MELM: Data Augmentation with Masked Entity Language Modeling for Low-Resource NER

Ran Zhou1,2 Xin Li1 Ruidan He1 Lidong Bing1 Erik Cambria2 Luo Si1 Chunyan Miao2
1DAMO Academy, Alibaba Group  2Nanyang Technological University, Singapore
{ran.zhou, xinting.lx, ruidan.he, l.bing, luo.si}@alibaba-inc.com
{cambria, ascy@miao}@ntu.edu.sg

Abstract

Data augmentation is an effective solution to data scarcity in low-resource scenarios. However, when applied to token-level tasks such as NER, data augmentation methods often suffer from token-label misalignment, which leads to unsatisfactory performance. In this work, we propose Masked Entity Language Modeling (MELM) as a novel data augmentation framework for low-resource NER. To alleviate the token-label misalignment issue, we explicitly inject NER labels into sentence context, and thus the fine-tuned MELM is able to predict masked entity tokens by explicitly conditioning on their labels. Thereby, MELM generates high-quality augmented data with novel entities, which provides rich entity regularity knowledge and boosts NER performance. When training data from multiple languages are available, we also integrate MELM with code-mixing for further improvement. We demonstrate the effectiveness of MELM on monolingual, cross-lingual and multilingual NER across various low-resource levels. Experimental results show that our MELM presents substantial improvement over the baseline methods.1

1  Introduction

Named entity recognition (NER) is a fundamental NLP task which aims to locate named entity mentions and classify them into predefined categories. As a subtask of information extraction, it serves as a key building block for information retrieval (Banerjee et al., 2019), question answering (Fabbri et al., 2020) and text summarization systems (Nallapati et al., 2016) etc. However, except a few high-resource languages / domains, the majority of languages / domains have limited amount of labeled data.

Since manually annotating sufficient labeled data for each language / domain is expensive, low-resource NER (Cotterell and Duh, 2017; Feng et al., 2018; Zhou et al., 2019; Rijhwani et al., 2020) has received increasing attention in the research community over the past years. As an effective solution to data scarcity in low-resource scenarios, data augmentation enlarges the training set by applying label-preserving transformations. Typical data augmentation methods for NLP include (1) word-level modification (Wei and Zou, 2019; Kobayashi, 2018; Wu et al., 2019; Kumar et al., 2020) and (2) back-translation (Sennrich et al., 2016; Fadaee et al., 2017; Dong et al., 2017; Yu et al., 2018).

Despite the effectiveness on sentence-level tasks, they suffer from the token-label misalignment issue when applied to token-level tasks like NER. More specifically, word-level modification might replace an entity with alternatives that mismatch the original label. Back-translation creates augmented texts that largely preserve the original content. However, it hinges on external word alignment tools for propagating the labels from the original input to the augmented text, which has proved to be error-prone.

To apply data augmentation on token-level tasks, Dai and Adel (2020) proposed to randomly substitute entity mentions with existing entities of the same class. They avoided the token-label misalignment issue but the entity diversity does not increase. Besides, the substituted entity might not fit into the original context. Li et al. (2020a) avoided the token-label misalignment issue by only diversifying the context, where they replaced context (having ‘O’ label) tokens using MASS (Song et al., 2019) and left the entities (i.e. aspect terms in their task) completely unchanged. However, according to the NER evaluations in Lin et al. (2020), augmentation on context gave marginal improvement on pretrained-LM-based NER models.
Our preliminary results on low-resource NER (see Figure 1) also demonstrate that diversifying entities in the training data is more effective than introducing more context patterns. Inspired by the aforementioned observations, we propose Masked Entity Language Modeling (MELM) as a data augmentation framework for low-resource NER, which generates augmented data with diverse entities while alleviating the challenge of token-label misalignment. MELM is built upon pretrained Masked Language Models (MLM), and it is further fine-tuned on corrupted training sentences with only entity tokens being randomly masked to facilitate entity-oriented token replacement. Simply masking and replacing entity tokens using the finetuned MLM is still insufficient because the predicted entity might not align with the original label. Taking the sentence shown in Figure 2b as an example, after masking the named entity “European Union” (Organization), the finetuned MLM could predict it as “Washington has”. Such prediction fits the context but it is not aligned with the original labels. To alleviate the misalignment, our MELM additionally introduces a labeled sequence linearization strategy, which respectively inserts one label token before and after each entity token and regards the inserted label tokens as the normal context tokens during masked language modeling. Therefore, the prediction of the masked token is conditioned on not only the context but the entity’s label as well.

