CS5560 Knowledge Discovery and Management

Problem Set 4

June 26 (T), 2017

Name: Rashmi
Class ID: 29

I. N-Gram

Consider a mini-corpus of three sentences
<s> I am Sam </s>
<s> Sam I am </s>
<s> I like green eggs and ham </s>

An N-gram is a sequence of N words. N-gram. words: a 2-gram (or bigram) is a two-word sequence of words like “please turn”, “turn your”, or ”your homework”, and a 3-gram (or trigram) is a three-word se- quence of words like “please turn your”, or “turn your homework”.

1) **Compute the probability of sentence “I like green eggs and ham” using the appropriate bigram probabilities.**

For bigram,

P(wi | wi-1) = count (wi-1, wi) / count (wi-1) means that probability that wordi-1 is followed by wordi = [Num times we saw wordi-1 followed by wordi] / [Num times we saw wordi-1]

- s = beginning of sentence
- /s = end of sentence

####Given the following corpus:
s I am Sam /s
s Sam I am /s
s I do not like green eggs and ham /s

P(I/s)=2/3
P(like/I)= 1/3
P(green/like)=1/1

P(eggs/green)= 1/1
P(and/eggs)= 1/1
P(ham/and)= 1/1

2) **Compute the probability of sentence “I like green eggs and ham” using the appropriate trigram probabilities.**

Formula for trigram is :

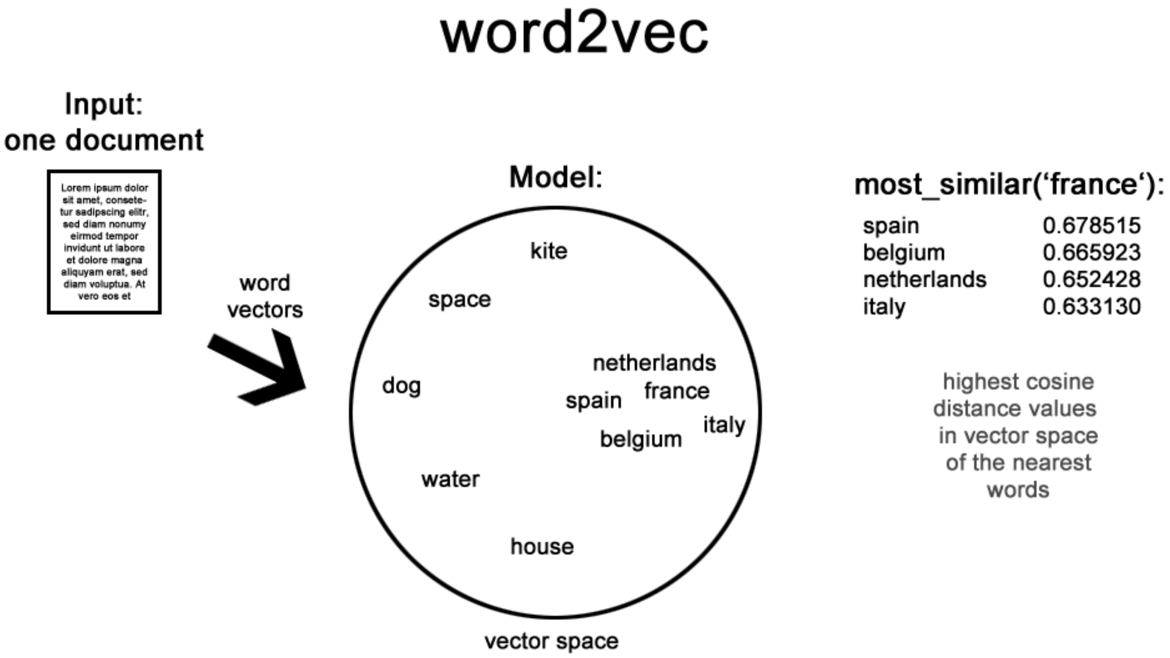
$$\mathbf{P(w_i \mid w_{i-1}\ w_{i-2}) = count (w_i, w_{i-1}, w_{i-2}) / count (w_{i-1}, w_{i-2})}$$

