SemEval 2023: MultiCoNER II Midterm Report

Asal Shavandi Chahyon Ku Josh Spitzer-Resnick London Lowmanstone IV
University of Minnesota
{shava006,ku000045,spitz123,lowma016}@umn.edu

Abstract

The MultiCoNER II shared task aims to detect noisy and out-of-domain complex named entities for multiple languages. While knowledge-retrieval-based systems such as DAMO-NLP and USTC-NELSLIP were successful for the previous version of the task, MultiCoNER I, still had shortcomings such as sensitivity to noise (typos) and lower performance on out-of-knowledge-base named entities. To alleviate these shortcomings, we propose data augmentation techniques to augment training data with noisy and diverse-domain entities.

1 Introduction

The named entity recognition task is a critical part of information extraction in which every word in a sentence is classified into named entity types such as names of people, organization, location, etc. (Nadeau and Sekine, 2007). Since it was first organized in 1996 at the Sixth Message Understanding Conference, many mono- and multilingual tasks, such as the CoNLL 2003 (Sang and Meulder, 2003), Ontonotes corpus v5 (Pradhan et al., 2013), and WNUT 2017 Emerging Entities (Derczynski et al., 2017) were organized to tackle its challenges.

Among named entities, complex named entities are the more syntactically complex named entities–often names of creative works–that existing systems have a hard time recognizing (Ashwini and Choi, 2014). Complex named entities are more challenging to detect than traditional ones, because they are rarer in data, newly created more frequently, and have more diverse syntactic structures.

Even though knowledge-retrieval-based systems achieved great results at the MultiCoNER shared task from SemEval 2022 (Malmasi et al., 2022), they are sensitive to noisy and out-of-domain entities. The MultiCoNER II shared task proposes new tasks to address the shortcomings of top performing models from the MultiCoNER shared task.

Our group, Team Polygots, attempt to propose improvements to the baseline NER model by trying various data augmentations on the training data.

2 Related Works

2.1 Named Entity Recognition

Named Entity Recognition (NER) is a core natural language processing (NLP) task (Chen et al., 2022) that has a lot of applications in academia, marketing, medical and security domains. Transformer-based pretrained language models have achieved great success in almost every NLP task (Kalyan et al., 2021) including NER. These models learn universal language representations from large volumes of text data using self-supervised learning and transfer this knowledge to downstream tasks (Kalyan et al., 2021). Multilingual BERT (M-BERT), released by (Devlin et al., 2019) as a single language model pre-trained from monolingual corpora in 104 languages, is shown to be very good at cross-lingual model transfer (Pires et al., 2019). XLM-RoBERTa (Conneau et al., 2019) is another pretrained multilingual language model at scale that has led to significant performance gain for a wide range of cross-lingual transfer tasks.

2.2 Multilingual Language Models

Fine-tuning pretrained contextual embedding is a useful and effective approach to many NLP tasks (Wang et al., 2022) and recently many researchers have put their effort into training fine-tuned multilingual models such as mBERT and XLM-RoBERTa to improve their model's performance. (Malmasi et al., 2022) designed a NER system using XLM-RoBERTa on MultiCoNER I dataset that computes a representation for each token which was then used to predict the token tag using a Conditional Random Field (CRF) classification layer (Sutton and McCallum, 2010). Their system resulted in a F1 score of 0.478.

2.3 Data Augmentation for Natural Language Processing

There are four main categories of data augmentation methods: translation, substitution, generation, and mix-up. Translation-based methods, such as MulDA (Liu et al., 2021), translates a sentence to another language and often back to the source language to introduce variance. Substitution-based methods, such as MELM (Zhou et al., 2022), replaces characters, words, or phrases based on heuristics or language models. Generation-based methods, such as DAGA (Ding et al., 2020) trains a language model on the training data and randomly sample from the language model to generate new data. Mix-up-based methods, such as SeqMix (Zhang et al., 2020), linearly interpolate between pairs of samples to generate novel sentences.

