table 5 and Table 6. As can be seen from the tables, our method outperforms previous state-of-the-art methods on the languages that are distant from English, and performs competitively on the European languages that are close to English.

	CT	g-CLDG-CT	BWET	g-CLDG-BWET
ar	42.73	47.09	33.44	35.64
ak	46.33	50.28	36.56	45.38
tr	67.06	68.33	66.78	69.33
wo	35.69	39.69	37.24	36.39
yo	44.9	47.85	39.45	41.51
uz	57.22	55.55	52.89	58.18
Avg	48.99	51.47	44.39	47.74

Table 8: NER on different dictionaries (one seed). "g-CLDG-XX" indicates producing training data with the **general setting** described in §4.3 using XX dictionary.

4.4.1 Languages Similar to & Distant from English

Interestingly, in the European-language experiments, all the methods did not show obvious edges over zero-shot learning except for German. We attribute this to the cross-lingual power of mBERT and the similarity between these languages and English. Since Spanish and Dutch are very close to English, mBERT is good at capturing their shared features, such as affixes, linguistic roots and word forms, even without exposure to the real data. These features might be good enough for NER already. Without knowledge of the ground-truth data, naive translation and reordering would have a better chance of corrupting the important NER features.

We verify this by conducting experiments using BERT instead (Table 7). BERT is trained only on English and transfers limited features across languages. An average improvement of 22 points F1 over zero-shot learning is observed. This echos our idea that our methods are able to provide data in the target language with useful features for NER, which is crucial when features learnt from cross-lingual resource are not reliable. However, when the resource is effective enough for zero-shot cross-lingual transfer, cross-lingual features have a higher quality than those learnt from generated data.

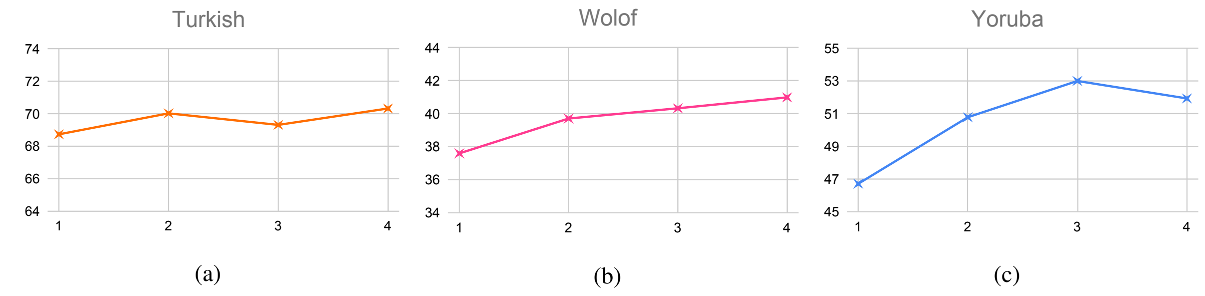

Figure 4: Learning curve of data generation. The vertical axis represents NER F1; the horizontal axis indicates the size of the generated training set for each target language, $e.g.$ $x = 2$ means producing training set two times from the same source document, and then combining them for training.

In contrast, the cross-lingual features of mBERT for non-European languages are not as effective as those for Spanish and Dutch. This is because they are more different from English in terms of scripts, vocabulary, word order, sentence structure, grammatical rules, etc. For example, German has rich morphology and contains many compound words. Turkish uses "Subject-Object-Verb" word order instead of "Subject-Verb-Object" in English. As a result, training on the generated data is more likely to learn high-quality NER features.

Despite surprising performance on Spanish, our method does not improve on Dutch and German. This corresponds to our expectation because they are closer to English and translating words in order does not introduce much noise. Wu et al. (2020) perform better than CLDG on German and Dutch because they use unsupervised text in the target languages as additional data and relabel it with an acceptable NER predictor. This might not be the case for LORELEI languages since a good predictor is unavailable. In contrast, CLDG can work much better in low-resource languages. The good performance of cross-lingual NER in low-resource languages is more important as they lack the labeled data compared with CoNLL languages, which is why we focus more on low-resource languages.

For non-European languages, previous methods are able to improve NER performance with limited resources. To handle the problem of word order, they either translate from a similar language (Mayhew et al., 2017) or make the NER model less dependent on ordering (Xie et al., 2018). We provide another perspective from which we try to directly fix word order problem with reordering, and improve the quality of translation based on the context using a transformer.

4.4.2 Generation Settings Ablation

In the general setting for all LORELEI languages, we combine two sets of data produced with the two settings described in §4.3, in order to avoid overfitting the generated training data. Observing an average improvement of 2.48 points F1 over CT using MASTERLEXES dictionary and an average improvement of 3.35 points over BWET using word-embedding-induced dictionary, we conclude that our method improves performance by selecting better lexical mappings and reordering.

