Boosting Relation Classification Model Performance with Domain-Specific Information: A Comparative Analysis of Special Tokens and Dataset Embeddings

Franciszek Liszka and **Jannik Elsäßer** and **Juraj Septak** and **Mirka Katuscakova**
IT University of Copenhagen
{frli, jels, juse, katu}@itu.dk

Abstract

This project explores the integration of domain-specific information into pre-trained Natural Language Processing (NLP) models to enhance their performance in various tasks. Domain knowledge is crucial in capturing relevant nuances and context, and this study investigates two techniques for incorporating domain-specific information: special tokens and dataset embeddings. Our experimental results show that both techniques, special tokens and domain embeddings, can indeed improve the performance of the NLP models. [1]

1 Introduction

Natural Language Processing (NLP) tasks often require domain-specific information to achieve optimal performance, as the domain knowledge helps models capture relevant nuances and context. This project investigates the incorporation of domain-specific information into pre-trained models and its effect on their performance in various tasks.

Fine-tuning pre-trained models on domain-specific datasets is a popular approach for adapting models to different domains. Although these models may perform well on their respective domains, they might not generalize well to new domains due to the lack of explicit domain information representation. As a result, the model's performance could be deficient when applied to novel domains.

We assessed two distinct techniques for overcoming these limitations:

1. Special tokens: By introducing special tokens that explicitly represent domain information, we wanted to allow models to learn domain-specific information during training. With this approach we aimed for a more flexible and robust model that can adapt to new domains more effectively.
2. Dataset embeddings: Alternatively, we examined representing each domain with a learned vector (i.e., domain embedding). This vector should capture more nuanced information about the domain, such as specific topics or language use, which may lead to more accurate and adaptable models without requiring additional training data.

Our main research question can be formulated as follows:

RQ Given a multi-domain training setup, is it possible to improve the performances by adding information about the domain itself?

2 Related work

In the context of multilingual parsing and monolingual analysis using treebanks, there are several studies thatinvestigate the benefits of incorporating dataset embeddings into the model architecture to improve the accuracy and performance of NLP tasks across different languages and domains.

Smith et al. (2018) demonstrated the benefits of training models on clusters of related languages using dataset embeddings. They observed substantial accuracy gains over training individual models for each language. Stymne et al. (2018) explored dataset embeddings in the monolingual context using heterogeneous treebanks. They compared the dataset embedding method to other techniques but did not specifically focus on incorporating domain-specific information. In our research, we aim to explicitly introduce domain information into the model architecture, allowing for better adaptation to specific domains. In their study, they found the dataset embedding method to be superior to all other methods.

Wagner et al. (2020) investigated whether dataset embeddings can be useful in an out-of-domain scenario where the treebank of the target data is un-

[1] Source code is freely available at: https://github.com/jannik-el/CrossRE-Exp.git

known. They found that it is possible to predict dataset embeddings for such target treebanks, making the method useful in this scenario.

van der Goot and de Lhoneux (2021) investigated the effectiveness of dataset embeddings in the context of training multilingual parsing models and explored different methods of incorporating dataset embeddings into the encoder of LM-based parsers. They have shown that encoder embeddings (incorporating dataset embeddings into the language model parameters at the token level, allowing them to be considered throughout the transformer layers) have outperformed the decoder embeddings (concatenating the dataset embedding to the output of the language model for each wordpiece before passing it to the decoder). This project has also utilized this approach.

In the field of protein toxicity assessment, ToxDL, proposed by (Pan et al., 2020), utilizes domain embeddings along with primary protein structure for predicting protein toxicity. One protein contains multiple associated domains that determine its function, and as some domains may frequently co-occur to function together, the authors incorporate protein domain embeddings to a convolutional neural network for variable-length input sequences.

Contrary to this project, the authors trained a Skip-gram model to learn domain embeddings. The final embedding vector for a protein was obtained by averaging the embeddings of all domains associated with that protein.

3 Methodology

3.1 Data description

For evaluation, the CrossRE dataset was chosen on account of its suitability for cross-domain evaluation of RE models. The benchmark includes 6 domains: news, politics, natural science, music, literature, and artificial intelligence. The specific choice of the domains in CrossRE was influenced by previous work, as they show to contain diverse vocabularies, and thus can be used for evaluating the performance of RE models across a wide range of contexts (Bassignana and Plank, 2022a).

