

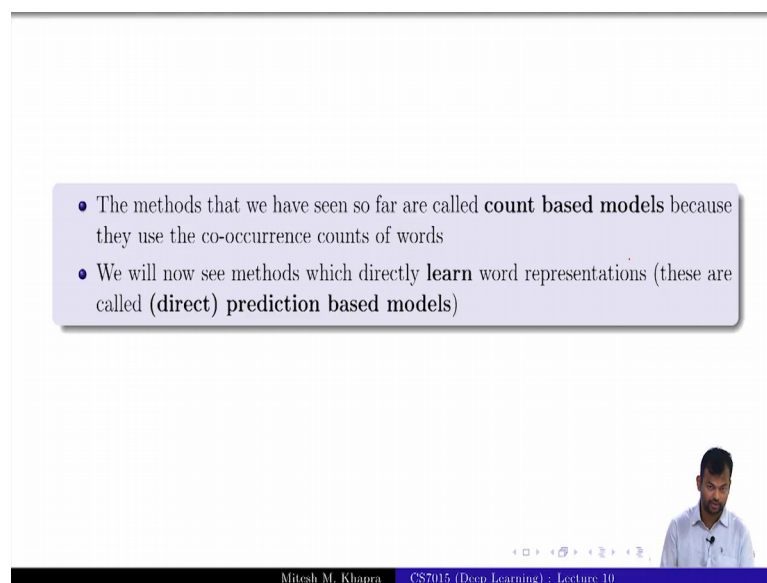
**Deep Learning**  
**Prof. Mitesh M Khapra**  
**Department of Computer Science Engineering**  
**Indian Institute of Technology, Madras**

**Module – 10.4**  
**Lecture – 10**  
**Continuous bag of words model**

So, from here on so, none of this that we covered had anything to do with neural networks say, but it was important to understand the context and I will tell you why it was important to go over the traditional way of learning word representations and then we will see how it ties to the modern way or the neural network way of learning representations right.

So, we will start with the first neural network based model for learning word representation, which is known as the Continuous bag of words model.

(Refer Slide Time: 00:37)



- The methods that we have seen so far are called **count based models** because they use the co-occurrence counts of words
- We will now see methods which directly **learn** word representations (these are called **(direct) prediction based models**)

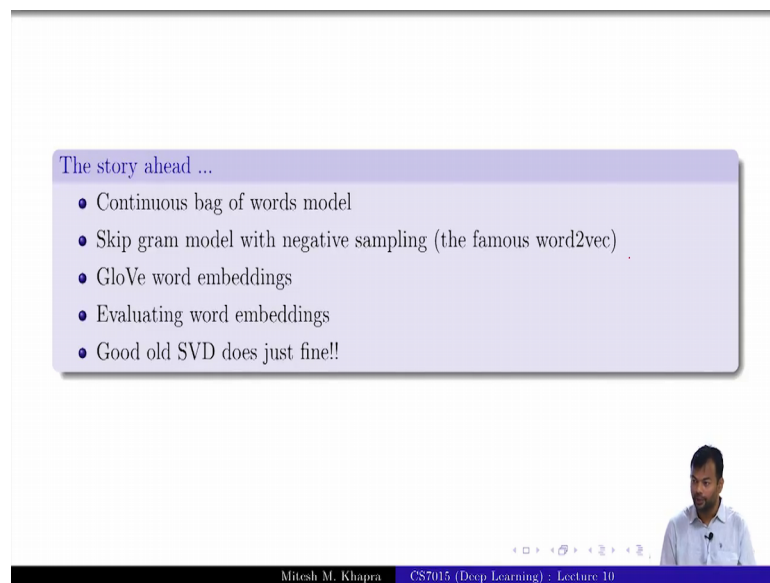
Navigation icons

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

So, just to set the context the methods that we have seen so far are known as count based models because they rely on these co occurrence counts for learning representations of words, and the methods that we are going to see now are called prediction based models and it will become clear shortly why the term prediction, and how they learn the word representations right.

So, in a way in the original thing there was no learning involved of course, you can say that you were trying to learn these eigenvectors and eigenvalues and so on, but it was not in the same way as it be as we have been learning parameters of a neural network and so on right it was not in the same spirit. But now once I do these second type of models, this distinction would become very clear one why is there a learning involved and why they are prediction based models.

