CSCI-5832 Project on Legal Named Entity Recognition

Preetham Maiya Manoor Satish, Chethan Kavaraganahalli Prasanna, Gülce Kardeş, and Robby Green
`{prma6536;chka2443;guka4411;rogr2023}@colorado.edu`

Abstract

Named Entity Recognition (NER) is the task of finding and characterizing named entities in unstructured text. In legal contexts, named entities might consist of the names of courts, parties, case numbers, judges, references to laws, and more. In this work, we present some results we found in attempting to solve the NER task for Indian legal documents.

1 Introduction

In this paper, we attempt to recognize named entities in Indian legal documents from the data set in [8]. We try four different methods for NER: Conditional Random Field (CRF) [12] with hand-crafted features, CRF with BERT [4], CRF with bidirectional long short-term memory (Bi-LSTM) [5] using GloVe [13] and Flair embeddings [2], CRF with bidirectional long short-term memory (Bi-LSTM) using BERT embeddings [6].

Performing the task of legal NER on this particular data set is more challenging than generic NER tasks, since our data set has a larger number of more granular label types. In particular, there are 14 types of named entities in this task, whereas generic NER tasks consider only around four entity types, namely PERSON (Person names), GPE (Geo-Political Entities), LOC (Locations), and MISC (Miscellaneous). The data set in [8] provides two types of training and validation data: the first set is associated with judgements which include data points from the "judgement" section of a legal proceeding document, and the remaining set obtained from the preamble section of such documents. (See Figure 1 for the named entity types and their distributions.)

The remainder of the paper is organized as follows. In Section 2, we give a brief overview of the methods we used. In Section 3, we present the results obtained by using the methods from Section 2, and discuss our results qualitatively. We conclude with a general description of our work in Section 4. For more details, we refer the reader to the code available at our github repo.

2 Methods

A survey paper [11] inspired our selection of the four models we chose to use. According to this paper, the models that achieve state-of-the-art performance for NER on the CoNLL-2003 data set [14] are the following:

1. Bi-LSTM + CRF with character and Cloze-style LM embeddings,
2. Bi-LSTM BERT + CRF with character and GloVe embeddings
3. BERT + Softmax and Dice loss

Our models are built using FlairNLP [1], a simple-but-powerful framework for natural language processing broadly, and for sequence labelling specifically. With FlairNLP, we construct multiple representations of input data (word embeddings, character embeddings, and contextual embeddings) and we investigate hybrid architectures for implementing NER.

2.1 CRF with hand-crafted features

Our baseline model was a CRF with hand-crafted features [1]. Below is the full list of features we considered in our analysis:

- The current word,
- The previous word,
- The next word,
- The presence of capital letters,
- The presence of digits,

[1]Recall that this is the same methodology we used to tag parts of speech in Hindi in Assignment 3 of CSCI-5832, Fall 2022.

Entity	Judgment Count	Preamble Count
COURT	1293	1074
PETITIONER	464	2604
RESPONDENT	324	3538
JUDGE	567	1758
LAWYER		3505
DATE	1885	
ORG	1441	
GPE	1398	
STATUTE	1804	
PROVISION	2384	
PRECEDENT	1351	
CASE_NUMBER	1040	
WITNESS	881	
OTHER_PERSON	2653	
Total	17485	12479

Figure 1: Distribution of the labeled entities in the data set.

- The presence of special characters, e.g., tabs or carriage returns,
- The lengths of words,
- Prefix and suffix of words.

To implement this CRF, we first generated a vector of these features for each word in a sentence, and concatenated those vectors to form the feature vector of the sentence. We then trained the CRF on these feature vectors. The hyper-parameters we used include a learning rate of 0.09 with a batch size of 16, trained on 40 epochs. With hand-picked features, our F1 score ended up being around 70%.

2.2 Bi-LSTM CRF with GloVe and Flair embeddings

Next, we constructed a hybrid model which consists of Bi-LSTM network with a CRF layer at the output for sequence labelling, whose architecture is provided in Fig 2.

