Machine Learning - Assignment 2

Aditya Sujith Gudimetla - Reg No: 207

***Abstract*— During this assignment we attempt to apply different topic modelling techniques to extract important information from the given dataset. The dataset consists of 94 bangla documents that are already POS tagged. The language we have used is python 3 along with multiple modules. The source code is publicly available at https://github.com/CaptainLazarus/banglaNLP.**

I. Preparing the Dataset

The dataset consists of 94 documents that are POS tagged. Each line contains a word, along with its letters and its POS tag at the end. An empty line indicates a new sentence. In the given programs, the following preprocessing was done:-

- We filtered words based on their occurrence and their POS tags. Words that appeared only once and words that appeared in more than 10% of the total corpora were removed.
- POS tags XC, XO, WQ were removed.
- Punctuation in the dataset was already removed.

II. Algorithms used

For topic modelling, we used 2 algorithms in the following assignment. They are

A. Latent Dirichlet Allocation

Algorithm 1 How LDA works

for document d_d in corpus D **do**
  Choose $\theta_d \sim (\alpha)$
  for position w in d_d **do**
    Choose a topic $z_w \sim (\theta_d)$
    Choose a word w_w from $p(w_w|z_w, \beta)$, a multinomial distribution over words conditioned on the topic and the prior β.
  end for
end for

- LDA is a topic model that generates topics based on word frequency from a set of documents.
- Useful for finding accurate mixtures of topics with documents.

B. Latent Semantic Ananlysis

- LSA is a topic model that analyzes relationships between a set of documents and the terms they contain by producing a set of concepts related to the documents and terms.
- LSA assumes that words that are close in meaning will occur in similar pieces of text

Algorithm 2 How LSI works

for document d_d in corpus D **do**
  Create a word document matrix A
  Using SVG decomposition, generate a word topic matrix U, a document matrix for topic V^T and a topic strength matrix S
  Multiply S and V^T
end for

III. Discussion

1) Differences:

- LSI learns latent topics by performing a matrix decomposition (SVD) on the term-document matrix. LDA is a generative probabilistic model, that assumes a Dirichlet prior over the latent topics.
- LDA is usually faster but is less accurate than LSI. This can change based on the parameters.
- LDA can predict topics for unseen documents. Should the documents have a similar structure, topics in the documents can be predicted.

2) Similarities:

- Both LDA and LSI need a lot of fine tuning to correctly predict topics.
- Topics are usually unlabelled, meaning that a human has to label the topics based on the words assigned to it.

IV. Code

A. data.py

Used for processing data from documents

```
import pickle
import os

def getData(fileName , rem):
    data = []
    idx=0
    for file in os.scandir(fileName):
        data.append([])
        with open(file , "r") as f:
            for line in f:
                raw_text = list(line.
                    ↪ split("\t"))
                word = raw_text[0]
                if word == "\n":
                    pass
```

```
                if rem is None:
                    data[idx].append(
                        ↪ word)
                else:
                    pos = raw_text
                        ↪ [-1][:-1]
                    if pos in rem:
                        pass
                    else:
                        data[idx].append
                            ↪ (word)
        idx+=1
    with open(fileName+".txt" , "wb") as
        ↪  f:
        pickle.dump(data , f)
```

B. *main.py*

Main file Used for LDA and LSI

```
import data
import pickle
import os
import argparse
import pyLDAvis.gensim
import pyLDAvis
import gensim
import json

def lda(fileName):
    with open(fileName , 'rb') as f:
        a = pickle.load(f)
        dictionary = gensim.corpora.
            ↪ Dictionary(a)
        dictionary.filter_extremes(
            ↪ no_below = 2 , no_above
            ↪ =0.1)


        corpse = [dictionary.doc2bow(
            ↪ text) for text in a]
        lda = gensim.models.ldamodel.
            ↪ LdaModel(corpse ,
            ↪ num_topics = 8 , id2word=
            ↪ dictionary , passes=10)
        lsi = gensim.models.LsiModel(
            ↪ corpse ,id2word=dictionary
            ↪  , num_topics=8)

    name = list(list(fileName.split('/')
        ↪ )[-1].split("."))[-2]
    # print(name)

    #LDA
    lda_name = "../output/"+name+"_lda.
        ↪ json"
    with open(lda_name , 'w') as f:
        f.write(json.dumps(lda.
            ↪ print_topics(-1), indent
            ↪ =4))

    web = pyLDAvis.gensim.prepare(
        ↪ topic_model=lda , corpus=
        ↪ corpse , dictionary=dictionary
        ↪ )
    htmlName = lda_name+".html"
    pyLDAvis.save_html(web , htmlName)

    #LSI
    lsi_name = "../output/"+name+"_lsi.
        ↪ json"
    with open(lsi_name , 'w') as f:
        f.write(json.dumps(lsi.
            ↪ print_topics(-1), indent
            ↪ =4))

    web = pyLDAvis.gensim.prepare(
        ↪ topic_model=lsi , corpus=
        ↪ corpse , dictionary=dictionary
        ↪ )
    htmlName = lsi_name+".html"
    pyLDAvis.save_html(web , htmlName)


if __name__ == "__main__":
    parser = argparse.ArgumentParser(
        ↪ description="Processing␣data")
    parser.add_argument("-rem" , metavar
        ↪ ="R" , nargs="*" , type=str ,
        ↪ help="List␣of␣POS␣tags␣to␣
        ↪ remove")
    args = parser.parse_args()

    fileName = "../data/94-documents"

    if not os.path.isfile(fileName):
        data.getData(fileName , args.rem
            ↪ )
        # print("\nEntered\n")
    lda(fileName+".txt")
```

