Language Understanding Systems - Mid Term Project
Sequence Labelling with WFST

Sebastiano Chiari
220527
sebastiano.chiari@studenti.unitn.it

Abstract

This document describes the approach used to complete the mid-term project of the Language Understanding System class.

The aim of this project is to develop a **SLU** (Spoken Language Understanding) module capable of assigning Movie Domain tags to words and to analyse its performances with different configurations.

All the code can be found on Github [1].

1 Introduction

A **concept tagger** is a module capable of assigning a label (a tag or a concept) to words belonging to a given sentence.

who plays luke on star wars new hope

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

Table 1: Concept tagging output

Each of the sentences' words is tagged according to the **IOB format**:

- **O**, words out of context
- **B-concept.type**, words that indicates the beginning of a concept (actor, movie, director, etc.). Concepts composed by only one word will have this tag.
- **I-concept.type**, words inside the concept (the ending word of a concept has this tag too)

[1] https://github.com/sebastianochiari/LUS-midterm-project

2 Data Analysis

The main dataset used to train and test the main features of the proposed model is a modified version [2] from the Microsoft Research NL2SparQL dataset. The dataset is already split into train and test sets following the original NL2SparQL splits.

Metric	Train	Test
Total Words	21,453	7,117
Total Utterances	3,338	1,084
Unique Words	1728	1039
IOB-tags	41	39

2.1 Concept Distribution

The table 2 describes the concept distribution and their related frequencies inside the dataset:

The large majority of the tags (not reported into the table above) are **O** tags, out-of-concept (72% both for the training and the test sets). This is an helpful insight in order to make some further improvements to the basic model (section 3.2).

It is also important to underline that some tags (such as movie.type, movie.description, person.nationality, etc.) are not present in both train and test sets: this is a crucial information in order to deal with unknown tags. The **Out Of Vocabulary ratio** indicates how many words in the test set are not present in the train set lexicon. In this case, the OOV ratio of the test set is 23.67% over 1039 unique tokens of the test set.

2.2 More datasets

More datasets were given in order to perform comparisons with the model developed and trained over the main dataset NL2SparQL.

All the following datasets have been preprocessed in order to convert the given raw data into

[2] https://github.com/esrel/NL2SparQL4NLU

Concept	Train	Test
movie.name	3157	1030
director.name	455	156
actor.name	437	157
producer.name	336	121
person.name	280	66
movie.subject	247	59
rating.name	240	69
country.name	212	67
movie.language	207	72
movie.release_date	201	70
movie.genre	98	37
character.name	97	21
movie.gross_revenue	34	20
movie.location	21	11
award.ceremony	13	7
movie.release_region	10	6
actor.nationality	6	1
actor.type	3	2
director.nationality	2	1
movie.description	2	0
person.nationality	2	0
movie.star_rating	1	1
award.category	1	4
movie.type	0	4

Table 2: NL2SparQL4NLU concept distribution

the needed file format required as input for the proposed model.

More insights about dataset size, concept distribution, OOV etc. are available in the corresponding data-analysis folder that can be found inside the repository of this project [3].

2.2.1 ATIS dataset

The ATIS dataset [4] is a collection of air travel information. This is the most interesting dataset because of its higher number of utterances used for training (4978 sentences, compared to the 3338 of the NL2SparQL). Due to this discrepancy, it can be expected a boost in performance thank to the higher amount of training data, that should produce a better sequence labelling during testing.

2.2.2 Corpora for evaluating NLU services

The following datasets is a collection of multiple corpora [5]: **AskUbuntu Corpus** (162 questions and answers from https://askubuntu.com.); **Web Applications Corpus** (89 questions and answers from https://webapps.stackexchange.com.) and **Chatbot Corpus** (206 questions from a Telegram chatbot for public transport in Munich).

Due to the lack of training and testing data (the proposed model performs best with large dataset which allow an intensive training phase) and due to the specific characteristics of this data collection (the aim of the benchmark and the structure of the annotations are towards intent recognition, not sequence labelling), these corpora are considered as non useful for this project purpose and thus they are not going to be considered.

2.2.3 Snips Voice Platform NLU benchmark

Within the work done during the 2017 improvement while analyzing Snips Voice Platform [6], benchmarks were made for seven chosen intents: more than 2000 queries have been generated for each intent with crowdsourcing methods. Although the goal of this work was to compare natural language understanding services offering custom solutions according to intent recognition, the collected corpora are sufficiently articulated and such as to allow the creation of a concept tagger based on these.

For time reasons, only two of these seven intents (chosen randomly) were considered and the results can be found in Section 4.4.