After injecting label information and finetuning on the label-enhanced NER data, our MELM can exploit rich knowledge from pre-training to increase entity diversity while greatly reducing token-label misalignment. Code-mixing (Singh et al., 2019; Qin et al., 2020; Zhang et al., 2021) achieved promising results by creating additional code-mixed samples using the available multilingual training sets, which is particularly beneficial when the training data of each language is scarce. Fortunately, in the scenarios of multilingual low-resource NER, our MELM can also be applied on the code-mixed examples for further performance gains. We first apply code-mixing by replacing entities in a source language sentence with the same type entities of a foreign language. However, even though token-label alignment is guaranteed by replacing with entities of the same type, the candidate entity might not best fit into the original context (for example, replacing a government department with a football club). To solve this problem, we propose an entity similarity search algorithm based on bilingual embedding to retrieve the most semantically similar entity from the training entities in other languages. Finally, after adding language markers to the code-mixed data, we use them to fine-tune MELM for generating more code-mixed augmented data.

To summarize, the main contributions of this paper are as follows: (1) we present a novel framework which jointly exploits sentence context and entity labels for entity-based data augmentation. It consistently achieves substantial improvement when evaluated on monolingual, cross-lingual, and multilingual low-resource NER; (2) the proposed labeled sequence linearization strategy effectively alleviates the problem of token-label misalignment during augmentation; (3) an entity similarity search algorithm is developed to better bridge entity-based data augmentation and code-mixing in multilingual scenarios.

2 Method

Fig. 2c presents the work flow of our proposed data augmentation framework. We first perform labeled sequence linearization to insert the entity label tokens into the NER training sentences (Section 2.1). Then, we fine-tune the proposed MELM on linearized sequences (Section 2.2) and create augmented data by generating diverse entities via
Figure 2: Comparison of different data augmentation methods, color printing is preferred. (a) augmentation with pretrained MLM (b) augmentation with MELM without linearization (c) augmentation with MELM masked entity prediction (Section 2.3).

The augmented data undergoes post-processing (Section 2.4) and is combined with the original training set for training the NER model. Algorithm 1 gives the pseudo-code for the overall framework. Under multilingual scenarios, we propose an entity similarity search algorithm as a refined code-mixing strategy (Section 2.5) and apply our MELM on the union set of gold training data and code-mixed data for further performance improvement.

2.1 Labeled Sequence Linearization
To minimize the amount of generated tokens incompatible with the original labels, we design a labeled sequence linearization strategy to explicitly take label information into consideration during masked language modeling. Specifically, as shown in Figure 2c, we add the label token before and after each entity token and treat them as normal context tokens. The yielded linearized sequence is utilized to further finetune our MELM so that its prediction is additionally conditioned on the inserted label tokens. Note that, we initialize the embeddings of label tokens with those of tokens semantically related to the label names (e.g., “organization” for \( \langle \text{B-ORG} \rangle \)). By doing so, the linearized sequence is semantically closer to a natural sentence and the difficulty of finetuning on linearized sequence could be reduced (Kumar et al., 2020).

2.2 Fine-tuning MELM
Unlike MLM, only entity tokens are masked during MELM fine-tuning. At the beginning of each fine-tuning epoch, we randomly mask entity tokens in the linearized sentence \( X \) with masking ratio \( \eta \).

Then, given the corrupted sentence \( \tilde{X} \) as input, our MELM is trained to maximize the probabilities of the masked entity tokens and reconstruct the linearized sequence \( X \):

\[
\max_\theta \log p_\theta(X|\tilde{X}) \approx \sum_{i=1}^{n} m_i \log p_\theta(x_i|\tilde{X}) \quad (1)
\]

where \( \theta \) represents the parameters of MELM, \( n \) is the number of tokens in \( \tilde{X} \), \( x_i \) is the original token in \( X \), \( m_i = 1 \) if \( x_i \) is masked and otherwise \( m_i = 0 \). Through the above fine-tuning process, the proposed MELM learns to make use of both contexts and label information to predict the masked entity tokens. As we will demonstrate in Section 4.1, the predictions generated by the fine-tuned MELM are significantly more coherent with the original entity label, compared to those from other methods.

