Data Augmentation in 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 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, we will conduct a fine-tuning phase 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. Our results shows that all the approaches employed in the project had a positive impact on the performance of the model.

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, 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 made 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, 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, adding too much random data can actually harm the performance of the model. 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. This uncertainty can come from ambiguity in the word or ambiguity in the class. In NER, uncertainty sampling means selecting the data points that the model is unsure about and has a hard time giving a confident classification for.
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 1.75 was kept to focus on all the uncertain words in the training set of the target domain that were above the given threshold. We chose that threshold because we wanted to re-annotate about 10% of the training data to capture the most uncertain words.
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. If the group was not aligned on a label we chose to use its golden label and included it into the re-annotated data. In the hypothetical case that the data wouldn't have the golden labels, we think a good approach would be to discuss between the annotators. If after discussion an agreement still cannot be reached the data point should be labeled as "O".

3.3 Similarity criterion

Adding data based on a similarity criterion was another attempted approach in this project. The feature vectors of the data point in the target domain and those in the other domains were computed. Then a cross-table (Appendix, fig 6) 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 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 knowl-

edge from related domains from which the model can learn patterns that are relevant to the target domain, improving its ability to make accurate predictions.

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 (Liu et al., 2020) dataset, which contains labeled data spanning over five diverse domains: AI, music, science, literature, and politics with domain-specific named entities. Additionally, it also includes a high-resourced 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 Data Preprocessing

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. This means that we used parts of the MultiCoNER II Taxonomy (Shervin Malmasi, 2022) 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.

4.3 Evaluation metric

Since the datasets we use in the project are imbalanced and we are predicting relational-based data, the span-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. The span-f1 score is the harmonic mean of precision and recall, which gives much more weight to minority classes.

4.4 Random data results

The first method to try and improve performance is to add more training data to our model. The non-biased way to do that is by adding random data. We merge all the data together and shuffle the sentences, not the words so that we don’t lose positional information. As we know machine learning algorithms usually have a learning curve. For that reason, we chose to test multiple amounts of data augmentation. Firstly training our algorithm with 10% more data. We tested adding data in increments of 10% all the way to 60% because that is the maximum amount of data augmentation our datasets would allow. Here is the learning curve in question :

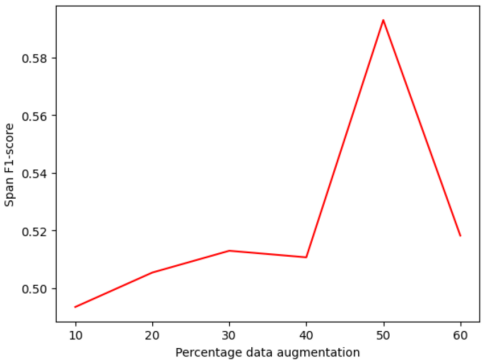

Figure 2: Learning curve for random data augmentation

The learning curve of our baseline model in Figure 2 shows that adding 50% of additional random data was optimal. The biggest impact in performance happened from no added data to 10% more training data (+0.12 span f1-score) and our best-performing model is the model with 50% more training data which has a span f1-score of 0.59. Surprisingly, after adding 60% more training data we get performance that is similar to adding 40% more data. That behavior could be due to the fact that with 60% more training data, we are generalizing too much and we are introducing too much diversity into the model.

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).
From our previous data augmentation method we saw that adding more training data does not necessarily result in better predictions. Now we need to find the best way to improve our score without having to add 50% more training data. We tested four different tasks: adding only the most similar or dissimilar and then the two most similar or dissimilar datasets. Our experiments revealed that adding the two most dissimilar training datasets makes the baseline model's span f1-score go from 0.38 to 0.56. The results of all experiments are shown here:

	Most Similar	Most Dissimilar
Single Dataset	0.48228	0.49470
Two Datasets	0.50089	0.56336

Figure 3: Comparing entropy-based data augmentation

Compared to adding random data our model performs the same but instead of having 21.211 training sentences we only use 14.541 (with the baseline having 14.141). That makes a big difference for our task because we are trying to improve performance in low-resourced domains, which means that we don't have access to a lot of annotated data. We can confidently say that adding data based on a high Euclidean distance between the two datasets is more effective than adding random data. Indeed we are adding only 3% more training data compared to the 50% of the random data.

4.6 Re-annotating uncertain words using entropy results

For our last experiment, we calculated the entropy score of the predictions made on the development dataset. We re-annotated the predictions with the highest entropy score and fed them as training data. Our final test was done on the testing dataset so data that was never seen by the algorithm before. For us to be able to compare the results we also had to run our baseline model on the testing data to get an idea of the base f1-score. Additionally, we trained our baseline model with the predictions made on the development dataset without the re-annotations and ran it on the testing data to see how much the sole re-annotation improves the model. Our baseline had an f1-score of 0.39, the baseline trained with the development dataset performed with a score of 0.47, and the final model with 0.54.