3 Methodology

The common pipeline for a sequence labelling model can be express in the following way:

$$\lambda = \lambda_W \circ \lambda_{W2T} \circ \lambda_{*LM}$$

- λ_W, Finite State Acceptor (FSA) representation of an input sentence
- λ_{W2T}, Finite State Transducer (FST) to translate words into output labels (e.g. iob + type)
- λ_{*LM}, FSA Ngram Language Model to score the sequences of output labels

[3] https://github.com/sebastianochiari/LUS-midterm-project

[4] https://github.com/howl-anderson/ATIS_dataset

[5] https://github.com/sebischair/NLU-Evaluation-Corpora

[6] https://github.com/sonos/nlu-benchmark

More, the proposed model uses a Maximum Likelihood Estimation approach. The FST λ_{W2T} has been modified in order to take into account also the relation between input and output, by estimating $p(w_i|t_i)$ from data.

3.1 Base model implementation (*minimum*)

While generating the lexicon, **cutoff** can be applied (*by default it is set equal to 2*), where words with lower frequency of appearance are not included inside the lexicon.

In order to handle unknown words, the probability for each possible pair `<unk>-tag` (which is not considered when computing the MLE for all the possible pairs `word-tag`) has been computed in the following way:

$$p(< unk >, tag) = -log(\frac{1}{\#tags})$$

The base model implementation has been trained over 5 different ngram order, from 1 to 5 (the average length of the queries in the dataset is 6 so there is no reason to proceed with further ngram orders) and for each smoothing method, namely *absolute*, *katz*, *kneser_ney*, *presmoothed*, *unsmoothed* and *witten_bell*. The **n-gram order** takes into account how many of the previous tags are considered while computing the probability of the current tag. The **smoothing method** allows to add some probability to unseen events, avoiding the whole sequence from getting 0 probability.

3.2 'O' tags removal (*2018 improvement*)

Another possible improvement is dealing with the O tags, by removing it from the training set. As shown in section 2.1, the majority of the tags are O tags, out-of-concept, and this brings no valuable information to an important slice of the dataset used during training.

A solution can be replace the O tag with the word itself (i.e. (who, O) becomes (who, O-who)).

who plays thor on the avengers

who	O-who
plays	O-plays
thor	B-character-name
on	O-on
the	B-movie-name
avengers	I-movie-name

By reducing the impact of the O-tags, the model (as shown in section 4.2) gains more knowledge about the other tags during the training phase, allowing for better results in the prediction one. This improvement has been applied over the base model discussed in section 3.1.

3.3 HMM (*2020 improvement*)

Sequence labeling for language understanding could be also approached using Hidden Markov Models (similar to Part-of-Speech tagging) through a joint-distribution modelling. The following expression describes how the probability of a certain tag given a certain word is computed:

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

From an implementation perspective, the biggest shift is the implementation of a world-tag pair system.

4 Performance Evaluation

Evaluation has been performed through the given script `evaluate.py` and the listed results are compared with the most informative metric available, the **F1 score**, which combines precision and recall and usually is more meaningful than single metrics such as accuracy when comparing different models' performances.

$$F_\beta = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

All the results can be found in the corresponding method folder inside the project repository [7]. In this report, only the relevant performances are discussed.

4.1 Base model performance

First, it is useful to take a base-line performance in order to compare all the future edits and improvements done at the model. For this task, the model with **unsmoothed method** and **n-gram order** equal to **1** was chosen.

Method	Precision	Recall	F1 Score
unsmoothed	0.5492	0.5728	0.5607

In Table 3 the model performances with different methods and same n-gram order (equal to 2) are highlighted.

[7] `https://github.com/sebastianochiari/LUS-midterm-project`

Method	Precision	Recall	F1 Score
absolute	0.7592	0.7167	**0.7373**
katz	0.7580	0.7149	**0.7358**
knesery_ney	0.7572	0.7149	**0.7355**
presmoothed	0.7599	0.7167	**0.7377**
unsmoothed	0.5492	0.5728	0.5607
witten_bell	0.7601	0.7176	**0.7383**

Table 3: Smoothing Method Variations with the base model

In Table 4, the model performances with different n-gram orders are highlighted, using the same method, absolute, which turned out to be the best in terms of performances. Table 4 also shows that the n-gram order achieving the best results is 2: this can be explained with the fact that most of the sentences are not very long and the bigram structure is powerful enough to model this kind of data.

N-Gram order	Precision	Recall	F1 Score
1	0.5492	0.5728	0.5607
2	0.7592	0.7167	**0.7373**
3	0.7226	0.7213	0.7220
4	0.7227	0.7241	0.7234
5	0.7276	0.7250	0.7263

Table 4: N-Gram Order Variations with the base model

As can be seen from the results obtained (Table 3 and Table 4), the smoothing method does not have a great impact on the results, while the effect produced by the modification of the n-gram order is decidedly greater.

