CS5560 Knowledge Discovery and Management

Problem Set 4

June 26 (T), 2017

Name: Mudunuri Sri Sai Sarat Chandra Varma

Class ID: 14

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>

1) Compute the probability of sentence "I like green eggs and ham" using the appropriate bigram probabilities.
2) Compute the probability of sentence "I like green eggs and ham" using the appropriate trigram probabilities.

Answer:

1) **Probability of sentence "I like green eggs and ham" using the appropriate bigram probabilities:**

"I like green eggs and ham"

P(I | <s>) = 2/3 = 0.67

P(like | <s>) = 1/3 = 0.33

P(green | like) = 1/3 = 0.33

P(eggs | green) = 1/3 = 0.33

P(and | eggs) = 1/3 = 0.33

P(ham | and) = 1/3 = 0.33

P(</s> | ham) = 1/3 = 0.33

P(am | </s>) = 1/3 = 0.33

P(sam | </s>) = 1/3 = 0.33

2) **Probability of sentence "I like green eggs and ham" using the appropriate Trigram probabilities:**

"I like green eggs and ham"

P(I | <s> | like) = 1/3 = 0.33

P(like | I |green) = 1/3 = 0.33

P(green | like | eggs) = 1/3 = 0.33

P(eggs | green | and) = 1/3 = 0.33

P(and | eggs | ham) = 1/3 = 0.33

P(ham | eggs | </s>) = 1/3 =0.33

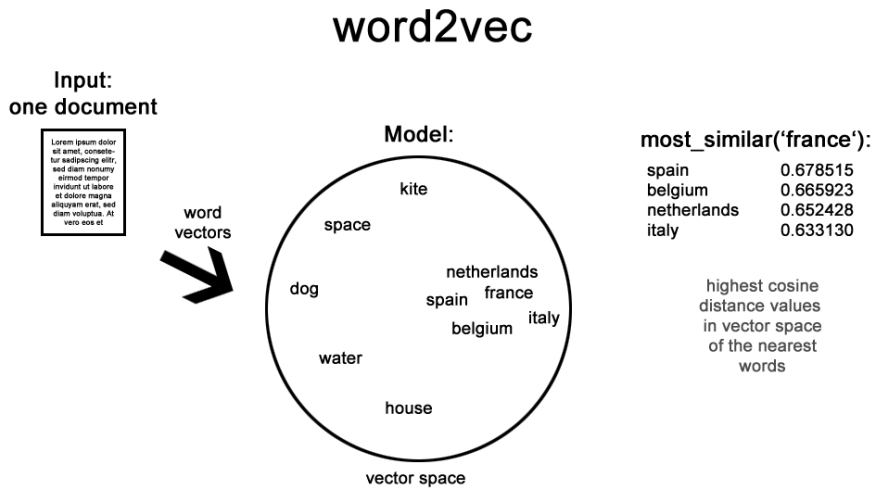
P(</s> | and | ham) = 1/3 = 0.33

P(ham | </s> | and) = 1/3 = 0.33

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.



a. Describe the word2vec model
b. Describe How to extend this model for multiple documents. Also draw a similar diagram for the extended model.

Answer:

a. Word2vec model:

Word2vec is a group of related models that are used to produce word embeddings. These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. Word2vec takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space. Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located in close proximity to one another in the space.