(Refer Slide Time: 01:24)



The story ahead ...

- Continuous bag of words model
- Skip gram model with negative sampling (the famous word2vec)
- GloVe word embeddings
- Evaluating word embeddings
- Good old SVD does just fine!!

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

So, the story is we are going to look at continuous bag of words model, then something known as skip gram. So, this is the famous word 2 vec model which you guys have already started looking at then we look at GloVe word embeddings which is some kind of a hybrid between the count based models and the prediction based models. And then we see how to evaluate word embeddings and then end with this depressing note that good old SVD is just fine right. So, all the progress that has happened in the past 5-6 years, you could just use SVD and still go by, but if you do that you will probably not get a job right you have to learn these things.

(Refer Slide Time: 01:59)

- Consider this Task: Predict  $n$ -th word given previous  $n-1$  words
- Example: he sat on a chair

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

So, now let us start with the continuous bag of words model. So, consider this task and just bear me for a few slides that when why this is connected to our problem and all that. So, I am going to consider ka task, we are here to predict the  $n$  th word given the previous  $n$  minus 1 words right. As this is something that you do regularly sometimes even in the class when you are whatsapping or smsing. So, this is what you do right you start typing he sat on her and you get this prompt that the next word should be chair or something like that right.

Now, you can think of this as a classification problem; tell me why you can think of this as a classification problem, can you tell me what is. So, remember that we have always thought of this that there is always a  $y$  there is always an  $x$ , and then we are trying to learn this relation from  $x$  to  $y$ . So, given this example can you tell me what is  $x$  here and what is  $y$  here?

Student: (Refer Time: 02:51).

Everyone is clear about that right. So, this is  $x$  and this is  $y$  now; I made a statement that I could think of this as a classification problem right. So, the minute I say classification what is the  $y$  that comes to your mind or dash hot by.

Student: (Refer Time: 03:07).

Y not I, anything else would have been inappropriate, but. So, one hot y is what you would expect there right and now what is the size of this one hot vector?

Student: Size of (Refer time: 03:19).

Size of the vocabulary. So, we are trying to predict one of the words in the vocabulary. So, you see why this is a multi class classification problem you see that there are many classes and you want to select one of these class. Now the moment I say classification I give you an x and y, I will start asking me who will give me the training data for this. So, can you think of training data for this any corpus similar to the one that you are creating right?

(Refer Slide Time: 03:46)

Sometime in the 21st century, Joseph Cooper, a widowed former engineer and former NASA pilot, runs a farm with his father-in-law Donald, son Tom, and daughter Murphy. It is post-truth society (Cooper is reprimanded for telling Murphy that the Apollo missions did indeed happen) and a series of crop blights threatens humanity's survival. Murphy believes her bedroom is haunted by a poltergeist. When a pattern is created out of dust on the floor, Cooper realizes that gravity is behind its formation, not a "ghost". He interprets the pattern as a set of geographic coordinates formed into binary code. Cooper and Murphy follow the coordinates to a secret NASA facility, where they are met by Cooper's former professor, Dr. Brand.

Some sample 4 word windows from a corpus

- Consider this Task: Predict  $n$ -th word given previous  $n-1$  words
- Example: he sat on a chair
- Training data: All  $n$ -word windows in your corpus
- Training data for this task is easily available (take all  $n$  word windows from the whole of wikipedia)
- For ease of illustration, we will first focus on the case when  $n = 2$  (i.e., predict second word based on first word)

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

In specific what you will do is, suppose you have framed it as the following problem that you are given for words and you want to predict the fifth word. So, in general I have call it as that you are given  $n$  minus 1 words and you want to predict the  $n$  th word the  $n$  that I am considering here is 4.

Now, what is going to be the training data for this? If you take any corpus that you have built anything right consider all five word windows from there do not get too engrossed in the story. So, there are four the first four words you can treat as x and the fifth word would be your y right. So, you can construct many such x comma y pairs from the raw corpus that you are creating, any 5 word window and you can keep sliding this window

right what I mean by that? This is your first training instance  $x$  comma  $y$ , this could be your second training instance. So, this could be these overlapping training instances you keep sliding this window and you will get many many training instances you see that.