2.3 Data Generation
To generate augmented training data for NER, we apply the fine-tuned MELM to replace entities in the original training samples. Specifically, given a corrupted sequence, MELM outputs the probability of each token in the vocabulary being the masked entity token. However, as the MELM is fine-tuned on the same training set, directly picking the most probable token as the replacement is likely to return the masked entity token in the original training sample, and might fail to produce a novel augmented sentence. Therefore, we propose to randomly sample the replacement from the top \( k \) most probable components of the probability distribution. Formally, given the probability distribution
Algorithm 1 Masked Entity Language Modeling (MELM)

Given $D_{\text{train}}$, $M$

$D_{\text{masked}} \leftarrow \emptyset$, $D_{\text{aug}} \leftarrow \emptyset$

for $\{X, Y\} \in D_{\text{train}}$

$X \leftarrow \text{FINETUNE}(X, Y)$

$\hat{X} \leftarrow \text{FINETUNE}(X, \eta)$

$D_{\text{masked}} \leftarrow D_{\text{masked}} \cup \{\hat{X}\}$

end for

$M_{\text{finetune}} \leftarrow \text{FINETUNE}(M, D_{\text{masked}})$

for $\{X, Y\} \in D_{\text{masked}}$

repeat $R$ times:

$X \leftarrow \text{LINEARIZE}(X, Y)$

$\tilde{X} \leftarrow \text{GENMASK}(\hat{X}, \mu)$

$X_{\text{aug}} \leftarrow \text{RANDCHOICE}(M_{\text{finetune}}(\tilde{X}), \text{Top } k = 5)$

$D_{\text{aug}} \leftarrow D_{\text{aug}} \cup \{X_{\text{aug}}\}$

end for

$D_{\text{aug}} \leftarrow \text{POSTPROCESS}(D_{\text{aug}})$

return $D_{\text{train}} \cup D_{\text{aug}}$

$\triangleright$ Given gold training set $D_{\text{train}}$ and pretrained MLM $M$

$\triangleright$ Labeled sequence linearization

$\triangleright$ Randomly mask entities for fine-tuning

$\triangleright$ Fine-tune MELM on masked linearized sequences

$\triangleright$ Labeled sequence linearization

$\triangleright$ Randomly mask entities for generation

$\triangleright$ Generate augmented data with fine-tuned MELM

$\triangleright$ Post-processing

$P(x_i | \tilde{X})$ for a masked token, we first select a set $V_k \subseteq V$ of the $k$ most likely candidates. Then, we fetch the replacement $\hat{x}_i$ via random sampling from $V_k$. After obtaining the generated sequence, we remove the label tokens and use the remaining parts as the augmented training data. For each sentence in the original training set, we repeat the above generation procedure $R$ rounds to produce $R$ augmented examples.

To increase the diversity of augmented data, we adopt a different masking strategy from train time. For each entity mention comprising of $n$ tokens, we randomly sample a dynamic masking rate $\epsilon$ from Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$, where the Gaussian variance $\sigma^2$ is set as $1/n^2$. Thus, the same sentence will have different masking results in each of the $R$ augmentation rounds, resulting in more varied augmented data.

2.4 Post-Processing

To remove noisy and less informative samples from the augmented data, the generated augmented data undergoes post-processing. Specifically, we train a NER model with the available gold training samples and use it to automatically assign NER tags to each augmented sentence. Only augmented sentences whose predicted labels are consistent with the original labels are kept. The post-processed augmented training set $D_{\text{aug}}$ is combined with the gold training set $D_{\text{train}}$ to train the final NER tagger.

2.5 Extending to Multilingual Scenarios

When extending low-resource NER to multilingual scenarios, it is straightforward to separately apply the proposed MELM on language-specific data for performance improvement. Nevertheless, it offers higher potential to enable MELM on top of code-mixing techniques, which proved to be effective in enhancing multilingual learning (Singh et al., 2019; Qin et al., 2020; Zhang et al., 2021). In this paper, with the aim of bridging MELM augmentation and code-mixing, we propose an entity similarity search algorithm to perform MELM-friendly code-mixing.

Specifically, given the gold training sets $\{D'_{\text{train}} | \ell \in \mathbb{L}\}$ over a set $\mathbb{L}$ of languages, we first collect label-wise entity sets $\mathbb{E}_{\ell,y}$, which consists of the entities appearing in $D'_{\text{train}}$ and belonging to class $y$. To apply code-mixing on a source language sentence $X_{\text{src}}$, we aim to substitute a mentioned entity $e$ of label $y$ with a target language entity $e_{\text{sub}} \in \mathbb{E}_{\ell,y}^\ell$, where the target language is sampled as $\ell_{\text{tgt}} \sim \mathcal{U}(\mathbb{L} \setminus \{\ell_{\text{src}}\})$. Instead of randomly selecting $e_{\text{sub}}$ from $\mathbb{E}_{\ell,y}^\ell$, we choose to retrieve the entity with the highest semantic similarity to $e$ as $e_{\text{sub}}$. Practically, we introduce MUSE bilingual embeddings (Conneau et al., 2017) and calculate the entity’s embedding $\text{Emb}(e)$ by averaging the embeddings of the entity tokens:

