Name Entity Recognition

Candidate number: 246743

May. 2023

Problem outline

Name entity recognition is actually a kind of sequence labelling, so what we are actually doing is labelling a series of variables. For example, Brighton, McDonald, can be labelled as location name, organization name. These phrases are often multi-word. This is essentially a prediction for semantic structuring.

NER is the basis for performing downstream tasks in NLP. Google's search engine box, automated question answering bots, search and retrieval of articles, etc., all require an accurate NER recognition system. The most common NER benchmarks are People, Organization, Location and Miscellaneous, which are the most basic taxonomies proposed by CoNll-2003.

However, there are still some difficulties with name entity recognition. One of them is the uncertainty of the boundary, which is often forgotten in the detection of boundaries in English. For example, in the case of the phrase "the New York Times", whether or not "the" should be defined as part of this entity. Another point is the ambiguity of the type of entity, for example in the case of "Steve Jobs", "Jobs" is simply the surname of the person, but at the same time, it can be used as a noun. This ambiguity can be described as an entity-noun. Similarly, for the surname "Hemsworth" in the personal name "Chris Hemsworth", which is actually the name of a town in England. This ambiguity is described as entity-entity.

BIO encoding was invented to represent the label of a word, B is for beginning a chunk, I is for inside a chunk and O is for outside a chunk. With this nomenclature, a span identification problem can be transformed into a sequence labeling problem.

The most common approach to the NER problem is to use a supervised approach. The training material is artificially labelled, and then encoded it in BIO way, trained using classifiers (HMM, SVM, MEMM, LSTM, etc.), and finally evaluated.

Introduction for different approaches

Hidden Markov Model (HMM)

For the Hidden Markov Model, there are two assumptions:
1. each observation depends only on the Markov chain state at that time node and is independent of the other states.
2. The current state is only related to the previous state.

Markov Models

To talk about the Hidden Markov Model, one needs to start with the Markov Model. The random variable X_t at some moment $t \geq 1$ has a conditional distribution $P(X_t|X_{t-1})$ with respect to the

random variable X_{t-1} at the previous moment. If X_t depends only on X_{t-1} , and not on the past random variables $\{X_1, X_2, ..., X_{t-2}\}$, then this property can be referred to as Markovianity.

In other words, the distribution of any random variable in the sequence is independent of earlier variables given its predecessor.

Expressed in the formula, this can be written as:

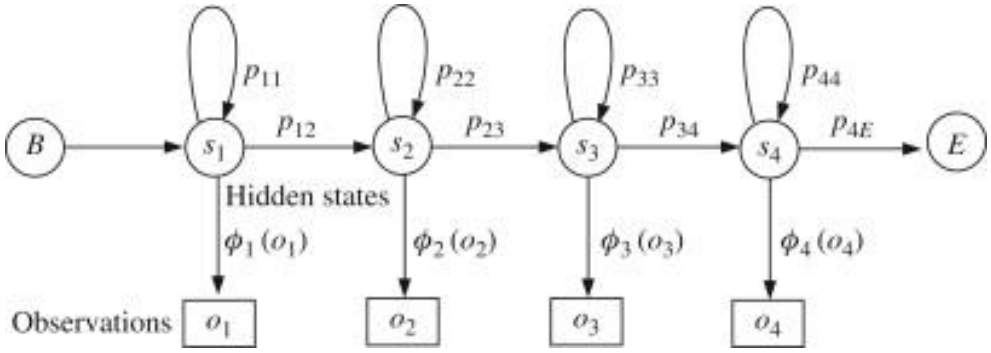
$$P(X_t \mid X_1, X_2, ..., X_{t-2}) = P(X_t \mid X_{t-1})$$

A chain of numbers in a sequence is called a Markov chain if all variables to follow the Markov principle.

$$X = \{X_1, X_2, ..., X_t\}$$

Hidden Markov Model example

X_t on such a Markov chain can be determined by the computation of X_{t-1} with the transfer probability distribution (also called the transfer matrix). The HMM is built on such a Markov chain. Each X is here a state ("s" for short)



As depicted in the figure above, each state (s) is Markovian, i.e., Given present, the future is independent with the past. Each state is constantly updated by the transfer probability P. Under each state is a corresponding observation which is computed from the emission probability ϕ .

For both of the transfer probability P and the mission probability ϕ , they are essentially presented as matrices. Because of the temporal nature of Markov chains, the initial probability matrix is given, and subsequent states are derived step by step from the transfer probability matrix.