Recall that bi-LSTM networks are recurrent neural networks (RNNs) with purpose-focused memory cells [7]. These networks consist of two LSTM layers (shown in Figure 2), a forward LSTM that processes sequential input from 'left to right', and a backward LSTM that process the input from 'right to left' (see [15] for more detailed description). This effectively increases the amount of context information available to the network. As a result, more context information is made available to the relational CRF layer, and this leads to an overall better modelling of dependencies for the output labels [10].

In this model, we provide the input by concatenating GloVe embeddings [13] (static word embeddings) with Flair embeddings (contextualized string embeddings). In [2], the authors report that contextualized string embeddings can lead to state-of-the-art results for sequence labelling tasks, especially NER. Here, bi-LSTM takes GloVe embeddings and Flair embeddings of the tokenized input sequence as input, and produces a hidden state, which in turn is provided as input to the CRF layer, and the CRF layer does the sequence labelling. Our result aligns with the theoretical expectation that the hybrid bi-LSTM CRF might perform better in finding dependencies between words: we got an F1 score of around 83%, which is indeed higher than the F1 score from only the CRF.

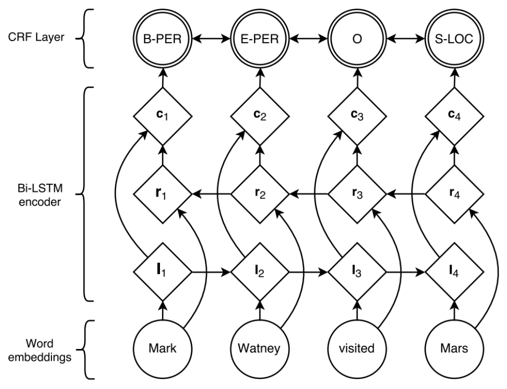

Figure 2: Bi-LSTM CRF with GloVe and Flair embeddings, excerpt from [10].

2.3 Bi-LSTM CRF with BERT embeddings

This model is quite similar to the previous one, except instead of using GloVe and Flair embeddings, we use contextual embeddings generated by BERT. BERT is a bidirectional transformer language model, whose architecture comprises of multiple self-attention layers stacked on top of each other [3, 18]. BERT jointly conditions on both left and right context, as shown in Figure 3, and is trained using Masked Language Model (MLM) and Next Sentence Prediction (NSP) as the objectives. BERT generates contextual embeddings for each token in the input sequence. These contextual embeddings are powerful representations of the input and have been used to produce state-of-the-art performance in many NLP tasks [9]. Moreover, in [11], the authors emphasize that contextualized embeddings generated from transformer-based language models trained on a large corpus can provide significant benefits in tasks like NER.

In our method, we implement a pre-trained BERT model, trained on the CoNLL-2003 data set for NER [17]. We chose this model because it was already trained on a similar task (NER) and thus thought it could give us relevant contextual embeddings for NER in the legal setting. We would like to note that while fine tuning the preamble data set, we ran into time and memory constraints, and thus could not report scores for that.

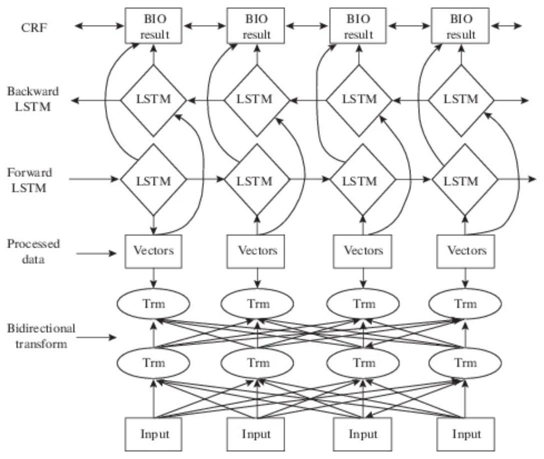

Figure 3: Bi-LSTM CRF with BERT embeddings, excerpt from [16].

Here, the bi-LSTM takes in BERT embeddings as input, and creates a hidden layer which is then given as input to a CRF, as illustrated in Figure 3. Apart from bi-LSTM's property of memory cells, the (strength of the) contextual embeddings provided by BERT results in a better representation of sequences. This gives a better performance with an improved micro F1 score of about 82% without fine-tuning, and around 86% with fine-tuning.