5 Analysis

We have found that all our experiments help to improve our predicting performance. By adding a lot of random data we are adding diversity and improving our model by 68% in the best case. For the dis/similarity data augmentation, we can see that the two assumptions we had were true with the dissimilar data having a slight advantage. Using a lot less data we increase our f1-score by 54% from our baseline. Finally, our entropy-based data augmentation yielded results that were also in the same range as the other two experiments slightly better than the similarity-based data augmentation with an improvement of 63% from the baseline.
The accuracy of our evaluation 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 overall performance, we created a confu-

sion matrix (Figure 4) of the predicted labels. We are also able to observe our model's performance for individual classes and identify patterns. There are differences in how our models recognize the beginning (B) and the inside (I) of entities, which are not always consistent. Analyzing the confusion matrix, we clearly notice a class imbalance between our labels. When we created a confusion matrix ignoring the majority class "O" (Appendix - Figure 6) we got a clearer picture of our model's ability to differentiate between named entities.

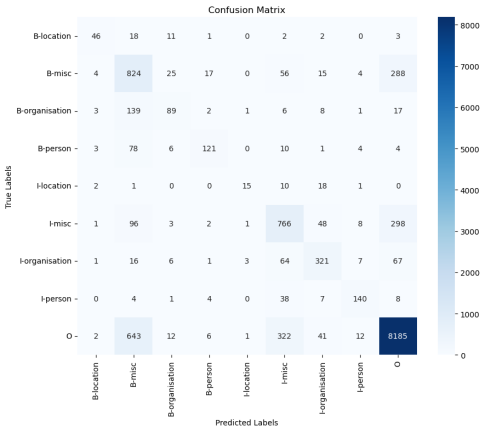

Figure 4: Confusion matrix

In order to obtain a more comprehensive understanding of how well our model performs, we made a classification report (Figure 5) 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 "B-organisation" label has an f1 score of 0.42, which indicates a defective trade-off between precision and recall. On the other hand, there is also a disproportion between the number of words outside of named entities (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 202 instances, 140 were correctly classified.

	precision	recall	f1-score	support
B-location	0.74	0.55	0.63	83
B-misc	0.45	0.67	0.54	1233
B-organisation	0.58	0.33	0.42	266
B-person	0.79	0.53	0.64	227
I-location	0.71	0.32	0.44	47
I-misc	0.60	0.63	0.61	1223
I-organisation	0.70	0.66	0.68	486
I-person	0.79	0.69	0.74	202
O	0.92	0.89	0.90	9224
accuracy			0.81	12991
macro avg	0.70	0.59	0.62	12991
weighted avg	0.83	0.81	0.81	12991

Figure 5: Classification report

6 Conclusion

Our project investigated different approaches to improve the performance of cross-domain NER models on a low-resource target domain. We used transfer learning techniques, including pre-training and fine-tuning, to get knowledge from high-resource source domains. In order to improve the model's generalization across domains and capture important patterns and features, we also investigated the effect of adding data from diverse domains. Our results showed that adding random data from multiple domains improved the model's performance, with the optimal improvement achieved when adding 50% more training data. We also found that adding data from dissimilar domains had a more positive impact on performance compared to adding data from similar domains. Furthermore, we explored active learning by re-annotating uncertain words that the model was unsure of its predictions. This approach resulted in a slight improvement in model performance, indicating the effectiveness of this strategy in enhancing NER performance. Overall, our findings demonstrate the potential of transfer learning and multiple-domain training in improving cross-domain NER performance.

7 Appendix

	conll	ai	literature	music	politics	science
conll	0.000000	11997.009752	11932.657416	11971.230137	11578.892218	11676.832790
ai	11997.009752	0.000000	145.161978	157.391232	530.475259	383.701707
literature	11932.657416	145.161978	0.000000	150.552316	468.542421	322.547671
music	11971.230137	157.391232	150.552316	0.000000	505.155422	367.381273
politics	11578.892218	530.475259	468.542421	505.155422	0.000000	301.948671
science	11676.832790	383.701707	322.547671	367.381273	301.948671	0.000000

Figure 6: Cross-table of Euclidean distance between domains

7.1 Group Contributions

The work in this project was done mostly as a group sitting next to each other. There were no big discrepancies in the workload as we all helped each other with the tasks. We divided the writing of the report but besides that, we all worked on the experiments.

7.2 Usage of AI

For this project, we didn't make use of AI chatbots apart for the creation of the ReadME file inside our GitHub.

References

Zihan Liu, Xu Yan, Tiehzheng Yu, Wenliang Dai, Ziwei Ji, Samuel Cahyawijaya, Andrea Madotto, and Pascale Fung. 2020. Crossner: Evaluating cross-domain named entity recognition.

Besnik Fetahu Sudipta Kar Oleg Rokhlenko Shervin Malmasi, Anjie Fang. 2022. Multiconer: A large-scale multilingual dataset for complex named entity recognition.

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 Veselin Stoyanov. Roberta Yinhan Liu, Myle Ott. 2019. A robustly optimized bert pretraining approach. arxiv preprint arxiv.