*Universität des Saarlandes*
*WiSe 2016/17*
*Computational Linguistics*

# Classes and methods

## IBM_Model1.py

An 'IBM_Model1' class is implemented:

- **IBM_Model1:**
  *__init__(self,source_sents,target_sents,iterations):* an IBM_Model1 object is instantiated with the source and target sentences. The __EM function is called.

  *__EM(self):* runs the Expectation-Maximization algorithm for the number of iterations specified in self.iterations.

  *most_probable_alignments(self,output_filename):* writes a file with the best alignments, one sentence per line, in the format index(target word)-index(source word). Positions are 0-indexed, the NULL alignment is not specified.

#NOTE: In running the model, we assumed a translation from English to French. Presumably, the gold manual alignment did not assume any direction of translation, so that doesn't make a real difference since we will just compare the alignments.

## gizapp_to_score_alignments_format.py

*convert_gizapp_to_score_alignments_format(gizapp_alignments_file,output_filename):* converts the GIZA++ alignment file in the Hansards alignment gold format to run the score-alignment code.

# How to run the program

You can run the program following these steps:

1) Open IBM_Model1.py in python3.

2) Specify the number of iterations.

3) The best alignments for the Hansards corpus are written to the file 'ibm_model1_alignments'.

*Francesca Capuano*

# SCORES

We used different models to calculate the best alignments of the Hansards corpus, and compared them to a gold alignment of the first 37 sentences. We first used the baseline aligner (trained with 1.000,10.000 and 100.000 sentences). The results are reported under 'Baseline aligner'.
Then, we used 5, 10 and 20 iterations of IBM Model 1 with GIZA++, IBM Model 3 in GIZA++, a combination of the two, and our implementation of IBM Model 1.
The AERs (Alignment Error Rate) were calculated and reported in Table 1.

#NOTE: In GIZA++ Model 1 is trained first, and the result is used to start Model2 training. Then Model2 is transfered to Model3. Model3 viterbi training follows. In IBM Models 1+3 with GIZA++, we make 5, 10 and 20 iterations of Model 1, which ultimately feed the 5, 10 and 20 iterations of Model 3 respectively. Note that, combining the models, the performance improves significantly.

## Baseline Aligner

trained with 1.000 sentences
AER = 0.680563

trained with 10.000 sentences
AER = 0.681997

trained with 100.000 sentences
AER = 0.681684

| | **IBM Model 1 with GIZA++** | **IBM Model 3 with GIZA++** | **IBM Models 1+3 with GIZA++** | **IBM Model 1 with my implementation** |
|---|---|---|---|---|
| 5 ITERATIONS | 0.411765 | 0.392272 | 0.328720 | 0.373574 |
| 10 ITERATIONS | 0.369524 | 0.361176 | 0.313793 | 0.361987 |
| 20 ITERATIONS | 0.358095 | 0.337237 | 0.264472 | 0.352153 |

*Table 1*. AER with 5, 10 and 20 iterations of IBM Model 1, IBM Model 3, their combination and IBM Model 1 with my implementation.

## Results

The baseline system reports a very high error rate, which doesn't seem to lower when we train it on a larger number of sentences.
The performances of all IBM models improve with the increasing number of iterations. Model 1 performs worse than Model 3, which performs worse than the

combination of the two.

In the folder 'alignment visualizations', you can find the alignment visualization for the first sentence of the Hansards corpus of the gold alignment (the blue squares indicate the certain alignments, the ones in yellow are the probable alignments).
You can also find the alignment visualization with my implementation of IBM Model 1 with 20 iterations (in blue), in comparison with the Baseline aligner trained on 100.000 sentences, and with the 20 iterations versions of GIZA++ Model 1, GIZA++ Model 3, and GIZA++ Model 1+3 (in yellow). The green squares indicate overlap in the outputs of the models.
The visualizations were obtained with the online word alignment visualization tool from the University of Southern California.