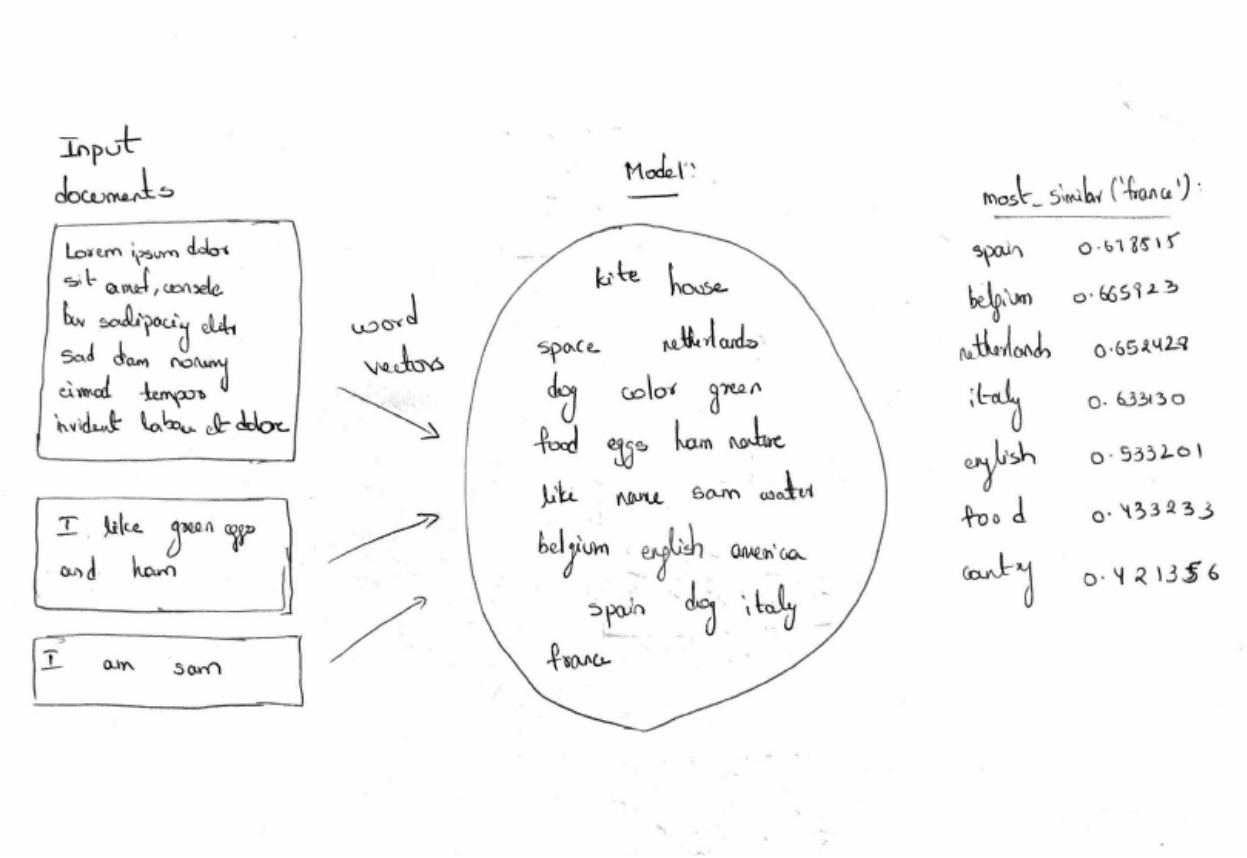
Word2vec was created by a team of researchers led by Tomas Mikolov at Google. The algorithm has been subsequently analysed and explained by other researchers. Embedding vectors created using the Word2vec algorithm have many advantages compared to earlier algorithms like Latent Semantic Analysis.

Word2vec can utilize either of two model architectures to produce a distributed representation of words: continuous bag-of-words (CBOW) or continuous skip-gram. In the continuous bag-of-words architecture, the model predicts the current word from a window of surrounding context words. The order of context words does not influence prediction (bag-of-words assumption). In the continuous skip-gram architecture, the model uses the current word to predict the surrounding window of context words. The skip-gram architecture weighs nearby context words more heavily than more distant context words. According to the authors' note, CBOW is faster while skip-gram is slower but does a better job for infrequent words.

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

The word2vec model can be extended for multiple documents by doc2vec. Doc2vec is an unsupervised algorithm to generate vectors for sentence/paragraphs/documents. [1405.4053] Distributed Representations of Sentences and Documents . The algorithm is an adaptation of word2vec which can generate vectors for words.The vectors generated by doc2vec can be used for tasks like finding similarity between sentences/paragraphs/documents.Unlike sequence models like RNN, where word sequence is captured in generated sentence vectors, doc2vec sentence vectors are word order independent.For sentence similarity tasks, doc2vec vectors may perform reasonably well. However if the input corpus is one with lots of misspellings like tweets, this algorithm may not be the ideal choice. One may be better off generating word vectors constructed from character n grams using Fasttext fastText , and then adding up the word vectors to compose a sentence vector.Skip thought vectors Skip-Thought Vectors is another option to consider based on problem being solved. It is also an unsupervised algorithm to generate vectors where the vector for a sentence is generated by predicting

adjacent sentences, that are assumed to be semantically related. All the methods mentioned above are unsupervised algorithms requiring no training data.



