Concept Tagging for Movie Domain Data

Anh Tu Phan - 1^{st} year student
Master of Computer Science
anhtu.phan@studenti.unitn.it

April 2021

Concept Tagging (entity extraction) is one of common tasks in Language Understanding. This report will try to segment the input which is a sentence talk about movies into constituents and label them using IOB-schemes.

There are several approach to solve concept tagging problem. This report will try to experiment Weighted Finite State Machines. Following the instruction from Language Understanding System Lab [1], the main contribution of this report are:

1. Try to preprocessing dataset
   - Remove stop word with tag O
   - Generalization of input: number generalization, lemmatize and stem word.
2. Evaluation and error analysis with different kind of dataset and different parameters of ngram model
3. Experiment with Joint Distribution Modeling

The implementation is published on https://github.com/anhtu95/lus-concept-tagging

1 Data Analysis

1.1 Data set

This report using *NL2SparQL4NLU Data Set* [2] which contains list of sentence talk about movies which is derived from NL2SparQL data set from Microsoft Research and split into training and test set. There are two type of format in data set which are utterances only:

```
who plays luke on star wars new hope
```

and concept tags (each token is assigned to one tag using IOB-scheme):

```
who     O
plays   O
luke    B-character.name
on      O
star    B-movie.name
wars    I-movie.name
new     I-movie.name
hope    I-movie.name
```

The overview of data set is described in Table 1; Most frequent words is reported in Table 2 and frequency of tags is shown in 3

Metric	Train	Test
Total Words	21453	7117
Total Utterances	3338	1084
Total Lexicon	1729	1039

Table 1: Some statistics of data set

Train		Test	
Word	**Frequency**	**Word**	**Frequency**
the	1337	the	406
movies	1126	movies	367
of	607	movie	200
in	582	of	194
movie	564	in	184

Table 2: Most frequent words

Tag	Train	Test
O	6360	5135
movie.name	1402	473
director.name	237	78
actor.name	227	80
rating.name	200	61
producer.name	195	73
country.name	190	62
movie.language	174	69
movie.subject	172	44
person.name	151	34
movie.genre	94	36
movie.release_date	87	28
character.name	49	15
movie.location	17	7
award.ceremony	10	7
movie.release_region	9	4
movie.gross_revenue	8	5
actor.nationality	5	1
actor.type	3	2
director.nationality	2	1
person.nationality	2	0
movie.description	2	0
movie.star_rating	1	1
award.category	1	2

Table 3: Frequency of Tags

1.2 Preprocessing data

On the one hand, it is clear that the number of word with tag O is major in data set (15391 words in training set). In addition stop word is common words in a language and it appears many times in data set. As a result, removing stop words with tag O is considered in this report. Because common words is removed, frequency cut off method will not be used to avoid losing words or tags with low occurrences in data set (for example removing word with tag *movie.star_rating* in training set). After removing stop word with tag O, most frequent words of training set is shown in Table 4. The number of word with tag O in training set reduce from 15391 to 6360.

Word	Frequency
movies	1126
movie	564
show	494
the	371
find	312

Table 4: Most frequent words after removing stop word with tag O

On the other hand, there are many different kinds of the same word. For example *to be* (is, are); words in third person; verbs in past and future tense ($was, were$). Therefore words will be lemmatized ($was, were, is, are \rightarrow be$, $plays \rightarrow play$, ...) and the reduction of word to their root (stem) ($movies \rightarrow movi$) is also consider in this report. In addition all number in dataset will be replaced by <*num*>. The below is some example of preprocessed sentence.

```
"who plays luke on star wars new hope"
-> "who play luke on star war new hope"

"who was the main character in men in black 3"
-> "who be the main charact in men in black <num>"
```

Table 5 shows statistics of data set after removing stop words and normalize words, the most frequent words is shown in Table 6.

