INT304: Pattern Recognition In Computer Vision **S2** 2021-2022

Assignment/Lab 1: Feature Selection

Lecturer: Xi Yang *Unit: INT Dept. of Intelligent Science*

Disclaimer:

1. *Lab reports deadlines are strict. University late submission policy will be applied.*
2. *Collusion and plagiarism are absolutely forbidden (University policy will be applied).*
3. *Report is due 21 days from the date of running this lab.* (March 29th, 2022)

1.1 Objectives

- Master the concepts and knowledge on feature selection.
- Be familiar with the text processing

1.2 Introduction

Text categorization is the task on classifying a set of documents into categories from a set of predefined labels. Texts cannot be directly handled by our model. The indexing procedure is the first step that maps a text d_j into a numeric representation during the training and validation. The standard **tfidf** function is used to represent the text. The unique words from English vocabulary are represented as a dimension of the dataset. The high dimensionality of the word space may be problematic for our classification methods. In this case, we will choose a subset by the feature selection methods to reduce the dimensionality of the feature space.

1.3 Tasks

1.3.1 Data Preprocessing

- (**10 marks**) Download the text dataset and read the documents: http://qwone.com/ jason/20Newsgroups/. The training examples of the version **20news-bydate.tar.gz** are used for our experiments, where the duplicates are removed.
- (**10 marks**) Remove the stopwords (Stopword.txt), which are frequent words that carry no information. Convert all word into their lower case form. Delete all non-alphabet characters from the text. Hint: using python set, regex and dict
- (**10 marks**) Perform word stemming, which means the remove the word suffix.
  - install the library: nltk (python language)
  - usage: see the following code on how to use Porter stemmer (https://www.nltk.org/howto/stem.html).

```
from nltk.stem.porter import *
stemmer = PorterStemmer()
plurals = ['caresses', 'flies', 'dies', 'mules', 'denied',
'died', 'agreed', 'owned', 'humbled', 'sized',
'meeting', 'stating', 'siezing', 'itemization',
'sensational', 'traditional', 'reference', 'colonizer',
'plotted']
singles = [stemmer.stem(plural) for plural in plurals]
```

1.3.2 Indexing

The documents are represented as the vector space model. In the vector space model, each document is represented as a vector of words. A collection of documents are represented by a document-by-word matrix A

$$A = (a_{ik}) \tag{1.1}$$

where a_{ik} is the weight of word k in document i.

1.3.3 Feature Selection

Feature selection try to remove non-informative words from the document in order to improve categorization effectiveness and reduce the computational complexity.

- (**10 marks**) Remove the low-frequency words

  The document frequency for a word is the number of documents in which the words occurs. You should compute the document frequency for each word in the training dataset and removes those words whose document frequency is less than some predefined threshold (the setting $df < 5$).

- (**40 marks**) Choose features with **information gain**

  Information gain measures the number of bits of information by knowing the presence or absence of a word in a document.

  (**10 marks**) Let $c_1, c_2, \cdots, c_K$ denote the set of possible categories. The information gain of a word w is defined to be

$$\begin{aligned} IG(w) &= H(C) - H(C|w) \\ &= -\sum_{j=1}^{K} P(c_j)\log(P(c_j)) + P(w)\sum_{j=1}^{K} P(c_j|w)\log P(c_j|w) + P(\bar{w})\sum_{j=1}^{K} P(c_j|\bar{w})\log P(c_j|\bar{w}) \end{aligned}$$

  - **5 marks** $P(c_j)$: the fraction of documents in the total collection that belongs to class c_j
  - **5 marks** $P(w)$: the fraction of documents in which the word w occurs
  - **10 marks** $P(c_j|w)$: the fraction of documents from class c_j that have at least one occurrence of word w
  - **10 marks** $P(c_j|\bar{w})$: the fraction of documents from class c_j that does not contain the word w

In the end, we choose 1000 word with maximum IG values by sorting all words.

1.3.4 (20 marks) TFIDF Representation

TFIDF representation assigns the weight to word i in document k in proportion to the number of occurrences of the word in the document, and inverse proportion to the number of documents in the collection for which the word occurs at least once.

$$a_{ik} = f_{ik} * \log(N/n_k) \tag{1.2}$$

- f_{ik}: the frequency of word k in document i
- N: the number of documents in the training dataset
- n_k: the total number of times word k occurs in the training dataset called the document frequency.

Taking int account the length of different documents, we normalize the representation of the document as

$$A_{ik} = \frac{a_{ik}}{\sqrt{\sum_{j=1}^{1000} a_{ij}^2}} \tag{1.3}$$

The training set can be represent as a matrix $A_{N\times 1000}$. Once the features are chosen, the test set can be converted into another matrix $B_{M\times 1000}$, where M is the size of the test dataset.

1.4 Lab Report

- Write a short report which should contain a concise description of your results and observations.
- **Please insert the clipped running image into your report for each step.**
- Submit the report and the source code to electronically into LearningMall.
- Python is strongly suggested.
- The report is strongly suggested to be written with the latex typesetting language.
- The report in pdf format and python source code of your implementation should be zipped into a single file. The naming of report is as follows:

  e.g. StudentID_LastName_FirstName_LabNumber.zip (123456789_Einstein_Albert_1.zip)

1.5 Hints

Please refer to the paper for more details: K Aas and L. Eikvil, Text Categorisation: A Survey, 1999.

- Latex IDE: texstudio
- Python IDE: pycharm
- Use the python set, dict and list collections flexibly.