So, this task the advantage is that, given the size of the web and so on at least for popular languages the training data almost comes for free right. Compare this to MNIST or any other task where you have to actually acquire these labels that this is an apple, this is a banana and so on here you get the training data for free just need to scrape it from the web know. So, window size is something that you will set right whether you want to learn four word windows or what do you mean we do not know the window size no.

So, this again there is a lot of existing literature in the traditional NLP, where various and lot of work has been done to figure out what is the right  $n$ . So, in most NLP task right if you want to predict the next word, a three word window is enough actually. If you know the last three words and you can try this as a mental exercise right; if you know three words you do not really need to know the words before that. So, this is the mark of assumption with where this is a trigram dependency in the words right.

So, this  $n$  is not really so difficult, and in the default tool that you guys would try probably they take the value of  $n$  is 7 that is an overkill, but that is again it comes from a lot of existing literature in NLP right this is not a this task is not deep learning broad right this task is a simple language modeling task, which has existed for many many years right from probably 1950s or 60s or something.

So, this is all  $n$  word windows in your corpus as I said training data comes for free and for ease of illustration we will now focus on the case when  $n$  is equal to 2; that means, I am given one word and I want to predict the next word ok.

(Refer Slide Time: 06:18)

We will now try to answer these two questions:

- How do you model this task?
- What is the connection between this task and learning word representations?

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

And we will see how to model this using a neural network. So, these are the 2 questions which I need to tell you how to model this task and what is the connection between this task and our original task of learning word representations these are the 2 things that I am going to answer.

(Refer Slide Time: 06:28)

We will model this problem using a feedforward neural network

- **Input:** One-hot representation of the context word
- **Output:** There are  $|V|$  words (classes) possible and we want to predict a probability distribution over these  $|V|$  classes (multi-class classification problem)

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

So, we will model this problem using a feed forward neural network, what is the input? One word, so say the word is sat I am going to represent it using a one hot vector and what is the output? I want to predict a distribution over all the words in the vocabulary

and I want to predict I want to pick the word which has the maximum probability that is how you did. So, for example, in the case when you had this classification problem of banana, apple, orange, mango, you predicted a distribution over these 4 classes and then picked the one which had the highest probability exact same idea here it just that instead of 4 classes now you have  $V$  classes and your  $V$  is very large but it is trying to learn a distribution over there.

And you know that in this case or the example that you are considering on is the actual next word. So, you type sat and the next word is on, and probably leading to sat on the chair or something like that. So, this is what you would want to maximize. I have given you the input, I have given you the output give me a neural network to model this, there are lot of hints in the diagram itself right you see some space between the input and the output.

So, what will you put in the middle layer we will put a middle layer there right. Is this a way of modeling this task. I have an input I want to predict an output, so, I just use a regular feed forward neural network and let us analyze these parameters a bit more carefully right. So, I am something known as  $W$  context, I have something known as  $W$  word I am already using some notations from the SVD lecture. There at the end we ended with  $W$  word and  $W$  context right it is not clear why I am using the same notations, but it will become clear in some time, but let us look at their dimensions right.

So, we have this one hot vector I have a parameter  $W$  context which I am going to learn right and its size is  $k$  cross  $V$ . So, what does that mean? This matrix is going to multiply by the vector and give me a  $k$  dimensional output right is that clear. So, I have this is of size  $V$  because always keep surprising me I do not know why you cannot do this  $R$  this is a  $V$  dimensional vector, you multiply it by a  $k$  cross  $V$  vector. So, you do  $W$  into  $x$ . So, you will get a  $k$  dimensional vector. So, this is  $k$  dimensional you have a  $k$  dimensional hidden representation.

And from there now having captured this hidden representation, you are trying to predict which is the next possible class. This is the same as any other thing right if you had done the image classification or the  $n$  th digit classification, you had this 784 dimensional input vector, you pass it through a hidden layer and then you predicted one of the 10 classes, there is nothing magic here it is the same thing that you have done seen before.

