CRF Models for Spoken Language Understanding

Festo Owiny
festo.owiny@studenti.unitn.it
LUS Final Project

I. Abstract

In this paper we present conditional random fields (*CRF*), a framework for building probabilistic models for sequence labeling. We relax the assumption that the data points are independently and identically distributed (i.i.d.) by moving to a scenario of structured prediction, where the inputs are assumed to have temporal or spacial dependencies hence sequential models, which correspond to a chain structure. The project task is to create Spoken Language Understanding Module with *CRF* model using different features such as; word feature, pos and lemmas. We also evaluate this model in comparison with FST-based SLU from the first project. The project has been developed using CRF++ toolkit.

II. Introduction

The need to segment and label sequences arises in many different problems in several scientific fields. Hidden Markov models (HMMs) are particular instances of directed probabilistic graphical models and is one of the widely used and most understood models for sequence labeling. HMMs are generative models, assigning a joint probability to paired observation and label sequences; the parameters are typically trained to maximize the joint likelihood of training examples. In particular, it is not practical to represent multiple interacting features or long-range dependencies of the observations, since the inference problem for such models is intractable.

This difficulty is one of the main motivations for the discriminative approach (conditional models) as an alternative. The difference between the approaches is that generative approach attempts to model the probability distribution of the data, $P(X, Y)$, whereas discriminative ones only model the conditional probability of the sequence, given the observed data, $P(Y/X)$. The generative model makes strong independence assumptions whereas whereas the discriminative approach we are not tied anymore to some of these assumptions. In particular:

- We may use "overlapping" features, such as n-gram features; prefixes, suffixes combined features.
- We may use features that depend arbitrarily on the entire input sequence.

The purpose of this project is to investigate the differences in performance between the *FST*-based *SLU* from the previous project and *CRF*-based *SLU* using the same data set. A second purpose is to identify the differences of performance on the second *SLU* with different features: in this case with Lemmas compared to tokens and with or without pos tagging.

III. Dataset

The data is NL-SPARQL Data Set containing preprocessed and annotated training and test set from the movie domain, presented in the IOB data format. There are 22 entity names identifiable in our dataset, a total of $7,117$ tokens with $1,091$ phrases. The data is distributed with training and test sets having $3,338$ and $1,084$ sentences respectively; Approximately a ratio of $3:1$. The data represents a bunch of queries in a movie domain. Queries search answers for common questions like name of film, year of production, name of actor, etc; Every word is associated with its pos, lemma and ner-IOB-tag. Therefore, a token as four columns.

Furthermore, other features (prefixes and suffixes) were extracted and added into the datasets.

IV. Conditional Random Fields (CRF)

Conditional Random Fields are discriminative Sequential Classifiers observed as an extension of Maximum Entropy models to structured problems. They are globally normalized models: the probability of a given sentence is given by *Equation.1*. Instead of computing the posterior marginals $P(Y = y/X = x)$ for all possible configurations of the output variables (which are exponentially many), it assumes the model decompose into "parts", and it computes only the posterior marginals for those parts, $P(Y_i = y_i/X = x)$ and $P(Y_i = y_i, Y_{i+1} = y_{i+1}/X = x)$. Instead of updating the features for all possible outputs $y^{'} \in V^N$, we exploit the decomposition into parts above and update only "local features".

$$\arg\max_{y \in V^N} P(Y = y/X = x) = \arg\max_{y \in V^N} w.f(x, y) \tag{1}$$

$$\hat{w} = \arg\max_{w} \sum_{m=1}^{M} \log P_w(Y = y^m/X = x^m) - \frac{\lambda}{2} \parallel w \parallel^2 \tag{2}$$

To avoid overfitting, it is common to regularize with the Euclidean norm function as above.

CRF++ is a simple, customizable, and open source implementation of CRFs for segmenting/labeling sequential data. CRF++ is designed for generic purpose and can be applied to a variety of NLP tasks, such as Named Entity Recognition, Information Extraction and Text Chunking.

V. Implementation

Since CRF++ is designed as a general purpose tool, we specified the feature templates. These templates correspond to

the different features previously discussed such as; pos, word, lemma and n-gram features. Each line in the template file denotes one template. There are two types of templates specified with the first character of templates; Unigram template: first character, *'U'* and Bigram template: first character, *'B'*. In this experiment we run with 18 feature templates categorized under; *unigram*, *bigram* and *trigram* respectively. In each of these categories we organized 6 templates namely; *word*(only word fetures), *pos*(only pos features), *lemma*(only lemma features), *word+pos*(both word and pos used as features), *word+lemma*(both word and lemma used as features) and *word+pos+lemma*(all features).

These features are then used in training and testing. Finally, an evaluation was carried out for each output result of the test set using the *conlleval.pl* file. The output of CRF++ is compatible to CoNLL 2000 shared task. This therefore allows us to use the perl script conlleval.pl to evaluate system outputs. This script gives us a list of F-measures with respect to our feature templates. The script also evaluates the result based on the accuracy measures; Global accuracy, precision and recall.

Finally, I performed analysis with additional features by extracting prefixes(P_1, P_2, P_3, P_4) and suffixes (S_1, S_2, S_3, S_4) from the train and test sets. The subscript indicate the length of prefix/suffix.