2.4 Best Models from MultiCoNER I

The first ranked team from MultiCoNER I shared task, DAMO-NLP (Wang et al., 2022), which got the highest F1 score for the multilingual task, took a different approach and introduced a knowledge-based NER system which used (Malmasi et al., 2022) system as their baseline and added a knowledge retrieval module to enhance their performance. The knowledge retrieval module takes an input sentence as a query and retrieves top-k related paragraphs from Wikipedia which will be then concatenated and fed into the NER module. The output of the NER module which is a token representation of the input sentence will be fed into a linear-chain CRF to produce the label predictions. This method has shown an F1 score of 0.853.

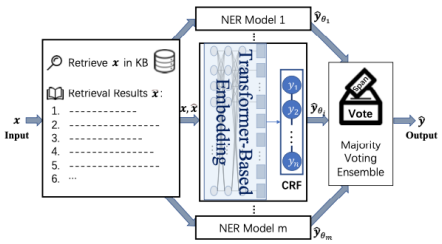

Figure 1: The architecture of DAMO-NLP knowledge-based NER system

The second-ranked team from MultiCoNER I, USTC-NELSLIP (Chen et al., 2022), also used the same concept of a knowledge retrieval system but instead of retrieving top-related paragraphs from Wikipedia, they introduced a gazetteer-adapted integration network (GAIN). This system first adapts the representation of the gazetteer network built from Wikidata to the XLM-RoBERTa model by minimizing KL divergence between them. After adaptation, these networks will be integrated for backend-supervised NER training. This method has also shown an F1 score of 0.853.

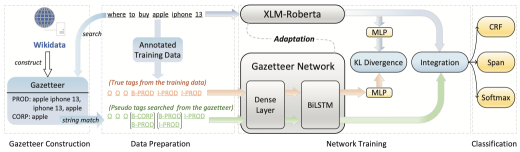

Figure 2: The architecture of USTC gazetteer-adapted integration network

3 Problem Formulation

3.1 Task

MultiCoNER II is a Multilingual Complex Named Entity Recognition shared task, offered as part of SemEval-2023, The 17th International Workshop on Semantic Evaluation. Given a sentence, the task is to detect and categorize all named entities in the sentence. To be more concrete, each word in the sentence is tagged with Beginning-Inside-Outside tags. The beginning tag denotes the first word in a named entity, inside denotes other words in a named entity, and outside means the word is not a part of a named entity. Then, each beginning and inside tags are categorized into one of 36 fine-grained labels, which are organized into 6 categories according to the WNUT 2017 (Derczynski et al., 2017) taxonomy entity types: person, group, corporation, location, product, and creative work. The dataset consists of 12 languages, where each language has between 9k-16k training sentences and 500-900 development sentences.

3.2 Dataset

MultiCoNER II is divided into 12 languages, including 7 of the same languages from MultiCoNER I (Bangla, Chinese, English, Farsi, German, Hindi, and Spanish) and 5 new languages (French, Italian, Portuguese, Swedish, and Ukrainian). The second iteration of this task leaves out 4 previous languages (Dutch, Korean, Russian, and Turkish) as well as the multilingual and code-mixed tasks of the previous competition. For our efforts, we have simplified our exploration to look at the English and French tasks.

Rather than the 6 coarse categories present in MultiCoNER I common of standard NER systems,

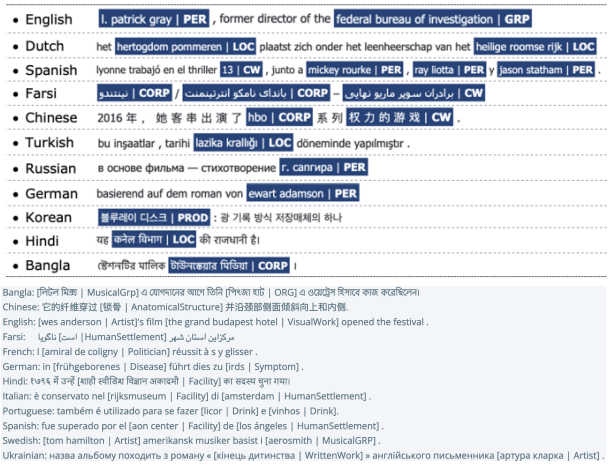

Figure 3: Example sentences and labeled named entities from MultiCoNER I (Malmasi et al., 2022) and MultiCoNER II

there are instead 36 defined granular labels which are then organized into 6 categories (see figure 4). For each language in the dataset in both train and development sets however, OtherCW, OtherCORP, and TechCORP never occur, thus limiting the dataset to 33 observed labels. Notably, MultiCoNER II introduces a new medical (MED) category, and lumps the previous year's corporation (CORP) category into the group (GRP) category. Examples of the difference in granularity of labeling between the two tasks can be seen in figure 3.

