Chinese Event Extraction Based on Hidden Markov Model and CRF

Linyang He(15307130240)

December 9, 2017

Abstract

Sequence labeling is an interesting subfield of machine learning. It involves the algorithmic assignment of a class label to each member of the a certain sequence. It is widely used in natural language processing and speech recognition. In NLP field, it can used in such as information extraction, part of speech labeling, event extraction, ect. In this project, we will focus on Chinese event extraction. Hidden Markov Model(HMM) is a widely used algorithm in sequence labeling. We will see more details about this kind of model. While conditional random field(CRF) is a kind of graphic model which can also handle some sequence labeling problems. In this project, we will use both algorithms to execute a Chinese event extraction task. This task is divided into two subtasks: "trigger" labeling and "argument" labeling. We will discuss the different performance based on different tasks and with the usage of different algorithms.

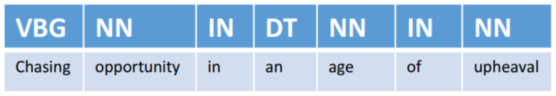
1 Introduction

Event Extraction is the core problem of the information extraction task. It pays attention on how to extract the information that users are interested in from unstrucural text to express the information with a structure form instead of natural language. It has significance application in Weibo or Twitter text processing.
Hidden Markov Model is a kind of transition state machine model. Different from the basic Markov model, the HMM model has an extra hidden state. It focuses on how to use what we can observe to estimate the hidden state sequence. There are three main algorithms uesd in HMM: Forward algorithm for evaluation, Viterbi algorithm for decoding the hidden state and Baum-Welch algorithm for estimating the parameters. In this project, we mainly need to decodee the hidden state, so we will focus on the Viterbi algorithms basicly. Besides, the greedy algorithm sometimes is also used in decoding. Though it cannot back off to get the real global optimal hidden state, computing of this model is quite easy than the Viterbi model. So we will discuss the balance between the accuracy and speed based on these two different models.
CRF model considers the advantage of both maxent model and HMM, is a non-directional graphic model. In this project, we will use the CRF++ toolkit to train and test our model using the CRF algorithm. Also, we should know that CRF is a discriminative model while HMM is a generative model.
The rest of the report is structured as following: Section 2 introduces the background we may need. Section 3 describes the data and the experimental setup. Section 4 presents the experiment result of the F1 score, type-correct score and consuming time varying from different model and different training data. Section 5 summarizes related work while Section 6 concludes our report.

2 Background

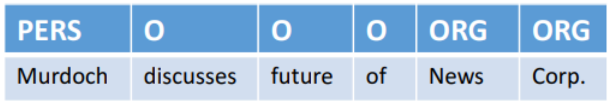
2.1 Sequence Labeling

There are some example application of sequence labeling as following.



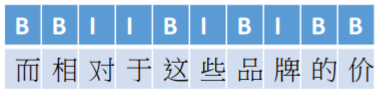
VBG	NN	IN	DT	NN	IN	NN
Chasing	opportunity	in	an	age	of	upheaval

Figure 1: POS Tagging



PERS	O	O	O	ORG	ORG
Murdoch	discusses	future	of	News	Corp.

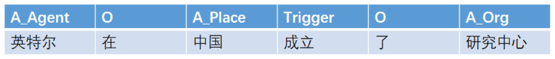
Figure 2: Named entity recognition



B	B	I	I	B	I	B	I	B	B
而	相	对	于	这	些	品	牌	的	价

Figure 3: Word segmentation

As for our project's goal:



A_Agent	O	A_Place	Trigger	O	A_Org
英特尔	在	中国	成立	了	研究中心

Figure 4: Event Extraciton

As we can see from the graph, we label each word in the sentence with trigger or A_argument or O. Trigger means that the main word expresses the occurence of the event while the A_argument denotes some entities which plays a certain role in the event. And O means the others. We have 8 kinds of triggers and 35 kinds of arguments, separately. Our goal is to use the labeled training dataset to learn a model that can predict the trigger or argument role in the test set.

