Projekt-CL

Anders Björkelund

anders@ims.uni-stuttgart.de
IMS, Stuttgart

Overview

Schedule

Cue classification and cross-validation

Linear classifiers

Table of Contents

Schedule next few weeks

- Preliminary report Dec. 10th

Schedule next few weeks

- Preliminary report Dec. 10th
  - Results on cue classification
  - Dev set and cross-validated

Schedule next few weeks

- Preliminary report Dec. 10th
  - Results on cue classification
  - Dev set and cross-validated
- Labs
  - Wed Nov. 28th - cross-validation and affix cue classifier
  - Wed Dec. 5th - preperation for report
  - Wed Dec. 12th - reproduce results in report

After Dec. 12th

- Machine-learning to do scope resolution
- Mid-January:
  - cross-validation and preliminary results for scopes
  - compare baseline with machine-learning

Table of Contents

Cue classification

- Both groups should already have cue detection systems:

Cue classification

- Both groups should already have cue detection systems:
- Treat affixal cues and full token cues separately
  - Full tokens - list of forms extracted from training data
  - Affixes - Maximum Entropy classifier trained on training data

Small development set

- Only 33 affixal cues in development set
- Makes it difficult to assess real improvements of the classifier
- Cross-validation on training set

Cross-validation

Split the training set in n parts: $\{p_1, p_2, ..., p_n\}$

for $k \in [1..n]$
$train_k = \bigcup_{i \neq k} p_i$
$test_k = p_k$
train on $train_k$, test on $test_k$

In practice

- Create the split once
- Pairs of files: $(train_k, test_k), k \in [1..n]$

How to split

- The training set consists of 14 chapters
- Make 7 folds consisting of chapters:

How to split

- The training set consists of 14 chapters
- Make 7 folds consisting of chapters:

k	$train_k$	$test_k$
1	(**!**), 3, 4, ... , 13, 14	1, 2
2		
3		
4		
5		
6		
7		

How to split

- The training set consists of 14 chapters
- Make 7 folds consisting of chapters:

k	$train_k$	$test_k$
1	(**!**), 3, 4, ... , 13, 14	1, 2
2	1, 2, (**!**), 5, 6, ..., 13, 14	3, 4
3		
4		
5		
6		
7		

How to split

- The training set consists of 14 chapters
- Make 7 folds consisting of chapters:

k	$train_k$	$test_k$
1	(**!**), 3, 4, ... , 13, 14	1, 2
2	1, 2, (**!**), 5, 6, ..., 13, 14	3, 4
3	1, 2, 3, 4, (**!**), 7, 8, ..., 13, 14	5, 6
4		
5		
6		
7		

How to split

- The training set consists of 14 chapters
- Make 7 folds consisting of chapters:

k	$train_k$	$test_k$
1	(**!**), 3, 4, ... , 13, 14	1, 2
2	1, 2, (**!**), 5, 6, ..., 13, 14	3, 4
3	1, 2, 3, 4, (**!**), 7, 8, ..., 13, 14	5, 6
4	...	...
5	...	...
6		
7		

How to split

- The training set consists of 14 chapters
- Make 7 folds consisting of chapters:

k	*train$_k$*	*test$_k$*
1	(**!**), 3, 4, ... , 13, 14	1, 2
2	1, 2, (**!**), 5, 6, ..., 13, 14	3, 4
3	1, 2, 3, 4, (**!**), 7, 8, ..., 13, 14	5, 6
4	...	...
5	...	...
6	1, 2, ..., 9, 10, (**!**), 13, 14	11, 12
7		

How to split

- The training set consists of 14 chapters
- Make 7 folds consisting of chapters:

k	$train_k$	$test_k$
1	(**!**), 3, 4, … , 13, 14	1, 2
2	1, 2, (**!**), 5, 6, …, 13, 14	3, 4
3	1, 2, 3, 4, (**!**), 7, 8, …, 13, 14	5, 6
4	…	…
5	…	…
6	1, 2, …, 9, 10, (**!**), 13, 14	11, 12
7	1, 2, …, 11, 12, (**!**)	13, 14

Running the system

- Use a bash for loop:

```
#!/bin/sh

for i in 1 2 3 4 5 6 7; do
   echo $i
done
```

Running the system

- Use a bash for loop:

```
#!/bin/sh

for i in 1 2 3 4 5 6 7; do
   echo $i
done
```

- Demo

Evaluation

With 7 output files, how to evaluate?

1. Compute the arithmetic mean of one specific metric over all 7 runs (macro-average)
   - Can fool you if instances are not uniformly distributed over the training set

Evaluation

With 7 output files, how to evaluate?