label	description
CORP /	Corporation
CW	Creative Work
GRP	Group
LOC	Location
PER	Person
PROD	Product

category	labels
Creative Work (CW)	ArtWork, MusicalWork, OtherCW, Software, VisualWork, WrittenWork
Group (GRP)	AerospaceManufacturer, CarManufacturer, MusicalGRP, ORG, OtherCORP, PrivateCORP, PublicCORP, SportsGRP, TechCORP
Location (LOC)	Facility, HumanSettlement, OtherLOC, Station
Medical (MED) //	AnatomicalStructure, Disease, MedicalProcedure, Medication/Vaccine, Symptom
Person (PER)	Artist, Athlete, Cleric, OtherPER, Politician, Scientist, SportsManager
Product (PROD)	Clothing, Drink, Food, OtherPROD, Vehicle

Figure 4: Coarse labels present in MultiCoNER I (top) as compared to granular labels in MultiCoNER II (bottom)

The data itself is stored in the CoNLL format, where each token has a label of 'O' for outside if it is not part of an entity, 'B-<label>' if it begins an entity, and 'I-<label>' if it is inside of an entity after it begins. By modeling the samples in this sequential fashion, there is no issue in working with languages that read left to right or right to left. For languages that have a concept of capitalization, all samples have been made lowercase, which removes the ability to use capitalization as an aid in identifying entities.

The dataset was not released until late October, which limited our time to explore and analyze it. In contrast to the stated aims of the task to explore limitations of previous methods which were brittle to out-of-knowledge base entities and noise such as misspellings and typos, none of those challenges were introduced, thus leaving the novelty in the new dataset to a new set of languages and a finer-grained label set. However, as the dataset was created with weak supervision, we observe that not all of the annotations may be accurate.

While MultiCoNER I had a consistent 15,300 training and 800 development samples per language (with the multilingual task a simple combination of the 11 languages), MultiCoNER II has more variability in the exact number of training and development samples per language, however, the ratios remained proportional across languages. Each language has between 9k-16k training sentences and 500-900 development sentences, though there are notably fewer samples for Bangla, Chinese, German, and Hindi than there are for English, Farsi, French, Italian, Portuguese, Spanish, Swedish, and Ukrainian. Across each language and dataset, there are roughly 1.25-1.5 entities per sentence on average. The samples and number of entities in each dataset can be seen in figure 5.

It is worth noting the imbalanced distribution of the named entity types as shown in figure 6.

4 Proposed Methods

4.1 Python Packages

Originally, we proposed using Python packages to introduce spelling errors and other forms of standard data augmentations into our data. However, we 1. Decided to focus on implementing novel data augmentation methods and 2. Did not find any spelling errors in the training or validation data. When the competition organizers were contacted about this, the response was "This edition's main focus is identifying fine-grained entity types and some new languages." In other words, it seems as though they have drastically changed the purpose of the competition since our first proposal. As described below, we decided to continue working on MulDA, but unfortunately, we did abandon the idea of introducing spelling errors as it no longer