V. Pipeline

During data processing,

- Text is to be tokenized. This was done by taking the first word of each line in each document, disregarding the new lines and then adding it to a list. We finally have a list of lists that represents each document and the words in them.
- Stop words are to be removed. Since there was no list of available bengali stop words I tokenized words based on POS tag. (The program allows you to remove certain POS tags from dataset). I also removed words

that appeared in more than 10% of the all the documents or appeared less than 2 times.

- Lemmatizing/Stemming are important steps to be taken, however considering that I don't know bengali and that libraries that exists cannot do those processes for bengali words, they were not done.
- LDA and LSI was implemented using gensim package. We make a dictionary and corpus using inbuilt gensim packages and pass these as parameters to the models. The number of topics decided was 8 though this number can be changed. The data they require is the same, so they were used in one program for simplicity.

VI. Output

A. LDA output


```
[
    [
        0,
        "0.007*\"dAkwArabAbu\" +
            ↪ 0.003*\"XarmaxAsa\" +
            ↪ 0.003*\"Akabara\" +
            ↪ 0.002*\"kaKaKano\" +
            ↪ 0.002*\"bAbAjI\" +
            ↪ 0.002*\"Suwe\" + 0.002*\"
            ↪ bAzXa\" + 0.002*\"AiyZAra
            ↪ \" + 0.002*\"dAkwArabAbura
            ↪ \" + 0.002*\"kqRNa\""
    ],
    [
        1,
        "0.008*\"rAmaxAsa\" + 0.003*\"
            ↪ bAbujI\" + 0.002*\"SaSI\"
            ↪ + 0.002*\"xora\" +
            ↪ 0.002*\"curi\" + 0.002*\"
            ↪ mqNAlaxixi\" + 0.002*\"
            ↪ girISa\" + 0.002*\"josePa
            ↪ \" + 0.002*\"BARA\" +
            ↪ 0.002*\"sejaxi\""
    ],
    [
        2,
        "0.007*\"maXu\" + 0.003*\"lakRmI
            ↪ \" + 0.003*\"xaroyZAna\" +
            ↪  0.002*\"BEraba\" +
            ↪ 0.002*\"BErabera\" +
            ↪ 0.002*\"cAla\" + 0.002*\"
            ↪ bAzdZuyye\" + 0.002*\"
            ↪ rAmabAbu\" + 0.002*\"
            ↪ niSIWa\" + 0.002*\"mAstAra
            ↪ \""
    ],
    [
        3,
        "0.006*\"SaSI\" + 0.005*\"
            ↪ bINApANi\" + 0.003*\"pudZe
            ↪ \" + 0.002*\"suramA\" +
            ↪ 0.002*\"kabi\" + 0.002*\"
            ↪ ceyZAre\" + 0.002*\"
            ↪ rAkRusI\" + 0.001*\"Peliwe
            ↪ \" + 0.001*\"byAgatA\" +
            ↪ 0.001*\"byAga\""
    ],
    [
        4,
        "0.005*\"rAmabAbu\" + 0.004*\"
            ↪ BEraba\" + 0.003*\"
            ↪ rAmaxAsa\" + 0.002*\"
            ↪ jyATAmaSAi\" + 0.002*\"
            ↪ BErabera\" + 0.002*\"
            ↪ dAkwArabAbu\" + 0.002*\"
            ↪ suramA\" + 0.001*\"brAhma
            ↪ \" + 0.001*\"brAhmaxera\"
            ↪ + 0.001*\"Agunera\""
    ],
    [
        5,
        "0.019*\"jyATAimA\" + 0.010*\"
            ↪ biSbeSbarI\" + 0.005*\"
            ↪ mAsi\" + 0.004*\"sejaxi\"
            ↪ + 0.003*\"badZaxA\" +
            ↪ 0.003*\"yawIna\" +
            ↪ 0.003*\"jyATAimAra\" +
            ↪ 0.003*\"GoRAla\" +
            ↪ 0.002*\"sejaxA\" +
            ↪ 0.002*\"ka\u2019ro\""
    ],
    [
        6,
        "0.010*\"SaSI\" + 0.004*\"
            ↪ rAmabAbu\" + 0.003*\"
            ↪ raGubIra\" + 0.003*\"kabi
            ↪ \" + 0.003*\"pisimA\" +
            ↪ 0.003*\"hAH\" + 0.002*\"
            ↪ hIrA\" + 0.002*\"SaSIra\"
            ↪ + 0.002*\"siM\" + 0.001*\"
            ↪ jyATAmaSAi\""
    ],
    [
        7,
        "0.006*\"BEraba\" + 0.006*\"
            ↪ sanAwana\" + 0.004*\"Ji\"
            ↪ + 0.003*\"gopAla\" +
            ↪ 0.002*\"karuNAmayZI\" +
            ↪ 0.002*\"BajuyZA\" +
            ↪ 0.002*\"mACa\" + 0.002*\"
            ↪ binoxa\" + 0.002*\"
            ↪ xiximaNi\" + 0.002*\"
            ↪ cEwanya\""
    ]
]
```