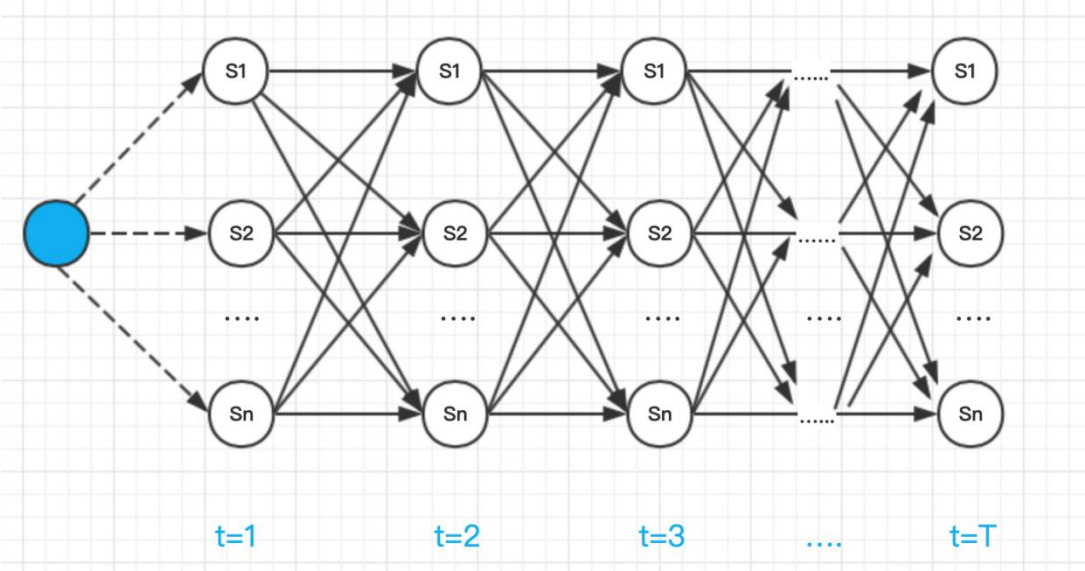
For example, if the value of state π (1) is required, let the initial state π (0) be [0.8 0.1 0.1], then:

$$\Pi(1) = P\Pi(0) = \begin{bmatrix} 0.3 & 0.5 & 0.2 \\ 0.5 & 0.2 & 0.3 \\ 0.2 & 0.3 & 0.5 \end{bmatrix} \times \begin{bmatrix} 0.8 \\ 0.1 \\ 0.1 \end{bmatrix} = \begin{bmatrix} 0.31 \\ 0.45 \\ 0.24 \end{bmatrix}$$

When training the HMM model, the model needs to calculate the initial probability, transmission

probability and emission probability, and then use these three probabilities to infer the required tasks to be completed.

In this model, we are currently concerned with three main questions.
1. at any moment, given the observation o, infer its state value s.
2. Given an observation o at any given moment, infer its state value transition.
3. Given an observation o, infer its most likely state value path.



Convolutional Neural Network (CNN)

Data pre-processing

Data pre-processing is required for any data model. For the NER task, annotated corpus for supervised learning is necessary. Next, this text data needs to be converted into vectors for representation. Finally for input to the CNN, the text and label sequences are converted to fixed length windows.

Build model

The CNN model starts with an input layer that accepts the previously pre-processed data. Afterwards, a convolutional layer is added, which serves to capture local features. At the same time, the activation function should be added to make the function evolve in a nonlinear direction. Each convolutional layer is often followed by a pooling layer, which is used to reduce the number of parameters of the model and to reduce the possibility of overfitting. This convolution-pooling cycle is repeated several times as needed, which increases the complexity of the model. A fully-connected layer is used at the end for closure, and the Softmax activation function is usually used to generate probability distributions for each entity class.

Training the model

The dataset will be divided into a training set and a validation set, or sometimes into a training

set, a validation set and a test set for further analysis. During training, appropriate loss functions and optimizers will be selected to assist in training. The model is continuously iterated during training and the weights are updated by minimizing the loss function. If a test set exists, it is used to evaluate the final performance of the model.

Comparison and Evaluation

Theoretical Comparison

Using the HMM model to accomplish the NER task, it is easy to understand and extend for tagging. It is also efficient in solving sequence tagging problems, especially for small data sets. In addition, the HMM model can take into account the dependencies between models, which makes it ideal for sequence tagging tasks like NER.

However, although HMM are simple to structure and train, the credibility of their models still relies heavily on correct labelling, which inadvertently adds a lot of manual work.

For CNN, as a deep learning model, although it was originally often used for processing images, it is gradually being applied to processing textual data due to its powerful ability to extract feature values. Unlike HMM, CNN require a large amount of annotated data for training, otherwise they are extremely prone to overfitting. However, the large number of datasets also gives the CNN a chance to learn more complex features and refine the model.