2.4 BERT with CRF

Finally, we use BERT and CRF together: BERT to generate the representations of the input data, and the CRF to perform the sequence labelling task, as sketched in Figure 4.

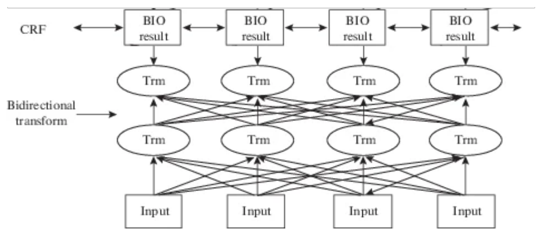

Figure 4: Sequence labelling with BERT followed by CRF, excerpt from [16].

3 Results and Discussion

In this section, we report the performance of each of the methods we used to solve the legal NER task. We present our results first for CRF with hand-crafted features in Section 3.1, next for the bi-LSTM CRF in Section 3.2 and Section 3.3, and finally for BERT with CRF in Section 3.4. Finally, we provide general comments to compare the performances achieved by CRF and BERT in Section 3.5. The performance metrics we consider are: false positive, true positive, false negative, true negative, precision, recall, F1, micro-F1, macro-F1 [7].

3.1 CRF with hand-crafted features

CRF with hand-crafted features (shown in Table 2 in the rightmost column) provides the results obtained by implementing the CRF with hand-crafted features. Here we observe that the model performs fairly well on classes COURT, DATE, STATUTE, PROVISION, and PRECEDENT. The model performs poorly on classes PETITIONER, RESPONDENT, JUDGE, ORG, GPE, WITNESS, and OTHER_PERSON. We believe that the high performance on the previous classes is due to a strong correlation with the corresponding subset of features. (In particular, we note that the prior word for entities in the COURT class is usually the same across all examples). Moreover, the DATE class has digits and special characters, and this allows CRF to identify dates with high confidence. The PROVISION and PRECEDENT classes have similar next and previous words, hence, they result in a higher F1 score.

For the preamble data, CRF does well for the classes LAWYER, JUDGE and COURT; whereas for the other classes, RESPONDENT and PETITIONER, the model has lower F1 scores. We believe that this is due to having a fixed structure of words surrounding the LAWYER, JUDGE and COURT tags in most documents. (For example, the JUDGE tag is usually preceded by "Honorable", and the LAWYER is usually preceded by "Advocate".)

3.2 Bi-LSTM CRF with GloVe and Flair embeddings

Table 2 provides the results obtained by Bi-LSTM CRF with GloVe and Flair embeddings. We observe that this model performs very well on certain classes like PROVISION, DATE, STATUTE and COURT. We suspect that this is due to the strong structure of individual words in those entity classes – i.e. dates have numbers, and courts and statutes often have

Class	BERT CRF	BERT CRF Fine Tuned	BERT BiLSTM CRF	BERT BiLSTM CRF Fine Tuned	BiLSTM CRF Glove Flair	CRF
OTHER_PERSON	0.6876	0.8713	0.8275	0.8775	0.9120	0.4056
PROVISION	0.9075	0.9524	0.8930	0.9198	0.8214	0.7879
STATUTE	0.9144	0.9732	0.9677	0.9817	0.9763	0.7867
DATE	0.9457	0.9159	0.9122	0.9492	0.9474	0.8299
PRECEDENT	0.5552	0.9098	0.7638	0.7452	0.6627	0.7190
COURT	0.8858	0.7273	0.8227	0.8345	0.8255	0.6667
GPE	0.7042	0.8095	0.7945	0.9343	0.6973	0.3883
ORG	0.6540	0.7935	0.9313	0.7518	0.9344	0.3773
CASE_NUMBER	0.6569	0.7983	0.7872	0.8173	0.7940	0.5239
WITNESS	0.4065	0.9163	0.7931	0.9508	0.8743	0.0566
JUDGE	0.5366	0.8800	0.9153	0.8472	0.8661	0.2009
PETITIONER	0.1176	0.7966	0.6863	0.8718	0.7250	0.1772
RESPONDENT	0.0870	0.7848	0.5079	0.6479	0.7547	0.1224
micro avg	0.7484	0.8736	0.8482	0.8754	0.8496	0.6303
macro avg	0.6199	0.8561	0.8156	0.8561	0.8301	0.4648

Table 1: F1 scores of methods on the Legal NER task for Judgements data.