(Refer Slide Time: 09:17)

- We will model this problem using a feedforward neural network
- **Input:** One-hot representation of the context word
- **Output:** There are  $|V|$  words (classes) possible and we want to predict a probability distribution over these  $|V|$  classes (multi-class classification problem)
- **Parameters:**  $W_{context} \in \mathbb{R}^{k \times |V|}$  and  $W_{word} \in \mathbb{R}^{k \times |V|}$  (we are assuming that the set of words and context words is the same: each of size  $|V|$ )

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10 27/70

How many if you get this and what are the parameters  $W_{context}$  and  $W_{word}$ . And we are going to focus on these parameters and understand what they actually mean.

(Refer Slide Time: 09:26)

- What is the product  $W_{context}x$  given that  $x$  is a one hot vector
- It is simply the  $i$ -th column of  $W_{context}$

$$\begin{bmatrix} -1 & 0.5 & 2 \\ 3 & -1 & -2 \\ -2 & 1.7 & 3 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.5 \\ -1 \\ 1.7 \end{bmatrix}$$

- So when the  $i^{th}$  word is present the  $i^{th}$  element in the one hot vector is ON and the  $i^{th}$  column of  $W_{context}$  gets selected
- In other words, there is a one-to-one correspondence between the words and the columns of  $W_{context}$
- More specifically, we can treat the  $i$ -th column of  $W_{context}$  as the repr context  $i$

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

So, what is the product  $W_{context}$  into  $x$  given that  $x$  is a one hot vector. So, I will tell you this suppose the  $i$  th entry is hot here, how many if you say it is the  $i$  th column of  $W_{context}$ . So, it is simply the  $i$  th column of  $W_{context}$  why? Because you have this  $W_{context}$  matrix you take a one hot vector which has the second entry as hot, if you do this



multiplication you basically get the second column of  $W$  and you can just see it everyone gets this now how many if you get this now?

So, if you have a one hot vector if its  $i$ th entry is on  $u$  multiply it by a matrix, you will get the  $i$ th column of the matrix. So, if the  $i$ th word is present in the input then the  $i$ th element of the one hot vector is on and the  $i$ th column of  $W$  context can be would be selected. So, then can what can you tell me about the  $i$ th column of  $W$  context? You see there is this one to one correspondence between words in your vocabulary and columns of the  $W$  context matrix; how many columns has  $W$  context have?  $V$  columns how many words are there in your vocabulary?  $V$ . Any one word is on only one column will get selected and that is a unique column it is not going to change right. So, there is a one to one mapping between the columns of  $W$  context and the words in your vocabulary; that means, the columns of  $W$  context are the are the vector representations.

Do you know these vector representations? No these are parameters of your network. So, they will they will be learned how we will see. So, you see the intuition for  $W$  context setting it up this way. So, now, I have set up the problem in a way that by parameter matrix directly gives me the word representations, but any kind of learning has to be driven by some objective. So, what is that objective it is already clear to a lot of you, but we will just do that in a bit more detail fine. So, this is exactly what I have just said.

(Refer Slide Time: 11:18)

• How do we obtain  $P(\text{on}|\text{sat})$ ? For this multi-class classification problem what is an appropriate output function? (**softmax**)

$$P(\text{on}|\text{sat}) = \frac{e^{W_{\text{word}}h[\text{on}]}}{\sum_j e^{W_{\text{word}}h[j]}}$$

Handwritten notes:  $W_{\text{word}} h$  (with  $N \times k$  and  $R$  dimensions), and a diagram of a softmax function.

Now, how do you obtain P on given sat no no. So, for a given training instance so, when you, so you could so, I will. So, for a given training instance you said that your corpus has been divided into those training windows right. So, it is possible that engineer sometimes the word does not and is not the next word, but for this training instance what is it. So, that is what you have to predict right is that fine.

(Refer Slide Time: 11:48)

The diagram illustrates the input to a neural network for word prediction. It shows a sequence of words: "he", "char", "man", and "sat". Above each word is a probability value:  $P(\text{he}|\text{sat})$ ,  $P(\text{char}|\text{sat})$ ,  $P(\text{man}|\text{sat})$ , and  $P(\text{on}|\text{sat})$ . The word "sat" is highlighted in red, and its corresponding probability  $P(\text{on}|\text{sat})$  is circled in red. Below the diagram is a one-hot representation of the word "sat" as a vector:  $[0, 1, 0, \dots, 0, 0, 0]$ .