language	MultiCoNER I train	dev	test	MultiCoNER II train	dev	test
Bangla (bn)	15,300	800	133,119	9,708	507	...
Chinese (zh)	15,300	800	151,661	9,759	506	...
Dutch (nl) *I*	15,300	800	217,337			
English (en)	15,300	800	217,818	16,778	871	...
Farsi (fa)	15,300	800	165,702	16,321	855	...
French (fr) *II*				16,548	857	...
German (de)	15,300	800	217,824	9,785	512	...
Hindi (hi)	15,300	800	141,565	9,632	514	...
Italian (it) *II*				16,579	858	...
Korean (ko) *I*	15,300	800	178,249			
Portuguese (pt) *II*				16,469	854	...
Russian (ru) *I*	15,300	800	217,501			
Spanish (es)	15,300	800	217,887	16,453	854	...
Swedish (sv) *II*				16,363	856	...
Turkish (tr) *I*	15,300	800	136,935			
Ukrainian (uk) *II*				16,429	851	...
Multilingual (multi) *I*	168,300	8,800	471,911			
Code mixed (mix) *I*	1,500	500	100,000			

language	MultiCoNER I train	dev	test	MultiCoNER II train	dev	test
Bangla (bn)	15,305	800	135,634	13,223	676	...
Chinese (zh)	23,831	1,281	173,183	15,228	773	...
Dutch (nl) *I*	22,331	1,157	265,942			
English (en)	23,553	1,230	272,922	25,449	1,296	...
Farsi (fa)	22,794	1,213	192,194	23,662	1,236	...
French (fr) *II*				26,377	1,352	...
German (de)	23,126	1,239	271,979	15,951	841	...
Hindi (hi)	15,956	828	144,940	12,870	684	...
Italian (it) *II*				26,443	1,398	...
Korean (ko) *I*	24,643	1,302	217,281			
Portuguese (pt) *II*				24,439	1,292	...
Russian (ru) *I*	19,840	1,042	249,458			
Spanish (es)	22,528	1,176	265,748	23,907	1,231	...
Swedish (sv) *II*				25,414	1,391	...
Turkish (tr) *I*	23,305	1,245	149,369			
Ukrainian (uk) *II*				21,956	1,135	...
Multilingual (multi) *I*	237,212	12,513	570,611			
Code mixed (mix) *I*	1,777	610	117,830			

Figure 5: Samples (top) and entities (bottom) in each dataset in MultiCoNER I and II

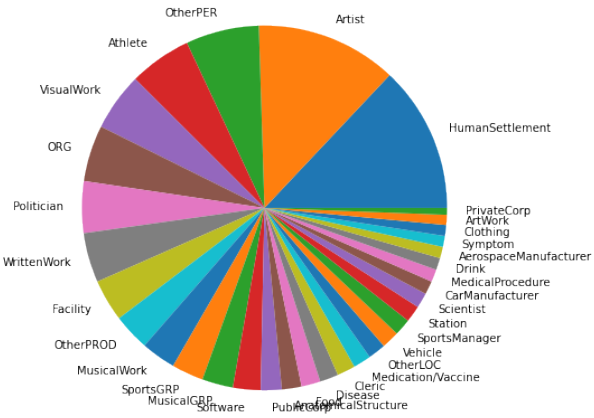

Figure 6: Distribution of label classes in MultiCoNER II across all 12 languages including the train and dev sets

seemed to be a productive use of our time, given the new scope of the competition.

4.2 MulDA

4.2.1 Translation

MulDA (Liu et al., 2021) was one of our proposed ideas which is a data augmentation technique that focuses on multilingual NER. MulDA uses off-the-shelf Google Cloud API as its translation tool which supports more than 100 languages. MulDA introduced a 3-step translation method that replaces named entities with contextual placeholders before sentence translation and then after translation, it replaces placeholders in translated sequences with the corresponding translated entities. See Figure 7 for a detailed example of how this is done.

4.2.2 Generation

In addition to translation, the MulDA paper goes on to use a linearization technique introduced by DAGA (Ding et al., 2020) that adds entity types before sequence tokens after the translation. It then trains an LSTM-based language model based on linearized sequences. This augmentation technique helps to increase diversity by generating synthetic labeled data in multiple languages (Liu et al., 2021). While we imitated the linearization aspect of DAGA in our translations, we do not attempt to do additional language generation. Thus, when referring to MulDA in the rest of the paper, we are referring only to section 4.2.1.

Labeled sentence in the source language:
[PER Jamie Valentine] was born in [LOC London].