Describe the differences of the following approaches

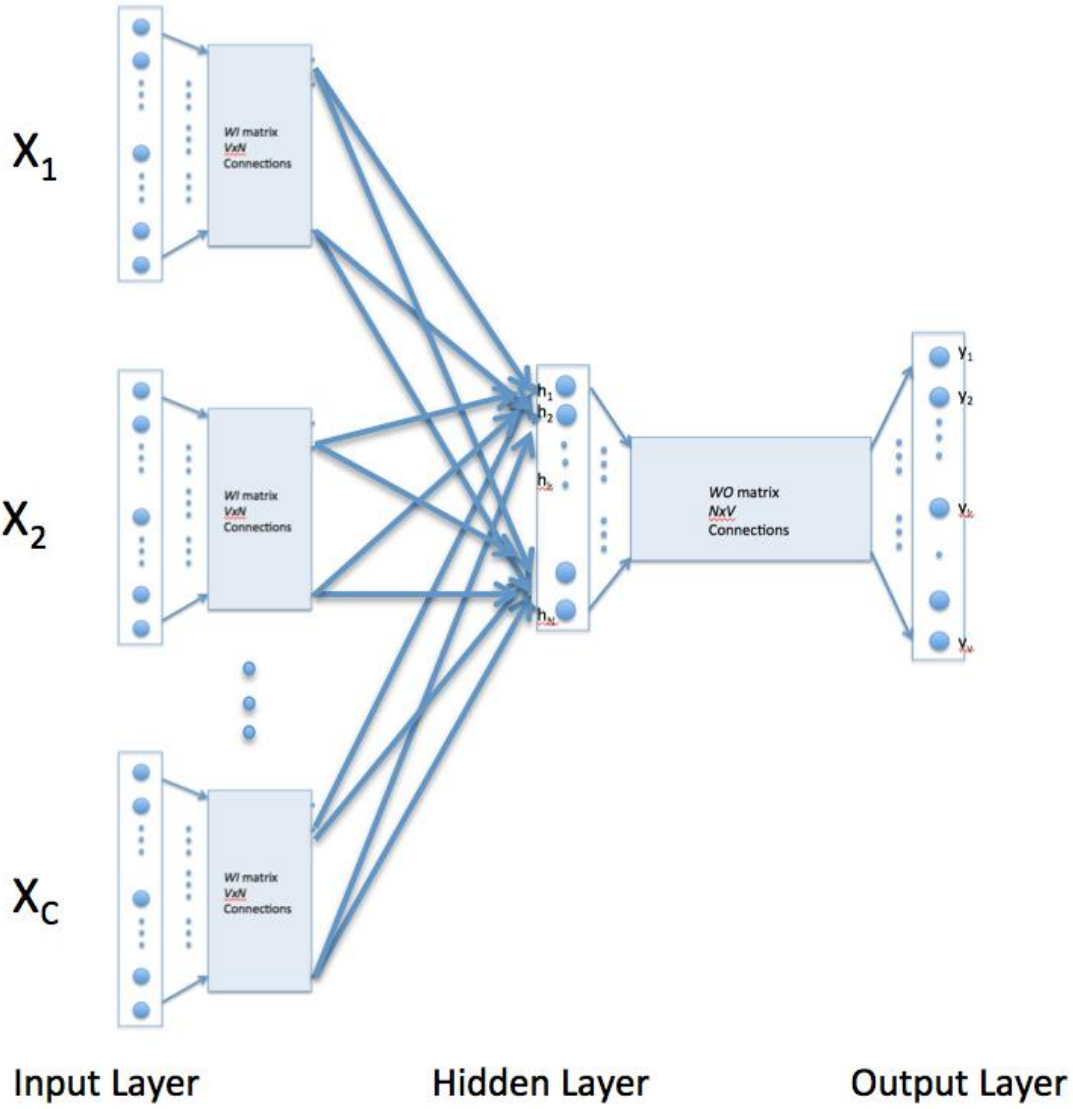
- Continuous Bag-of-Words model,
- Continuous Skip-gram model

Answer:

<u>Continous Bag-of-Words model:</u>

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.

