# Cross-Domain Named Entity Recognition

**Mathis Gravil**
mvgr@itu.dk

**Katarina Kraljevic**
katkr@itu.dk

**Maria Momanu**
mmom@itu.dk

**Sneha Shrestha**
snsh@itu.dk

**Abstract**

Named Entity Recognition (NER) plays a crucial role in extracting and classifying named entities from unstructured text data. However, NER performance can be limited in low-resource domains where labeled data is scarce. In this project, we propose a transfer learning framework for cross-domain NER to address this challenge. Our approach leverages labeled data from a high-resource source domain to improve the performance of NER in a low-resource target domain. We employ a pre-training phase where a model is trained on the source domain data. Furthermore, we also aim to exploit datasets from multiple other domains and find the best ways to add data from cross-datasets to train a pre-trained model for better results. Subsequently, a fine-tuning phase is conducted using the limited labeled data available from the target domain. By adapting the pre-trained model to the target domain, we aim to enhance the model's ability to identify and classify named entities accurately. We evaluate the effectiveness of our transfer learning approach on the target domain and demonstrate significant improvements in NER performance.

## 1 Introduction

Named Entity Recognition (NER) is a fundamental task in natural language processing (NLP) that involves automatically identifying and categorizing named entities, such as names of individuals, organizations, locations, and other entities of interest, in unstructured text data. Despite significant advancements in NER, the performance in low-resource domains remains challenging due to limited labeled data availability. Transfer learning techniques, such as pre-training and fine-tuning, have emerged as promising approaches to address this issue by leveraging knowledge from high-resource source domains. Additionally, incorporating training data from multiple domains has shown the potential in improving NER performance (Xiang Chen, 2023). In this paper, we investigate the effectiveness of transfer learning and multi-domain training in NER. We evaluate different ways of adding efficient training data to a pre-trained model from multiple domains. Therefore from this study, we aim to answer the following research question: How can we improve the performance of cross-dataset NER models on a low-resource target dataset by using highly-resourced domain adaptation?

## 2 Related Work

The lack of annotated data for training a robust model for easy domain adaptation is a weakness of cross-domain NER that has been addressed by a few researchers. One research paper from 2023 (Xiang Chen, 2023) shows that transferring knowledge from multiple source domains instead of only one improves the accuracy of a model tested on the target dataset. (Yinhan Liu and Roberta, 2019) study shows that although the pre-trained language models have made waves in NLP, achieving overwhelming performance in widespread NER datasets, yet they require many domain-related labeled data for training when adapting to the target domains. Although current research in cross-domain NER provides valuable approaches for improving the adaptability of NER models, it is still difficult to find an efficient way to utilize the data available for other domains to train a pre-trained model for low-resource domains. In this project, we examine different approaches to transfer knowledge from highly-sourced datasets and multiple domains for training a NER model to test its performance on low-resource target domains. If successful, such an approach can be used as a guide for future work with low-resource domains.

## 3 Methods

**Baseline -** As a baseline model, RNNs offer the advantage of effectively handling sequential data, making them suitable for NER tasks where the order of words in a sentence is essential for identifying and classifying named entities. RNNs process the input text sequentially, enabling them to capture dependencies between words and make predictions based on the information seen so far. RNNs can be trained on labeled data from a variety of domains from which they can learn to generalize across domains and capture common characteristics of named entities. Therefore we decided to use RNN as a baseline model for this project. To solve the problem of some domains being low-resourced, we aimed to make the most of datasets from other domains following some transfer learning techniques. Initially, a model is pre-trained on a large labeled dataset of the source domain, where NER annotations are available. This pre-trained model learns general linguistic and contextual features that are relevant across domains. As we had five other domains that were smaller as compared to the source domain, we wanted to look for the domain with the most limited data and for which the baseline model performs the worst. Therefore, the baseline model is further trained each time with the training set of the five other domains and then evaluated on the development set. The worst-performing domain is considered the target domain.

After finding our target domain, the next step for us was to find ways to make use of data from other domains to make our baseline model (trained on the source domain and fine-tuned with limited data from the target domain) better. With the goal to more precisely predict the words in the target domain. We emphasized testing effective methods to add the available data for training.

### 3.1 Adding random data - Domain Concatenation

As the dataset for our target domain was small and biased, adding random concatenated data from multiple domains to this limited dataset was our first approach. We expect this to increase the diversity of the data and improve the performance of our baseline model as diverse data helps in generalization. However, keeping in mind that adding too much random data can actually harm the performance of the model and it's important to strike a balance between diversity and relevance.

