Language Understanding Systems
Mid-term project

Kateryna Konotopska
198234
kateryna.konotopska@studenti.unitn.it

Abstract

The aim of this report is to document main concepts and results of the mid-term project developed for the Language Understanding Systems course in the academic year 2017/2018.
The objective of this project is to create a concept tagger for the movie domain. Considerations on provided data, main concepts of concept tagging, issues encountered and proposed solutions are described in this report.

1 Introduction

Given a set of sentences, the task consists in assigning for each word in a sentence a concept tag from the movie domain.
For this purpose firstly train and test datasets were analyzed, extracting from test and train some statistics and comparing their distributions of words. Subsequently a basic solution was build using open-source tools for working with formal grammars and finite state transducers: OpenGrm and OpenFst libraries. During this phase some issue points in dataset were noticed such as poor information contained in one of the tags and high OOV rate present in the test set. To manage better problems caused by these aspects some improvements of the basic model were implemented. Lastly all the models were compared with simple baseline models and afterwards the most evident errors were investigated.
The following sections will explain better each stage described above.

2 Methods

2.1 Data analysis

Analyzing a dataset is important before starting to work on a problem like the given one since it allows to have a quick look on the complexity of the problem, dimensions and differences of train and test datasets.
The dataset used for this project is called NLSPARQL, train and test files consists of sentences as input and concept tags as output. Additionally files containing POS tags and unique lemmas of each word in test and train were provided.
In Table 2 you can look at some basic statistics of the test and train files.

Statistics	Train	Test
# sentences	3338	1084
# tokens	21443	7117
# unique tokens	1728	1039
# unique concept tags	24	23
max. sentence length	21	22
average sentence length	6.4	6.6

Table 1: Simple statistics on the provided dataset.

Distributions of tokens and concept tags are similar and follow the Zipf's law.

Of the 7117 tokens in test 260 tokens were not in the dictionary of which 246 are unique. The OOV rate is approximately 23.7 %. Out-of-vocabulary tokens contain a lot of typos, some of them are proper nouns.

2.2 Basic model

Formally the problem can be described in the following way: given a sequence of words $\mathbf{w} = (w_1, w_2, ..., w_n)$, find the most probable sequence of tags $\mathbf{t} = (t_1, t_2, ..., t_n)$:

$$\mathbf{t} = \underset{\mathbf{t}}{\operatorname{argmax}}\, P(\mathbf{t}|\mathbf{w}) \tag{1}$$