1. Translate sentence with placeholders:
src: PER0 was born in LOC1.
tgt: PER0 nació en LOC1.

2. Translate entities with context:
PER0
src: [Jamie Valentine] was born in London.
tgt: [Jamie Valentine] nació en Londres.

LOC1
src: Jamie Valentine was born in [London].
tgt: Jamie Valentine nació en [Londres].

3. Replace placeholders with translated entities:
[PER Jamie Valentine] nació en [LOC Londres].

Figure 7: MulDA's labeled sentence translation where **src** and **tgt** are the source and target languages respectively

5 Data Augmentation Techniques

5.1 Full: Linearized Labeled Sequence Translation

We were introduced to DAGA (Ding et al., 2020) through MulDA (Liu et al., 2021) paper, which is an augmentation method with language models trained on linearized labeled sentences. Linearization is the process of inserting entity tags before the corresponding word as shown in figure 8 (Ding et al., 2020).

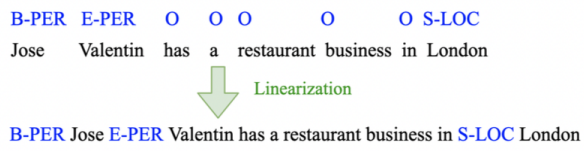

Figure 8: An example of labeled sentence linearization

We took step 3 of MulDA and DAGA's linearization idea and made our translations based on that. For each training sample, we built a linearized sequence, more specifically we used brackets to mark the span of each entity and then translated the sequence to the target language. Figure (6) displays a linearized sequence with brackets that will be sent to the Google cloud for translation. This linearization method helped us to debug the translated texts and find the corresponding words easier. In order to avoid having tags translated, we replaced them with "UNK" tokens and save the tags and their corresponding words in a dictionary and retrieve them after translation.

src:
the county seat is [B-HumanSettlement saint-georges]
What is sent to Google Cloud:
the county seat is [UNK saint-georges]
tgt:
le chef-lieu est B-HumanSettlement saint-georges

Figure 9: An example of using brackets in a linearized sequence

5.2 Partial: Linearized Labeled Sequence Translation

We attempted an additional technique that does not translate the entities. For this technique, we first linearized the text, but instead of putting brackets around the linearized text, we instead put quotes and additional span tags surrounding the text which indicated that the entities should not be translated by Google Translate. For an example of how this was done, see Figure 10.

src: heron was born in [HumanSettlement welwyn garden city] in 1949.

sent to Google Translate:
heron was born in "B-HumanSettlement welwyn" "I-HumanSettlement garden" "I-HumanSettlement city" in 1949.

tgt: heron est né à "B-HumanSettlement welwyn" "I-HumanSettlement garden" "I-HumanSettlement city" en 1949.

output: heron est né à [HumanSettlement welwyn garden city] en 1949.

Figure 10: How the partial method translates text

5.3 Stabilized: Linearized Labeled Sequence Translation

Our final method of data augmentation, Stabilized, is the closest to the original MulDA paper and generates two augmented datasets (Stabilized Full and Stabilized Partial) which can be directly compared with one another.

In the Stabilized data augmentation method, we do not complete Step 1 of MulDA (illustrated in Figure 7). Instead, we begin with Step 2, putting brackets around entities and translating via Google Translate. Then, after translating the full sentence once for each entity, we check to see if the translations match each other. If not, we discard the example, and don't include it in either of the output datasets. We theorize that this "stabilizes" Google Translate, leading to better performance than Full or Partial. That is, if Google Translate gives two different translations just due to brackets around different words in the source text, this likely indicates that Google Translate is not particularly stable or proficient at translating that sentence. Thus, by removing results where Google Translate is not stable, we increase the quality of the dataset.

5.3.1 Stabilized Full

If Google Translate is stable (gives the same translation regardless of which entity is bracketed), then we create two new augmented examples, one for each output augmented dataset. The example for the Stabilized Full dataset simply uses the translated entities from the output of Google Translate. We convert the translated sentence back to the original CoNLL format and add it as an example to the Stabilized Full dataset. This Stabilized Full dataset is the closest dataset to MulDA.