B. LSI output


```
[
    [
        0,
        "-0.650*\"SaSI\" + -0.193*\"kabi
            ↪ \" + -0.162*\"hAH\" +
            ↪ -0.120*\"SaSIra\" +
            ↪ -0.108*\"jyATAimA\" +
            ↪ -0.083*\"sabyasAcIra\" +
            ↪ -0.079*\"rAmaxAsa\" +
            ↪ -0.076*\"hIrA\" +
            ↪ -0.071*\"dAkwArabAbu\" +
            ↪ -0.069*\"SaSi\""
    ],
    [
        1,
        "-0.751*\"jyATAimA\" + -0.372*\"
            ↪ biSbeSbarI\" + 0.184*\"
            ↪ SaSI\" + -0.118*\"
            ↪ jyATAimAra\" + -0.095*\"
            ↪ bAbAjI\" + -0.081*\"
            ↪ badZaxA\" + -0.073*\"xInu
            ↪ \" + -0.070*\"beNIbAbu\" +
            ↪  -0.070*\"cAbi\" +
            ↪ -0.069*\"BAzdZAra\""
    ],
    [
        2,
        "0.433*\"rAmaxAsa\" + -0.305*\"
            ↪ SaSI\" + -0.211*\"jyATAimA
            ↪ \" + 0.196*\"dAkwArabAbu\"
            ↪  + -0.112*\"kabi\" +
            ↪ -0.109*\"biSbeSbarI\" +
            ↪ -0.107*\"hAH\" + 0.102*\"
            ↪ bAbujI\" + 0.086*\"
            ↪ raGubIra\" + 0.082*\"xora
            ↪ \""
    ],
    [
        3,
        "0.420*\"rAmaxAsa\" + -0.270*\"
            ↪ BEraba\" + -0.222*\"
            ↪ raGubIra\" + -0.189*\"mAsi
            ↪ \" + 0.158*\"jyATAimA\" +
            ↪ -0.148*\"rAmabAbu\" +
            ↪ -0.124*\"XarmaxAsa\" +
            ↪ -0.113*\"wAriNI\" +
            ↪ -0.102*\"BErabera\" +
            ↪ 0.099*\"biSbeSbarI\""
    ],
    [
        4,
        "0.317*\"raGubIra\" + -0.276*\"
            ↪ BEraba\" + -0.218*\"mAsi\"
            ↪  + -0.181*\"rAmaxAsa\" +
            ↪ 0.158*\"rAmabAbu\" +
            ↪ -0.136*\"XarmaxAsa\" +
            ↪ -0.132*\"wAriNI\" +
            ↪ -0.117*\"hAH\" + -0.109*\"
            ↪ SaSI\" + 0.107*\"mAijI\""
    ],
    [
        5,
        "-0.289*\"raGubIra\" + 0.143*\"
            ↪ saByawAra\" + -0.139*\"
            ↪ rAmabAbu\" + -0.135*\"hAH
            ↪ \" + 0.126*\"BEraba\" +
            ↪ 0.123*\"saByawA\" +
            ↪ 0.111*\"bAfalAra\" +
            ↪ 0.107*\"SAnwi\" + 0.099*\"
            ↪ iuropera\" + -0.098*\"hIrA
            ↪ \""
    ],
    [
        6,
        "-0.335*\"raGubIra\" + 0.331*\"
            ↪ dAkwArabAbu\" + -0.309*\"
            ↪ rAmaxAsa\" + -0.114*\"
            ↪ bAbujI\" + -0.111*\"mAijI
            ↪ \" + 0.092*\"AiyZAra\" +
            ↪ -0.084*\"BEraba\" +
            ↪ 0.078*\"kqRNa\" + 0.075*\"
            ↪ paWera-xAbI\" + 0.070*\"
            ↪ sejaxi\""
    ],
    [
        7,
        "0.315*\"dAkwArabAbu\" +
            ↪ 0.304*\"raGubIra\" +
            ↪ -0.255*\"rAmabAbu\" +
            ↪ -0.203*\"sejaxi\" +
            ↪ -0.146*\"mqNAlaxixi\" +
            ↪ -0.143*\"harira\" +
            ↪ -0.136*\"jyATAmaSAi\" +
            ↪ -0.117*\"suramA\" +
            ↪ -0.099*\"sejaxA\" +
            ↪ 0.098*\"mAijI\""
    ]
]
```