Chinese Event Extraction Report

Cheng Zhong 16307110259

Abstract:

In this project, we are required to implement two sequence labeling models (HMM, CRF) for Chinese event extraction. The task is separated as two subtasks: Identify and classify the trigger word (8 types) and arguments (35 types) in the sentence. After using the HMM method, I can reach the accuracy of 95.21% (Bigram) and 95.26%(Trigram) in trigger dataset, 71.11% (Bigram) and 72.11% (Unigram) in argument dataset.

Index Terms: **Event extraction, HMM, CRF, N-gram**

1. Introduction:

Since many problems in NLP have data which is a sequence of characters, words, phrases, lines or sentences. We can think of our task as one of labeling each item based on some knowledge of probability. Because the corpus is context-sensitive, we can use Markov chain models to predict it.

2. Methods

Section 2.1 and Section 2.2 show the principle of the method to deal with the project. (HMM、CRF). Section 2.3 shows the Viterbi algorithm to calculate the probability of HMM model. Finally, Section 2.4 shows the N-gram method and add-k smoothing to pre-processing the dataset.

2.1 Hidden Markov Models (HMM)

HMM are underlying Markov chain over states X, however we can observe outputs (effects) at each time step. Under HMMs, we assume that the future depends on past via the present, and current observation independent of all else given current state. So that state independent of all past states and all past evidence given the previous state, and evidence is independent of all past states and all past evidence given the current state. Denote the state as X and evidence as E, the joint distribution over X, E can be written as

$$\mathrm{P}(\mathrm{X_1}, \mathrm{E_1}, \dots, \mathrm{X_T}, \mathrm{E_T})$$
$$= \mathrm{P}(\mathrm{X_1})\mathrm{P}(\mathrm{E_1}|\mathrm{X_1}) \prod_{\mathrm{t}=2}^{\mathrm{T}} \mathrm{P}(\mathrm{X_t}|\mathrm{X_{t-1}})\mathrm{P}(\mathrm{E_t}|\mathrm{X_t})$$

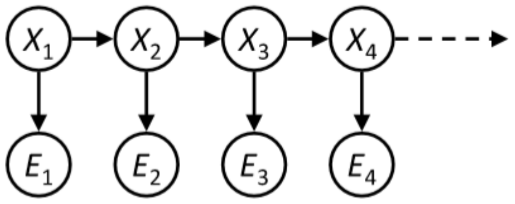

Fig. 1 Hidden Markov Models

2.2 Conditional Random Field (CRF)

Here we mainly introduce special condition random fields defined on linear chains (linear chain conditional random field).

Under the assumption that given a random variable sequence X, the conditional probability distribution P(Y|X) follows a conditional random field, which holds

$$\mathrm{P}(Y_i|\mathrm{X}, Y_1, \dots, Y_{i-1}, Y_{i+1}, \dots, Y_n)$$
$$= \mathrm{P}(Y_i|\mathrm{X}, Y_{i-1}, Y_{i+1})$$

Thus, it can be written as

$$P(y|x) = \frac{1}{Z(x)} \exp\left(\sum_{i,k} \lambda_k t_k(y_{i-1}, y_i, x, i) + \sum_{i,l} \mu_l s_l(y_i, x, i)\right)$$

Where the t, s are characteristic functions, λ, μ_l are weights and Z(x) is a normalization factor. So that we can use the training data set to obtain the conditional probability model, and use the maximum likelihood of the condition probability to generate the output sequence $\hat{y}$.

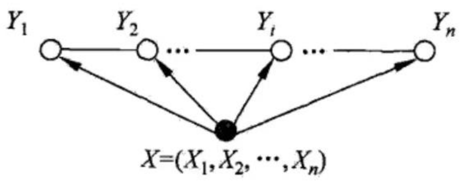

Fig. 2 Conditional Random Field

2.3 Viterbi Algorithm

Viterbi Decoding is an algorithm using dynamic programming to calculate the maximum/minimum probability of state t.

Under the assumption of HMM, we can summarize the algorithm as "When calculating the shortest path of the i+1th state, only the shortest path from the start to the current k state values and the shortest path of the current state value to the i+1th state value need to be considered."

Thus, we only need to follow the recursive formula as:

$$P_{t+1}(i) = \max_j X_i(E_{t+1})[P_t(j) \text{path}_{ji}]$$

2.4 N-gram and add-k smoothing

N-Gram is a language model commonly used in large vocabulary continuous speech recognition. This model assumes that the appearance of the Nth word is only related to the N-1 words before it, and not related to any other words. At the same time, to solve the problem of zero probability, we apply add-k smoothing method.