Probability that we saw wordi-1 followed by wordi-2 followed by wordi = [Num times we saw the three words in order] / [Num times we saw wordi-1 followed by wordi-2]
P(green/I like)=count(I like green)/count(I like)= 0/0=0
Similarly
P(eggs/like green)=count(like green eggs)/count(like green)= 1/1=1
P(and/green eggs)=count(green eggs and)/count(green eggs)= 1/1=1
P(ham/eggs and)=count(eggs and ham)/count(eggs and)= 1/1=1

II. Word2Vec

Word2Vec reference: https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/

Consider the following figure showing the Word2Vec model.


word2vec
Input:
one document
Lorem ipsum dolor sit amet, consete-tur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et
word vectors
Model:
kite
space
dog
netherlands
spain
france
italy
belgium
water
house
vector space
most_similar('france'):
spain 0.678515
belgium 0.665923
netherlands 0.652428
italy 0.633130
highest cosine distance values in vector space of the nearest words

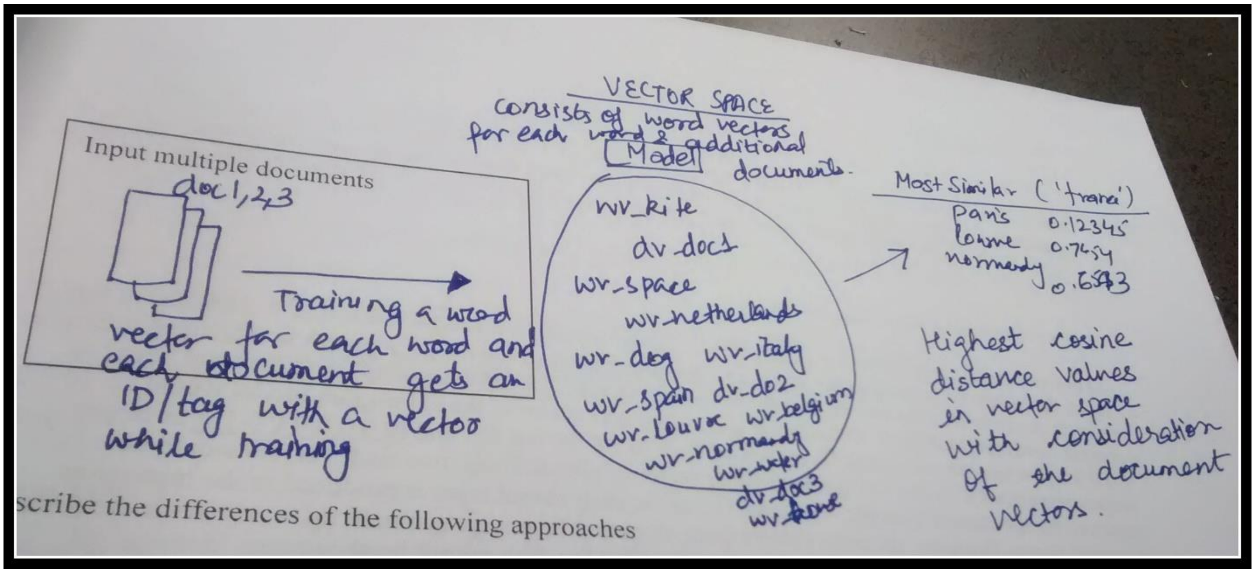
a. Describe the word2vec model

Word2vec is a two-layer neural net that processes text. Its input is a text corpus and its output is a set of vectors: feature vectors for words in that corpus. While **Word2vec** is not a deep neural network, it turns text into a numerical form that deep nets can understand.

Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them. Thecosine of 0° is 1, and it is less than 1 for any other angle.

b. Describe How to extend this model for multiple documents. Also draw a similar diagram for the extended model.

It does so in one of two ways, either using context to predict a target word (a method known as continuous bag of words, or CBOW), or using a word to predict a target context, which is called skip-gram. We use the latter method because it produces more accurate results on large datasets.