$$\text{Emb}(e) = \frac{1}{|e|} \sum_{i=1}^{|e|} \text{MUSE}_{\ell_{\text{src}}, \ell_{\text{tgt}}}(e_i)$$

where $\text{MUSE}_{\ell_{\text{src}}, \ell_{\text{tgt}}}$ denotes the $\ell_{\text{src}} - \ell_{\text{tgt}}$ aligned embeddings and $e_i$ is the $i$-th token of $e$. Next, we obtain the target-language entity $e_{\text{sub}}$ semantically closest to $e$ as follows:

$$e_{\text{sub}} = \arg\max_{\hat{e} \in \mathbb{E}_{\ell,y}^\ell} f(\text{Emb} (e), \text{Emb} (\hat{e}))$$
$f(\cdot, \cdot)$ here is the cosine similarity function. The output entity $e_{sub}$ is then used to replace $e$ to create a code-mixed sentence more suitable for MELM augmentation. To generate more augmented data with diverse entities, we further apply MELM on the gold and code-mixed data. Since the training data now contains entities from multiple languages, we also prepend a language marker to the entity token to help MELM differentiate different languages, as shown in Figure 3.

Figure 3: Applying MELM on gold and code-mixed data. Language markers (e.g., <Español>) are inserted during linearization.

### 3 Experiments

To comprehensively evaluate the effectiveness of the proposed MELM on low-resource NER, we consider three evaluation scenarios: monolingual, zero-shot cross-lingual and multilingual low-resource NER.

#### 3.1 Dataset

We conduct experiments on CoNLL NER dataset (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003) of four languages where $L = \{\text{English (En)}, \text{German (De)}, \text{Spanish (Es)}, \text{Dutch (Nl)}\}$. For each language $\ell \in L$, we first sample $N$ sentences from the full training set as $D_{\ell, train}^{\ell,N}$, where $N \in \{100, 200, 400, 800\}$ to simulate different low-resource levels. For a realistic data split ratio, we also downscale the full development set to $N$ samples as $D_{\ell, dev}^{\ell,N}$. The full test set for each language is adopted as $D_{\ell, test}^{\ell,N}$ for evaluation.

For monolingual experiments on language $\ell$ with low-resource level $N \in \{100, 200, 400, 800\}$, we use $D_{\ell, train}^{\ell,N}$ as the gold training data, $D_{\ell, dev}^{\ell,N}$ as the development set and $D_{\ell, test}^{\ell,N}$ as the test set. For zero-shot cross-lingual experiments with low-resource level $N \in \{100, 200, 400, 800\}$, we use $D_{\ell, train}^{\ell,N}$ as the source language gold training data, $D_{\ell, dev}^{\ell,N}$ and $D_{\ell, test}^{\ell,N}$ as development set and $D_{test}^{\ell,N}$ as the target language test set. Under multilingual settings where $N$ training data from each language is available ($N \in \{100, 200, 400\}$), we use $\bigcup_{\ell \in L} D_{\ell, train}^{\ell,N}$ as the gold training data, $\bigcup_{\ell \in L} D_{\ell, dev}^{\ell,N}$ as the development set and evaluate on $D_{test}^{\ell,N}$ respectively.

#### 3.2 Experimental Setting

**MELM Fine-tuning** We use XLM-RoBERTa-base (Conneau et al., 2020) with a language-modeling head to initialize MELM parameters. MELM is fine-tuned for 20 epochs using Adam optimizer (Kingma and Ba, 2015) with batch size set to 30 and learning rate set to $1e-5$.

**NER Model** We use XLM-RoBERTa-Large (Conneau et al., 2020) with CRF head (Lample et al., 2016) as the NER model for our experiments. We adopt Adam optimizer (Loshchilov and Hutter, 2019) with learning rate set to $2e-5$ and set batch size to 16. The NER model is trained for 10 epochs and the best model is selected according to dev set performance. The trained model is evaluated on test sets and we report the averaged Micro-F1 scores over 3 runs.

