INF5830, H2015 – Semantic Role Labeling (SRL) (2 of 2)
Part 2: Classification

Deadline: November 13th

In this assignment you will be working to solve the task of **argument classification**, an integral subtask within the larger task of SRL. We will assume that predicates and arguments have been identified and will focus on the task of labeling these arguments with semantic roles. We will be working with the original data set from the CoNLL08 shared task on syntactic and semantic parsing for English. The task will be solved as a supervised classification task. In doing so, you will need to process the data to extract relevant features, format these appropriately and experiment with different machine learning algorithms, in order to arrive at your final solution. The assignment is divided into two parts:

1. Feature extraction
2. **Classification and evaluation**

The requirement for the whole assignment (parts 1 *and* 2) is to submit a written report of 3-6 pages which provides details on your experiments and addresses the questions posed in the assignment. Your report should form a coherent text (you do not need to refer to section numbers). In order to document your work, you should also submit a sample of your input data (more on this below). The report and data should be submitted in Devilry before the deadline (November 13th, 23:59).

For this part of the assignment (part 2) you should report on sections 6 and 9, and *choose* between section 7 or 8.

5 scikit-learn

- Obtain scikit-learn (if you are using Anaconda, scikit-learn is included). Work through the general tutorial:
  `http://scikit-learn.org/stable/tutorial/basic/tutorial.html`
- As you know from the tutorial, scikit-learn requires a specific input format for the feature vectors. It does however have support for conversion of feature vectors represented as lists of dictionaries. Have a look at the `DictVectorizer` class:
  `http://scikit-learn.org/stable/modules/feature_extraction.html`

  and use this to convert your feature vectors from part 1.
- Please note that you need to use the same vectorizer for the conversion of training and development/test data. In order to apply the vectorizer to the development/test data, you should simply use `transform` instead of `fit_transform`.

6 Baseline classifier

We will start out by training a baseline system, using a small number of simple features and a SVM classifier. Please train on data taken from the training data set (`train.closed`), develop on the development set (`devel.closed`) and do your final testing on the held-out test set (`test.wsj.closed.GOLD`). This means that you should refrain from testing on the final test set until you have optimized your system wrt features and algorithms at the end of the assignment (section 9).

- Train a classifier using the baseline feature set obtained in the previous assignment on feature extraction. You can in principle follow the setup described in the scikit-learn tutorial, however you should opt for a linear SVM, which is much more efficient with large feature sets: `svm.LinearSVC()`
- The model may take some time to train so you should consider saving it for later usage (e.g. final testing). In order to do so, have a look at the final sections in the tutorial and the information about `joblib`. You can also save the vectorizer in the same way (using `joblib`). In this way you can easily fit the vectorizer to the training data, save it and apply it again to development and test data.
- You will now need to evaluate your classifier. Examine the documentation on classification metrics: `http://scikit-learn.org/stable/modules/model_evaluation.html`
- Summarize your experimental setup (data, machine learning algorithm, settings, etc.) in a short text.
- You should report the following:
  - What is the accuracy of your classifier?
  - Which five semantic roles obtain the highest F-Measures?
  - Which five semantic roles obtain the lowest F-Measures?
- Train another classifier using the extended feature set obtained in the previous assignment on feature extraction. What is the accuracy of the new classifier? Does your feature set improve the classifier?

7 Machine learning algorithm

Choose between the following:

1. Experiment with different machine learning algorithms and evaluate their performance. You should try out at least 2 different machine learning algorithms available in scikit-learn, in addition to the SVM classifier. Make sure that the algorithms you choose are suitable for supervised classification problems (i.e. read the documentation).
   - briefly present the machine learning algorithm
   - evaluate the effect in terms of classifier performance
   - can you say anything about which algorithm is best suited for this task?
2. Tune the parameter settings of your SVM classifier. Have a look at the grid search facility: `http://scikit-learn.org/stable/modules/grid_search.html#grid-search` which allows you to perform a search for the best parameters for your classifier. Please detail the results of your tuning and conclude on the best setting for your classifier.

8 Nominal predicates

Train a separate classifier for nominal predicates, using the set of features obtained in section 5. You are free to add or remove features, but this is not a requirement.

- Briefly describe your classifier, focusing on the modifications made to your verb classification set-up
- Evaluate your classifier:
  - What is the accuracy of the classifier?
  - Which five semantic roles obtain the highest F-Measures?
  - Which five semantic roles obtain the lowest F-Measures?
  - Which differences do you observe compared to the verb classifier?

9 Final testing on held-out data

Choose a final configuration of your system based on the development results and test it on the held-out test data.

- Evaluate your final classifier:
  - What is the accuracy of your classifier?
  - Which five semantic roles obtain the highest F-Measures?
  - Which five semantic roles obtain the lowest F-Measures?