Natural Language Understanding for Spoken Data

Wencan Luo
University of Pittsburgh
Pittsburgh, PA 15260, USA
wencan@cs.pitt.edu

Lin Zhao
Bosch Research and Technology Center
Palo Alto, CA 94304, USA
lin.zhao@us.bosch.com

Abstract

This document is a short summary of what we have done for the Natural Language Understanding (NLU) project during 2013 Summer.

1 Introduction

Let a computer understand what people say is a long term goal. However, no general solution has been proposed for all purposes. In this project, the domain is constrained within spoken data when speakers are driving. In term of natural language understanding, we would like to identify what specific topic the speaker is talking about and the specific slots that related to the topic. The topics and slots will be used for dialog manager.

2 Task

2.1 Topic/Frame Prediction

Given the sentence, we want to first identify what a user is talking about. This is the task of topic/frame prediction. For example, when speaking "find my friend peter"[1], it is about "friendlocation"; "is it raining in palo alto" is about "checkweather". Usually, a topic determines what should be responded to the speaker. For example, "friendsearch" should be responded "who and/or where are my friends"; "localsearch" should be given "where is the point of interest".

2.2 Slot Identification

In addition to identify the topic, we should understand what specific information a speaker wants and what constraints given by the speaker. Take "where is the best taco bell in palo alto" for example, the speaker wants to find a restaurant named "taco bell" in the city "palo alto" and the restaurant should be ranked "best" in term of the reviews. Here, "restaurant name", "city name", "rank order" are the slots that we want to identify and "taco bell", "palo alto", "best" are the corresponding values.

3 Related Work

Eun at al.(2005) showed that combining several different classifiers promoted the performance of natural language understanding.

Both generative and discriminative models work pretty well for spoken language understanding (Raymond and Riccardi, 2007).

However, majority of people work on human-transcribed text but rather than directly on speech recognition results. However, without ignoring the speech recognition errors, the performance is expected to decrease a lot.

4 Corpus

4.1 TextData

This data is collected by Amazon Mechanical Turk. The turkers task is to response what they want to say when given a topic. Totally, there are 3364 distinct ways [2] for 11 topics and their frequencies follow power law. The number of ways for each topic is shown in Table 1 and the topics are grouped into four domains. These topics are chosen because we believe these are the most popular scenarios when people driving.

To simplify the problem, we present some dummy words to the turkers. The complete dummy words are shown in Table 2. When people refer to such things, they should always pick up the words from the given dummy words. For example, a city name can only be "Palo Alto". It means that if a speaker would like to refer to a city, she/he shall use "Palo Alto" only. However,

[1] all letters are lowercase since we assume speech recognition engine is not case-sensitive

[2] A way to say something about a topic is a sentence.

even given such constraints, turkers might not follow the rules. For instance, two of the turkers use other cities like “San Francisco”.

The challenge for natural language understanding under such conditions is the fact that although we limit what to say, we don’t constrain how to say. For example, there are a couple of ways to express the meaning of ”my current location” – “my current location”, “my location”, “nearby”, “me”, “where i am”, “my present location”, “close me”, etc. For another example, “restaurant” can be represented as “restaurant”, “restaurants”, “place to buy food”, “where can i get food”, “where i can get food”, “place to eat”, “food”, “taco bell”, “chevron”, etc.

The topics and slots in this dataset are annotated by human.

Domain	Topic	#
Social	friendsearch	268
	friendactivitysearch	207
	friendlocation	102
	friendactivity	82
	updatesocialstatus	434
POI	localsearch	793
	propertyquery	244
Route	planroute	800
	addmidpoint	181
	removemidpoint	98
Weather	checkweather	155
	Total	3364

Table 1: Number of Topics and Domains in TextData

content	meet you there
socialname	facebook
	twitter
cityname	palo alto
distance	5 miles
poiname	taco bell
	chevron
friendname	peter
poitype	restaurant
	gas station
streetname	el camino real
	1234 main street
cuisinetype	chinese
	mexican

Table 2: Dummy words used in TextData

4.2 Fake Data

To some extent, the TextData is far from practical. Therefore, we extracted some grammar patterns and generated lots of fake data. In total, 371312 sentences are generated. We believed these fake data are more representative than TextData. However, the distribution of topics among the fake data is very different from TextData. At the same time, the differences between each other within the Fake Data are very small. The performance by choosing only 1000 sentences as the train set and testing on others can achieve very good results. Thus, fake data is not treated as very useful.

4.3 Spoken Data

To approximate the real environment, 1870 of sentences from TextData are transcribed to speech by a native speaker.