5.3.2 Stabilized Partial

The example for the Stabilized Partial dataset takes the original entities from the source language and uses them instead of the translated entities in the target language. We still keep the rest of the sentence translated; it is only the entities that we replace with the source language entities. The rationale behind doing this is that it will increase diversity in the entities found in the dataset since now entities must be recognized even if they are from a language different than the rest of the text.

5.3.3 Stabilized Summary

Overall, stabilization had a significant effect on the number of examples introduced to the two Stabilized datasets. Out of 16778 examples, a full 4149 examples were dropped due to Google Translate not translating the examples in a stable fashion. (An additional 72 sentences were dropped due to invalid bracketing before translation, and 52 sentences were dropped due to brackets not being found after translation, leading to a total of 4273 dropped examples.) Since the dropped examples are exactly the same between Stabilized Full and Stabilized partial, the two datasets provide a direct contrast to one another; in the Stabilized Full dataset, entities are translated, whereas in the Stabilized Partial, they are not.

5.3.4 Bracket Choice

We also ran a small-scale experiment to determine which type of brackets led to the most stabilization. We iterated through square brackets, curly braces, double quotes, angular brackets, and parentheses. Ultimately, the experiment determined that using double quotes (" ", akin to Partial in Section 5.2) worked the best for stabilization, with only 3481 examples needing to be dropped due to stability in comparison to square quotes' 4149 examples. However, we decided to use square brackets for our full report because 1. It more closely matched the MulDA technique (which uses square brackets) and 2. When translating from English to French, Google Translate modified the quotes in 1823 examples, often times converting them into French quotes known as "les guillemets": « ». Since the quotes were modified, the algorithm could not find the boundaries between the translated entities and the rest of the text, and therefore these examples needed to be dropped as well. Thus, overall, translating using square brackets led to more examples being added to the datasets, despite the decrease in stability.

5.4 Masked Entity Language Modelling

Motivated by the success of masked entity language modeling (Zhou et al., 2022) in low-resource settings (less than 1000 ground truth samples), we propose to use the same technique in a complex named entity setting to provide more variation in the samples. This method substitutes named entities to different ones by formulating it as a variant of masked language modeling. To specify, we will finetune a pretrained XLM-RoBERTa-base to predict the named entities given the input where all named entities are masked out. We further condition this task by including named entity tags before and after the corresponding word as shown in figure 11. During generation, masks in a sentence are greedily recovered one by one to generate coherent samples conditioned on previous predictions.

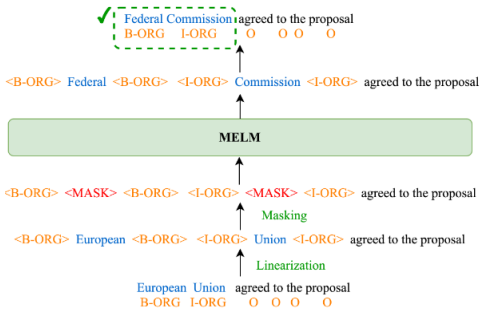

Figure 11: Diagram from MELM (Zhou et al., 2022) explaining their method: masked entity language modeling with linearization.

6 Preliminary Results

We fine-tune a pre-trained named entity recognition model on various sets of data. The named entity recognition model consists of a pre-trained XLM-RoBERTa-base (Conneau et al., 2019) model with a conditional random field classier on top. This setup is derived directly from the baseline model from MultiCoNER 1 and was utilized by all top-performing teams. Per convention, we trained the model using the AdamW optimizer with a learning rate of 1e-5. We trained each model for 20 epochs, which took about 2.5 hours with a single A40 GPU on the Minnesota Supercomputing Institute's Agate cluster. We used macro-averaged validation F1 score as the main evaluation metric for comparing the performance of models trained on various datasets. For the preliminary results, we focused on two languages: French and English.

6.1 Translation

6.1.1 Quantitative Results

	F1	P	R
EN	0.802	0.800	0.804
EN-F	0.794	0.796	0.792
EN-P	0.777	0.766	0.789
EN-S-F	0.805	**0.810**	0.801
EN-S-P	**0.809**	0.795	**0.823**
FR	0.827	0.825	0.829
FR-F	0.817	0.811	0.822
FR-P	0.822	0.828	0.817
FR-S-F	**0.835**	**0.836**	**0.835**
FR-S-P	0.818	0.820	0.816

Table 1: Best validation F1 score, precision at that epoch, and recall at that epoch for French/English + Full/Partial/Stabilized Full/Stabilized Partial.