For all the results illustrated up to now, a **cutoff** equal to 2 has been used by default. In this way, words that appear with a low frequency are eliminated. However, given the nature of the dataset, having many particular names (such as film names, actors, directors, etc.) which however appear few times, the cutoff could worsen the performance of the model, eliminating portions of data that are actually fundamental. For this reason, it has been tested no cutoff with the best combination obtained so far, method absolute and n-gram order equal to 2, and the results are better, as expected.

cut-off	Precision	Recall	F1 Score
no	0.7869	0.7415	**0.7635**
2	0.7592	0.7167	0.7373

Table 5: Cutoff effect with the base model

For the base model, the best result obtained is the one trained with n-gram order equal to **2**, smoothing method **absolute** and **no-cutoff** applied.

4.2 No O-tags model performance

As expected, the model gains more knowledge about the other tags during training and it improves the overall performances in concept tagging, as can be seen in Table 7.

Exploiting the knowledge acquired with the base model and the assumptions drawn during the analysis of the results, we could expect that the best performances are obtained also in this case with a model that uses the best configuration found in the previous one. However, the best configuration for this model is with smoothing method **kneser_ney**, n-gram order equal to **4** and **no-cutoff** applied.

method	ngo	cutoff	Precision	Recall	F1 Score
absolute	2	no	0.7827	0.7992	0.7909
		2	0.7651	0.8001	0.7822
kneser_ney	4	no	0.8280	0.8340	**0.8310**
		2	0.8199	0.8304	0.8251

Table 6: No-O tags model performances

4.3 HMM perfomances

No substantial improvements can be found even using the Hidden Markov Model implementation. As can be seen from the table below, the results are on average in line with those of the basic model and are even lower than the without O-tags improvement.

ngo - method	basic	no-O	HMM
1 - unsmoothed	**0.5607**	0.5290	0.5600
2- katz	0.7358	**0.7820**	0.7238
3- unsmoothed	0.7207	**0.7793**	0.7253
4 - witten_bell	0.7236	**0.8125**	0.7258
5 - witten_bell	0.7301	**0.8143**	0.7265

Table 7: F1 score comparison between the base model, no-O improvement and HMM

4.4 Other datasets on base model performance

4.4.1 ATIS dataset

As expected, having almost twice as many utterences available in the testing phase creates a considerable boost in performance. The base model trained on the ATIS dataset outperforms both the

NL2SparQL trained over the base model and also over any improvement.

method	ngo	Precision	Recall	F1 Score
kneser_ney	4	0.8856	0.8815	0.8835
absolute	5	0.8879	0.8794	0.8836
kneser_ney	5	0.8897	0.8843	0.8870
witten_bell	5	0.8882	0.8794	0.8838

Table 8: ATIS dataset best results over the base model

4.4.2 Snips Voice Platform NLU benchmark

The NLU benchmark dataset gives us a new perspective over how performance can be affected by the composition of the corpora. The base model has been tested with two intents: BookRestaurant and GetWeather. Despite the fact that they are both corpora with fewer utterances than NL2SparQL, the general performances are better and the cause is to be found in the composition of the corpora itself: in fact, we find a much lower percentage of OOV words (by looking at the BookRestaurant, we can find a 57% of OOV words, while in the NL2SparQL the OOV words percentage is 71%). This allows the concept tagger module to learn better in the training phase and produce better results during testing.

method	ngo	Precision	Recall	F1 Score
absolute	4	0.8593	0.8566	0.8580
kneser_ney	4	0.8625	0.8598	0.8611
witten_bell	4	0.8593	0.8566	0.8580
kneser_ney	5	0.8482	0.8535	0.8509

Table 9: BookRestaurant best results over the base model

method	ngo	Precision	Recall	F1 Score
absolute	5	0.8688	0.8760	0.8724
kneser_ney	5	0.8588	0.8801	0.8693
unsmoothed	5	0.8641	0.8677	0.8659

Table 10: GetWeather best results over the base model

5 Conclusions

The original goal of developing an SLU for concept tagging in the movie domain has been achieved. Several models have been implemented and their performances have been commented on. In some cases (section 4.2), the improvements have led to a significant increase in performance (regarding the F1 score and the ability to perform better concept tagging).

More, other datasets were used with the same models and the hypothesis initially formulated in the theoretical introduction part was confirmed (section 4.4): the proposed models performs better when a large dataset is used to train the system, that produces a better sequence labelling during testing.