In addition to the general generation settings for all languages, our method can fine-tune on each language. Take Yoruba as an example. Yoruba is a West African language spoken by around 50 million people. It is very under-resourced. Even Yoruba Wikipedia contains only about 66K sentences. For Yoruba, by training on the data generated with a fine-tuned setting ($i.e.$ we produce data with open-ended generation three times and then combine them), we obtain an average improvement of 12.86 over zero-shot learning and an average improvement of 6.03 points over CT (see Table 5). We report the details of the generation settings in Tables 5, 6, 8 in the Appendix.

4.4.3 Generation Size Ablation

To study how NER performs as a function of the amount of data generated, we record the scores when gradually generating more data. Fig. 4 shows that generally the more data we produce, the better NER can be. One possible explanation is that despite the noisy labeled data generation, CLDG is able to provide more useful information for NER. However, when the amount of data achieves an upper bound – usually this upper bound is 3 or 4 according to experiments – the noise may overtake the beneficial signals and thus corrupt the performance.

4.4.4 Dictionary Ablation

By comparing the results using different dictionaries (Table 8), we observe that the performance of our method depends on the dictionary quality. For example, in Akan, BWET performs much worse than CT. Although our method is able to beat BWET with a margin of 8.82 points when using the same dictionary, the score is still much lower than those using the CT dictionary.

Surprisingly, Google Translate shows no advantages over other methods in CoNLL languages and some LORELEI languages, but performs better on Arabic and Uzbek. There are several reasons. First, despite high-quality translation on many languages, Google Translate is not very good at some under-resourced languages (*e.g.* Yoruba). Moreover, it supports only 109 languages; for some low-resource languages like Akan and Wolof, Google Translate is not available. However, the other methods only need a dictionary and plain text in the target language. Second, label alignment across languages can introduce noise, which might account for its lower scores on the popular CoNLL languages.

5 Conclusion and Discussion

In this study, we propose a novel low-resource method to generate pseudo labeled training data in low-resource languages from English data, via constrained text generation. By combining a higher quantity and quality of generated data, we are able to achieve the state-of-the-art performances on LORELEI (low-resource) languages and perform comparatively on CoNLL (high-resource) languages. Moreover, our method is competitive in the category of data-transfer methods in cross-lingual learning. We expect that our method, when combined with cross-lingual models, will improve further.

Acknowledgments

This work was partly supported by Contracts FA8750-19-2-0201 and HR0011-18-2-0052 with the US Defense Advanced Research Projects Agency (DARPA). The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher 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.

Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 600–609, Portland, Oregon, USA. Association for Computational Linguistics.

Mostafa Dehghani, Arash Mehrjou, Stephan Gouws, Jaap Kamps, and Bernhard Schölkopf. 2018. Fidelity-weighted learning.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. Bert: Pre-training of deep bidirectional transformers for language understanding.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019b. 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.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Meng Fang and Trevor Cohn. 2016. Learning when to trust distant supervision: An application to low-resource POS tagging using cross-lingual projection. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 178–186, Berlin, Germany. Association for Computational Linguistics.

William Fedus, Ian Goodfellow, and Andrew M. Dai. 2018. Maskgan: Better text generation via filling in the .

X. Feng, Ming Liu, J. Liu, B. Qin, Yibo Sun, and T. Liu. 2018. Topic-to-essay generation with neural networks. In *IJCAI*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546, Vancouver, Canada. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *CoRR*, abs/1904.09751.

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.

Fuli Luo, Peng Li, Jie Zhou, Pengcheng Yang, Baobao Chang, Zhifang Sui, and Xu Sun. 2019. A dual reinforcement learning framework for unsupervised text style transfer.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Stephen Mayhew, Snigdha Chaturvedi, Chen-Tse Tsai, and Dan Roth. 2019. Named entity recognition with partially annotated training data. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 645–655, Hong Kong, China. Association for Computational Linguistics.

Stephen Mayhew, Chen-Tse Tsai, and Dan Roth. 2017. Cheap translation for cross-lingual named entity recognition. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2536–2545, Copenhagen, Denmark. Association for Computational Linguistics.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 62–72, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Jian Ni and Radu Florian. 2016. Improving multilingual named entity recognition with Wikipedia entity type mapping. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1275–1284, Austin, Texas. Association for Computational Linguistics.

Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy, and James R. Curran. 2013. Learning multilingual named entity recognition from wikipedia. *Artificial Intelligence*, 194:151 – 175. Artificial Intelligence, Wikipedia and Semi-Structured Resources.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

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, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324, New Orleans, Louisiana. Association for Computational Linguistics.