Then, two speech recognition engines are used to recognize the speech to text: Google and Vocon. Both the two engines give N-Best outputs with confidence scores.

5 Approach

5.1 Topic/Frame Prediction

The topic prediction is one type of text classifications. For this task, sentences are classified into topics and one sentence can only belong to one topic.

5.2 Slot Identification

The slot identification is formed as a sequence labeling problem. For example, “where is the best taco bell in palo alto” will be annotated as “where/O is/O the/O best/B-psrh taco/B-ppn bell/I-ppn in/O palo/B-lcn alto/I-lcn”. The BIO tags are used here, where ‘B’ indicates the beginning of a slot; ‘I’ means the inside of a slot; ‘O’ means the ending of a slot. The slot name “psrh” is short of “property sorting rating high”, “ppn” is short of “poiname” and “lcn” is short for “locationconstraint cityname”. The complete slot names together with their short names are shown in Table 7.

6 Experiments

In the experiments, we randomly sampled 700 sentences from the spoken data to be the test set and 2596 sentences from the TextData to be train set. All the test samples will not be included in the train set.

For the training, the correct transcripts are used; for the testing, we have three different setting: reference text (human transcripts), Google speech recognition outputs and Vocon speech recognition outputs.

For the topic recognition, only unigram is used. We also tried bigram, trigram, but they are not better than the unigram baseline.

For the slot identification, we tried several features produced by Senna (Collobert et al., 2011): part-of-speech (POS), chunking (CHK), name entity recognition (NER) and semantic role labeling (SRL)

6.1 Topic Prediction

Top 1

In this experiment, only the top ranked speech recognition result are used. The performance is shown in Table 3. The classifier is SVM, implemented in Weka (Hall et al., 2009) with default parameters.

Three different feature sets are used here: "Uni", "Uni+Lem" and "Uni+Lem+hasNeg". Respectively, they mean unigram, unigram + lemma and unigram+lemma+hasNegative, where "lemma" means to use the lemmatized form of words instead of unigram; "hasNegative" means to include a binary feature that whether the sentence has words such as "not, 't, avoid, skip".

Feature	Reference	VoCon	Google
Uni	0.973	0.656	0.935
Uni+Lem	0.982	0.674	0.948
Uni+Lem+hasNeg	0.976	0.671	0.94

Table 3: Topic prediction accuracy with top-ranked speech recognition result

N-Best

There are three voting methods to combine the N-Best Results.

Upper Bound: If the correct one appears in any of the N-Best, it is a correct prediction.

Majority Voting: Use the majority as the prediction. Choose a random one if there are more than one of them.

Weighted Voting: "Majority Voting" does not consider the speech recognition ranking. The top SR should get more weight. In this approach, each of the SR gets the weight $(i+1)/i$, i is the SR rank. In this way, the top rank gets more weights.

6.2 Slot Identification

In addition to the three voting methods above, we introduced a new method "First Non-Empty".

First None-Empty: Pick up the first non-empty slot as the final prediction. The results are shown in Table 5.

7 Discussion

7.1 Differences between Vocon and Google

The Vocon SR relies on human-written grammar, and the sentences given by Vocon follow the grammar. The weakness is that users have to provide a grammar, which might not have a good coverage. Besides, a word list for proper nouns has to be provided to Vocon, too. In advance, the larger the word list, the worse of the Vocon SR performs.

Different from Vocon, the Google SR is for general purpose. It is good for words that do not appear in provided proper-noun list.

Although Vocon outputs good-grammar sentences, it might not make sense for humans. Google SR outputs bad-grammar sentences, but the semantic sense is much better than Vocon. Thus, the topic and slot predicted results for Google are much better than Vocon.

7.2 Class-Based Model

We tried to replace certain proper noun slot values to their representative words, shown in Table 6.

We used representative words rather than the slot names because they do not affect POS and these words can be recognized by other tools without re-training.

However, the performance is only slightly better than the previous model. One of the reasons might be that for the TextData, there are not many choices among the slots. For example, replacing all "palo alto" to "Chicago" does not gain much. Another reason might be for these slots, the sentences themselves are not very complex and most of the errors happen when sentences have no such slots.

8 Implementation

The sequence labeling model for slot identification is done by CRF++ (2013). In addition, you can also use Yamcha (2005) for this task.

For the feature extraction, we use Senna (Collobert et al., 2011), including POS, CHK, NER and SRL.