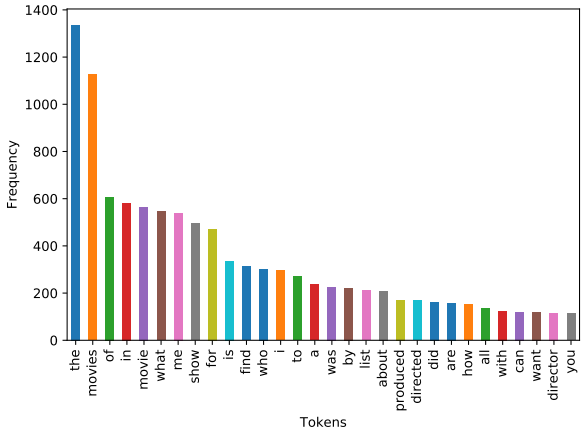

Figure 1: Train lemmas distribution

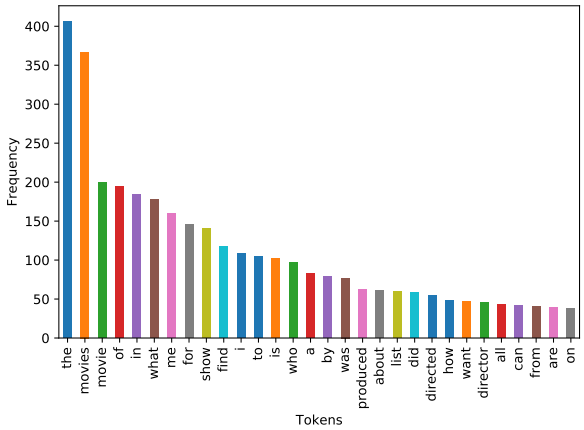

Figure 2: Test lemmas distribution

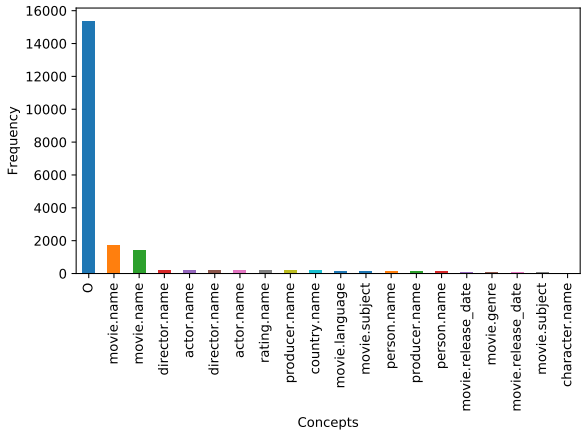

Figure 3: Train concepts distribution

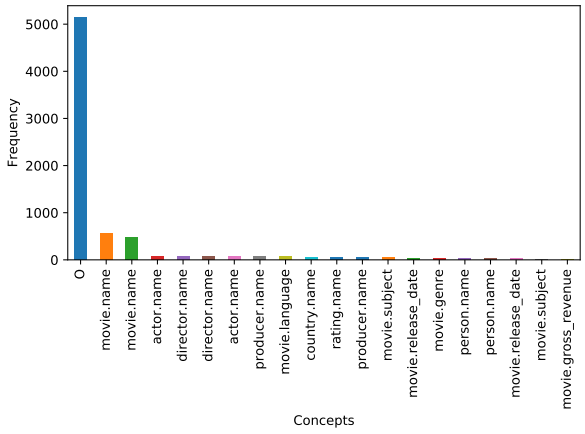

Figure 4: Test concepts distribution

According to Bayes' rule:

$$\mathbf{t} = \underset{\mathbf{t}}{\operatorname{argmax}} \frac{P(\mathbf{w}|\mathbf{t})P(\mathbf{t})}{P(\mathbf{w})} \tag{2}$$

Since P(**w**) is fixed for all tags we have:

$$\mathbf{t} = \underset{\mathbf{t}}{\operatorname{argmax}}\, P(\mathbf{w}|\mathbf{t})P(\mathbf{t}) \tag{3}$$

We will approximate this formulation making some simplifying assumptions:

- probability of the word depends only on it's own tag:

$$P(\mathbf{t}) = \underset{t_1,t_2,\ldots,t_n}{\operatorname{argmax}}\, P(t_1|w_1)P(t_2|w_2)...P(t_n|w_n) \tag{4}$$

- first order Markov assumption:

$$P(\mathbf{t}) = P(t_2|t_1)P(t_3|t_2)...P(t_n|t_n-1) \tag{5}$$

In case of trigram model we will consider second Markov assumption and so on. To recap:

$$\mathbf{t} = \underset{t_1,t_2,\ldots,t_n}{\operatorname{argmax}} \prod_{i=1}^{N} P(w_i|t_i)P(t_i|t_i-1) \tag{6}$$

Firstly P($w_i|t_i$) and P($t_i|t_i-1$) were calculated as :

$$P(w_i|t_i) = \frac{C(t_i, w_i)}{C(t_i)} \tag{7}$$

$$P(t_i|t_i-1) = \frac{C(t_i-1, t_i)}{C(t_i-1)} \tag{8}$$

where $C(x)$ is count of x.

Final state transducer was used to encode (6). It was composed of word:concept transducer encoding (7)

and concept language model containing all the conditional probabilities of (8). All the components were implemented using OpenGrm and OpenFst tools, which permit to compute quickly argmax as transition that minimizes the costs of the final transducer, where costs are weights of arcs of the final model and are represented by negative logs of probabilities. Final state acceptor was prepended to encode one specific sentence at a time.