**Hyperparameter Tuning** The masking rate $\eta$ in MELM fine-tuning, the Gaussian mean $\mu$ for MELM generation and the number of MELM augmentation rounds $R$ are set as 0.7, 0.5 and 3, respectively. All of these hyperparameters are tuned on the dev set with grid search. Details of the hyperparameter tuning can be found in Appendix A.1

#### 3.3 Baseline Methods

To elaborate the effectiveness of the proposed MELM, we compare it with the following methods:

**Gold-Only** The NER model is trained on only the original gold training set.

**Label-wise Substitution** Dai and Adel (2020) randomly substituted named entities with existing entities of the same entity type from the original training set.

**MLM-Entity** We randomly mask entity tokens and directly utilize a pretrained MLM for data augmentation without fine-tuning and labeled sequence linearization as used in MELM. The prediction of a masked entity token does not consider label information but solely relies on the context words.

**DAGA** Ding et al. (2020) firstly linearized NER labels into the input sentences and then use them to train an autoregressive language model. The language model was used to synthesize augmented

---

2https://github.com/allanj/pytorch_neural_crf
data from scratch, where both context and entities are generated simultaneously.

MulDA Liu et al. (2021) fine-tuned mBART (Liu et al., 2020) on linearized multilingual NER data to generate augmented data with new context and entities.

3.4 Experimental Results

3.4.1 Monolingual and Cross-lingual NER

As illustrated on the left side of Table 1, the proposed MELM consistently achieves the best averaged results across different low-resource levels, demonstrating its effectiveness on monolingual NER. Compared to the best-performing baselines, our MELM obtains 6.3, 1.6, 1.3, 0.38 absolute gains on 100, 200, 400 and 800 levels, respectively. Cross-lingual NER results are shown on the right side of Table 2. Again, on each of the designed low-resource levels, our MELM is superior to baseline methods in terms of the averaged F1 scores. We also notice that, given 100 Nl training samples, the Gold-Only method without data augmentation almost fails to converge while the monolingual F1 of our MELM reaches 66.6, suggesting that data augmentation is crucial for NER when the annotated training data is extremely scarce.

To assess the efficacy of the proposed labeled sequence linearization (Section 2.1), we directly fine-tune MELM on masked sentences without linearization (as shown in Figure 2b), denoted as MELM w/o linearize in Table 1. We observe a considerable performance drop compared with MELM, which proves the label information injected via linearization indeed helps MELM differentiate different entity types, and generate entities compatible with the original label.

Taking a closer look at the baseline methods, we notice that the monolingual performance of Label-wise is still unsatisfactory in most cases. One probable reason is that only existing entities within the training data are used for replacement and the entity diversity after augmentation is not increased. Moreover, randomly sampling an entity of the same type for replacement is likely to cause incompatibility between the context and the entity, yielding a noisy augmented sample for NER training. Although MLM-Entity tries to mitigate these two issues by employing a pretrained MLM to generate novel tokens that fit into the context, the generated tokens might not be consistent with the original labels. Our MELM also promotes the entity diversity of augmented data by exploiting pretrained model for data augmentation.

In the meantime, equipped with the labeled sequence linearization strategy, MELM augmentation is explicitly guided by the label information and the token-label misalignment is largely alleviated, leading to superior results in comparison to Label-wise and MLM-Entity.

We also compare with DAGA (Ding et al., 2020), which generates augmented data from scratch using an autoregressive language model trained on gold NER data. Although DAGA is competitive on low-resource levels of 400 and 800, it still underperforms the proposed MELM by a large margin when the training size reduces to 100 or 200. We attribute this to the disfluent and ungrammatical sentences generated from the undertrained language model. Instead of generating augmented data from scratch, MELM focuses on modifying entity tokens and leave the context unchanged, which guarantees the quality of augmented sentences even under extremely low-resource settings.