Input multiple documents
doc1,2,3
Training a word vector for each word and each document gets an ID/tag with a vector while training
VECTOR SPACE
consists of word vectors for each word & additional documents.
Model
wv_kite
dv_doc1
wv_space
wv_netherlands
wv_dog
wv_italy
wv_spain
dv_do2
wv_louvre
wv_belgium
wv_normandy
wv_water
dv_doc3
wv_france
Most Similar ('france')
paris 0.12345
louvre 0.7454
normandy 0.6543
Highest cosine distance values in vector space with consideration of the document vectors.
scribe the differences of the following approaches

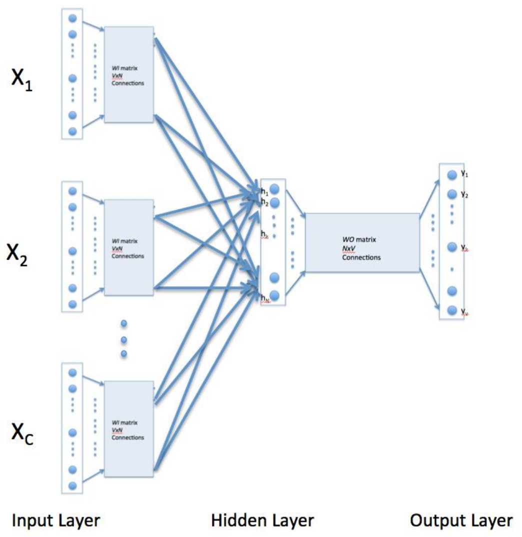
Describe the differences of the following approaches

- Continuous Bag-of-Words model,

The bag-of-words model is a simplifying representation used in natural language processing and information retrieval (IR). In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity. The bag-of-words model has also been used for computer vision.[1]

The bag-of-words model is commonly used in methods of document classification where the (frequency of) occurrence of each word is used as a feature for training a classifier.

In the continuous bag of words model, context is represented by multiple words for a given target words. For example, we could use “cat” and “tree” as context words for “climbed” as the target word. This calls for a modification to the neural network architecture. The modification, shown below, consists of replicating the input to hidden layer connections *C* times, the number of context words, and adding a divide by *C* operation in the hidden layer neurons.



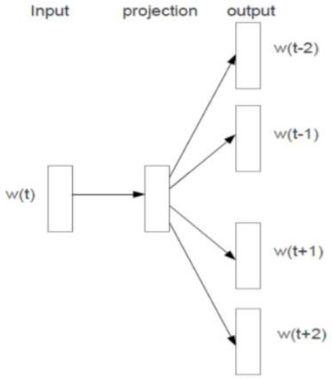
With the above configuration to specify *C* context words, each word being coded using 1-out-of-V representation means that the hidden layer output is the average of word vectors corresponding to context words at input. The output layer remains the same and the training is done in the manner discussed above.

- Continuous Skip-gram model

Skip-gram model reverses the use of target and context words. In this case, the target word is fed at the input, the hidden layer remains the same, and the output layer of the neural network is replicated multiple times to accommodate the chosen number of context words. Taking the example of "cat" and "tree" as context words and "climbed" as the target word, the input vector in the skim-gram model would be [0 0 0 1 0 0 0 0]t, while the two output layers would have [0 1 0 0 0 0 0 0] t and [0 0 0 0 0 0 0 1]t as target vectors respectively. In place of producing one vector of probabilities, two such vectors would be produced for the current example. The error vector for each output layer is produced in the manner as discussed above. However, the error vectors from all output layers are summed up to adjust the weights via backpropagation. This ensures that weight matrix *WO* for each output layer remains identical all through training.

For the sentence "morning fog, afternoon light rain,"

- Place the words on the skip-gram Word2Vec model below.



- Draw a CBOW model using the same words.