- We will model this problem using a feedforward neural network
- **Input:** One-hot representation of the context word
- **Output:** There are  $|V|$  words (classes) possible and we want to predict a probability distribution over these  $|V|$  classes (multi-class classification problem)

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

And at test time so you are saying that what you are saying is more practical that when I have typed sat in the whatsapp message, I do not want on as the always the answer. So, we get these 5 options right 3 to 5 options. So, what could be that you have this probability distribution pick the top 5 from there and show it as options so, is that fine? So, we are done with this now how do you compute P on given sat, what is the actual operation happening there, what is the appropriate output function? This is a multi class classification problem softmax.

This is what softmax looks like. So, the property if suppose on is the  $i$ th word in your vocabulary, then I am saying that the probability of on given sat is going to be this quantity; how many of you agree with that? I mean those who agree is fine I am asking why the others do not agree what is not clear about this? I do not know how to explain this I mean it is just so plain obvious, what is the softmax function? First of all you will do this aggregation so, you will do this  $W$  word into  $h$  that is fine right. So, for you will compute this vector consisting of  $W$  word into  $h$  fine what is the dimension of that, what

is the dimension of that mod this is k dimensional this is V cross k or k cross V depending on how you multiply it. So, what is the output going to be V. So, you have V entries.

These are dash entries the options are normalized unnormalized, unnormalized now what does softmax do?

Student: Normalization.

Normalization that is exactly what this formula is doing right. You want for the i th word you see what was the end this product right this gave you a V dimensional vector, you look at the i th entry there right that is what you are doing here raise it to an exponent and divided by the summation of all these entries. Come on guys this is highly disappointing I cannot teach softmax, I had in the tenth lecture eleventh lecture of the course right what is wrong? How many if you get this now just have to ask of it tangle you.

So, you see this right this is what is happening here. So, you get this V dimensional vector and you just convert into a probability distribution using the softmax function. So, now, this value how did he compute this value actually? You computed this product which is W word into h and then you took the i th entry of that and then this some transformation on that, the softmax transformation you see that.

(Refer Slide Time: 14:21)

How do we obtain  $P(on|sat)$ ? For this multi-class classification problem what is an appropriate output function? (**softmax**)

Therefore,  $P(on|sat)$  is proportional to the dot product between  $j^{th}$  column of  $W_{context}$  and  $i^{th}$  column of  $W_{word}$

$$P(on|sat) = \frac{e^{(W_{word}h)[i]}}{\sum_j e^{(W_{word}h)[j]}}$$

Handwritten notes:  $W_{word}h[i]$ ,  $Wh[i]$ ,  $W_{word}h$ .

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

So, now I can say that P on given sat is actually proportional to the dot product between the j th column of W context and i th column of W word why am I saying that?

So, remember that this was the i th word in your vocabulary and on was the j th word in your vocabulary. So, can you explain the meaning of this sentence doing? First let us look at the first part what is h? It is a j th column of W context oh sorry this should be i this should be j. So, this you already saw that h is the jet column of W context because I am multiplying a one odd vector with the matrix is that fine and what is the i th column of W word? So, why what is this product actually equal to if I say W word into h W word into h that is a vector, and then I am taking the i th entry of that. So, I am saying that is the same as taking the i th column of W word and multiplying it by h; how many if you get this is basically in your algebra right.

Now, these 4 different ways of multiplying matrices I am just using one of those right. So, if I multiply a matrix with a vector and then take the i th entry of that, that is the same as multiplying the i th column of the matrix with the vector. Just go back and verify this just take my word for it for now. So, now, what is happening is that it is proportional to the product between the j th column of W context and the i th column of W word is that clear now, everyone gets this.

