Language Understanding Systems first mid-term project

Alessia Tovo - 189192
University of Trento - DiSi
alessia.tovo@studenti.unitn.it

Abstract

This document describes the work done for the first mid-term project of Language Understanding Systems course. The project consists of develop a Spoken Language Understanding (SLU) Module for Movie Domain using NL-SPARQL Data Set.
The first section of this report contains a short data analysis. Then in the following section there is the description of the project implementation and as last section there are all the results obtained.

1 Introduction

This document has been written with the aim to explain the work done for the first midt-term project of LUS course.
The purpose of this project is to develope a Spoken Language Understanding Module for Movie Domain.
In order to develop this module different technologies have been used:

- Python scripts
- Bash scripts
- OpenFST
- OpenGRM

The SLU Module consists of take in input sentences and give to each word of each sentence a predicted concept tag, on the basis of the train model.
Section 2 contains a description of the Data Set given and a short analysis of it.
Section 3 describe the first SLU Module created with the basic features.
Sections 4, 5 and 6 explain the work done using the advanced features.
In the last section all the evaluation results have been reported in order to understand better the results obtained from the different modules.
All the automata created are compiled with the **OpenFST** tools that allow us to create easily the transducers given a *lexicon* file.
The **OpenGRM** libraries helped us to make n-gram language model combining different smoothing methods (katz, witten bell, absolute, kneser ney, presmoothed and unsmoothed) and different sizes of ngram model (from 1 to 5).
In the sections 8 is possible to see how this parameters affect the correctness of outputs.

2 Data Analysis

The Data Set used to train and test the SLU Module built is the **NL-SPARQL Data Set**.
It contains two group of data:

- basic features composed by *NLSPARQL.train.data* and *NLSPARQL.test.data*;
- advanced features composed by *NLSPARQL.train.feats.txt* and *NLSPARQL.test.feats.txt*.

The first group is used to build the basic SLU Module, while a combination of the two groups is used to develop the other modules.
The basic features are basically a sequence of sentences where each word is assigned to a certain *IOB tag*. An *IOB tag* format is a common way to tag tokens respect to a certain domain, in this case the Movie Domain.
The train set contains 21453 words, which 1728 are unique, and each word is assigned to one of the 41 different *IOB tags*.
In the test set there are 7117 words, which 1039

are unique, and each wors is assigned to one of the 39 different *IOB tags*.
In table 1 is possible to see which are the couple *word - IOB tag* most frequent in the train set with basic features. As can be seen, the more frequent couple are the ones composed by a word and the IOB tag "O", which means that the word is *Outside of span*.
A table for less frequent couple *word - IOB tag* is not shown beacuse the frequency counter is equal to 1, but in table 3 is possible to see how many words have low frequency. In the train set there are 1166 couple *word - IOB tag* that have this frequency.

Word	IOB tag	Occurencies
movies	O	1126
the	O	966
movie	O	563
what	O	544
me	O	534
of	O	518
in	O	498
show	O	493
for	O	463
find	O	312

Table 1: Most frequent occurencies word-IOB.

It is interesting to know which are the word most and less frequent into the train set, in order to develop a different Modules that takes care of this information. In the table 2 is possible to see which are the word with greater frequency than others and in table 3 the word, grouped per category, that appears less frequent into the document.

Word	Occurencies
the	1137
movies	1126
of	607
in	582
movie	564
what	545
me	539
show	494
for	472
is	335

Table 2: Most frequent words.

Word per frequency	Occurencies
Word with frequency 1	778
Word with frequency 2	237
Word with frequency 3	153
Word with frequency 4	103
Word with frequency 5	55
Word with frequency 6	46
Word with frequency 7	36
Word with frequency 8	29
Word with frequency 9	19
Word with frequency 10	22

Table 3: Less frequent words per category.

In the table 4 is possible to see which are the 10 more frequent IOB tag in the train set.

IOB tag	Occurencies
O	15391
I-movie.name	1755
B-movie.name	1402
B-director.name	237
B-actor.name	227
I-director.name	218
I-actor.name	210
B-rating.name	200
B-producer.name	195
B-country.name	190

Table 4: Most frequent IOB tags.

3 Basic SLU Module

The first Module has been developed using only the basic features, so the information about words and IOB tags.
The automa created is an automa with a single state 0 and transitions of type $Word \rightarrow IOBtag$, like in the figure 1. The weight of the transition is computed using a simplify assumption of the *Bayesan's rule*, so the weight of the arc is computed as in figure 2.

Figure 1: Automa

$$P(w_i|t_i) = \frac{c(w_i, t_i)}{c(t_i)}$$

Figure 2: Weight transitions

The automa created map each known word to a set of possible tags, while the unknown words are mapped to every tags found in the train set with equal probaility, thus $1/number\_of\_tags$.
Than, from the automa a Weighted Final State Transducer has been generated using *OpenFST* tools. Since the transducer contains multiple path, for some words, to map word to IOB tag, the `fstshortestpath` command has been used in order to take only the path with higher probability and consequenlty less cost.
To the transducer is given in input all the sentences contained in the test set, in order to assign to each word of each sentence the most probable IOB tag. Than, an evaluation has been made to understand how much correct is the result obtained.

4 Lemma and PoS tag transitions

With the same technique explained in section 3 other two automata and WFST have been developed using, this time, also the advanced features.