3.4.2 Multilingual NER

For multilingual low-resource NER, we firstly directly apply MELM on the concatenation of training sets from multiple languages. As shown in Table 2, MELM-gold achieves substantial improvement over the Gold-only baseline, which is consistent with monolingual and cross-lingual results. We compare with MulDA (Liu et al., 2021) as a baseline data augmentation method. MulDA generates augmented data autoregressively with an mBART model, which is fine-tuned on NER data with inserted label tokens. At the low-resource levels in our experimental settings, MulDA is less effective and even leads to deteriorated performance. The unsatisfactory performance mainly results from the discrepancy between pretraining and fine-tuning due to the inserted label tokens. Given very few training samples, it is difficult to adapt mBART to capture the distribution of the inserted label tokens, and thus MulDA struggles to generate fluent and grammatical sentences from scratch. In comparison, our proposed method preserves the original context and introduces less syntactic noise in the augmented data. To further leverage the benefits of code-mixing in multilingual NER, we experiment with two code-mixing methods: (1) Code-Mix-random, which randomly substitutes entities with existing entities of the same type from other languages, and (2) Code-Mix-ess, which adopts
Gold Method | Monolingual | Cross-lingual
---|---|---
| En | De | Es | Ni | Avg | En→De | En→Es | En→Nl | Avg | En→De | En→Es | En→Nl | Avg |
---|---|---|---|---|---|---|---|---|---|---|---|---|---|
Gold-Only | 50.57 | 39.47 | 42.93 | 21.63 | 38.65 | 39.54 | 37.40 | 39.27 | 38.74 |
Label-wise | 61.34 | 55.00 | 59.54 | 27.85 | 50.93 | 45.85 | 43.74 | 50.51 | 46.70 |
MLM-Entity | 61.22 | 50.96 | 61.29 | 46.59 | 55.02 | 47.96 | 45.42 | 49.34 | 47.57 |
DAGA | 68.06 | 59.15 | 69.33 | 45.64 | 60.54 | 52.95 | 46.72 | 54.63 | 51.43 |
MELM w/o linearize | 70.01 | 61.92 | 65.07 | 59.76 | 64.19 | 48.70 | 49.10 | 53.37 | 50.39 |
MELM (Ours) | 75.21 | 64.12 | 75.85 | 66.57 | 70.44 | 56.56 | 53.83 | 60.62 | 57.00 |

Table 1: Left side of table shows the results of monolingual low-resource NER. Right side of table shows the results of cross-lingual low-resource NER with English as source language. **Avg** on left side and right side are the averaged result over all languages and all transfer pairs, respectively.

<table>
<thead>
<tr>
<th>#Gold</th>
<th>Method</th>
<th>En</th>
<th>De</th>
<th>Es</th>
<th>Ni</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gold-Only</td>
<td>74.64</td>
<td>62.85</td>
<td>72.64</td>
<td>55.96</td>
<td>66.52</td>
<td>54.95</td>
</tr>
<tr>
<td>Label-wise</td>
<td>76.82</td>
<td>67.31</td>
<td>78.34</td>
<td>66.52</td>
<td>72.25</td>
<td>55.01</td>
</tr>
<tr>
<td>MLM-Entity</td>
<td>79.16</td>
<td>70.01</td>
<td>78.45</td>
<td>66.69</td>
<td>73.58</td>
<td>60.44</td>
</tr>
<tr>
<td>DAGA</td>
<td>79.11</td>
<td>69.82</td>
<td>78.95</td>
<td>68.53</td>
<td>74.10</td>
<td>59.58</td>
</tr>
<tr>
<td>MELM w/o linearize</td>
<td>81.77</td>
<td>71.41</td>
<td>80.43</td>
<td>72.92</td>
<td>76.63</td>
<td>62.57</td>
</tr>
<tr>
<td>MELM (Ours)</td>
<td>82.91</td>
<td>72.71</td>
<td>80.46</td>
<td>77.02</td>
<td>78.27</td>
<td>65.01</td>
</tr>
</tbody>
</table>

Table 2: Results of multilingual low-resource NER. Gold training set contains the same number of training samples from each language. **Avg** is the averaged result over all languages.

The proposed entity similarity search algorithm in Section 2.5 as the code-mixing strategy.

Experimental results in Table 2 show that both methods are able to achieve improved performance over Gold-Only. This observation suggests that code-mixing techniques, either random code-mixing or code-mixing via our entity similarity search, are indeed helpful for multilingual NER. Comparing these two methods, the performance gains brought by Code-Mix-*ess* are more significant and consistent across different low-resource levels, which demonstrates the effectiveness of our proposed entity similarity search algorithm. Applying MELM on both gold data and code-mixed data from Code-Mix-*ess*, the multilingual NER results are further improved. In summary, our proposed MELM is well-suited for multilingual NER, which can be integrated with our code-mixing technique to achieve further improvement.

4 Further Analysis

4.1 Case Study

Apart from the quantitative results, we further analyze the augmented data to demonstrate the effectiveness of our MELM in maintaining the consistency between the original label and the augmented token. Table 3 presents examples of the top-5 predictions from pretrained MLM, MELM w/o linearize and MELM. As we can see, the pretrained MLM, which does not introduce any design or constraint on data augmentation, tends to generate high-frequency words such as “the”, “he” and “she”, and the majority of generated words do not belong to the original entity class. Being finetuned on NER data with entity-oriented masking, MELM...
Table 3: Examples of the top-5 predictions by MLM, MELM w/o linearize and MELM. Predictions that do not belong to the original class are highlighted in red.

