DESIGN DOCUMENT

Mustansir Mama	2016B4A70533P
Anubhav Bansal	2016B4A70824P
Manan Agarwal	2016B5A70607P
Akshika Goel	2016B5A70672P
Aayush Agarwal	2016B5A70716P

Documentation

Pre-Processing of data:

All punctuation, extra spaces, and numbers were removed from the dataset, including all sequences containing one or more digits. This was achieved by regex matching. Punctuation is not required during translation and all numbers can be directly replicated, hence neither need to be included while training.

Flow of Program:

Training:

- vocab(sentence_pairs):Takes sentence pairs as input and extract the Dutch and English vocabularies.
- vocab_size(english_words,foreign_words):Computes vocabulary size of English and Dutch words for assigning initial translation probability.
- init_prob:initialize probabilities of all possible word pair translations in corresponding sentences.
- Packages pandas, numpy, matplotlib.pyplot imported.
- Training data imported.
- Read and stored pairs of sentences from file into memory. Our model was trained twice, over 50,000 lines and 100,000 lines.
- Main method to call vocab, vocab_size, and init_prob methods. All English-Dutch pairs were initialised to the same value.
- An optimization of IBM Model and EM Algorithm was run to compute translation hash probabilities until convergence. Details of algorithm and its proof of correctness are given further below.
- For Dutch to English translation, each Dutch word was mapped to its most likely English translation as predicted by our computed t hash values.
- The final Dutch to English mapping was saved for testing.
- For English to Dutch translation, the entire process remains identical, but the initial t hash is computed from English to Dutch instead, and we finally obtain a map from English to Dutch.

Testing:

- preprocesses(input file): cleans input by removing punctuation, sequences containing one or more numbers, and extra spaces.
- cos(s1, s2): computes cosine similarity between s1 and s2, where s1 and s2 are list of words.

- jaccard_correlation(s1, s2): computes Jaccard correlation between s1 and s2, where s1 and s2 are list of words.
- The main method receives document to be translated as input (Dutch-to-English or English-to-Dutch must be specified) and loads the corresponding weights file.
- The input data is cleaned using the pre-processes routine, and the map obtained post training is used to translate the document word-by-word.
- The correct translation file, also provided as input, is cleaned using preprocesses routine.
- Both the correct translation file and translated document are converted to word-count vectors.
- The word-count vectors are passed to jaccard_correlation and cos routines to obtain the Jaccard correlation and cosine similarities scores respectively.

Algorithm:

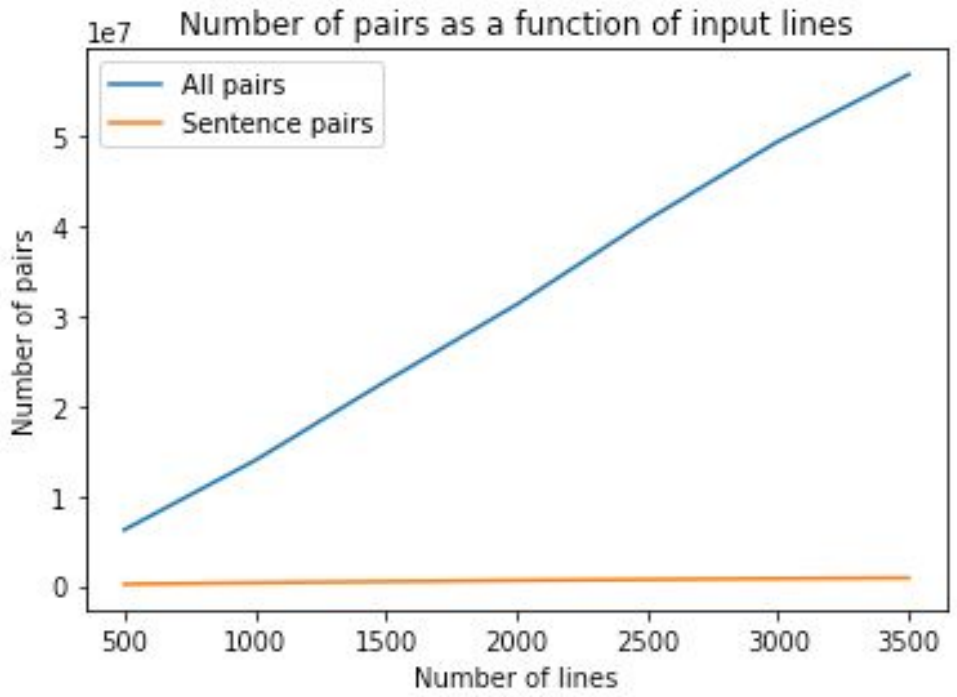
- An optimized modification of IBM Model 1 using the EM Algorithm was implemented. While the logic provided by the pseudo-code in the slides was followed, instead of initializing and iterating over all possible English-Dutch pairs, probability values of only those words which occurred together in a sentence pair were calculated. This greatly reduced both space and time complexity and allowed us to train over a larger dataset.
  The algorithm was run until convergence for
  both 50,000 and 100,000 pairs of sentences.
- The probability values of a Dutch word mapping to an English word were obtained. We then find the Dutch-English t hash value with the highest probability value, and assume that the English word in the pair is the translation of the corresponding Dutch word. This allows us to create a map from each Dutch word to an English word.
- To compute our translations accuracy, we computed the cosine similarity and Jaccard correlation between the word-vectors of the translated sentence of our model and the correct translation as provided in the corpus.

