Programming task: Language recognition using distributed high dimensional representations.

The goal with this task is to be able to classify the language of an input text. The language recognition will be done for 21 European languages. The list of languages is as follows: Bulgarian, Czech, Danish, German, Greek, English, Estonian, Finnish, French, Hungarian, Italian, Latvian, Lithuanian, Dutch, Polish, Portuguese, Romanian, Slovak, Slovene, Spanish, Swedish. The training data is based on the Wortschatz Corpora:

- https://corpora.uni-leipzig.de/en?corpusId=deu_newscrawl-public_2018

The test data is based on the Euro Parliament Parallel Corpus:

- http://www.statmt.org/europarl/

T1. Encoding of using d-dimensional {+1,-1} representation

Use encoding procedure for n-grams in {+1;-1} coordinates as described in https://www.dropbox.com/s/qr6g8e4g5e81zke/Kleyko2019_Chapter_DistributedRepresentationOfN-g.pdf?dl=0.

- Use n=3 (tri-grams). Use length of HD vectors d=1000.

An HD vector for a particular input text of certain language is computed by adding all the n-gram vectors. Since we consider 21 European languages at the end of the training phase we will have 21 d-dimensional language HD vectors stored in an array.

In the test phase for an unknown text sample first a query vector in the same fashion as you constructed language vectors in the training phase. To determine the language of this text sample compare its query vector to all the (21) language vectors w.r.t **cosine similarity metric**. Present confusion matrix, compute accuracy and F1-score.

T2. Encoding using complex numbers.

The previous task was a kind of warm up and could be used as a reference result in this task. This task is important since complex representations will be used in the project.

In this task we will use complex vectors as generalization of encoding in {+1;-1} system. Random vectors will have coordinates that are elements of the unit circle group U(1) of complex numbers whose modulus is 1.

The following operations must be used (if needed read https://eprints.qut.edu.au/54258/15/typed-vectors.pdf for more detailed description and corresponding references)

To measure similarity:

- In polar mode, normalized sum of differences between each pair of corresponding phase angles.
- In Cartesian mode, the cosine similarity of the corresponding real vectors: in other words, the real part of the standard Hermitian scalar product.

To superimpose n-grams into a text vector:

- In polar mode, the weighted average of the corresponding phase angles.
- In Cartesian mode, standard complex vector addition.

To permute vectors (for n-gram construction):

- In polar mode, circular convolution of vector with itself n-times. The key observation here is that, because the representation is already in a phase angle form, circular convolution is simply the addition of phase angles.
- In Cartesian form, permutation of coordinates (circular shift) is used instead.