The dataset contains over 5,000 sentences that were hand-annotated using 17 relation types and allowing for multi-label annotation. Across all domains, 6% of relations make use of multi-labels.

	SENTENCES				RELATIONS			
	train	dev	test	tot.	train	dev	test	tot.
📰	164	350	400	**914**	175	300	396	**871**
🏛	101	350	400	**851**	502	1,616	1,831	**3,949**
🌿	103	351	400	**854**	355	1,340	1,393	**3,088**
♫	100	350	399	**849**	496	1,861	2,333	**4,690**
📕	100	400	416	**916**	397	1,539	1,591	**3,527**
🤖	100	350	431	**881**	350	1,006	1,127	**2,483**
tot.	**668**	**2,151**	**2,446**	**5,265**	**2,275**	**7,662**	**8,671**	**18,608**

Table 1: Distributions of sentences and relations across domains in CrossRE dataset. Please see icon legend in Appendix C

3.2 Data preparation and pre-processing

The dataset was in a JSON format, from which the entity and relation information had to be extracted during pre-processing:

```
{
  "doc_key": "ai-dev-1",

  "sentence": ["Here", ",",
  "accuracy","is", "measured",
  "by", "error", "rate", ",",
  "which", "is","defined",
  "as", ":"],

  "ner": [[2, 2, "metrics"],
  [6, 7, "metrics"]],

  "relations":
  [[6, 7, 2, 2, "part-of",
  "", true, false]]
}
```

The entity information was extracted from the "ner" key of the input JSON. In the above-mentioned case, there are two entities:

Entity 1: The entity spans from token index 2 to 2 ("accuracy") and has the label "metrics".

Entity 2: The entity spans from token index 6 to 7 ("error rate") and also has the label "metrics".

The relation information was extracted from the "relations" key. In this case, there is one relation: Entity 1 is a part of Entity 2. The relation label is "part-of". Afterwards, all possible entity pairs were generated, and for each of them a marked sentence was built with entity start and end markers - `'<E1:entity_1>'` and `'</E1:entity_1>'` into the original sentence. The resulting marked sentence represents the position of the entities within the sentence:

```
"Here, <E1:metrics> accuracy
```

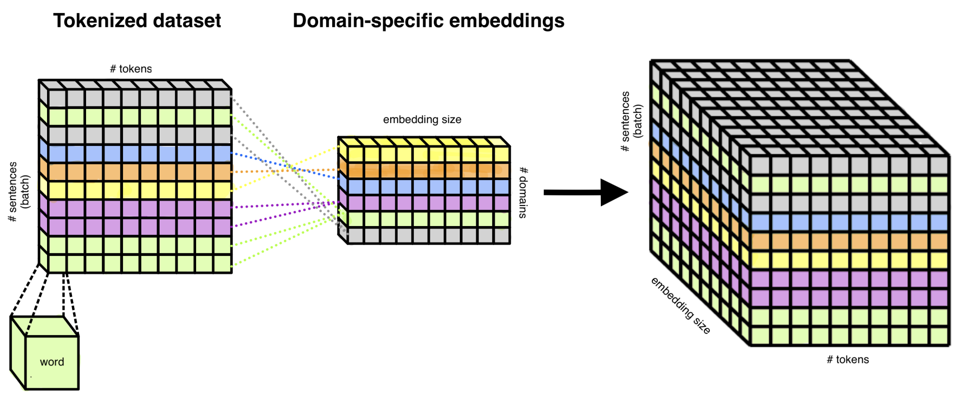

Figure 1: Domain-aligned embedding tensor construction.

```
</E1:metrics> is measured
by <E2:metrics> error rate
</E2:metrics> , which is
defined as :"
```

Finally, the relation label was retrieved and associated with the entity pair. Each marked sentence, the indices of entity markers within the sentence, and the relation label are then appended to their respective lists. If multiple labels were present, the relation label was converted to a one-hot encoded vector and appended to the relations list. Otherwise, a unique identifier of the label was appended.

3.2.1 Dataset Experiments

To properly measure how well each experiments model learns each domain from another, and also generalizes well across certain domains, a decision was made to create train, dev and test datasets where domains were combined. The results of these tests are described as combined dataset results. The BiLSTM model was not run on this combined dataset since it did not prove effective, and there it's results are based on training for each domain, similar to the CrossRE (Bassignana and Plank, 2022a) paper's experiment setup.