& Text & EU & rejects & German & call to boycott & British & Lamb \\
Label & B-ORG & O & B-MISC & O & O & B-MISC & O \\
MLM & Britain, EU, UK, Trump, US & US & a, UN, the, UK & the, a, black, white, young & \\
MLM w/o linearize & EU, Australia, US, UN, Israel & German, Indian, the, Washington, Union & Chinese, British, raw, California, Australian & \\
MELM & EU, Greenpeace, Amnesty, UN, Reuters & German, British, Dutch, French, EU & African, British, Guinean, white, French & \\
Label & Text & Clinton aide resigns, NBC says & my, his, My, When, her & & & he, she, it, and, who & \\
MLM & French, German, British, Swiss, Russian & Reuters, Pompeo, Blair Hill, AFP & & & & \\
MLM w/o linearize & French, White, Walker, Ferguson, David & NBC, AFP, Greenpeace, BBC, Anonymous & & & & \\

Figure 4: Comparison between the number of unique valid entities introduced by different methods

4.2 Number of Unique Entities

As demonstrated in Lin et al. (2020) and our preliminary experiments in Figure 1, introducing unseen entities can effectively provide more entity regularity knowledge, and helps to improve NER performance. Therefore, we examine the amount of unique entities introduced by different methods. As there might be token-label misalignment in the augmented data, we firstly train an ‘oracle’ NER model on the full CoNLL dataset and then use it to tag training data of MELM and different baseline methods. For each method, we count the total number of unique entities whose labels match the labels assigned by the ‘oracle’ model. As shown in Figure 4, while many augmented entities from MLM-Entity, DAGA and MELM w/o linearize are filtered out due to token-label misalignment, we note that MELM introduces a significantly larger number of unseen entities in the augmented data. Therefore MELM is able to provide richer entity regularity knowledge, which explains its superiority over the baseline methods.

5 Related Work

On sentence level tasks, one line of data augmentation methods are built upon word-level modifications, which can be based on synonym replacement (Wei and Zou, 2019), LSTM language model (Kobayashi, 2018), MLM (Wu et al., 2019; Kumar et al., 2020), auto-regressive pretrained LM (Kumar et al., 2020), or constituent-based tagging schemes (Zhong et al., 2020). However, these methods suffer from token-label misalignment when applied to token-level tasks such as NER, which requires sophisticated post-processing to remove noisy samples in augmented data (Bari et al., 2021; Zhong and Cambria, 2021).

Existing works avoid token-label misalignment by replacing entities with existing entities of the same class (Dai and Adel, 2020), or only modifying context works and leaving entities / aspect terms unchanged (Li et al., 2020a). Others attempt to produce augmented data by training / fine-tuning
a generative language model on linearized labeled sequences (Ding et al., 2020; Liu et al., 2020).

Backtranslation (Sennrich et al., 2016; Fadaee et al., 2017; Dong et al., 2017; Yu et al., 2018) translates source language sentences into a target language, and subsequently back to the source language, which preserve the overall semantics of the original sentences. On token-level tasks, however, they hinge on external word alignment tools for label propagation, which are often error-prone (Tsai et al., 2016; Li et al., 2020b).

6 Conclusion

We have proposed MELM as a data augmentation framework for low-resource NER. Through labeled sequence linearization, we enable MELM to explicitly condition on label information when predicting masked entity tokens. Thus, our MELM effectively alleviates the token-label misalignment issue and generates augmented data with novel entities by exploiting pretrained knowledge. Under multilingual settings, we integrate MELM with code-mixing for further performance gains. Extensive experiments show that the proposed framework demonstrates encouraging performance gains on monolingual, cross-lingual and multilingual NER across various low-resource levels.

Acknowledgements

This research is partly supported by the Alibaba-NTU Singapore Joint Research Institute, Nanyang Technological University. Erik Cambria would like to thank the support by the Agency for Science, Technology and Research (A*STAR) under its AME Programmatic Funding Scheme (Project #A18A2b0046).

References


A Appendix

A.1 Hyperparameter Tuning

Masking hyperparameters. To determine the optimal setting for fine-tune mask rate $\eta$ and generation masking parameter $\mu$, we conduct a grid search on both hyperparameters in range $[0.3, 0.5, 0.7]$. We finetune MELM and generate English augmented data on CoNLL following our method in Section 2. The augmented data is used to train a NER tagger and its performance on English dev set is recorded. As shown in Table 4, we achieve the best dev set F1 when $\eta = 0.7$ and $\mu = 0.5$, which is adopted for the rest of this work.