### 3.2 Active learning by re-annotating uncertain words

Uncertainty sampling is a popular active learning strategy, which involves selecting the data points that have the highest entropy or uncertainty according to the model's prediction. In NER, this means selecting the data points that the model is most uncertain about. This uncertainty can come from ambiguity in the word. Therefore the probability of a word belonging to a certain class was calculated for each possible named entity label and finally the entropy for each data point was computed by summing the negative logarithms of the probabilities. A threshold of 2 was kept to focus on all the uncertain words in the training set of the target domain that were above the given threshold. Then, the uncertain words were re-annotated by the group members, and the data was fed again into the model expecting the model to improve its overall performance.

### 3.3 Similarity criterion

Adding data based on a similarity criterion was another attempted approach in this project. For this, the feature vectors of the data point in the target domain and those in the other domains were computed. Then a cross-table was generated that outputs the Euclidean distance between the target domain and the other domains. We fed the model with both the most similar and most dissimilar domains to evaluate the similarity and relevance of the added data. An assumption made is that adding data from dissimilar domains improves the generalization ability of the model as introducing diverse data points can help the model learn to identify and classify a wider range of patterns and features, which can make it more robust to different types of inputs. Another assumption is that adding data from a similar domain helps in leveraging knowledge from related domains from which the model can learn patterns that are relevant to the target domain, improving its ability to make accurate predictions.

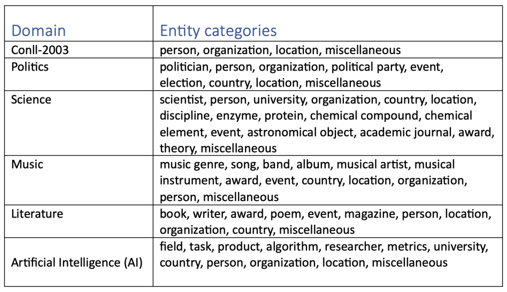
### 3.4 Normalizing domain-specific labels

In the context of NLP tasks, including NER, the presence of domain-specific labels poses challenges for model generalization and transferability. Domain-specific labels are tailored to specific applications or domains, which hinders the application of NER models to new domains. To address this limitation, we normalized domain-specific labels and transformed them into a standardized set of more general labels that are applicable across domains.

## 4 Experiments and Results

### 4.1 Data

As we were working on cross-domain NER and required datasets from multiple domains, we decided to use the CrossNER (Cro) dataset, which contains labeled data spanning over five diverse domains like AI, music, science, literature, and politics with domain-specific named entities. Additionally, it also includes a big dataset conll-2003 which is taken as the source domain in our project. For our source domain, the named entities classes are such: Location (B/I), Person (B/I), Organization (B/I), Misc (B/I), and anything else is labeled as O. B signifies the beginning of the area of interest and I signifies Inside the area of interest but is not the first word. As an example, for “United States” United would be classified as B-location and States as I-location. The entity classes for the other domains look as follows:



| Domain | Entity categories |
|---|---|
| Conll-2003 | person, organization, location, miscellaneous |
| Politics | politician, person, organization, political party, event, election, country, location, miscellaneous |
| Science | scientist, person, university, organization, country, location, discipline, enzyme, protein, chemical compound, chemical element, event, astronomical object, academic journal, award, theory, miscellaneous |
| Music | music genre, song, band, album, musical artist, musical instrument, award, event, country, location, organization, person, miscellaneous |
| Literature | book, writer, award, poem, event, magazine, person, location, organization, country, miscellaneous |
| Artificial Intelligence (AI) | field, task, product, algorithm, researcher, metrics, university, country, person, organization, location, miscellaneous |

Figure 1: Domain-specific entity categories

For each domain, we have three sets of data – training, development, and test set. The training set is used to train the model, the development set to assess the model’s parameters, and the test set to finally test the model’s performance. As part of the pre-processing, we decided to penalize the weight of the majority class O in our datasets as we had an imbalance in the labels.

### 4.2 Evaluation metric

Since the datasets we use in the project are imbalanced, f1 score was chosen as an evaluation metric to measure performance. In imbalanced datasets, the accuracy score can be misleading because the model may achieve high accuracy by simply predicting the majority class for the majority of instances while performing poorly on the minority class. F1 score is the harmonic mean of precision and recall, which gives much more weight to minority classes.