4.1 $Lemma \rightarrow IOBtag$ transition

The first automa, instead of consider words, uses their lemmatization. So each transition has as input the lemma and as output the IOB tag. The weight of each transition is computed as before, but instead of consider the couple *word-IOB tag*, the couples *lemma-IOB tag* are considered.
From this automa the appropriate WFST has been generated and the test set is given in input to it. As before the purpose is to assign to each lemma the correct IOB tag and a final evalutaion has been made to understand the transducer's efficiency.
Since there are less lemma than words, we expect that the perfomance decrease, but all the details are reported in the last section of the document.

4.2 $PoS \rightarrow IOBtag$ transition

The second automa has been developed as an experiment with very low expectations. In fact, the automa use the *Part of Speach tags* as input feature and the IOB tag as output.
In the train set there are only 49 PoS tag that are mapped to 41 IOB tags. The weight of each transition has been computed like the two before, with the approssimation of the Bayesan's rule. This time, the couple *PoS - IOB* are consider in the nominator of the formula.
Also in this case the purpose is to assign the most correct IOB tag to the input and an evaluation has been made to understand the perfomance of this development decision.

5 Lexicon cut-off

In this part of the project, the transitions applied to the automa are still $Word \rightarrow IOBtag$ but are not included words that appear more or less than a certain frequency. In other words it has been applied a *lexicon cut-off*. It is possible to decide which is the lower and the upper bounds, but two different automata are generated. This decision has been made since apply both cut-off to the automa could generate too loss of information.
So, a first automa is refering to the cut-off of word that appears less in the train and a second automa does not consider words that appear in the document too much. Tables 2 and 3 could help to decide the most appropriate hyperparameters.
This cut-off generates a different approach in computing the weight of transitions. In fact, in previous automata the occurencies of each couple *word-IOB tag* have been divided by the total number of the IOB tag considered. Since some words have been deleted, also the related IOB tag has been deleted from the train set and the number to the denominator changes.
Some IOB tags could be associated only to those words with very low frequency, like 1. So the total number of tags changes and the transtitions $< unk > \rightarrow IOBtag$ became less.
Two separated test hs been applied to the two automata in order to obtain to different output, one with a lower cut-off approach and another with a upper cut-off approach. Then both outputs have been evaluated how you can see section 8.

6 Lemma_pos to IOB transition

In order to make more "forced" the transition $word \rightarrow IOBtag$ we consider the word as a couple composed by the lemma and its PoS tag.
The sequence of words of the train set has been

change into a sequence of *lemma_pos* words. These informations came from a combination of the two type of data, the basic and the advanced ones.This experiment has been made to see how much words are tied up with their lemmas and their PoS tags.
The weight of the transitions has been computed like previous one, with the approssimation of *Bayesan's rule*. At the nominator we have the number of triple *lemma - PoS- IOB* and at the denominator we have the number of the IOB tag considered. We do not consider the PoS tag at the denominator because they are part of the input of transition and are linked with lemmas. The IOB tag is the only part of the output.
The same type of test and evaluation has been made to this Module, like previous ones.

7 Independent features

This experiment breaks every link between words, PoS, lemmas and IOB tags.
Three different automata have been generated:

- First automa: transformation $Word \rightarrow lemma$ has been generated. Weights are computed with the *Bayesan's rule*.
- Second automa: transformation $lemma \rightarrow PoStag$ has been generated. Weights are computed with the *Bayesan's rule*.
- Second automa: transformation $PoS \rightarrow IOBtag$ has been generated. Weights are computed with the *Bayesan's rule*.

A composition of the three automata generate a fourth automa that map words to IOB tags. Unlike the Basic Automa (3), which always map words to IOB tag with a certain probability, here the probabilities change totally, because of the `fstcompose` operation. Compose the three automata means associate to a lemma, which is the output of the first automa, to a PoS tag, which is the ouput of the second automa. To make the compose we firstly apply the `fstshortestpath`, so the lemma is associated to a PoS tag different from the one contained in the train set. Then, the result of this composition, is compose to the third automa. Also in this case the PoS tag, output of the composition done, is mapped to an IOB tag, which may be different from the relation written in the train set.
With this experiment we expect that the performance will decrease and in the following section more detail are reported.

8 Evaluation

To each transducer has been applied the OpenGRM methods to model the n-gram language models and then predict the correct IOB tag for words contained in the test set. Different combinations of all smoothing methods and ngram size has been used to generate output. Then they have been evaluated to understand how much correct is the results.
In table 5 is possible to see, for each implementation, which is the best results. The evaluation is given by the *F-measure score*
.

Implementation	**Smooth. Method**	**Size**	**Score**
Basic	Abs. - Wit. bell	2	76.37
Lemma	Absolute	2	75.82
Lemma_PoS	Absolute	5	74.99
Up. cut-off (1000)	Witten bell	5	74.55
Low. cut-off (1)	Absolute	4	73.67
Low. cut-off (10)	Absolute	4	52.81
Ind. features	Witten bell	5	25.01
PoS	Kneser ney	5	21.59

Table 5: Best F-measure for each implementation.

As can be seen the best performance is obtained taking all the words and map them directly to the IOB tag. This beacuse there is more distinction between one word and another.
Future development could be understand how much correct is the prediction of tags I and B, than check their movie domain.