<table>
<thead>
<tr>
<th>$\eta$</th>
<th>$\mu$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0.3$</td>
<td>$76.90$</td>
</tr>
<tr>
<td></td>
<td>$75.64$</td>
</tr>
<tr>
<td></td>
<td>$78.08$</td>
</tr>
<tr>
<td>$0.5$</td>
<td>$76.16$</td>
</tr>
<tr>
<td></td>
<td>$78.06$</td>
</tr>
<tr>
<td>$0.7$</td>
<td>$75.94$</td>
</tr>
<tr>
<td></td>
<td>$78.09$</td>
</tr>
<tr>
<td></td>
<td>$78.37$</td>
</tr>
</tbody>
</table>

Table 4: Dev set F1 for masking hyperparameter tuning.

Number of augmentation rounds. Merging augmented data from multiple rounds increase entity diversity until it saturates at certain point. Continuing adding in more augmented data begins to amplify the noise in augmented data and leads to decreasing performance. To determine the optimum number of augmentation rounds $R$, we merge different amount of augmented data with English gold data to train a NER tagger, with $R$ ranging from 1 to 6. As shown in Table 5, dev set F1 increases with increasing amount of augmented data until $R=3$, and starts to drop further beyond. Therefore, we choose $R = 3$ for all of our experiments.

<table>
<thead>
<tr>
<th>$R$</th>
<th>Dev F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>92.35</td>
</tr>
<tr>
<td>2</td>
<td>92.36</td>
</tr>
<tr>
<td>3</td>
<td>92.84</td>
</tr>
<tr>
<td>4</td>
<td>92.72</td>
</tr>
<tr>
<td>5</td>
<td>92.59</td>
</tr>
<tr>
<td>6</td>
<td>92.39</td>
</tr>
</tbody>
</table>

Table 5: Dev set F1 for number of augmentation rounds.

A.2 Statistics for Reproducibility

In this section, we present the validation F1 averaged among 3 runs of MELM under different languages and low-resource levels. We also summarize the estimated time for fine-tuning MELM and the number of parameters used. We separately show the statistics of monolingual (Table 6), cross-lingual (Table 7) and multilingual (Table 8) NER.

<table>
<thead>
<tr>
<th>#Gold</th>
<th>En</th>
<th>De</th>
<th>Es</th>
<th>Nl</th>
<th>time</th>
<th>#Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>82.38</td>
<td>71.11</td>
<td>71.77</td>
<td>71.01</td>
<td>~ 7min</td>
<td>270M</td>
</tr>
<tr>
<td>200</td>
<td>85.93</td>
<td>77.96</td>
<td>83.25</td>
<td>79.53</td>
<td>~ 10min</td>
<td>270M</td>
</tr>
<tr>
<td>400</td>
<td>89.01</td>
<td>82.95</td>
<td>85.10</td>
<td>81.40</td>
<td>~ 15min</td>
<td>270M</td>
</tr>
<tr>
<td>800</td>
<td>92.01</td>
<td>84.82</td>
<td>86.65</td>
<td>85.61</td>
<td>~ 20min</td>
<td>270M</td>
</tr>
</tbody>
</table>

Table 6: Validation F1 for MELM under monolingual settings

<table>
<thead>
<tr>
<th>#Gold per language</th>
<th>dev F1</th>
<th>time</th>
<th>#Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>83.21</td>
<td>~ 20min</td>
<td>270M</td>
</tr>
<tr>
<td>200</td>
<td>84.83</td>
<td>~ 30min</td>
<td>270M</td>
</tr>
<tr>
<td>400</td>
<td>87.07</td>
<td>~ 45min</td>
<td>270M</td>
</tr>
</tbody>
</table>

Table 7: Validation F1 for MELM under cross-lingual settings

<table>
<thead>
<tr>
<th>#Gold per language</th>
<th>dev F1</th>
<th>time</th>
<th>#Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>92.35</td>
<td>~ 7min</td>
<td>270M</td>
</tr>
<tr>
<td>200</td>
<td>92.36</td>
<td>~ 10min</td>
<td>270M</td>
</tr>
<tr>
<td>400</td>
<td>92.84</td>
<td>~ 15min</td>
<td>270M</td>
</tr>
<tr>
<td>800</td>
<td>92.72</td>
<td>~ 20min</td>
<td>270M</td>
</tr>
</tbody>
</table>

Table 8: Validation F1 for MELM under multilingual settings

A.3 Computing Infrastructure

Our experiments are conducted on NVIDIA V100 GPU.