VI. RESULTS

Tables I,II and III show accuracy values interms of Global accuracy, precission, Recall and F-measure. Global accuracy refers to the number of positively classified tags in relation to the total tags. The global accuracy is high across all the smoothing terms since it also includes the *O-tag* which is the most common tag across any NER task. This *O-tag* is excluded while calculating precission, recall and f-measure and therefore is ignored during our remarks. As shown, combined bigram features of word and lemma performed best with F-measure of 82.27. In general, table results show highest performance when all the features were incorporated. Bigram feature results in Table II is higher than corresponding unigram/trigram feature results in tables I/III. Unigram features of Table I have the lowest performance since it doesn't consider dependency between terms as bigrams and trigrams. It is also observable that *lemma* features performed best, followed by *word* features and *pos* features. *pos* features result to very low F-measure across all the templates.

Finally, in the additional tasks carried out using prefixes and suffixes, the result is shown in Table V with higher accuracy compared to the previous results.

UNIGRAM	global acc	precision	recall	$F_{\beta=1}$
word	93.59%	74.74%	73.24%	73.98
pos	78.39%	26.96%	27.68%	27.32
lemma	93.58%	75.21%	74.52%	74.86
word+pos	93.65%	75.98%	74.24%	75.10
word+lemma	94.10%	76.51%	76.44%	76.48
word+pos+lemma	94.32%	77.40%	77.54%	77.47

Table I
ACCURACY RESULTS ON TEST SET W.R.T UNIGRAM FEATURE TEMPLATES.

BIGRAM	global acc	precision	recall	$F_{\beta=1}$
word	93.59%	88.19%	74.61%	80.83
pos	83.20%	52.40%	47.11%	49.61
lemma	93.80%	87.34%	75.89%	81.22
word+pos	94.31%	85.80%	77.54%	81.46
word+lemma	94.37%	86.46%	78.46%	82.27
word+pos+lemma	94.37%	86.52%	78.28%	82.19

Table II
ACCURACY RESULTS ON TEST SET W.R.T BIGRAM FEATURE TEMPLATES.

TRIGRAM	global acc	precision	recall	$F_{\beta=1}$
word	93.48%	87.64%	74.06%	80.28
pos	84.01%	53.25%	46.56%	49.68
lemma	93.87%	87.29%	75.53%	80.98
word+pos	93.68%	85.67%	75.62%	80.33
word+lemma	93.86%	87.13%	76.35%	81.39
word+pos+lemma	94.21%	85.90%	77.64%	81.56

Table III
ACCURACY RESULTS ON TEST SET W.R.T TRIGRAM FEATURE TEMPLATES.

CRF-based	global acc	precision	recall	$F_{\beta=1}$
Unigram	94.32%	77.40%	77.54%	77.47
Bigram	94.37%	86.52%	78.28%	82.19
Trigram	94.21%	85.90%	77.64%	81.56
Average	94.30%	83.27%	77.82%	80.41

FST-based	global acc	precision	recall	$F_{\beta=1}$
Unigram	88.84%	55.51%	60.04%	57.68
Bigram	92.69%	78.51%	74.34%	76.37
Trigram	92.65%	76.67%	74.70%	75.67
Average	91.39%	70.23%	69.69%	69.90

Table IV
BEST EVALUATION RESULTS FROM THE TWO MODELS W.R.T. N-GRAM FEATURES.

ALL-FEAT	global acc	precision	recall	$F_{\beta=1}$
Bigram	94.67%	87.87%	79.65%	83.56

Table V
RESULTS FOR ALL FEATURES INCLUDING PREFIXES, SUFFIXES.

VII. FST-BASED VS. CRF

Table IV shows model performances across unigram, bigram and trigram with their average values . The best results for FST-based models of assignment 1 were those of "absolute smoothing" method. We select these values and compare with that of discriminative CRF approach. While the baseline model outperforms our models, CRF model performs better than FST-based appraoch of assignment 1. CRF evaluation result is better w.r.t all the accuracy measures (F-measure, precision and Recall). The average values also display a similar trend of resulta. Note that recall values are lower than the corresponding precision, F-measure across both approaches since in supervised learning, LUS systems often suffer from low recall, which is caused by lack of both resource and context. In contrast, CRF models are computationally more expensive than the corresponding FST model since CRF models rely on combination of more features in order to realize better accuracy.

VIII. Conclusion

From the above result we deduce that lemma and word features are very effective in LUS tasks. In general, performance improves as more features are added as depicted in Tables I,II,III. Furthermore, the additional work introduced involving use of prefixes and suffixes as other features registered the best results. Therefore, It is clear that if more features such as *orthogonal* and *gazetteer* are added to the template our accuracy will further improve. Without necessarily exploiting complex features we can realize good accuracy once we incorporate the above suggested or other features. Furthermore, the template files could be better tuned to realize higher accuracy for the CRF++ model. This model relies on feature engineering to add more features which attempts to improve on errors. More features will make training and decoding more expensive. Moreover, if features are very specific, such as the (previous word, current word, next word), they might occur very rarely in the training set, which leads to overfit problems.

References

[1] Truc-Vien T. Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. *Conditional Random Fields: Discriminative Training over Statistical features for Named Entity Recognition*. , 2013.

[2] Yashar Mehdad, Vitalie Scurtu, Evgeny Stepanov. *Italian Named Entity Recognizer Participation in NER task Evalita 09*. , 2016.

[3] Michael Collins, Columbia University. *Language Modeling*. , 2015.