Wordspace Exercise

Wordspaces for ML
Giuseppe Castellucci, Danilo Croce, Roberto Basili
Web Mining & Retrieval 2015/2016

Wordspaces

- Wordspaces are meant to acquire representations for lexical items
- They aim at representing the "meaning" of words in compact representations

OBJECTIVE

- Verify the contribution of a representation oriented to Word Spaces in a ML setting
  - can they help the generalization capability of ML algorithms?
- Verify whether they provide useful features when combined in a multiple kernel setting

Exercise

Question classification

- Given a *question* in natural language
- **classify it** with respect to 6 classes
- In the previous exercise you've seen
  - Tree Kernels and BOW
  - kernel combination
- In this exercise you'll
  - adopt a Word Space to get a new representation WS
  - combine WS with other kernels and measure its contribution

How to represent a question with a word space?

- A question q is made of $w_i,...,w_n$ words
- In a word space each w_j is represented through a vector
  - Let us define $\varphi(w_j)$ the function that given a word returns the vector associated to it
- An effective and simple way to represent q is to linearly combine the vectors of the word composing it

$$\vec{q} = \sum_{w_i \in q} \alpha(w_i)\phi(w_i)$$

- $\alpha(.)$ is a function that given a word returns a coefficient for the linear combination

Material

- You will find on the website a package containing
  - **`qc_train.klp`**: kelp training file of the previous exercise containing the original question in the "quest" representation
  - **`qc_test.klp`**: kelp test file of the previous exercise containing the original question in the "quest" representation
  - **`WmIRQuestionClassificationExample.java`**: the main class of the previous KeLP exercise
  - **`wordspace_qc.txt.gz`**: the wordspace you'll adopt composed of 8135 words represented through a 250-dimensional vector

Wordspace file format

- The first line contains
  - the number of represented words
  - the number of the dimensions of each word vector
  - e.g. 8135 250
- word vectors format:
  - word [TAB] 0 [TAB] 0 [TAB] vector_representation
- Example:

```
8135 250
run  0       0       -0.0732422,0.0839844,-0.00744629,0.0397949,…
associaton      0       0       0.246506,-0.032694, …
…
```

What you have to do (1)

1. Load the word space in memory
   1. maintain a data structure where to each word you associate a vector
   2. and it is efficient to retrieve a vector given a word
2. Load the train and test datasets with the KeLP functions
3. For each dataset, For each example e
   1. Retrieve the "quest" representation
      1. `String quest = e.getRepresentation("quest").toString().toLowerCase();`
   2. Tokenize it on the whitespace token
   3. Compute the linear combination of the word vectors in a vector v of type `double[]`
      1. Assume a() is 1 for all words
   4. Add v in a `DenseVector dv` to e
      1. `DenseVector dv = new DenseVector(v);`
      2. `e.addRepresentation("ws", dv);`

What you have to do (2)

- Now you "augmented" each example of the datasets with a new representation "ws" that is the linear combination of the word vectors composing a question
- Modify the main class WmIRQuestionClassificationExample.java
  - Write the proper kernel functions such that:
    - you can run a linear kernel on "WS", i.e. linear(WS)
    - you can run a linear combination of kernels with linear(BOW)+linear(WS)
    - you can run a linear combination of kernels with poly(2, BOW)+linear(WS)+SSTK(grct)
- Measure the differences in accuracy and report it Monday 16 to the teacher

Help with KeLP!

- If you have technical questions, please contact:
- Giuseppe Castellucci: **castellucci@ing.uniroma2.it**
- Danilo Croce: **croce@info.uniroma2.it**