2.2 Hidden Markov Model

Here's the basic hidden markov model as Figure 5 shows:

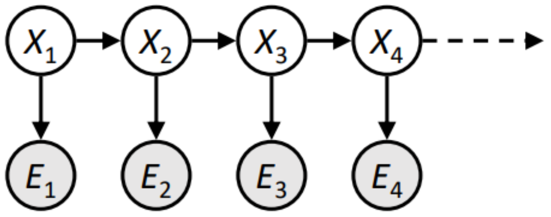


Figure 5: Event Extraciton

Here X_i denotes the hidden state, while E_i denotes the values we can observe. And any HMM can be defined by: Initial distribution probability $P(X_1)$ and transition probability $P(X_t|X_{t-1})$ and emission probability $P(E_t|X_t)$. What should be paid with attention is that HMMs have two important indepence properties:

- Future depeonds on past via the present.
- Current observation is related to only current state.

According to these two indepence properties, we can estimate the probability as:

$$P(X_1, E_1, ..., X_T, E_T) = \Pi_{t=1}^{T} P(X_t|X_{t-1})P(E_t|X_t)$$

In our specific project, what we need to do is to use the HMM to get a most likely explanation. Given a sentence or some words, to find the sequence of hidden states that is most likely to generate the observation values. That is to compute:

$$argmax_{x_{1:t}} P(x_{1:t}|e_{1:t})$$

The most naive method is to find all possible sequence of hidden states and then compute the most likely one. This method is too slow, we won't too much information about it. Instead, we will talk about the Viterbi algorithm and greedy algorithm.

2.2.1 Viterbi Algorithm

Instead of computing all the possible situation, we just choose the max likely situation. Specifically, if we have all possible last states' probability, we can compute the current state probability as:

$$m_t[x_t] = P(e_t|x_t) max_{x_{t-1}} P(x_t|x_{t-1}) m_{t-1}[x_{t-1}]$$

Here $m_t[x_t]$ denotes the current state probability, $m_{t_1}[x_{t-1}]$ denotes the last one. $P(e_t|x_t)$ denotes the emissioin probability, and $P(x_t|x_{t-1})$ denotes the transition probability. The algorithm is to iterate all possible transition ways based on the last state, and find the most possible one. Note that although the current state is generated by the max likely one, all the possiblitily of the current state should be stored because the future state needs them to find the next optimal one. In addition to the Viterbi algorithm, we have another algorithm: greedy algorithm.

2.2.2 Greedy Algorithm

As we can see from its name "greedy", everytime we consider the current state, we just consider the max likely one based on the transition probability. Different from the Viterbi algorithm, we don't need to store other possibility, because in greedy algorithm, we believe in the local optimal everytime.(Though it might not get the global optimal.) In this situation, we don't need to consider the last state, we just consider it as a constant. So we have:

$$m_t[x_t] = P(e_t|x_t) m_{t-1}[x_{t-1}] max_{x_{t-1}} P(x_t|x_{t-1})$$

We will implement all the HMM stuff by ourselves.

2.3 CRF

Conditional random field algorithm was first introduced by Lafferty et al. at 2001. They gave the discription as:
Let $G = (V, E)$ be a graph, such that Y is indexed by the vertices of the graph. We have (X, Y) is a conditinoal random field when the variables Y_v conditioned on X, obey the Markov property. So, actually, HMM is a special CRF when the graph is a linear sequence. We will use the CRF++ toolkit to implement the CRF algorithm. And this is how CRF++ works: we need a template file to desribe how our training dataset and test dataset construct. Besides, we need a training dataset file as well as a test dataset. In the template file, we use "%x[row,col]" to specify a certain token in the training or test data. More information can be found on their website's doc.