1. Compute the arithmetic mean of one specific metric over all 7 runs (macro-average)
   - Can fool you if instances are not uniformly distributed over the training set
2. Concatenate the output files for each run, and then run the evaluation script over this (micro-average)

Reports

- In the report for Dec. 10th, present cue-classification numbers

  1. On the dev set
  2. Cross-validated on the test set

Reports

- In the report for Dec. 10th, present cue-classification numbers
  1. On the dev set
  2. Cross-validated on the test set
- And for both cases,
  1. All cues
  2. Only the affixal cues

Reports

- In the report for Dec. 10th, present cue-classification numbers
  1. On the dev set
  2. Cross-validated on the test set
- And for both cases,
  1. All cues
  2. Only the affixal cues
- On Dec. 12th (lab session), run your programs and show Sina and me the same numbers

Table of Contents

Instances and feature vectors

- An *instance* to be classified is mathematically represented as a feature vector

$$\mathbf{x} \in X = \mathbb{R}^n$$

- n is large! "High-dimensional vector space"

Instances and feature vectors

- An *instance* to be classified is mathematically represented as a feature vector
$$\mathbf{x} \in X = \mathbb{R}^n$$
- n is large! "High-dimensional vector space"
- A binary classification problem looks at classifying instances into two classes
$$Y = \{-1, 1\}$$

Instances and feature vectors

- An *instance* to be classified is mathematically represented as a feature vector
$$\mathbf{x} \in X = \mathbb{R}^n$$
- n is large! "High-dimensional vector space"
- A binary classification problem looks at classifying instances into two classes
$$Y = \{-1, 1\}$$

- When training a classifier, we need a set of instances with their corresponding class
$$(\mathbf{x_i}, y_i) \in X \times Y, \ i \in [1..n]$$

Learning and classification

- Given the training instances

$$(\mathbf{x_i}, y_i) \in X \times Y, \; i \in [1..n]$$

- A *learning algorithm* learns a *weight vector*

$$\mathbf{w} \in \mathbb{R}^n$$

Learning and classification

- Given the training instances

$$(\mathbf{x_i}, y_i) \in X \times Y, \ i \in [1..n]$$

- A *learning algorithm* learns a *weight vector*

$$\mathbf{w} \in \mathbb{R}^n$$

- Classification is done by taking the scalar product and looking at the sign

$$sign(\mathbf{x} \cdot \mathbf{w})$$

- If > 0, then class is 1, else class is -1

Learning algorithms

- Multiple ways of computing the weight vector:
  - Maximum Entropy (MaxEnt)
  - Naive Bayes
  - Perceptron
  - (Linear) Support Vector Machines (SVMs)
- For now we stick with MaxEnt

Cue-classification classifiers

- In the cue-classification setting, the classes are

$$Y = \{Negation, Not\ Negation\}$$

- Feature vectors consist of **binary** “indicator functions”

$$\mathbf{x} = (f_1, f_2, f_3, ..., f_n)$$

- where every f_i is either 0 or 1

Feature functions

- Is the affix a prefix?
  $f_{238} = 1$ if affix is a prefix, 0 otherwise

Feature functions

- Is the affix a prefix?
  $f_{238} = 1$ if affix is a prefix, 0 otherwise
- Is the affix a suffix?
  $f_{239} = 1$ if affix is a suffix, 0 otherwise

Feature functions

- Is the affix a prefix?
  $f_{238} = 1$ if affix is a prefix, 0 otherwise
- Is the affix a suffix?
  $f_{239} = 1$ if affix is a suffix, 0 otherwise
- Is the affix "un-"?
  $f_{987} = 1$ if word starts with "un", 0 otherwise

Feature functions

- Is the affix a prefix?
  $f_{238} = 1$ if affix is a prefix, 0 otherwise
- Is the affix a suffix?
  $f_{239} = 1$ if affix is a suffix, 0 otherwise
- Is the affix "un-"?
  $f_{987} = 1$ if word starts with "un", 0 otherwise
- Is the first 4 characters of the base character n-gram "derl"?
  $f_{1029} = 1$ if base starts with "derl", 0 otherwise

Feature functions

- Is the affix a prefix?
  $f_{238} = 1$ if affix is a prefix, 0 otherwise
- Is the affix a suffix?
  $f_{239} = 1$ if affix is a suffix, 0 otherwise
- Is the affix "un-"?
  $f_{987} = 1$ if word starts with "un", 0 otherwise
- Is the first 4 characters of the base character n-gram "derl"?
  $f_{1029} = 1$ if base starts with "derl", 0 otherwise
- Are the word and its base antonyms according to wordnet?
  $f_{1282} = 1$ if antonyms, 0 otherwise

Feature functions

- Is the affix a prefix?
  $f_{238} = 1$ if affix is a prefix, 0 otherwise
- Is the affix a suffix?
  $f_{239} = 1$ if affix is a suffix, 0 otherwise
- Is the affix "un-"?
  $f_{987} = 1$ if word starts with "un", 0 otherwise
- Is the first 4 characters of the base character n-gram "derl"?
  $f_{1029} = 1$ if base starts with "derl", 0 otherwise
- Are the word and its base antonyms according to wordnet?
  $f_{1282} = 1$ if antonyms, 0 otherwise
- ...

Implementation details

- The packages you use for machine-learning (both Python and Java) abstract away from indexing in the feature vector
- Feature functions correspond to strings:

Feature function	String
f_{238}	`AffixType: Prefix`
f_{239}	`AffixType: Suffix`
f_{987}	`StartsWith: un`
f_{1029}	`BaseCharacter4Gram: derl`
f_{1282}	`AreWordNetAntonyms`

Questions

Questions