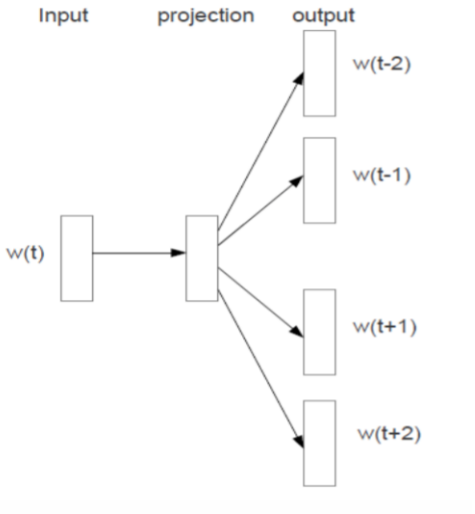
- **<u>Continuous Skip-gram model:</u>**

  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.

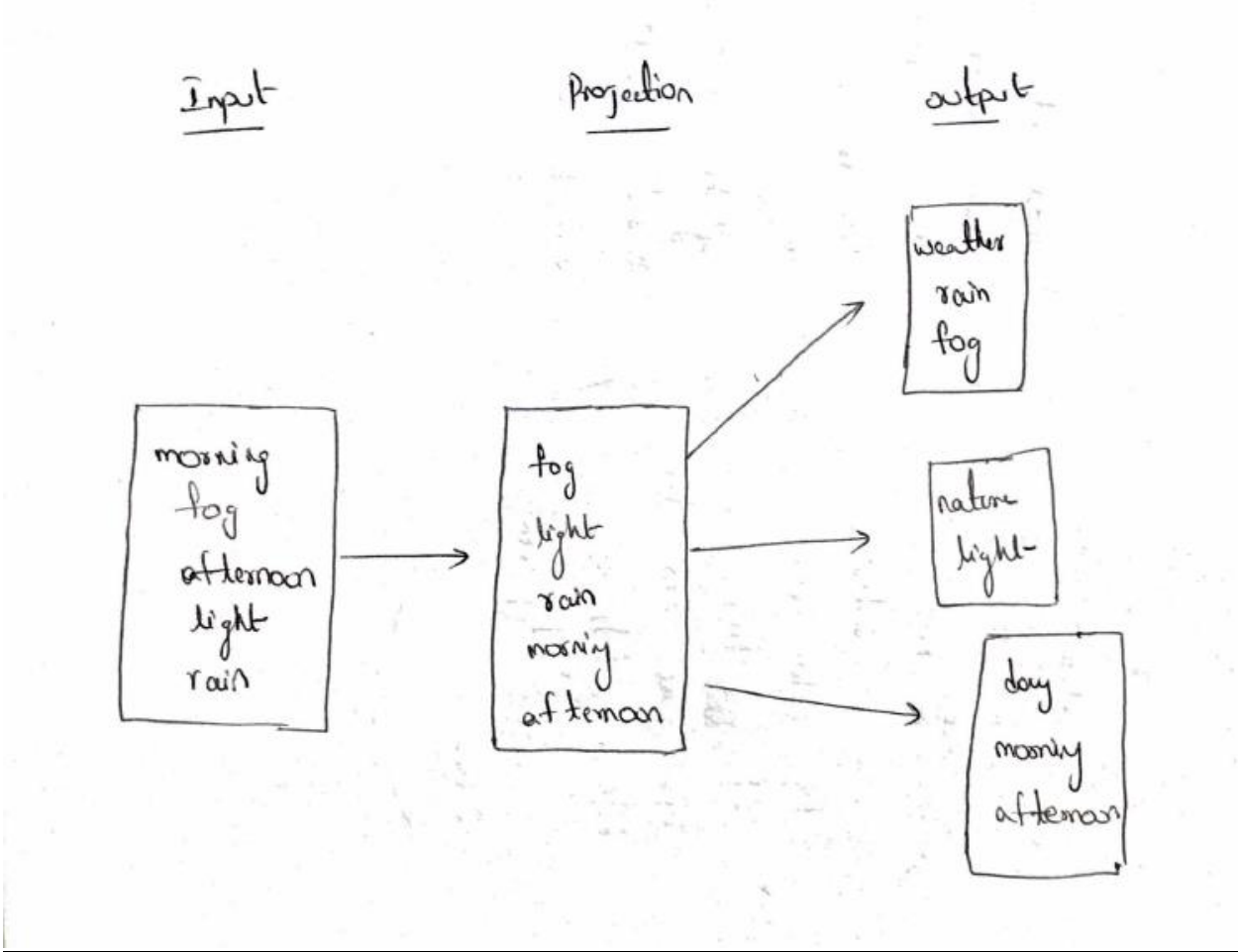
For the sentence “morning fog, afternoon light rain,”

- Place the words on the skip-gram Word2Vec model below.
- Draw a CBOW model using the same words.