3.3 Feature Engineering

3.3.1 Domain Tags

By adding special tokens ([MUSIC], [AI], [LITERATURE], [NEWS], [POLITICS], [SCIENCE]) the tokenizer is made aware of the domain-specific labels and concepts, allowing the model to recognize and process them during training.

The domain information has been extracted from the 'doc-key' field in the JSON input file. For these domain tags to work correctly with the transformer model, we specifically declare them as special tokens and add them to the tokenizer's vocabulary. In order for the language model to effectively handle the newly introduced tokens, the token embeddings of the language model were resized to match the updated tokenizer size. This approach allows our model to grasp the explicit context provided by the domain tags, possibly leading to better model performance.

3.3.2 Domain Embeddings

To enhance the model's understanding of domain context, a numerical representation of each domain was incorporated into the architecture, allowing it to be learned during the training process. The objective was to equip the model with the ability to differentiate between domain-specific nuances.

In the initial phase, each domain has been assigned a domain-specific embedding. An embedding layer has been initialized, producing the output shaped as per number of domains by the size of the embedding. Following this step, an empty domain aligned tensor has been constructed of a shape (batch size, maximum sentence length, embedding size). When iterating over each sentence's attention mask, if the mask signifies a token entry, the corresponding entry in the above mentioned tensor is substituted with the domain-specific embedding that aligns with the domain of the sentence, as shown in Figure 1.

After completing the embedding injection for the token entries, an element-wise addition between the domain aligned embedding tensor and the original token embeddings is conducted. This operation integrates the domain-specific embeddings into each token embedding. The enriched tensor then undergoes the pooling operation to yield sentence embeddings, which are later processed by the linear Label Model.

To make the domain embeddings more interpretable and provide an intuitive understanding of the spatial relationships between domains, Principal Component Analysis (PCA) has been used to reduce the dimensionality of these embeddings down to three dimensions (as shown in figure 2). Some domains appear more clustered together, suggesting similar contextual structutes, while other are positioned further apart, showing more distinct domain characteristics.

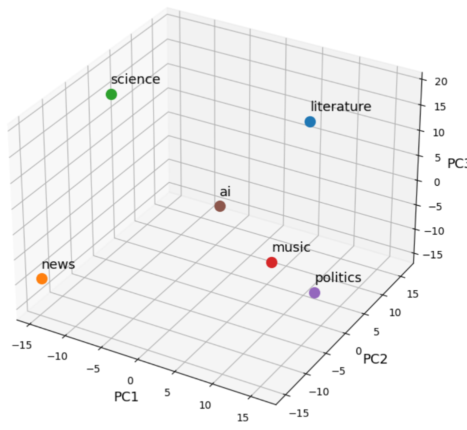

Figure 2: 3D Projection of Domain Embeddings

3.4 Model Training

For information on what parameters were used for training see "Reproducibility" (Appendix A).

3.5 Model Architecture

Our model follows the architecture used by Bassignana and Plank (2022a). The input sentences are first passed through the Embeddings Model, where a pre-trained BERT (Devlin et al., 2019) transformer (bert-base-cased) model is being used. This model transforms each sentence into a sequence of token embeddings, having dimensions determined with maximum sentence length by embedding dimension. Afterwards, a marker embeddings pooling is performed based on both entities in the sentence. Following the pooling step, the sentence embeddings are passed to a Label Model, a linear layer responsible for mapping the embeddings to logits. The output shape corresponds to the number of label types. The output of the model is the predicted labels, which are determined by selecting the highest logit value for each sentence.

3.6 BiLSTM layer

To improve model performance, a decision was made to implement a BiLSTM (Bidirectional Long-Short Term Memory) layer in the model architecture. BiLSTMs effectively increase the amount of information available to a model, improving the context available to the algorithm (e.g. knowing what words immediately follow and precede a word in a sentence). Importantly for us, a BiLSTM layer has improved model performance in other relation extraction experiments (Ni et al., 2021). Our hope was that the extra information created by the BiLSTM layer would allow the linear layer to make better relation type predictions.