	Upper Bound			Majority Voting			Weighted Voting		
TopK	vocon	google	combined	vocon	google	combined	vocon	google	combined
1	0.677	0.930	0.677	0.677	0.930	0.677	0.677	0.930	0.677
2	0.700	0.940	0.760	0.680	0.924	0.709	0.677	0.930	0.677
3	0.704	0.949	0.870		0.924	0.754	0.676	0.926	0.741
4		0.954	0.963		0.927	0.811		0.930	0.763
5			0.971			0.861		0.926	0.839
6			0.973			0.903			0.864
7			0.974			0.921			0.866
8						0.920			0.894
9						0.920			0.894

Table 4: Topic prediction accuracy with N-Best speech recognition result

	Upper Bound		Majority Voting		Weighted Voting		First Non-Empty	
TopK	vocon	google	vocon	google	vocon	google	vocon	google
1	0.486	0.810	0.486	0.810	0.486	0.810	0.486	0.810
2	0.500	0.859	0.470	0.694	0.486	0.810	0.494	0.830
3	0.503	0.874	0.459	0.693	0.469	0.724	0.496	0.833
4		0.893		0.670		0.729		0.836
5		0.897		0.671		0.693		0.837

Table 5: Slot prediction accuracy with N-Best speech recognition result

slot name	representive words
cityname	Chicago
cuisinetype	Chinese
friendname	Jacob
poiname	Costco
socialname	Facebook
statename	California
streetname	Main Street

Table 6: Representative words for certain slot

The SVM Classifier in Weka (Hall et al., 2009) is used to predict topics. For the N-Gram features in topic prediction, we used StringToWordVector in Weka.

9 Future Work

9.1 Hybrid Model for Topic and Slot Prediction

Slot Helps Topic: An experiment has shown that slots are useful for topic identification on FakeData. In that experiment, we add a binary value for each slot. If the sentence has a slot, then the binary value for that slot feature is '1'. We used the golden-standard slot here and it promotes the performance. Therefore, it is reasonable to infer predicted slot might be useful for topic prediction on TextData.

Topic Helps Slot: Whether topic helps slot identification or not is still questionable. However, since the topic identification accuracy is much higher than slot prediction, we believe the topic can be a positive effect.

9.2 Semantic Re-ranking Model for SR

The current two speech recognition engines rank their results in their own ways. However, it seems that they have not considered the semantics of their results. For example, Google recognized "friends on el camino real" as "friends on el camino rielle" and "friends on el camino real". The first one is ranked higher than the second one. However, we can know that "el camino real" is a street name. Thus, the second one should make more sense than the first one.

Following this idea, we can re-rank the N-Best with the objective function of maximizing the semantic meaning of sentences. Moreover, we are not trying to find the best recognition result which has the least word error rate, but to find a recognition that has a better semantics. Furthermore, we are not trying to find one sentence, but to select possible slot combination among the N-Best which maximizes the semantic meaning.

References

[Collobert et al.2011] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu and P. Kuksa. 2011. *Natural Language Processing (Almost) from Scratch.* Journal of Machine Learning Research (JMLR).

[Eun et al.2005] J. Eun, M. Jeong, G. Geunbae Lee 2005. *A Multiple Classifier-based Concept-Spotting Approach for Robust Spoken Language Understanding.* Eurospeech, Lisbon, Portugal, pp. 3441-3444

[Raymond and Riccardi2007] C. Raymond and G. Riccardi. 2007. *Generative and discriminative algorithms for spoken language understanding.* In Interspeech, pp. 16051608, Antwerp, Belgium, Aug. 2007.

[Hall et al.2009] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann and Ian H.Witten. 2009. *The WEKA data mining software: An update*. SIGKDD Explorations, 11(1).

[CRF++2013] 2013. *CRF++: Yet Another CRF toolkit*. http://crfpp.googlecode.com/svn/trunk/doc/index.html.

[YamCha2005] 2013. *YamCha: Yet Another Multipurpose CHunk Annotator*. http://chasen.org/ taku/software/yamcha/.

Slots and Short names

The slot names and shot names are shown in Table 7.

slot name	short name
cityname	lcn
cuisinetype	pct
current_loc	lcl
destination	ld
distance	pd
poiname	ppn
poitype_gas	ppt
poitype_grocery	ppt
poitype_lodging	ppt
poitype_parking	ppt
poitype_restaurant	ppt
propertytype_address	pt
propertytype_cuisine	pt
propertytype_review	pt
ref_area	lra
ref_line	lrl
ref_poi	lrpo
ref_point	lrpt
ref_vague	lrv
sorting_distance_low	psdl
sorting_price_high	psph
sorting_price_low	pspl
sorting_rating_high	psrh
statename	lstn
streetname	lsn
streetnumber	lsnb
friendname	f
ref_friend	rf
socialname	s
topic	t
propertytype	pt

Table 7: Slots Used in TextData