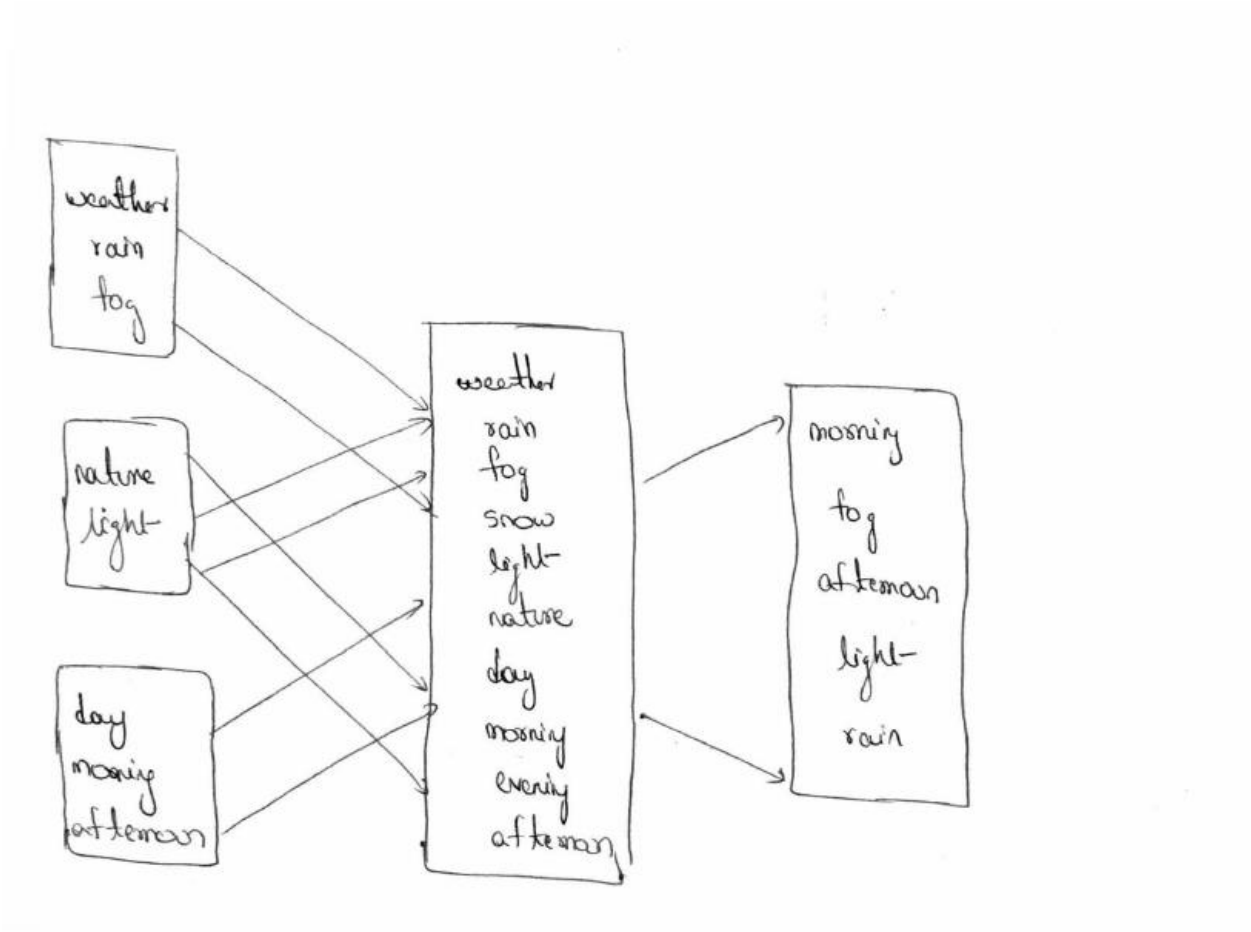


Answer:

skip-gram Word2Vec model:



CBOW model:


weather
rain
fog
nature
light-
day
morning
afternoon
weather
rain
fog
snow
light-
nature
day
morning
evening
afternoon
morning
fog
afternoon
light-
rain