### 4.3 Experiment Setup

For this project, we chose to run two experiments in parallel to get a better idea of what affects cross-domain NER. In the first experiment we used the raw data with all the domain-specific labels and we tested the three methods mentioned in the Methods section (random data augmentation, dis/similar data augmentation, and re-annotation based on uncertainty). In the second experiment, we wanted to show how our model would perform if we got rid of the complexity of having domain-specific labels. For that, we normalized the labels and ran the same data augmentation methods. To do the normalization we used parts of the MultiCoNER II Taxonomy (https://multiconer.github.io/) to bring our domain-specific labels to higher-level generic labels. For example in the science domain, we have a label for "B/I-scientist", all the samples in the data that were categorized as a scientist were then changed to "B/I-person". For all labels that were not about persons, locations, or organizations we categorized them as miscellaneous. By doing this we increased the f1-score of the worst-performing model by a factor of 3.

### 4.4 Random data results

The first method to try and improve performance is to just add more training data to your model. The non-biased way to do that is by adding random data. What we did is we merge all the data together and used the "random" package to pick random samples. We chose to add data-based percentages. So adding 10% more data all the way to the maximum of our datasets would allow us to do which is 60%. We did that to get a better understanding of the improvement in performance and what the optimal amount of added data would be. Here are the results :

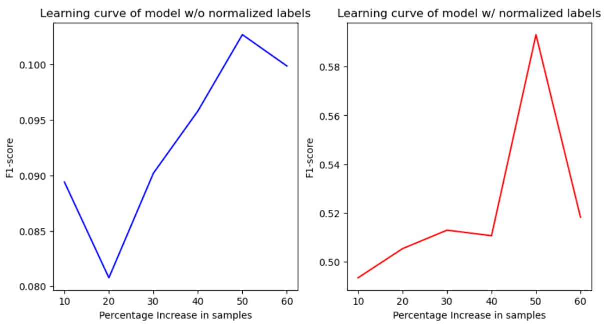


Figure 2: Learning curves

The learning curves of our baseline model in figure 2 shows that adding 50% of additional random data was optimal no matter we normalize the labels or not.

### 4.5 Adding most similar/dissimilar data results

Using the Euclidean distance we found out that the two most different datasets from AI are politics (530.5) and science (383.7). The most similar ones are literature (145.2) and music (150.6).
Our experiments revealed that adding dissimilar data makes the baseline model more efficient when the labels are not normalized however feeding the model with additional training data from the most similar domain was seen to be better after standardizing the entity labels. Despite this, the disparities in outcomes were minimal when the data quantity was small. Consequently, opting to include 50% more data, specifically from the most similar domains, after label normalization proved to be a beneficial decision.

### 4.6 Re-annotating uncertain words using entropy results

After using the uncertainty sampling, two models were compared on the test file: one is a baseline model trained with the re-annotated file, and the second model is trained with the same file re-annotation was done. Their performance was compared by calculating the accuracy score using the test file golden labels.
The baseline model trained with the file before re-annotation resulted in an 0.11 accuracy score, while the model trained with the re-annotated file achieved an accuracy score of 0.23. This shows that uncertainty sampling can be an effective technique in enhancing the NER models performance for low-resourced data.

### 4.7 Comparing the results

## 5 Analysis

Now that we have tried all our methods to improve performance we can analyze the results and prepare the answer to the research question. The method which improved our f1-score the most was normalizing the labels, in fact, that tripled our f1-score for the worst-performing model. That is pretty intuitive because in our dataset we have many domain-specific labels that are not present in the ConLL-2003 dataset nor in the 4 other smaller datasets. The accuracy of our model has also significantly improved. Accuracy is an important metric to measure a model's overall performance, but can at times be misleading if a class imbalance exists in the dataset. To better evaluate and visualize our model's performance, we created a confusion matrix (Figure 3) of the predicted labels. Analyzing it, we clearly notice a class imbalance between our labels.