(Refer Slide Time: 15:56)

$W_{word} \in \mathbb{R}^{k \times |V|}$   
 $h \in \mathbb{R}^k$   
 $W_{context} \in \mathbb{R}^{k \times |V|}$   
 $x \in \mathbb{R}^{|V|}$   
 $sat$

$$P(on|sat) = \frac{e^{(W_{word}h)[i]}}{\sum_j e^{(W_{word}h)[j]}}$$

- How do we obtain  $P(on|sat)$ ? For this multi-class classification problem what is an appropriate output function? (**softmax**)
- Therefore,  $P(on|sat)$  is proportional to the dot product between  $j^{th}$  column of  $W_{context}$  and  $i^{th}$  column of  $W_{word}$
- $P(word = i|sat)$  thus depends on the  $i^{th}$  column of  $W_{word}$
- We thus treat the  $i$ -th column of  $W_{word}$  as the representation of word  $i$
- Hope you see an analogy with SVD! (there we had a different way of learning  $W_{context}$  and  $W_{word}$  but we saw that the  $i^{th}$  column of  $W_{word}$  corresponded to the representation of the  $i^{th}$  word)
- Having understood the int  $W_{context}$  and  $W_{word}$ , our  $i$  learn these parameters

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

So, P word equal to I given sat does depends on the i th column of W word.

So, now what can you say. So, earlier we saw that the  $i$  th column of  $W$  context corresponds to a particular word now what can you say about the  $i$  th column of  $W$  word? It also corresponds to a particular word. So, now, why these 2 correspondences I already had a correspondence between  $W$  context and every word in my vocabulary, now I am saying that there is also correspondence between  $W$  word and every word in my vocabulary; how many of you first of all are comfortable with the sentence? That, every column of  $W$  word has a correspondence with some word in the vocabulary.

The second sentences every column of  $W$  context has a correspondence with some word in the vocabulary do you all of you agree with both these statements? Okay that is what we have try to prove so far. So, now, for every word; that means, I have 2 columns waiting for it, how do I deal with this situation, have you ever dealt with it before? The same thing happened in SVD also right. SVD also gave you this  $u$  sigma which was  $W$  word and then  $V$  which was  $W$  context. So, you can always learn 2 different representations for the words one is when the word appears as a context word and the other is when the word appears as the target would you get that, you see why we have 2 different representations fine.

And as I said hope you see the analogy with SVD right you already saw there that there were these two representation. Now given all this set up and please do not disappoint me can you learn these parameters with some tweaks to the code that you have written for MNIST, can you use the same code to learn these parameters? How many if you say yes. So, what is the tweaks, what are the tweaks? The input changes instead of the image input you have this  $V$  dimensional input, what else changes?

Student: Output.

The output changes instead of a 10 dimensional output you have a  $V$  dimensional output all of you are absolutely clear about this and what is the training algorithm?

Student: (Refer Time: 17:59).

Back propagation what is the loss function? Cross entropy good.

(Refer Slide Time: 18:05)

$\bullet$  We denote the context word (sat) by the index  $c$  and the correct output word (on) by the index  $w$

$\bullet$  For this multiclass classification problem what is an appropriate output function ( $\hat{y} = f(x)$ )? **softmax**

$\bullet$  What is an appropriate loss function? **cross entropy**

$$\mathcal{L}(\theta) = -\log \hat{y}_w = -\log P(w|c)$$

$$h = W_{context} \cdot x_c = u_c$$

$$\hat{y}_w = \frac{\exp(u_c \cdot v_w)}{\sum_{w' \in V} \exp(u_c \cdot v_{w'})}$$

$u_c$  is the column of  $W_{context}$  corresponding to context word  $c$  and  $v_w$  is the column of  $W_{word}$  corresponding to the word  $w$

30/70

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

So, for some I will do some more stuff on this because there is some interesting interpretations of the gradient descent update rule here. So, I will refer to the word sat by the index  $c$ , and the word on by the index  $w$ . And you already saw that the appropriate loss output function is softmax, the appropriate loss function is cross entropy. So, let me just look at this right. So,  $w$  was the index of the output word. So, my cross entropy formula would just boil down to this everyone is fine with this? I will just try to maximize the  $w$ th entry in my  $\hat{y}$ , how many of you are fine with this? Okay and that is nothing, but the probability of the word given the context.

