# Report Of Assignment 3— Chinese Event Extraction

## 1 Task Description

Part of speech tagging is a common example of sequence labeling task which seeks to assign a part of speech to each word in an input sentence or document. In this task, we implement both Hidden Markov Model (HMM) and Conditional Random Field (CRF) to identify the trigger words and arguments in a given sentence. To achieve a better result, we try both bigram and trigram for HMM. The best test accuracy for trigger words and for arguments is 0.9534 and 0.7336, respectively.

## 2 Method

- Dataset: A sequence of segmented Chinese words.
- HMM: We use viterbi algorithm.
  - Bigram
    - Initialize $\sigma_0(s) = 1$ for $s$ the start state, and $\sigma_0(s) = 0$ for all other states (this is equivalent to having only the start state in the trellis at position zero)
    - For each value $i = 1, ..., n$, calculate:
      - $\sigma_i(s) = max_{s_{i-1}} P(s_i|s_{i-1})P(w_{i-1}|s_{i-1})\sigma_{i-1}(s_{i-1})$......(1)
      - $\phi_i(s) = argmax_{s_{i-1}} P(s_i|s_{i-1})P(w_{i-1}|s_{i-1})\sigma_{i-1}(s_{i-1})$ ......(2)
      - $s$means state, and $w$ means word.
    - Finally, fill out the end state of the trellis (position $n+1$) using the rules in (2) above.
  - Trigram
    - After extending to trigram, the probability of a tag depends on the previous two tags:
      - $\sigma_i(s) = max_{s_{i-1}} P(s_i|s_{i-1}, s_{i-2})P(w_{i-1}|s_{i-1})\sigma_{i-1}(s_{i-1})$
      - $\phi_i(s) = argmax_{s_{i-1}} P(s_i|s_{i-1}, s_{i-2})P(w_{i-1}|s_{i-1})\sigma_{i-1}(s_{i-1})$
    - We estimate the probability $P(s_i|s_{i-1}, s_{i-2})$ by a weighted sum of the unigram, bigram, and trigram probabilities:
      - $P(s_i|s_{i-1}, s_{i-2}) = \lambda_1 P(s_i) + \lambda_2 P(s_i|s_{i-1}) + \lambda_3 P(s_i|s_{i-1}, s_{i-2})$
    - We also implement Beam search to accelerate the running process.
- CRF:

  - We use the toolkit CRF++-0.58.

- Smoothing Method:
  - Add-$\lambda$-smoothing
  - Directly assign a value **M** to $P(w_{i-1}|s_{i-1})$ if $w_{i-1}$ is an unknown word and a value **N** to $P(s_i|s_{i-1})$ if it is zero in the training set.

# 3 Evaluation Result

## 3.1 Bigram HMM

We find out that Smoothing Method 2 is better than Smoothing Method 1 generally. With Smoothing Method 1, the best accuracy for trigger and argument in bigram HMM is 0.8184 and 0.7148. With Smoothing Method 2, the best accuracy for trigger and argument in bigram HMM is 0.9534 and 0.7336. Since there are a large amount of unknown words in the testing set, we find it reasonable for Smoothing Method 2 to work well. (Smoothing Method 2 assigns a greater probability for unknown words.)

Tab. 1: Bigram HMM with Smoothing Method 1

| Type | lambda | type correct | accuracy | precision | recall | F1 |
|---|---|---|---|---|---|---|
| Trigger | 0.001 | 0.8219 | 0.818 | 0.2998 | 0.8526 | 0.4436 |
| | 0.0001 | 0.8025 | **0.8184** | 0.3005 | 0.854 | 0.4445 |
| | 1e-5 | 0.8188 | 0.8184 | 0.3005 | 0.854 | 0.4445 |
| Argument | 1e-06 | 0.3035 | 0.7143 | 0.6091 | 0.803 | 0.6928 |
| | 1e-07 | 0.3036 | **0.7148** | 0.6098 | 0.8028 | 0.6931 |
| | 1e-08 | 0.3037 | 0.7147 | 0.6098 | 0.8024 | 0.693 |

Tab. 2: Bigram HMM with Smoothing Method 2

| Type | M | N | type correct | accuracy | precision | recall | F1 |
|---|---|---|---|---|---|---|---|
| Trigger | 1e-05 | 0.001 | 0.9616 | 0.9513 | 0.7708 | 0.6088 | 0.6803 |
| | 1e-05 | 0.01 | 0.9609 | 0.9532 | 0.774 | 0.635 | 0.6977 |
| | 1e-06 | 0.01 | 0.9611 | **0.9534** | 0.7748 | 0.638 | 0.6998 |
| Argument | 1e-05 | 0.001 | 0.4252 | 0.7289 | 0.7008 | 0.566 | 0.6262 |
| | 1e-05 | 0.01 | 0.3855 | 0.7312 | 0.6887 | 0.602 | 0.6425 |
| | 1e-06 | 0.01 | 0.3946 | **0.7336** | 0.6959 | 0.5969 | 0.6426 |

## 3.2 Trigram HMM

Counterintuitively, we find out that the result of Trigram HMM is not better than that of Bigram HMM. This phenomenon may be explained by two reasons: 1) We accelerate the running process by implementing Beam Inference, however, the globally best sequence can fall off the beam; 2) The part of word relies more on unigram and bigram probability than trigram probability in this particular dataset. We can see from Table 3 that, the greater the $\lambda_1$, the better the result is.

Tab. 3: Trigram HMM with Beam width $\beta = 5$

| Type | $\lambda_1$ | $\lambda_2$ | $\lambda_3$ | type correct | accuracy | precision | recall | F1 |
|---|---|---|---|---|---|---|---|---|
| Trigger | 0.1 | 0.2 | 0.7 | 0.5087 | 0.8096 | 0.2018 | 0.419 | 0.2724 |
| | 0.6 | 0.3 | 0.1 | 0.5018 | 0.85 | 0.2554 | 0.3985 | 0.3113 |
| | 1.0 | 0.0 | 0.0 | 0.4961 | **0.8606** | 0.2698 | 0.3737 | 0.3133 |
| Argument | 0.1 | 0.2 | 0.7 | 0.268 | 0.6023 | 0.5042 | 0.5273 | 0.5155 |
| | 0.6 | 0.3 | 0.1 | 0.3352 | 0.6033 | 0.5075 | 0.3783 | 0.4335 |
| | 1.0 | 0.0 | 0.0 | 0.3981 | **0.615** | 0.5432 | 0.2544 | 0.3465 |

## 3.3 CRF

Tab. 4: CRF with $f$ 3 $c$ 1.5

| Type | type correct | accuracy | precision | recall | F1 |
|---|---|---|---|---|---|
| Trigger | 0.9553 | 0.9471 | 0.7834 | 0.5226 | 0.627 |
| Argument | 0.4839 | 0.7366 | 0.7425 | 0.5257 | 0.6156 |

# 4 Conclusion

We implement Hidden Markov Model and Conditional Random Field for this Chinese event extraction task. We find out that Bigram HMM outperforms other models in this task. For future works, we can try more various combinations of parameters, or explore more suitable smoothing methods to achieve the best performance for each model.

Student ID: 16307090185 Name: Lin Yawen