Nima Pourdamghani, Nada Aldarrab, Marjan Ghazvininejad, Kevin Knight, and Jonathan May. 2019. Translating translationese: A two-step approach to unsupervised machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3057–3062, Florence, Italy. Association for Computational Linguistics.

A. Radford, Jeffrey Wu, R. 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.

Marc'Aurelio Ranzato, S. Chopra, M. Auli, and W. Zaremba. 2016. Sequence level training with recurrent neural networks. *CoRR*, abs/1511.06732.

Nils Reimers and Iryna Gurevych. 2017. Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 338–348, Copenhagen, Denmark. Association for Computational Linguistics.

Leanne Rolston and Katrin Kirchhoff. 2016. Collection of bilingual data for lexicon transfer learning. Technical report, University of Washington.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. Insertion transformer: Flexible sequence generation via insertion operations.

Stephanie Strassel and Jennifer Tracey. 2016. LORELEI language packs: Data, tools, and resources for technology development in low resource languages. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3273–3280, Portorož, Slovenia. European Language Resources Association (ELRA).

Bowen Tan, Zichao Yang, Maruan AI-Shedivat, Eric P. Xing, and Zhiting Hu. 2020. Progressive generation of long text.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Chen-Tse Tsai, Stephen Mayhew, and Dan Roth. 2016. Cross-lingual named entity recognition via wikification. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 219–228, Berlin, Germany. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Mengqiu Wang and Christopher D. Manning. 2014. Cross-lingual projected expectation regularization for weakly supervised learning. *Transactions of the Association for Computational Linguistics*, 2:55–66.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Qianhui Wu, Zijia Lin, Börje F. Karlsson, Biqing Huang, and Jianguang Lou. 2020. Unitrans : Unifying model transfer and data transfer for cross-lingual named entity recognition with unlabeled data. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3926–3932. ijcai.org.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. 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 833–844, Hong Kong, China. Association for Computational Linguistics.

Jiateng Xie, Zhilin Yang, Graham Neubig, Noah A. Smith, and Jaime Carbonell. 2018. Neural cross-lingual named entity recognition with minimal resources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 369–379, Brussels, Belgium. Association for Computational Linguistics.

A Dataset Statistics

We report the dataset statistics of our supervised learning experiments (see Tables 5 and 6) below.

Language	Number of Words (k)
ak	67.3
ar	55.9
tr	61.9
wo	67.3
yo	57.3
uz	12.7
de	207.5
es	264.7
nl	202.9

Table 9: Sizes of the gold training sets used for supervised learning experiments.

B Pseudo-code of CLDG

The pseudo-code of CLDG algorithm is described in the below Algorithm table.

C Generation Settings

In this section, we report different generation settings in Table 11. Notations of parameters are described in Table 10.

Algorithm: Pseudocode for CLDG

```
CONSTRAINED_SEARCH(model, input, srcConstraints, max_unconstraint_len, k)
Grid[:][:] ⇐ ∅
start_hyp.token ⇐ model.bos_token_id
start_hyp.constraint.srcConstraints ⇐ srcConstraints
start_hyp.constraint.phraseCloseWordList ⇐ ∅
start_hyp.constraint.closeTokenList ⇐ ∅
Grid[0][0] = start_hyp
$t = 1$
foreach $t < maxLen$ do
    foreach $c < maxC$ do
        $gCands, cCands = \varnothing$
        foreach $hyp \in$ Grid$[t-1][c]$ do
            if $hyp.constraint.isOpen()$ and $hyp.ucpathLen < max\_unconstraint\_len$ and
              notEOS(hyp.token) then
                cands ⇐ model.generate(hyp,input)
                cands[:].ucpathLen ⇐ hyp.ucpathLen+1
                gCands ⇐ gCands⋃ cands // generate new open cands
        if $c > 0$ then
            foreach $hyp \in$ Grid$[t-1][c-1]$ do
                cands ⇐ hyp.constraint.generateCands()
                cCands ⇐ cCands ⋃ cands
        if $do\_sample$ then
            cands_for_sampling ⇐ n-argmax$_{h \in gCands}$ model.score(h)
            k-sampled_cands ⇐ score_weighted_sampling(cands_for_sampling)
            Grid[t][c] ⇐ k-sampled_cands ⋃ k-argmax$_{h \in cCands}$ model.score(h)
        else
            Grid[t][c] ⇐ k-argmax$_{h \in gCands \bigcup cCands}$ model.score(h)
finishedHyps ⇐ ∅
foreach $hyp \in$ Grid[:][:] do
    if hyp.constraint.isDone() and (isEOS(hyp.token) or hyp.ucpathLen = max_unconstraint_len) then
        finishedHyps ⇐ finishedHyps ⋃ hyp
$bestHyp$ ⇐ argmax$_{h \in finishedHyps}$ model.score(h)
return $bestHyp$
// continue on the next page
```