Class	BERT CRF	BERT BiLSTM CRF	BiLSTM CRF Glove Flair	CRF
LAWYER	0.7909	0.8815	0.8426	0.7289
RESPONDENT	0.3478	0.6606	0.7200	0.5166
PETITIONER	0.3636	0.8182	0.8837	0.4163
JUDGE	0.5333	0.6486	0.4865	0.6693
COURT	0.8696	0.9091	0.7826	0.7109
micro avg	0.6308	0.7991	0.7800	0.5792
macro avg	0.5811	0.7836	0.7431	0.6084

Table 2: F1 scores of methods on the Legal NER task for Preamble data.

"court" or "statute" in their name. On the other hand, we also observe that the model's performance suffers on the classes ORG and PRECEDENT. This can be attributed to the existence of generally longer names in these classes, which leads to higher error rate as a single word missed in a long sequence would result in a misclassification. (Note that this observation and argument match those provided in [8]).

Given the F1 scores, we note that the model is able to classify the classes PETITIONER, RESPONDENT and JUDGE fairly well – even though the judgement training data consists of fewer samples from these classes. Overall, for the task at hand, a macro F1 score of 83% is commendable, especially without any post-processing.

For even more insight, we ran the same model with the provided hyper parameters on the preamble data. The preamble has smaller number of sentences, though is richer in tags than the judgement data, so there are more B and I tags than O tags. For the preamble data, we reported an F1 score of 78%. In the preamble case, we see that our model performs well on the classes LAWYER, PETITIONER and COURT. We believe that it does so because of the abundance of training labels for these classes.

Notice that the preamble data is partially structured and labeled (i.e. it has headers and labels), so we expect higher F1 scores, since this partial structure helps CRF to identify tags with a fairly high confidence because of repeated sequence during training. (e.g., see COURT, which justifies higher F1 score for this tag). For the classes RESPONDENT and JUDGE, the model does not do as well. Since RESPONDENT and JUDGE include people's names given without consistent formatting, this makes sense.

3.3 Bi-LSTM CRF with BERT embeddings

Table 2 provides the results obtained from our Bi-LSTM CRF with BERT embeddings. We observe good performance on classes PROVISION, DATE, STATUTE and COURT, though the F1 score for classes PETITIONER and RESPONDENT is quite low. Comparing with the model in 3.2, we suspect that the bi-LSTM CRF with BERT embeddings is more vulnerable to having too few examples of those classes in the training data. In contrast to the BiLSTM-CRF-GloVe-Flair, however, we observe a higher performance on ORG and PRECEDENT here. We thus speculate that adding BERT embeddings makes the network predict longer tags more accurately, reducing misclassification. This model provides the strongest performance for the legal NER task at hand (without any post-processing).

Moreover, unfreezing the last layer of BERT for our downstream task provides a significant boost in performance, see Table 2. It appears that with such fine-tuning, BERT performs better still with classes occurring less frequently in the data set (e.g., improvement of F1 scores for classes like PETITIONER and RESPONDENT). This configuration produces the best results on the data set. For the preamble data, we obtain similar results and performance with respect to the Bi-LSTM CRF with GloVe and Flair embeddings. The classes LAWYER, RESPONDENT, PETITONER and COURT have good F1 score, but the class JUDGE has very low F1 score.

3.4 BERT with CRF

The performance of BERT with CRF is not spectacular, as seen in 2. This model performs quite well on the classes PROVISION, STATUTE and DATE; however, the performance on the class COURT drops. The model's performance especially suffers on classes PETITIONER and RESPONDENT. Furthermore, without having introduced the fine-tuned BERT weights, performance suffers on the classes PRECEDENT, ORG, CASE NUMBER, WITNESS, and JUDGE. Fine-tuning the last layer of BERT significantly improves results as, shown in Table 2. In fact, it results in the best results[2]. Before proceeding with the next section, we note that for resource constraints mentinoed in Section 2.3, we were not able to implement the task with fine tuning over preamble data.