3 Project

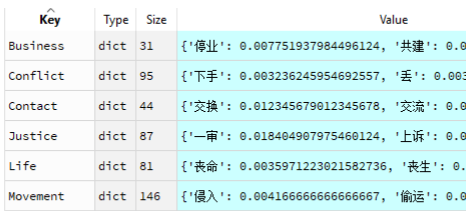
Consider that the CRF++ method just need to implement directly without programming by ourselves, we will focus on the HMM algorithm.

3.1 Dataset Preparation

We will use the pickle library to store all the frequency in advance.

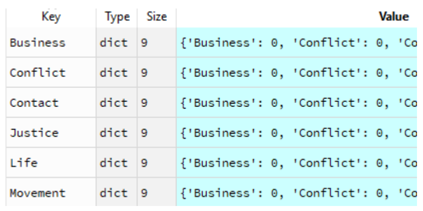
3.2 Encode

We will use the frequency to compute all the emission probability and transition probability. We take the first task as the example



Key	Type	Size	Value
Business	dict	31	{'停业': 0.007751937984496124, '共建': 0.0
Conflict	dict	95	{'下手': 0.003236245954692557, '丢': 0.003
Contact	dict	44	{'交换': 0.012345679012345678, '交流': 0.0
Justice	dict	87	{'一审': 0.018404907975460124, '上诉': 0.0
Life	dict	81	{'丧命': 0.0035971223021582736, '丧生': 0.
Movement	dict	146	{'侵入': 0.004166666666666667, '偷运': 0.0

Figure 6: Emission probability



Key	Type	Size	Value
Business	dict	9	{'Business': 0, 'Conflict': 0, 'Co
Conflict	dict	9	{'Business': 0, 'Conflict': 0, 'Co
Contact	dict	9	{'Business': 0, 'Conflict': 0, 'Co
Justice	dict	9	{'Business': 0, 'Conflict': 0, 'Co
Life	dict	9	{'Business': 0, 'Conflict': 0, 'Co
Movement	dict	9	{'Business': 0, 'Conflict': 0, 'Co

Figure 7: Transition probability

3.3 Decode

After we get the emission probability and transition probability, we could decode out test file as the observation to estimate the max likely explanation.

And at last, we will use the CRF++ toolkit. We need to write our own template based on our training dataset. It can be found in the attachment.

4 Result

We can have our results as Table 1 and Table2 shows.

Model	HMM-Viterbi	HMM-Greedy	CRF
Type_cor	**0.96**	**0.96**	**0.96**
Precision	0.59	0.59	0.76
Recall	0.48	0.48	0.59
Accuracy	0.93	0.93	**0.95**
F1	0.53	0.53	**0.67**
Time	**0.4s**	**0.4s**	5.6s

Table 1: Results for the Trigger task

Model	HMM-Viterbi	HMM-Greedy	CRF
Type_cor	0.30	0.32	**0.40**
Precision	0.68	0.70	0.70
Recall	0.58	0.44	0.58
Accuracy	0.72	0.70	**0.73**
F1	0.62	0.54	**0.63**
Time	**0.9s**	**0.9s**	170.5s

Table 2: Results for the Argument task

5 Summary

From the Table1, we can find that all the algorithms works well for the first task, that is only one word in the sentence needed to be labeld. And interestingly, the HMM-Viterbi and HMM-Greedy have the same performance. This is easy to understand. Since in the first task, we just need to tag only one word, then for the Greedy algorithm, the local optimal is exactly the global optimal. Besides, we find that our own model's consuming time is the least. So when we do the first task, the HMM works better.
For the second task, we find that the Viterbi algorithm works better than the greedy one, which is reasonable. Relatively speaking, the CRF model works better than our HMM model, but the time cost is much higher. With the time consuming increasing from 0.9s to 170.5s, then accuracy just increases by 0.01.

6 Conlusion

When we lable just one word, HMM is better. When we need to label several words with different tags, if we don't consider the time consuming, CRF is better, but if we do consider, HMM-Viterbi is the optimal one.