Assignment 2
Neural Networks 501582-3

Pattern Clustering with Kohonen Self-Organizing Maps (SOM)

Assessment information:

Weighting	15%
Deadline	1st June 2023
Submission Mode	Electronic via blackboard
Learning outcome assessed	(1) Demonstrate network abilities to cluster patterns. (2) Evaluate practical considerations in applying neural networks in different applications.
Purpose of assessment	This assignment assesses the understanding of the SOMs and competitive learning by implementing SOM network for word clustering.
Marking Criteria	Group work (max 4 students)

1. Objectives

This assignment requires you to implement SOMs with competitive learning using the Python programming language.

Note: NO credit will be given for implementing any other type of clustering algorithms. More importantly, no credit will be given also for using existing libraries instead of implementing it by yourself. However, you are allowed to use numpy, scipy or any other libraries that you might need to implement/evaluate the algorithm or to visualise the results (e.g., matplotlib). You must provide a README file describing how to run you code to re-produce the results.

2. Word Clustering using SOMs

Text clustering is one of the most important text mining research directions. Despite the loss of some details, clustering technology simplifies the structure of data set, so that people can observe the data from a macro point of view. After clustering process, the text data set can be divided into different clusters, making the distance between the individuals in the same cluster as small as possible, while the distance between the different categories as far away from each other as possible.

In the assignment, you are required to cluster words belonging to four categories: *animals*, *countries*, *fruits* and *veggies*. The words are arranged into four different files. You can use pre-trained GloVe[1] (Global Vectors) for word embeddings (i.e., representations). The first entry in each line is a word followed by 300 features (word embedding/representation) representing the meaning of that word. You can try to use 50 dimensional representations for the words by downloading the corresponding embedding file from the GloVe website.

Preamble

- Install *python 3*.
- Install linear algebra libraries numpy and scipy

[1] https://nlp.stanford.edu/projects/glove/

- For figures and visualisations, you can install *matplotlib* [2].
- You might need to install scikit-learn to perform some evaluations for the clusters[3].

Questions/Tasks

(1) Write a program to load the data instances to memory from the provided data file.
(2) Implement the SOM algorithm to cluster the training instances (i.e., words). Train SOM with *k* neurons (i.e., clusters).
(3) Vary the value of *k* (neurons) from 2 to 10 in the SOM and compute the precision, recall, and F-score for each set of clusters. Plot a figure that shows *k* in the horizontal axis and precision, recall and F-score in the vertical axis in the same plot.

3. Submission Instructions:

NO credit will be given for any copy/paste between groups. Please use your own words to prepare the assignment report and use proper citations with a list of reference at the end of the report.

Submit:

a) The python source code of your program (**do not provide ipython/ jupyter/colab notebooks, instead submit standalone code in a single python .py file**)
b) A README file describing how to compile/run your code to produce the results
c) a file providing the answers/discussions to the assignment's questions.

Evaluating clustering:

Many metrics can be used to evaluate the clustering result. **The Scikit-learn library has many built-in methods for clustering evaluation which you can use in this assignment.** Some of these metrics are explained below.

a) As you have the gold-labels for the words in the dataset (i.e, country, fruit, etc), you can label each cluster using the dominant label. Then you can measure the precision as the number of instances belongs to the cluster name to the total number of instances in the cluster. You perform this measure for all the clusters and compute the macro-average precision. You need to repeat the same process for the recall is the ratio of the number of relevant instance in a cluster to the total number of relevant instances in the dataset. Compute the recall for each cluster and then measure the macro-average recall. F-score is a harmonic mean of precision and recall.
b) Random Index (RI) and precision/recall/F-score from the contingency table is another way to evaluate the clusters as follows:

- Build a contingency table considering pairs of items in each cluster
  - Positive = same cluster
  - Negative = different clusters
  - True = same class
  - False = different classes
- TP = No. of item pairs that are in the same cluster and belong to the same class
- FP = No. of item pairs that are in the same cluster but belong to different classes
- TN = No. of item pairs that are in different clusters and belong to different classes
- FN = No. of item pairs that are in different clusters but belong to the same class

$$RI = \frac{TP + TN}{TP + FP + TN + FN}$$

(accuracy of the clustering)

contingency table	same cluster	different clusters
same class	TP	FN
different classes	FP	TN

$P = TP / (TP + FP)$

$R = TP / (TP + FN)$

$F = 2PR / (P + R)$

[2] https://matplotlib.org/stable/users/installing.html
[3] https://scikit-learn.org/stable/install.html

Useful link for SOM in Python (among many others):
https://www.youtube.com/watch?v=M272NC1PizE