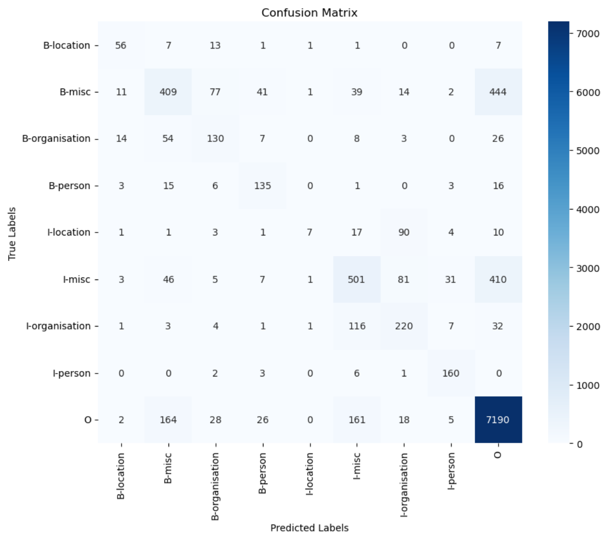
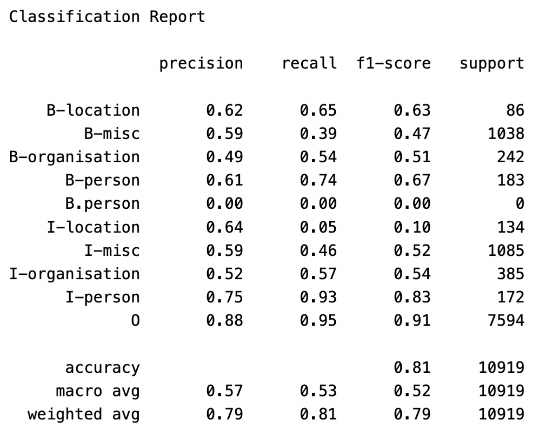


Figure 3: Confusion matrix

In order to obtain a more comprehensive understanding of how well our model performs, we made a classification report (Figure 4) in which we analyzed the important performance metrics for each label. While the general accuracy score we got is a fairly good one, we can see that our model lacks in correctly predicting certain labels. For example, the I-location label has an f1 score of 0.1, which indicates a defective trade-off between precision and recall. On the other hand, however, there is also a disproportion between the number of ordinary words (class O) and that of named entities, which is way smaller. However, we have good confidence in our model's ability to correctly differentiate between named entities and common words. Another label for which the model performed well is I-person: out of the 172 instances, 160 were correctly classified.



```
Classification Report

                precision    recall  f1-score   support

     B-location       0.62      0.65      0.63        86
         B-misc       0.59      0.39      0.47      1038
B-organisation       0.49      0.54      0.51       242
       B-person       0.61      0.74      0.67       183
       B.person       0.00      0.00      0.00         0
     I-location       0.64      0.05      0.10       134
         I-misc       0.59      0.46      0.52      1085
I-organisation       0.52      0.57      0.54       385
       I-person       0.75      0.93      0.83       172
              O       0.88      0.95      0.91      7594

       accuracy                           0.81     10919
      macro avg       0.57      0.53      0.52     10919
   weighted avg       0.79      0.81      0.79     10919
```

Figure 4: Classification report

## 6 Conclusion

From all the results gathered, we can conclude that the most important step to improve the performance of cross-dataset NER models on low-resource datasets with domain-specific labels is to normalize the labels to labels that are also present in your high-resource training dataset. This has such a big impact because adding labels adds complexity to the model. When we add random data to the model we end up with 80 unique labels that the model can assign. That is too much for our model not to get confused, therefore normalizing the labels is an interesting step to improve cross-dataset NER models. We could categorize this step as data transformation. In this project, we also looked at data augmentation techniques to improve our performance.
We have found out that for our model, which has a limited size of 14141 sentences as initial training data (ConLL-2003 train.txt &¨AI train.txt), adding the most amount of data in the training brings us the best performance improvement. In fact, adding 50% more training data gives us an f1-score of 0.59. In other words an improvement of 68.6% from our baseline.
Regarding the question about similarity-based data augmentation, we found out that adding dissimilar data helped our model when we normalized the model but when the model input domain-specific labels similar data helped the performance. In both cases adding the counterpart also improved our performance.
Additionally, the process of uncertainty sampling and file re-annotation, showed improvements in the resulting accuracy scores. The increased performance of the model says that this method can have positive results for improving NER models, especially for domains with scarce datasets.

## References

https://github.com/zliucr/crossner/tree/main/ner_data.

Shuofei Qiao Ningyu Zhang Chuanqi Tan Yong Jiang Fei Huang Huajun Chen Xiang Chen, Lei Li. 2023. One model for all domains: Collaborative domain-prefix tuning for cross-domain ner.

Naman Goyal Jingfei Du Mandar Joshi Danqi Chen Omer Levy Mike Lewis Luke Zettlemoyer Yinhan Liu, Myle Ott and Veselin Stoyanov. Roberta. 2019. A robustly optimized bert pretraining approach. arxiv preprint arxiv.