Now, remember that  $h$  is equal to  $W_{context}$  into  $x_c$ , I am going to call that as  $u_c$ . So, you see is the dash of the word sat title of the lecture? It is a vectorial representation of the word sat everyone is fine with that? Because that is exactly what this product is going to do and now my  $\hat{y}_w$  is equal to this because I already said it is the product of the  $c$ th column of  $W_{context}$  and the  $w$ th column of  $W_{word}$  fine.

(Refer Slide Time: 19:32)

- How do we train this simple feed forward neural network? backpropagation
- Let us consider one input-output pair  $(c, w)$  and see the update rule for  $v_w$

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

So, now I have a formula for  $\hat{y}_w$  what is the training algorithm that you will use? Gradient descent with back propagation. Now let us consider one such input output pair and see the update rule for  $V_w$ .

(Refer Slide Time: 19:44)

$$\mathcal{L}(\theta) = -\log \hat{y}_w$$

$$= -\log \frac{\exp(u_c \cdot v_w)}{\sum_{w' \in V} \exp(u_c \cdot v_{w'})}$$

$$= -(u_c \cdot v_w - \log \sum_{w' \in V} \exp(u_c \cdot v_{w'}))$$

$$\nabla_{v_w} = -(u_c - \frac{\exp(u_c \cdot v_w)}{\sum_{w' \in V} \exp(u_c \cdot v_{w'})} \cdot u_c)$$

$$\nabla_{v_w} = \frac{\partial}{\partial v_w} \mathcal{L}(\theta)$$

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

So, my loss function is this, this is actually this quantity. Now I can just rewrite it as this I have just expanded the log. So, the log of a by b is log a minus b now I want this quantity. Because this is the parameter of the network right  $v_w$  is one of the columns of  $W_{context}$  or is it  $W_{word}$   $w_{word}$ ;  $v_w$  is one of the columns of  $W_{word}$  and I want to

learn I want to learn it what are the what are these column entries so; that means, I am interested in this particular gradient.

So, I will start taking this. So, what is it going to be? So, only  $u_c$  will remain here, of all these summation terms only one of them would remain and then you can derive derivative right. So, this is what it is going to look like what is this quantity? The softmax machinery. So, this is what I get how many of you are comfortable with this? Okay good.

(Refer Slide Time: 20:42)

$$\mathcal{L}(\theta) = -\log \hat{y}_w$$

$$= -\log \frac{\exp(u_c \cdot v_w)}{\sum_{w' \in V} \exp(u_c \cdot v_{w'})}$$

$$= -(u_c \cdot v_w - \log \sum_{w' \in V} \exp(u_c \cdot v_{w'}))$$

$$\nabla_{v_w} = -(u_c - \frac{\exp(u_c \cdot v_w)}{\sum_{w' \in V} \exp(u_c \cdot v_{w'})} \cdot u_c)$$

$$= -u_c(1 - \hat{y}_w)$$

And the update rule would be

$$v_w = v_w - \eta \nabla_{v_w}$$

$$= v_w + \eta u_c(1 - \hat{y}_w)$$

32/70

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

So, now my gradient update rule is going to look like; everyone is fine with this I have derived this formula and I have just substituted that here, and this negative and this negative. So, now let us look at this update rule.



(Refer Slide Time: 20:57)

- This update rule has a nice interpretation

$$v_w = v_w + \eta u_c (1 - \hat{y}_w)$$

- If  $\hat{y}_w \rightarrow 1$  then we are already predicting the right word and  $v_w$  will not be updated
- If  $\hat{y}_w \rightarrow 0$  then  $v_w$  gets updated by adding a fraction of  $u_c$  to it

$$v_w = v_w + \eta u_c$$
$$w = w + x$$

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

So, this update rule has a very nice interpretation which allows us to understand, what does the continuous bag of words model actually learn. Now suppose  $\hat{y}_w \rightarrow 1$  what would that mean? Your prediction is very correct right you are almost predicting it has probability as 1 what would happen to the update in that case? There will be no update if it is one there will be no update if it is close to 1 there is going to be very very minimalistic update; that means, you have already learned the  $v_w$  well enough.