2.3 Tagging unknown words

Built model is capable of assigning labels only to words encountered in the train set. In order to deal with unseen words <unk> token was used: every time a model encounters an unknown word, it substitutes it with <unk> token. In order to decide how to assign concept tags to unknown words two strategies were applied. In particular we have to decide how to assign probabilities to all possible concept tags in transducer.

1. Uniform probability: all concept tags have the same probability to be assigned to unknown word:

$$P(t_i| < unk >) = \frac{1}{\#concepttags} \quad (9)$$

2. Cut-off: probability distribution of concept tags is taken from concept tags assigned to "unknown" words in the train set. Given θ, probability distribution can be calculated in the following way:
Let consider couples $S = \{(t, c)|C(t, c) < \theta\}$, where $C(t, c)$ is the number of occurrences of couple (t, c) in the training set; corresponding tags $Q = \{c|\exists t \text{ s.t. } (t, c) \in S\}$ and the set of tokens in training set: T. For each $c \in Q$ $P(c| < unk >)$ is given by:

$$P(c| < unk >) = \frac{\sum_{t \in T} C(t, c)}{\sum_{(t,c) \in S} C(t, c)} \quad (10)$$

2.4 Improvement of basic model

Analyzing test and train it was noticed that the majority of concepts were 'O' tags. This fact influences our model based on conditional probabilities since P('O') is higher then probabilities of other concepts and 'O' tags do not contain much information.
Two different improvements were used to overcome this problem:

1. For all tokens substitute 'O' tags with corresponding tokens

2. For all tokens substitute 'O' tags with concatenation of corresponding lemmas and POS tags provided as additional information to train and test sets

The project can be found on GitHub: https://github.com/katerynak/lus_project1

3 Results

Different versions of basic and improved models were compared with simple baseline models, which assign tags in three different ways:

- random: each concept tag has the same probability to be assigned to the given token
- majority: the most common concept in the training set was assigned to all tokens
- train distribution: the probability distribution of concept tags was token from the training set

All the baselines presented poor results.

Baseline	Accuracy	Precision	Recall	F1-score
random	2.19	0.32	2.02	0.56
majority	72.15	0.00	0.00	0.00
train dist.	53.17	1.43	2.57	1.84

Table 2: Results of baseline models evaluation.

Basic model was evaluated with different smoothing methods, ngram order sizes, unknown tokens handling. The most relevant results are reported in the following table.

smoothing	ngram order	F1-score
witten_bell	2	76.37
absolute	2	76.37
witten_bell	5	76.32
presmoothed	2	76.27
kneser_ney	2	76.27

Table 3: Best 5 results and corresponding configurations

Cut-off strategy did not improve the results and in some cases leaded to drop in terms of F1-score.

In order to evaluate improved models 10-Fold cross-validation was performed trying the same parameters of basic model and the best result was found with following configuration:

smoothing:	**kneser_ney**
ngram order:	**3**
F1-score:	**82.04**

Using test set as dev set different best score was obtained:

smoothing:	**kneser_ney**
ngram order:	**4**
F1-score:	**82.74**

4 Discussion

In order to analyze results of built SLU model output of the model with highest cross validation result was analyzed.

Concepts presenting the worst results are: award.category, movie.star_rating, movie.type, director.nationality, movie.release_region, movie.gross_revenue, award.ceremony.
Classes such as award.category, movie.star_rating, movie.type have one or none occurrences in training set. Director.nationality tag is confused with movie.language and vice versa. Only sequences containing words 'million' and 'dollars' are tagged with movie.gross_revenue since all training examples contained these tokens, in some cases it leads to confusion with movie.name tags. Movie.release_region is often tagged as country.name since they are similar and country.name tags are more frequent in training set. I-award.ceremony concept is not present in the test set and identical word sequences such as 'academy award' or 'oscar award' are tagged in different way in test and train set (with 'O' tag instead of 'I-award.ceremony' tag in test); this can be an annotation problem in test set.

References

Jurafsky, Daniel and Martin, James H. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, 2008.