Algorithm: Pseudocode for CLDG

```
FUNCTION constraint.generateCands():
cands ⇐ ∅
if isOpen() then
    foreach phrase ∈ $srcConstraints$ do
        if keep_order and index(phrase)>0 then
            break
        phraseCloseWordList ⇐ phrase
        new_cands ⇐generate_cands(phraseCloseWordList)
        new_cands[:].constraint.srcConstraints ⇐ srcConstraints\phrase
        cands ⇐ cands $\bigcup$ new_cands
else
    if closeTokenList $\neq$ ∅ then
        new_cand.token ⇐ closeTokenList[0]
        new_cand.constraint ⇐ clone(constraint)
        new_cand.constraint.closeTokenList ⇐ closeTokenList\new_cand.token
        cands ⇐ cands $\bigcup$ new_cand
    else
        if phraseCloseWordList $\neq$ ∅ then
            new_cands ⇐generate_cands(phraseCloseWordList)
            cands ⇐ cands $\bigcup$ new_cands
return cands

FUNCTION generate_cands(phraseCloseWordList):
cands ⇐ ∅
foreach word ∈ $phraseCloseWordList$ do
    if ph_keep_order and index(word) > 0 then
        break
    trgTokenSeqList ⇐ get_token_list_from_dict(word)
    foreach tokenSeq ∈ $trgTokenSeqList$ do
        new_cand.token ⇐ tokenSeq[0]
        new_cand.constraint ⇐ clone(constraint)
        new_cand.constraint.closeTokenList ⇐tokenSeq[1:]
        new_cand.constraint.phraseCloseWordList ⇐ phraseCloseWordList\word
        cands ⇐ cands $\bigcup$ new_cands
return cands

FUNCTION get_token_list_from_dict(word):
trgTokenSeqList ⇐ ∅
foreach trg_word ∈ $dict[word]$ do
    if prominence(trg_word) ≥ top_th then
        trgTokenSeq ⇐ model.tokenize(trg_word)
        trgTokenSeqList ⇐ trgTokenSeqList $\bigcup$ trgTokenSeq
return trgTokenSeqList

FUNCTION constraint.isOpen():
if phraseCloseWordList = ∅ and closeTokenList = ∅ then
    return True
else
    return False
```

notation	description	default value
$dict$	which dictionary is used	
$max\_unconstraint\_len$	the maximum number of unconstrained tokens to generate	
do_sample	whether or not sampling is used in generation	false
$keep\_order$	w/wo global reorder	false
ph_keep_order	w/wo local reorder within phrase	false
top_th	prominence threshold for multiple translation; 0 for all translation choices	0
num_beam and k	beam search size	
$srcConstraints$	phrase or word constraints in source sentence	

Table 10: Notation List.

	Parameters
The general setting for LORELEI languages	Combine the data produced by the setting a and b: (a) $num\_beam = 4$, $max\_unconstraint\_len = 0$, $srcConstraints$: all source words (b) $num\_beam = 4$, $max\_unconstraint\_len = 0$, $keep\_order = true$, $ph\_keep\_order = true$, $top\_th = 1$, $srcConstraints$: all source words
The general setting for CoNLL languages	$num\_beam = 8$, $max\_unconstraint\_len = 0$, $keep\_order = true$, $ph\_keep\_order = true$, $top\_th = 0.9$, $srcConstraints$: all source words
German	The general setting for CoNLL languages, $dict = CT$
Spanish	The general setting for CoNLL languages, $dict = BWET$
Dutch	The general setting for CoNLL languages, $dict = BWET$
Arabic	The general setting for LORELEI languages, $dict = CT$
Akan	The general setting for LORELEI languages, $dict = CT$
Turkish	Generate data with the setting c and d once, respectively; then combine them together with the data produced by the general setting with the BWET dict (c) $num\_beam = 8$, $max\_unconstraint\_len = 0$, $keep\_order = true$, $ph\_keep\_order = true$, $top\_th = 0.9$, $srcConstraints$: all source words, $dict = BWET$ (d) $num\_beam = 4$, $do\_sample = true$, $max\_unconstraint\_len = 1.5 * unconstrained\_src\_words\_num$, $srcConstraints$: named entities and neighbours in the source
Wolof	The general setting for LORELEI languages, $dict = CT$
Yoruba	Generate data with the setting e for three times and combine them together (e) $num\_beam = 4$, $do\_sample = true$, $dict = CT$, $max\_unconstraint\_len = 1.5 * unconstrained\_src\_words\_num$, $srcConstraints$: named entities and neighbours in the source
Uzbek	The general setting for LORELEI languages, $dict = BWET$

Table 11: Generation settings used in Tables 5, 6 and 8 of the paper.