Metric	Train	Test
Total Words	12422	3747
Total Lexicon	1405	889

Table 5: Some stastics of pre-processed dataset

Word	Frequency
movi	1690
show	505
find	396
the	371
produc	263

Table 6: Most frequent words of pre-processed dataset

2 Evaluation on original dataset (base line)

2.1 Evaluation

The base line pipeline of sequence labeling model λ is performed by composition of there components ($\lambda = \lambda_W \circ \lambda_{W2T_{MLE}} \circ \lambda_{LM_1}$). The first one is a Finite State Acceptor (FSA) λ_W which is represent an input sentence. The second one is a Finite State Transducer (FST) $\lambda_{W2T_{MLE}}$ to translate words into output labels. This FST will explore the relation between input and output. It will estimate probability of word given IOB tag $p(w_i|t_i)$ from data. The last one is FSA Language model to score the sequences of output labels. This model is build by using output label priors $p(t_i)$. Ngram model will be considered to build λ_{LM_1} and $\lambda_{W2T_{MLE}}$. The Table 7 show the result (average precision, recall and f1-score of all tags) of base line model with different parameters (ngram order and smooth method) of ngram model of λ_{LM_1} and $\lambda_{W2T_{MLE}}$. Table 8 show result in each tag with ngram order of λ_{LM_1} equals to 2 (value -1.0 mean that model does not label any word with this tag).

λ_{LM_1}	$\lambda_{W2T_{MLE}}$	Avg. Precision	Avg. Recall	Avg. F1
order=1, method=witten_bell	method=witten_bell	0.557	0.563	0.56
order=1, method=witten_bell	method=absolute	0.56	0.571	0.565
order=1, method=witten_bell	method=katz	0.549	0.573	0.561
order=2, method=witten_bell	method=witten_bell	0.761	0.717	**0.738**
order=3, method=witten_bell	method=witten_bell	0.726	0.726	0.726
order=4, method=witten_bell	method=witten_bell	0.723	0.729	0.726

Table 7: Base line with different parameters

2.2 Error Analysis and Discussion

The model could not label the word with tag rarely appear in training set. These words will be assigned to tag *O* or different tag which is similar meaning with correct tag in test set. For example:

- *award.category* appear one time with word *directing* in training set. Two word *best supporting* and *best director* are predicted as *O* in test set instead of *award.category* (The word *best* and *director* is assigned mainly to tag *O* in training set (*best*: 11/12 times, *director*: 115/115 times) and the word *supporting* does not appear in training set).

	precision	recall	F1	s
movie.name	0.804	0.808	0.806	473
director.name	0.603	0.543	0.571	81
actor.name	0.747	0.738	0.742	80
producer.name	0.811	0.589	0.683	73
movie.language	0.792	0.609	0.689	69
country.name	0.577	0.661	0.617	62
rating.name	0.951	0.951	0.951	61
movie.subject	0.75	0.682	0.714	44
movie.genre	1.0	0.722	0.839	36
person.name	0.513	0.588	0.548	34
movie.release_date	0.741	0.69	0.714	29
character.name	0.556	0.333	0.417	15
movie.location	0.0	0.0	0.0	7
award.ceremony	0.571	0.571	0.571	7
movie.gross_revenue	0.556	1.0	0.714	5
movie.type	-1.0	0.0	0.0	4
movie.release_region	-1.0	0.0	0.0	4
actor.type	1.0	1.0	1.0	2
award.category	-1.0	0.0	0.0	2
director.nationality	-1.0	0.0	0.0	1
actor.nationality	1.0	1.0	1.0	1
movie.star_rating	-1.0	0.0	0.0	1
Total	0.761	0.717	0.738	1091

Table 8: Base line with original data

- Word *turkish* is predicted as *movie.language* instead of *director.nationality* in test set. Because in training set *turkish* appear two time and is labeled as *movie.language*. The same error occurs with labeling *country.name* tag instead of *movie.release_region* tag.

In short, base line model is primarily statistically based on the number of time a word is assigned to a tag $p(w_i|t_i)$ and the probability of tag $p(t_i)$ in data set. This is a reason why the word does not appear in training set is assign to tag O and when a word is assign to different tag the model cannot label it correctly.

3 Evaluation on pre-processed dataset

3.1 Evaluation

The pipeline of sequence labeling model λ performed in this dataset is similar to base line model with added generalization component λ_G which applies pre-process method (removing stop word and normalize input as described in section 1.2). The pipeline now is $\lambda = \lambda_G \circ \lambda_W \circ \lambda_{W2T_{MLE}} \circ \lambda_{LM_1}$. The Table 9 show the result (average precision, recall and f1-score of all tags) of model with different parameters (ngram order and smooth method) of ngram model of λ_{LM_1} and $\lambda_{W2T_{MLE}}$. Table 10 show result in each tag with ngram order of λ_{LM_1} equals to 3.