4 Experiments

To properly compare the performance of the different models the ablation study (a quantitative form of analysis) was carried out. Within the scope of this investigation this specifically meant the following: A baseline study was carried out, where by the CrossRE Relation Classification model from Bassignana and Plank (2022a) was tested on a multi-domain dataset, and it's performance evaluated based not only on its prediction accuracy of the different relation types, but also it's accuracy across different domains, to see if the model was generalizing well across multiple domains. Subsequent attempts were then made to improve the model's performance through the implementation of two distinct feature engineering methods Domain Tags and Domain Embeddings. As such, in the results three different experiment names are shown:

- "Baseline", the RC model from Bassignana and Plank (2022a)
- "BiLSTM", the Baseline model but with a BiLSTM layer
- "Domain Tag", based on the baseline model but added with domain tag features
- "Domain Embedding", also based on the baseline model but with domain dataset embeddings added to the model input embeddings

4.1 Results

As shown in the Table 4, the implementation of domain tags demonstrates its effectiveness by improving performance across nearly all domains.

	Domain	Micro F1	Macro F1	Weighted F1
baseline	📖	**67.48**	37.68	**63.00**
	♫	**76.52**	40.36	**73.90**
	📰	42.73	15.97	33.12
	🏛	57.87	20.84	53.23
	🍃	40.27	25.62	36.38
biLSTM	📖	**67.04**	33.33	**61.70**
	♫	**74.30**	34.93	**71.58**
		40.56	8.38	26.66
	🏛	54.05	17.08	48.04
	🍃	36.74	19.58	31.35

Table 2: Baseline model outperforms baseline with LSTM across all domains.

Domain tags implementation does however not improve performance for the domain "news", specifically when it comes to the micro F1 and weighted F1 score (the macro F1 score is still the highest).

Domain tags is also slightly outperformed by the baseline model in the domain "politics", however only in the micro F1 scores.

In contrast, domain embeddings seem to not significantly improve performance, and in most cases worsen performance, with the exception of one noteworthy case. In the domain "news" and the metrics micro F1 and weighted F1, domain embeddings outperform both baseline and domain tags models.

Implementing the BiLSTM unfortunately also did not prove effective, and only worsened scores across all domains (see Table 2). It was also observed that training time increased, therefore this combined with no performance improvement lead to the decision being made to not test the BiLSTM on the combined domains dataset.

4.2 Analysis

The hypothesis for why implementing domain tags should help improve relation extraction models improvement, was that the domain tags would serve as a contextual cue for the model. By incorporating domain specific information, the model should be able to adapt to a domains definitive nuances and characteristics, helping it to better understand and leverage domain-specific knowledge.

As shown in results, this was indeed true, with the domain tagged model performing level best in all domains, with the exception of one case. The lower micro and weighted F1 scores within the "news" domain, are postulated to have been caused by the model being effected by imbalanced class distributions. As evident from Table 1, the news domain has a significantly lower amount of relations than all the other classes.

Likewise for domain dataset embeddings, it was expected they would have the same effect, with the added bonus of being embeddings that were learned by the model during training, allowing for a much more specific domain dataset embedding.

Domain embeddings did not perform well, with the exception of one noteworthy case. Domain embeddings outperformed all other models within the "news" domain. As previously mentioned, it is believed that the worsened performance of the other models was caused due to the domain tags model being effected by imbalanced class distributions.

5 Discussion

One of the outcomes of this study was the previously mentioned discovery that domain embeddings performs better than the baseline and domain tag models, where there are significant class imbalances. Finding this exhibits significant practical utility, in the context of current LLM development. A drive is currently being made to develop models which perform well, but on a significantly smaller datasets (Peng et al., 2020). The analysis suggests that this indicates that domain embeddings may perform quite well when working with significantly larger test sets that exhibit greater imbalanced class distributions.

6 Conclusions

In summary, drawing upon the results of the conducted experiments and tests, the following conclusions can be drawn:

- Domain tagging boosts relation classification model performance, and helps it generalize well across multiple domains.

- Domain dataset embedding does not significantly boost classification model performance, unless the model has significant class distribution imbalances. This finding suggests that the model exhibits favorable characteristics for larger datasets, enabling effective generalization to new domains even when trained on limited data.

- Adding a BiLSTM layer does not improve (Bassignana and Plank, 2022a) relation classification model's performance.

References

Elisa Bassignana and Barbara Plank. 2022a. CrossRE: A cross-domain dataset for relation extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3592–3604, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Elisa Bassignana and Barbara Plank. 2022b. What Do You Mean by Relation Extraction? A Survey on Datasets and Study on Scientific Relation Classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 67–83, Dublin, Ireland. 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.