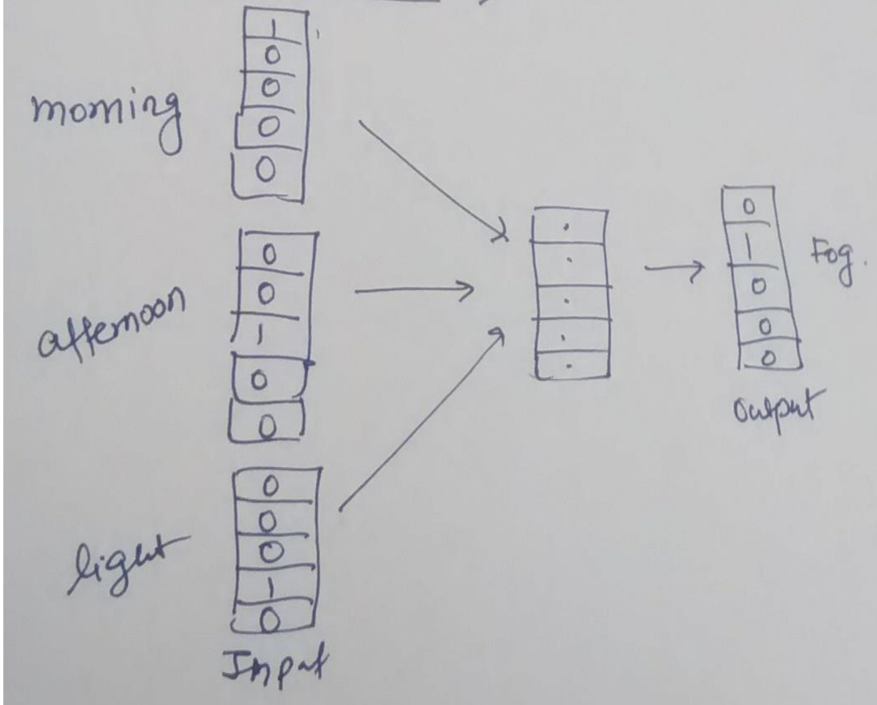
Consider window size = 1

INPUT	TRAINING SAMPLES
morning	(morning, fog) (morning, afternoon)
fog	(fog, morning) (fog, afternoon) (fog, light)
afternoon	(after-noon, morning) (afternoon, fog) (afternoon, light) (afternoon, rain)
light	(light, morning) (light, fog) (light, afternoon) (light, rain)
rain	(rain, morning) (rain, fog) (rain, afternoon) (rain, light)

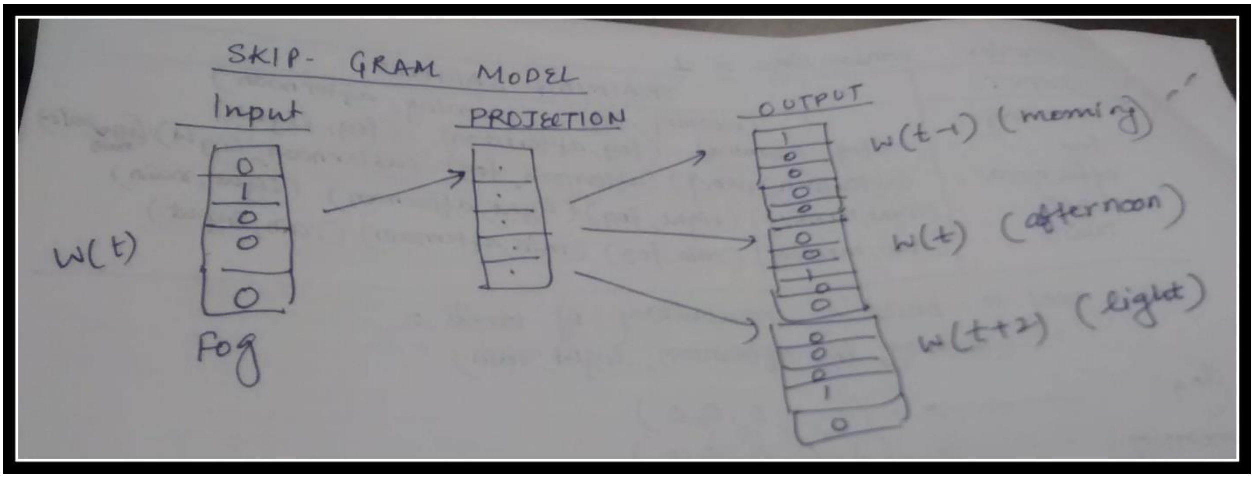
We need to build a vocabulary of words :
(morning, fog, afternoon, light, rain)

(fog) → (0,1,0,0,0)
(morning) → (1,0,0,0,0)
(afternoon) → (0,0,1,0,0)
(light) → (0,0,0,1,0)
(rain) → (0,0,0,0,1)

CBOW MODEL :→



As skig-gram is mirror image of the same :


SKIP- GRAM MODEL
Input
PROJECTION
OUTPUT
w(t)
0
1
0
0
0
Fog
1
0
0
0
0
w(t-1) (morning)
0
0
1
0
0
w(t) (afternoon)
0
0
0
1
0
w(t+2) (light)