The stabilized versions of the translation achieved about 0.05 to 0.1 point improvement over the baselines of either language trained on just the training set from MultiCoNER as shown in Table 1. However, to our surprise, our first two methods of translating worsened performance on the validation data. To specify, compared to a model trained only on the training set, the F1 scores for full and partial translations were worse by about 0.05 to 0.1 points for both languages.

6.1.2 Qualitative Results

The three most common failure modes were: predicting a shorter/longer span, completely missing a named entity, and misclassifying a span to a different label. An example of the first is the following: "eli lilly and company" is tagged as a "PublicCorp" in one of our samples, but only "eli lilly and" would be predicted to be "PublicCorp" The second error happens most often when a single word, such as "pulpit" is tagged as a named entity. There isn't a sufficient context for the model to identify that even such a common everyday word can be a named entity. The third error happens most often between tags such as "OtherPer" "Artist" and "Politician." As the 33 entity labels are very fine-grain, with multiple categories of people and groups, the model often correctly identifies that a span is some kind of person, but fail to realize exactly which type of person.

6.2 Masked Entity Language Modelling Results

For masked entity language modeling, we trained the XLM-RoBERTa-base using a masked entity language modeling objective for 40 epochs on the training set. Each took around 4 hours to train. The implementation is still in its early stages and we need to explore more sampling techniques to produce better results.

Contrary to our intuition that introducing different yet likely entities would make the NER model more robust and produce better results, it ended up hurting the performance by quite a bit. To specify, a model trained on French + MELM augmentation had a validation F1 score of 0.81, which is lower than that (0.817) of even the lowest-performing version of the translation (full). Figure 12 shows quite convincing samples produced by the MELM model, so we were surprised to find out such a bad performance.

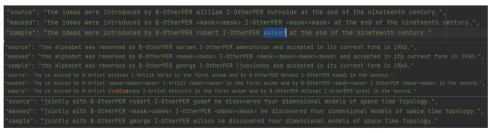

Figure 12: The first four augmented samples from the English data.

7 Limitations

7.1 Google Translate

It turned out that the translation tool we used (Google Cloud) possessed a few limitations that were reflected in our translated files. The first limitation was that we noticed tags such as “PROD-Vehicle" or “MED-Symptom" were also getting translated. To fix this problem, we made a dictionary of words and their tags in the source language, then replaced tags with the “UNK" token so it won’t get translated. Finally, we replaced “UNK" tokens with the correct tags of words from the dictionary. While debugging this problem we also noticed that the order of the words has also changed. For example, the word comes before the tag, therefore we used back translation for words before and after the tag to find the correct corresponding word.

The second limitation was that we realized all words inside brackets were capitalized after translation. The python .lower() method made this problem easy enough to handle. Other problems such as missing the plural “s" after translation and translating some words twice, depending on the language, were also seen through debugging process and were handled.

7.2 Environmental Risks

Increasing the size of the training dataset through data augmentation, not only adds to the development cost in terms of dollars but also to the model’s training time, which results in more CO_2 emissions (Feng et al., 2021).

While some of the cloud-compute companies use carbon credit-offset sources, the majority of their energy is not sourced from renewable sources and many energy sources in the world are not carbon neutral. For the task of machine translation where large LMs have resulted in performance gains, it is estimated that an increase in 0.1 BLEU score using neural architecture search for English to German translation results in an increase of $150,000 dollars compute cost (Bender et al., 2021).

7.3 Cost

Without batching, it takes around 5-6 hours to augment one training set which is incredibly high but we were able to reduce this time significantly by sending sentences in batches to Google Cloud for translation. The other limitation regarding the cost was that Google Cloud API offers free translation for only the first 500,000 characters. After that, you will be charged for $20 per month for every million characters.