Misha Malyshev, Alexey Goldin, and Qiang Zheng. The landscape of large language models.

Damianos P Melidis and Wolfgang Nejdl. 2021. Capturing protein domain structure and function using self-supervision on domain architectures. *Algorithms*, 14(1):28.

Tao Ni, Qing Wang, and Gabriela Ferraro. 2021. Explore BiLSTM-CRF-based models for open relation extraction.

Xiaoyong Pan, Jasper Zuallaert, Xi Wang, Hong-Bin Shen, Elda Posada Campos, Denys O Marushchak, and Wesley De Neve. 2020. ToxDL: deep learning using primary structure and domain embeddings for assessing protein toxicity. *Bioinformatics*, 36(21):5159–5168.

Xingchao Peng, Yichen Li, and Kate Saenko. 2020. Domain2vec: Domain embedding for unsupervised domain adaptation.

Aaron Smith, Bernd Bohnet, Miryam de Lhoneux, Joakim Nivre, Yan Shao, and Sara Stymne. 2018. 82 treebanks, 34 models: Universal Dependency parsing with multi-treebank models. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 113–123, Brussels, Belgium. Association for Computational Linguistics.

Sara Stymne, Miryam de Lhoneux, Aaron Smith, and Joakim Nivre. 2018. Parser training with heterogeneous treebanks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 619–625, Melbourne, Australia. Association for Computational Linguistics.

Rob van der Goot and Miryam de Lhoneux. 2021. Parsing with pretrained language models, multiple datasets, and dataset embeddings. In *Proceedings of the 20th International Workshop on Treebanks and Linguistic Theories (TLT, SyntaxFest 2021)*, pages 96–104, Sofia, Bulgaria. Association for Computational Linguistics.

Joachim Wagner, James Barry, and Jennifer Foster. 2020. Treebank embedding vectors for out-of-domain dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8812–8818, Online. Association for Computational Linguistics.

A Reproducibility

In Table 3 we report the hyperparameter setting of our models (see Section 3.4).

Parameter	Value
Encoder	bert-base-cased
Classifier	1-layer FFNN
Loss	Cross Entropy
Optimizer	Adam optimizer
Learning rate	$2e^{-5}$
Batch size	16
Seeds	4012, 5096, 8878, 8857, 9908

Table 3: **Hyperparameters Setting.** Model details for reproducibility.

All models were trained on an Intel Xeon E5-2687W, 32GB RAM and a GTX 980 GPU (with 4GB VRAM). The model was run on the CUDA cores of the GTX 980. To decrease running time the batch size was set to 16, as this was the was just under the VRAM limit of the GPU. The original run script from CrossRE was also altered, such that the script looped through all domains and all seeds.

B Group Contributions

All group members contributed equally to the realization of the project. There were no significant differences in the workload distribution among the group members. Each member actively participated in all aspects of the project, including brainstorming, research, coding, and finalizing the report. As a result, the project was a collaborative effort with shared responsibilities.

C Domain encoding

The 6 domains were encoded with icons for the purposes of readability in tables. These were news (📰), natural science (🍃), literature (📕), music (🎵), news (📰) and politics (🏛).

D Performance metrics

In Table 4 we report the performance metrics (Micro-F1, Macro-F1, Weighted F1) of the models across domains.

	Baseline			Domain embeddings			Domain tags		
	Micro F1	Macro F1	Weighted F1	Micro F1	Macro F1	Weighted F1	Micro F1	Macro F1	Weighted F1
🤖	46.11	30.34	39.23	42.79	26.29	36.34	48.51	33.87	43.74
📕	**67.72**	39.27	**62.97**	**62.54**	34.16	58.61	**68.52**	41.03	**64.28**
🎵	**73.14**	36.97	**69.26**	**69.63**	32.23	65.89	74.95	39.67	**71.55**
📰	50.45	14.38	44.45	51.36	13.69	46.25	48.79	15.22	42.71
🏛	58.41	26.40	54.08	57.15	24.28	52.89	58.11	29.35	54.23
🍃	44.86	32.79	41.30	40.37	29.34	37.14	47.51	36.40	44.29

Table 4: Performance metrics (Micro-F1, Macro-F1, Weighted F1) of the models across domains.