λ_{LM_1}	$\lambda_{W2T_{MLE}}$	**Avg. Precision**	**Avg. Recall**	**Avg. F1**
order=1, method=witten_bell	method=witten_bell	0.642	0.608	0.625
order=1, method=witten_bell	method=absolute	0.646	0.61	0.628
order=1, method=witten_bell	method=katz	0.63	0.606	0.617
order=2, method=witten_bell	method=witten_bell	0.783	0.726	0.753
order=3, method=witten_bell	method=witten_bell	0.771	0.752	**0.762**
order=4, method=witten_bell	method=witten_bell	0.767	0.748	0.757

Table 9: Model with different parameters on pre-processed dataset

3.2 Error Analysis and Discussion

The average F1 score has been improved compared to base line model. One of the reason is the number of O tags in training set has decreased (from 15391 to 6360) so the probability of word given a tag O is decrease. For example $p(watch|O) = 5.11$ and $p(watch|I - movie.name) = 6.13$ in pre-processed dataset while $p(watch|O) = 6.02$ and $p(watch|I - movie.name) = 6.16$ in original data, as a result, the word *watch* is labeled correctly as $I - movie.name$ instead of labeled as O. However, same as base line model, because this method just modify data input and dose not change two ngram model λ_{LM1} and $\lambda_{W2T_{MLE}}$, the error on the word with tag rarely appear still remain (ex: *award.category*, *director.nationality*, ...).

	precision	**recall**	**F1**	**s**
movie.name	0.83	0.867	0.848	466
director.name	0.59	0.568	0.579	81
actor.name	0.695	0.825	0.754	80
producer.name	0.776	0.625	0.692	72
movie.language	0.81	0.681	0.74	69
country.name	0.662	0.741	0.699	58
movie.subject	0.75	0.75	0.75	44
movie.genre	0.906	0.806	0.853	36
person.name	0.565	0.382	0.456	34
movie.release_date	0.792	0.655	0.717	29
rating.name	0.88	0.88	0.88	25
character.name	0.875	0.467	0.609	15
award.ceremony	0.571	0.571	0.571	7
movie.location	0.0	0.0	0.0	7
movie.gross_revenue	0.375	0.6	0.462	5
movie.release_region	0.0	0.0	0.0	4
movie.type	-1.0	0.0	0.0	4
award.category	-1.0	0.0	0.0	2
actor.type	1.0	1.0	1.0	2
director.nationality	-1.0	0.0	0.0	1
actor.nationality	1.0	1.0	1.0	1
total	0.771	0.752	0.762	1042

Table 10: Result on pre-processed dataset

4 Joint Distribution Modeling

In the previous model, the probability of tag given a word is calculated by measuring separately the probability of words and tags:

$$p(t_1^n|w_1^n) \approx \prod_{i=1}^{n} p(w_i|t_i)p(t_i)$$

In joint distribution modeling, the probability of tag with word is jointly calculated:

$$p(w_1^n, t_1^n) \approx \prod_{i=1}^{n} p(w_i t_i|w_{i-N+1}^{i-1} t_{i-N+1}^{i-1})$$

Table 11 show the result (average precision, recall and f1-score of all tags) of joint distribution modeling with different parameters of ngram on orginal dataset and result on pre-processed dataset is shown in Table 12.

Parameters	**Avg. Precision**	**Avg. Recall**	**Avg. F1**
order=2, method=witten_bell	0.75	0.689	0.718
order=2, method=absolute	0.75	0.689	0.718
order=2, method=katz	0.753	0.697	0.724
order=3, method=witten_bell	0.751	0.69	0.719
order=3, method=katz	0.751	0.696	0.722

Table 11: Joint distribution Modeling on original dataset

Parameters	**Avg. Precision**	**Avg. Recall**	**Avg. F1**
order=2, method=witten_bell	0.792	0.702	0.745
order=2, method=absolute	0.782	0.694	0.735
order=2, method=katz	0.764	0.687	0.724
order=3, method=witten_bell	0.789	0.701	0.742
order=3, method=katz	0.761	0.689	0.723

Table 12: Joint distribution Modeling on pre-processed dataset

References

[1] Language understading systems lab, https://github.com/esrel/LUS.

[2] Nl2sparql4nlu dataset, https://github.com/esrel/NL2SparQL4NLU.