In terms of modifiability, CNN are more difficult to interpret, and their model composition is still a black box for current humans, which leads to subsequent refinement of the model by adjusting hyperparameters, rather than optimising directly for the core of the problem. Furthermore, CNNs are powerful in extracting local features, but they are not ideal for handling global dependencies such as long paragraphs.

Practical Comparison

In the NER task, different languages can affect the success of the recognition task to some extent. For example, in English, capitalised words are often used to express specific titles or names of organisations, which can be used to help identify Named Entities, whereas in Hindi no such capitalised character exists. In addition, Hindi is a language that can have free order of words and the position of Name Entity can vary in different contexts. (Chopra D 2016)

The morphology can also vary from language to language. For English and Hindi, for a particular root word, different affixes can always be added to form a new word. For Chinese, however, there are no affixes, and each character is independent and immutable.

HMM

Morwal S 2012 presents an HMM based system adapted to a variety of Indian languages (Hindi, Urdu, Punjabi, etc.). The annotated corpus was used for training and testing. As far as the results are concerned, it obtained an accuracy of 90%. It is worth noting that since the parameters used are dynamic, this system is basically capable of performing other classification tasks as well. Currently, tags like "tree", "driver", "location", "country " have been tested.

Chopra D 2013 proposed an HMM model for English text with 8 tags: "PER", "ORG", "CO ", "MAGAZINE", "WEEK", "LOC", "PC ", "MONTH". The post-test F-Measure reached 73.8%. The prediction accuracy of the label "PER" reached 70%.

Fu G 2005 proposed alexicalized HMM model for NER. This is actually designed to solve the problem of unknown words. The model is divided into two parts, with the binary model first determining whether the word is known when performing the recognition task, and then predicting the label based on the lexicalised HMM. In terms of results, the lexicalised HMM has a higher performance than the standard HMM, with a high regression rate and a high precision, floating between 85% and 92% for tag PER and LOC. However, the recognition of ORG is dwarfed, basically only around 70%. I think this is due to the fact that Chinese cannot recognise specific organisations by capital case as English does.

CNN

Gui T, 2019 proposed a novel CNN model with a reflection mechanism to handle the NER task in Chinese. Unlike English, most characters in Chinese have multiple meanings, and these different meanings are called potential words. These potential words can lead to ambiguity when recognising utterances. It has been found that when the window size is adjusted to 2, all potential words in the utterance automatically correspond to the correct position. At the same time, this added reflection mechanism can also adjust the weight of embedded words and resolve possible conflicts between potential words through higher-level semantics.

The model was tested and evaluated on the MSRA, WEIBO, and Resume NER datasets, and the results showed significant improvements in precision, recall, and f1 compared to previous models. In addition, for different datasets, the running speeds are 1.5 - 3.21 times faster than the previous state-of-art method.

Conclusion

Each specific experiment is based on a different dataset, so it is difficult to make a direct comparison. However, in general, CNN perform relatively better than HMM, which should be related to the complexity of the model. CNN can always learn rich features from constant back-propagation, which increases the accuracy of the model. The HMM, on the other hand, is limited by its model complexity and can only take into account a limited amount of contextual information. However, in the case of some small datasets, CNN may face the risk of overfitting,

and this is where HMM can still be useful.

For different languages, NER takes different measures. For instance, Chinese is a single-character language and there is no significant interval between each character. Therefore, the word separation task is extremely important when performing NER on the Chinese corpus.

For further work, hybrid models are considered promising. Each of the different approaches can be used to optimise the whole project using the strengths of each. For example, HMM can be used to capture dependencies between sequence labels, while CNN can be used to summarise and extract local features. Similarly, pre-trained language models (such as BERT) can be added together to further improve the overall performance. (X. Wu, 2022)

Reference

Chopra D, Morwal S. Named entity recognition in english using hidden markov model[J]. International Journal, 2013.

Fu G, Luke K K. Chinese named entity recognition using lexicalized HMMs[J]. ACM SIGKDD Explorations Newsletter, 2005, 7(1): 19-25.

Gui T, Ma R, Zhang Q, et al. CNN-Based Chinese NER with Lexicon Rethinking[C]//ijcai. 2019: 4982-4988.

Morwal S, Jahan N, Chopra D. Named entity recognition using hidden Markov model (HMM)[J]. International Journal on Natural Language Computing (IJNLC) Vol, 2012, 1.

X. Wu, T. Zhang, S. Yuan and Y. Yan, "One Improved Model of Named Entity Recognition by Combining BERT and BiLSTM-CNN for Domain of Chinese Railway Construction," 2022 7th International Conference on Intelligent Computing and Signal Processing (ICSP), Xi'an, China, 2022, pp. 728-732, doi: 10.1109/ICSP54964.2022.9778794.