Innovation:

- Our algorithm maps English-Dutch words which only occur together in a sentence pair instead of naively mapping all English-Dutch words in the training set, and this greatly reduces the complexity.
- Complexity of naive IBM Model 1 was $O(n * m)$ where *n* is the total number of English words and *m* is the total number of Dutch words in out training set. Our model reduces the complexity to $O(\sum_{i \in N} n_i * m_i)$ where *N* is the set of all sentences

in our training set and n_i is the number of English words in i^{th} English sentence and m_i is the number of Dutch words in the i^{th} Dutch sentence.
A comparison of the two models is shown below:

Size of the training set (in no. of lines)	**Naive IBM Model 1 (in no. of English-Dutch pairs)**	**Optimized IBM Model 1 (in no. of English-Dutch pairs)**
500	6241158	199977
1000	13954090	356590
1500	22759850	489306
2000	31277439	611823
2500	40698246	720710
3000	49421878	814990
3500	56867346	903768



Correctness of Optimized IBM Model 1:
Despite greatly reducing the number of pairs while training the model, our model's accuracy is unaffected. This is illustrated by the following simple example.

Running the two models of English sentences "The book" and "The house" and their corresponding German translations "Das haus" and "Das buch", the English-Dutch word pairs formed were:

Using Naive IBM Model 1

Pair of English-German words	**Translation probability after convergence**
('the','das')	0.99995
('book','das')	0.00003
('house','das')	0.00003
('the','Buch')	0.03387
('book','Buch')	0.96613
('the','Haus')	0.03387
('house','Haus')	0.96613
('book','Haus')	0
('house','Buch')	0

Our model eliminates only those pairs which have a translation probability of 0, while all other values remain unchanged, as can be seen below:

Using Optimized IBM Model 1

Pair of English-German words	**Translation probability after convergence**
('the','das')	0.99995
('book','das')	0.00003

('house','das')	0.00003
('the','Buch')	0.03387
('book','Buch')	0.96613
('the','Haus')	0.03387
('house','Haus')	0.96613

Assumptions:

- Our model assumes there exists a direct translation for every Dutch word to an English word and vice versa.
- All sequences containing one or more digits were assumed to be mapped to themselves, without requiring translation.
- The Jaccard and Cosine was computed by converting entire document into one word-count vector. We are assuming that every word's translation in the i^{th} English sentence is in the i^{th} Dutch sentence, if it exists, and vice-versa.

Limitations:

- Our model can not predict the correct alignment of the translated sentence.
- The fertility problem which requires adding words which don't have a direct translation is not handled by our model.
- The model has not been trained over the entire corpus, but 50,000 and 100,000 lines instead.