7.4 Dataset Annotations

We later realized that some samples in the training sets don’t have correct annotations. For example in the sentence “he holds wins over tito ortiz masakatsu funaki yuki kondo semmy schilt and minoru suzuki,” all entities are athletes and should be considered athletes in context, but for some reason “yuki kondo” is left out as “OtherPER.” This error seemed most common among tags of people.

We brought this up to the competition organizers, who unfortunately informed us that “the dataset is created with weak supervision. So things like these are expected. The annotations are not 100% accurate all the time." We have requested an estimate for the number of inaccurate examples, but have yet to receive a concrete reply.

8 Plan Until the end of semester

8.1 Augmentation Quality

While we have analyzed the performance of the trained NER models, we have not worked much on measuring or guaranteeing the quality and reliability of the augmentations. We will hand-analyze our augmented samples more carefully while looking at model-based techniques such as T-SNE to visualize the distribution of the augmented samples compared to ground truth samples. Then, we will conduct more experiments on other variations of translation techniques (translation models instead of Google Cloud w/ attention map).

8.2 Error Analysis

We analyzed performance in terms of macro-averaged F1 scores, but we need to analyze more fine-grained metrics to identify more exact error modes. As there is an uneven distribution of named entity types, identifying exactly which entities are the easiest and hardest to predict would offer insight into how to improve the model further.

Role Assignments

1. Asal Shavandi: MulDA implementation, Full Linearized Labeled Sequence Translation, Limitations

2. Chahyon Ku: NER model impl. Translation (partial). MELM. Model training and hyperparameter tuning. Result and error analysis. Future work.

3. Josh Spitzer-Resnick: Broader impact. Dataset analysis.

4. London Lowmanstone IV: Translation (stable full and stable partial)

References

Sandeep Ashwini and Jinho D. Choi. 2014. Targetable named entity recognition in social media. *CoRR*, abs/1408.0782.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA. Association for Computing Machinery.

Beiduo Chen, Jun-Yu Ma, Jiajun Qi, Wu Guo, Zhen-Hua Ling, and Quan Liu. 2022. Ustc-nelslip at semeval-2022 task 11: Gazetteer-adapted integration network for multilingual complex named entity recognition.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147, Copenhagen, Denmark. Association for Computational Linguistics.

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.

Bosheng Ding, Linlin Liu, Lidong Bing, Canasai Kruengkrai, Thien Hai Nguyen, Shafiq Joty, Luo Si, and Chunyan Miao. 2020. DAGA: Data augmentation with a generation approach for low-resource tagging tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6045–6057, Online. Association for Computational Linguistics.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. 2021. Ammus : A survey of transformer-based pretrained models in natural language processing.

Linlin Liu, Bosheng Ding, Lidong Bing, Shafiq Joty, Luo Si, and Chunyan Miao. 2021. MulDA: A Multilingual Data Augmentation Framework for Low-Resource Cross-Lingual NER. 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 5834–5846, Online. Association for Computational Linguistics.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022. SemEval-2022 task 11: Multilingual complex named entity recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 1412–1437, Seattle, United States. Association for Computational Linguistics.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual bert?

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152, Sofia, Bulgaria. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *CoRR*, cs.CL/0306050.

Charles Sutton and Andrew McCallum. 2010. An introduction to conditional random fields.

Xinyu Wang, Yongliang Shen, Jiong Cai, Tao Wang, Xiaobin Wang, Pengjun Xie, Fei Huang, Weiming Lu, Yueting Zhuang, Kewei Tu, Wei Lu, and Yong

Jiang. 2022. Damo-nlp at semeval-2022 task 11: A knowledge-based system for multilingual named entity recognition.

Rongzhi Zhang, Yue Yu, and Chao Zhang. 2020. SeqMix: Augmenting active sequence labeling via sequence mixup. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8566–8579, Online. Association for Computational Linguistics.

Ran Zhou, Xin Li, Ruidan He, Lidong Bing, Erik Cambria, Luo Si, and Chunyan Miao. 2022. MELM: Data augmentation with masked entity language modeling for low-resource NER. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2251–2262, Dublin, Ireland. Association for Computational Linguistics.