Therefore, we can update the transition probability as (Trigram HMMs)

$$q(s|u,v) = \lambda_1 \times q_{ML}(s|u,v) + \lambda_2 \times q_{ML}(s|v) + \lambda_3 \times q_{ML}(s)$$

Where

$$\lambda_1 + \lambda_2 + \lambda_3 = 1$$

$$q_{ML}(s|u,v) = \frac{c(u,v,s) + d}{c(u,v) + d * |V|}$$

$$q_{ML}(x|s) = \frac{c(x,s) + d}{c(s) + d * |V|}$$

3. Training and Results

3.1 Bigram HMM

First, we use the Bigram model to generate HMM model, after choose different hyper-parameter k for smoothing, the results as follows:

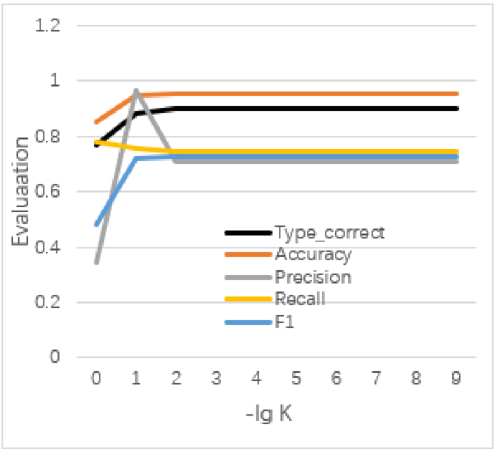

Fig. 3 Result of Bigram HMM in Trigger

Type Correct	Accuracy	Precision	Recall	F1
0.8982	0.9521	0.7068	0.746	0.7259

Tab. 1 Best Result of Bigram HMM in Trigger (k<10^ (-3))

From the table, we can see that the indicators tend to converge when the parameter K decreases, and model perform better on the Trigger dataset with less classifications than the Argument dataset.

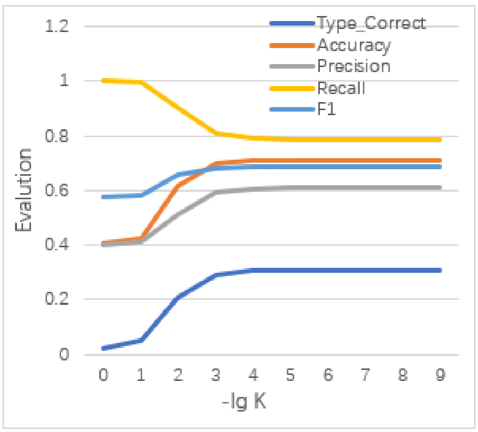

Fig. 4 Result of Bigram HMM in Argument

Type Correct	Accuracy	Precision	Recall	F1
0.3072	0.7111	0.6087	0.7839	0.6853

Tab. 2 Best Result of Bigram HMM in Argument (k<10^ (-6))

3.2 Trigram HMM

At the same time, we use the Trigram HMM with different hyper-parameter λ to combine the different N-gram Model. The $\lambda1$, $\lambda2$, $\lambda3$ are the weights of Trigram 、 Bigram and Unigram. The results are as follows:

$\lambda1:\lambda2:\lambda3$	Type_c	Acc	Precision	Recall	F1
1:1:8	0.8895	0.9374	0.6063	0.7533	0.6719
8:1:1	0.8847	0.9514	0.7065	0.7343	0.7201
1:8:1	0.8994	0.9523	0.7111	0.7401	0.7253
1:1:1	0.89	0.9526	0.7067	0.7562	0.7306

Tab. 3 Result of Trigram HMM in Trigger

$\lambda1:\lambda2:\lambda3$	Type_c	Acc	Precision	Recall	F1
1:1:8	0.3063	0.7211	0.6272	0.7515	0.6838
8:1:1	0.3021	0.7123	0.6112	0.7771	0.6842
1:8:1	0.3092	0.7094	0.6103	0.7622	0.6779
1:1:1	0.312	0.7142	0.614	0.7744	0.6849

Tab. 4 Result of Trigram HMM in Argument

Unfortunately, the accuracy of the model has not improved after applying Trigram method. We conclude that this may be due to the number of training sets, which results in fewer observable trigrams model, thus fails to improve the accuracy.

3.3 CRF-Toolkit

Finally, we use a toolkit (CRF++0.58) to apply CRF model to this project.

We set the parameter as "-f 3 -c 1.5". The result is as follows, from the result, the CRF perform well in Argument but poor in Trigger dataset. It's because Argument has more features and types so it's suitable for CRF, which focus on the feature design.

Dataset	Type Correct	Accuracy	Precision	Recall	F1
Trigger	0.5071	0.7296	0.7565	0.4805	0.5878
Argument	0.9602	0.9533	0.9478	0.4774	0.635

Tab. 5 Result of CRF

4. Conclusion

We have applied HMM and CRF method into two different Chinese datasets to do event extraction. In this project, we found that the HMM model has better performance than the CRF model in Trigger dataset and poor in Argument dataset. This may be due to the different principle of the method. The CRF focus on the feature design and correlation of sequence.

Also, we found that the model performs well in Trigger dataset, which have less type to predict.

However, the Trigram HMM haven't led to any improvement, this may due to lack of training set.

For higher accuracy, we can try more complex network structures like LSTM or Bert and training with larger datasets.