Gebze Technical University

Department of Computer Engineering

CSE 654 / 484
Fall 2020

AHMET MELİH YANALAK
151044044

Homework 03

REPORT

Part1 : Word2Vec Operation

In order to implement Naive Bayes Sentiment Analyser Program,we need to understand the function of Word2Vec.

In Word2Vec approach,words are represented as n-dimensional vectors which keep floating point values for each dimension.

Fort he Beyazperde Train and Test Sets,the files have been tranformed into vector files which kept as binary in word_vectors.bin.

```
time ./word2vec -train new_test.txt -output word_vectors.bin
-cbow 1 -size 200 -window 5 -negative 15 -hs 0 -sample 1e-4
```

We need to figure out what those parameters are first.Window size is the number that neighbours words are considered while representing center word as vector.For this operation CBOW(Contigious Bag of Words) is used,which is a technique in word2vec.

Size is the dimension number of vectors,we specified the size as 200 in the first place.

For the word “güzel” here are the closest words

```
Enter word or sentence (EXIT to break): guzel

Word: guzel  Position in vocabulary: 231

                                              Word       Cosine distance
----------------------------------------------------------------------------
                                              alip              0.245804
                                                 j              0.241390
                                           buradan              0.213036
                                     saçmalamislar              0.210716
                                           puanlama              0.210336
                                        çalisilmis              0.207009
                                          olacakmis              0.206393
                                              düsen              0.204139
                                             çilgin              0.203853
                                              çikan              0.199197
                                         alkislamak              0.199189
                                          gelmesini              0.197241
                                            reklami              0.195640
                                          aktörler               0.194267
```

Vectors with 200 dimension doesn’t give good results.Hence,vector size increased to 300.

```
Enter word or sentence (EXIT to break): guzel

Word: guzel  Position in vocabulary: 231

                                              Word       Cosine distance
----------------------------------------------------------------------
                                               cok              0.804453
                                             super              0.707123
                                       surukleyici              0.704120
                                         ozellikle              0.685258
                                         gercekten              0.675658
                                             jetli              0.639399
                                              icin              0.601202
                                              kotu              0.583393
                                           guzeldi              0.571943
                                             bugun              0.567221
                                               hic              0.559236
                                             butun              0.556157
                                             dovus              0.540638
                                             sezer              0.535082
                                            gorsel              0.517099
                                        yakisikli              0.516238
                                             gusel              0.504977
```

Fort he 300 dimension,results are similar to word “güzel”

Part2 : Merging Datasets

-In second part of the project,in order to merge the files for training(trwiki,beyazperde train and test) first punctuations are removed from the files.

-3 files are merged together as one.

- Then same operation as we did in part 1 is performed to this merged dataset file.

-Now we have vector representations of all the words in the beyazperde train-test set and the trwiki.

Part3 : Training Model and Testing

In the training part, dataset is read comment by comment.For each comment,average vector of words inside that comment calculated by average_vector function.

```python
def average_vector(model, comment):
    comment = re.sub(r"[,.:;@#?!&$]+\ *", " ", comment)
    words = comment.split()
    sum=[]
    sum = [0 for i in range(300)]
    for word in words:
        for i in range(300):
            sum[i] = average_vector_helper(model,word)

    return sum

def average_vector_helper(model,word):
    sum = 0
    count = 0
    for i in range(300):
            if(word in model.vocab):
                sum += model[word][i]
                count += 1
    if(count != 0):
        sum = sum / count
    return sum
```

Calculates the average of words in the comments

Probability Calculation 1:

In the training part,to get the p(w,c) calculation (We will use it in Naive Bayes calculation later),the distance of each vector from average vector of that comment is taken.Then all distances sum up and divided by 1000 in order the get smaller value.Then this value kept as p(w | c) in dictionary.

For example:

“ Film çok güzeldi ” class = 1

[x1,x2,x3….x300] [y1,y2,y3….y300] [z1,z2,z3….z300]

Film çok güzeldi

Avg vector = [(x1+y1+z1)/3 , (x2+y2+z2)/3 …. (x300+y300+z300)/3]

p(güzeldi | class = 1) = |average vector – güzeldi| / 1000

If same word is seen again,this calculated value is added to previous probabilities

p(güzeldi | class = 1) = p(güzeldi | class = 1) + |average vector – güzeldi| / 1000

Probability Calculation 2:

For the second choice Cosine Similarity has been used while assigning probability values.

Cosine Similarity:

$$\cos\theta = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\|\vec{b}\|}$$

$$\|\vec{a}\| = \sqrt{a_1^2 + a_2^2 + a_3^2 + \cdots + a_n^2}$$

$$\|\vec{b}\| = \sqrt{b_1^2 + b_2^2 + b_3^2 + \cdots + b_n^2}$$

We take cosine similarity between average vector and current word vector.Then this calculated value directly assigned to p(w|c) where w is current vector and c is assigned class.
Note: If there is already a value for that word in dictionary,then the value is added to previous value.

Testing the dataset:

In order to test the dataset,we need to apply the Naive Bayes rule which is shown below:

Likelihood

Class Prior Probability

$$P(c \mid x) = \frac{P(x \mid c)P(c)}{P(x)}$$

Posterior Probability

Predictor Prior Probability

$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Since we calculated the P(w|c) values in training part,we put the probabilities in formula and get the results for both classes.

Then the class with greater value is assigned as predicted.

Testing Results:

Get_accuracy function is used to calculate accuracy:

```
def get_accuracy(predicted_labels,true_labels):
    countMatch = 0
    for i in range(len(predicted_labels)):
        if true_labels[i] == predicted_labels[i]:
            countMatch += 1

    accuracy = (float)(countMatch / len(true_labels))
    return accuracy
```

By using the first probability calculation we have the %85 accuracy value

```
PS C:\Users\Dell\Desktop\HW3>
0.8522130532633159
```

By using the cosine similarity calculation we have the %50 accuracy value

```
PS C:\Users\Dell\Desktop\HW3>
0.5015003750937734
```