A: Theory

The algorithm used on this assignment was the implementation of the baseline merged with the ability of finding null word alignments and the posterior probability, accounting both $p(f|e)$ and $p(e|f)$.

B: Algorithm

The algorithm for the word alignment is given below:

Algorithm 1 Word Alignment

Input: French/German sentences f and English sentences e
Output: Aligned words

Get count of English words v_e and French/German words v_f
Initialize dictionaries t_1, t_2, $count_e$, $count_f$ $count_{fe}$ and $count_{ef}$
repeat
  Make NULL available
  for each pair of sentences (f, e) **do**
    for each f_i in f **do**
      Initialize $c = 0$ and $z = 0$
      for each e_j in e **do**
        normalize $z = 1.0/v_f$ **if** first iteration **else** $z+ = t_1[(f_i, e_j)]$
      for each e_j in e **do**
        normalize $c = (1.0/v_f)/z$ **if** first iteration **else** $c = t_1[(f_i, e_j)]/z$
        increment both $count_e[e_j]$ and $count_{fe}[(f_i, e_j)]$ with the value of c
    for each e_i in e **do**
      Initialize $c = 0$ and $z = 0$
      for each f_j in f **do**
        normalize $z = 1.0/v_e$ **if** first iteration **else** $z+ = t_2[(e_i, f_j)]$
      for each f_j in f **do**
        normalize $c = (1.0/v_e)/z$ **if** first iteration **else** $c = t_2[(e_i, f_j)]/z$
        increment both $count_f[f_j]$ and $count_{ef}[(e_i, f_j)]$ with the value of c
  for each pair (e, f) in $count_{fe}$ **do**
    $t_1[(f, e)] = (count_{fe}[(f, e)] + 0.1)/(count_e[e] + 0.1 * v_f)$
  for each pair (e, f) in $count_{ef}$ **do**
    $t_2[(e, f)] = (count_{ef}[(e, f)] + 0.1)/(count_f[f] + 0.1 * v_e)$
until # of iterations not 5
Print the best word alignments according to the instructions of the baseline and ignore the alignments defined as *NULL*