3.5 Comments regarding the performance of CRF and BERT

Given our mixed observations on the performance of BERT (without fine tuning) followed by a CRF, we aim to provide a closer look on our results. Recall that a BERT-only classifier makes independent, local label decisions, whereas a CRF makes global decisions by trying to maximize the probability of a whole sequence of local label decisions [12, 4]. BERT is capable of taking into account neighboring words, although it is not capable of taking into account previous decisions it has made. The question of effectiveness, then, is the question of whether the CRF is detecting any additional patterns which were not detected by BERT (e.g., whether there are specific label sequences that are logically advantaged or prohibited by the nature of the problem). This heavily depends on the semantics of the documents, the semantic domains in which those documents exist, and the semantics of the tasks we are asking our models to perform. We see that BERT-CRF does a much better job of recognizing LAWYER and COURT than it does of recognizing PETITIONER and RESPONDENT. From examining the data, we can see that the parts of the document containing the COURT and LAWYER are usually highly-structured and this helps the CRF decide a good tag sequence, whereas for the other less-structured tags, the CRF doesn't assign tag sequences effectively and doesn't detect patterns which were not already detected by BERT. We think that the F1 scores for the low performing classes can be increased by pre-processing the data to add more structure around the tags we consider. In addition, another reason for the poor performance of BERT-CRF in PETITIONER and RESPONDENT could be that BERT can't tackle such a low number of training samples whereas the BiLSTM we used is able to do so. (Observe that PETITIONER and RESPONDENT have the lowest judgement counts in the data set used.)

It is also worthy of note that we attempted the task with a BERT + Softmax architecture as well. However, this architecture produces very poor results (F1 $\approx 50\%$). This seems to further support the value added by using a CRF layer in sequence labelling tasks.

4 Conclusion

In this work, we evaluated a set of approaches for the extraction of semantic concepts from Indian legal documents. We identified 14 different semantic classes, and implemented CRF and bi-LSTM classes (augmented with BERT and Glove embeddings) corresponding to the state-of-the-art performance. Our results demonstrate the superiority of the Bi-LSTM CRF hybrid models with the provided F1 scores from Section 3. In particular, we see that BERT embeddings with a Bi-LSTM CRF architecture produces the best performance on the legal NER dataset used.

We believe that the general performance of our models could be improved by combining the judgement and preamble data sets. In this way, models would be provided more training samples for certain classes such as PETITIONER and RESPONDENT (see detailed discussion in Section 3.3 and 3.2). In this work, we did not consider data augmentation between judgement and preamble data sets due to resource constraints, but note that it is to be explored in future work.

[2] Fine-tuning even one of the weight layers of a pre-trained BERT model goes a long way in getting better performance on downstream tasks.

References

[1] flairnlp. https://github.com/flairNLP/flair. Accessed: 2022-12-12.

[2] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. 08 2018.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. In *Proceedings of the 2019 Conference of the North*. Association for Computational Linguistics, 2019.

[5] Alex Graves, Abdel rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, May 2013.

[6] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging, 2015.

[7] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall PTR, USA, 1st edition, 2000.

[8] Prathamesh Kalamkar, Astha Agarwal, Aman Tiwari, Smita Gupta, Saurabh Karn, and Vivek Raghavan. Named entity recognition in indian court judgments, 2022.

[9] M. V. Koroteev. Bert: A review of applications in natural language processing and understanding, 2021.

[10] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition, 2016.

[11] Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. A survey on deep learning for named entity recognition, 2018.

[12] Fuchun Peng and Andrew McCallum. Information extraction from research papers using conditional random fields. *Information Processing &amp Management*, 42(4):963–979, July 2006.

[13] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[14] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. 2003.

[15] M. Schuster and K.K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[16] Yuhang Song, Shengwei Tian, and Long Yu. A method for identifying local drug names in xinjiang based on bert-bilstm-crf. *Automatic Control and Computer Sciences*, 54:179 – 190, 2020.

[17] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147, 2003.

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.