On the other hand if I am very bad, if  $\hat{y}_w$  is close to 0 what would happen? Just tell me the case when  $\hat{y}_w$  is actually 0 what is the update rule? Have you seen something similar ever before? Have you seen something similar before where did you see this update rule? Perceptron what happened when you did this?  $W$  and  $x$  came closer to each other the angle between them actually decreased. So, the same thing is happening here right.

So, what you are trying to do is, you are trying to make your word representation closer to the context representation is that clear how many if you get this? It straight away follows from the update rule right because you are adding a fraction of your context vector to your word vector and we know that when we add 2 vectors they come close to each other the cosine between them decreases, that is what we proved in the word to it lecture in the perceptron lecture right.

(Refer Slide Time: 22:30)

- This update rule has a nice interpretation
 
$$v_w = v_w + \eta u_c (1 - \hat{y}_w)$$
- If  $\hat{y}_w \rightarrow 1$  then we are already predicting the right word and  $v_w$  will not be updated
- If  $\hat{y}_w \rightarrow 0$  then  $v_w$  gets updated by adding a fraction of  $u_c$  to it
- This increases the cosine similarity between  $v_w$  and  $u_c$  (How? Refer to slide 38 of Lecture 2)
- The training objective ensures that the cosine similarity between word ( $v_w$ ) and context word ( $u_c$ ) is maximized

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10 33/70

So, you can go back and refer to this slide on lecture 2. Now, so, the training objective is essentially ensuring that the cosine similarity between  $v_w$  and context word is maximized; between the word and the context word is maximized.

(Refer Slide Time: 22:47)

- What happens to the representations of two words  $w$  and  $w'$  which tend to appear in similar context ( $c$ )
- The training ensures that both  $v_w$  and  $v_{w'}$  have a high cosine similarity with  $u_c$  and hence transitively (intuitively) ensures that  $v_w$  and  $v_{w'}$  have a high cosine similarity with each other
- This is only an intuition (reasonable)
- Haven't come across a formal proof for this!

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

Now, what is the result of this? Now I want you to think go back with a starting example, where we said that we want to learn representations such that cat and dog are close to each other, but cat and truck are not close to each other.

I want you to think whatever you see on this slide, the conclusions that you drew from this slide, how do they help you to relate back to that initial goal. So, now, let us let me give you the intuition right. So, what happens to the representations of 2 words  $w$  and  $w'$  which tend to appear in the same context  $c$ ? So, say dog eats cat eats right. So, dog and cat are 2 words, which appear with the same context eats. So, what will happen to the representation of dog? It will come close to eats; what will happen to the representation of cat? Come close to eats. Not only that dog will also go close to pet animal sleeps right and so on and cat will also go close to these. So, transitively what will happen? Dog is going close to a certain point or certain sets of points; cat is also coming close to the same set of points. So, transitively dog and cat will come close together you get this intuition.

Anyone sees a problem with this? No. So, known objective and I said that dog comes close to eats is that what he wanted I mean, why should dog be close to eats; that means, if I find the nearest neighbors of dog, I will get words like eats, sleeps, barks and so on is that what I wanted? So, that is exactly what is happening? And based on that I convinced you that dog and cat will come close to each other, but there is a subtle gap here I want you to close that gap how many matrices do we have? 2 that is enough hint; we are going to either take columns of this matrix as the representations or the columns of this matrix as the representation not mixed.

So, now can you tell me? So, dog here will come close to eats, sleeps, barks, here word will come close to context word right? Cat here will come close to eat, sleeps, and so on right. So, transitively dog and cat will come close to here and this is the representation that you care about not representations across these 2 here ah. So, what I said is that the training rule ensures that the words representation comes close to the context words representation that is what we saw with the training update rule.

So; that means, dog will come close to any kind of context word that it appears with. So, dog I would expect it to appear with context words like eats, pet animals, dog, barks drinks and so on right. So, dog is coming closer to these words, and I expect cat also to come up here with these words and of course, I do not expect truck to appear with these words right. So, then cat will also come close to these set of words, dog will also come close to these set of words. So, transitively dog and cat will come close to each other

right all of them are coming close to each other, which is fine which was my original goal.

But my original goal was not that dog and eats should come close to each other, because eats and dog are neither synonyms when they do not have any semantics I mean they have a semantic relation, but that is not what I wanted. I wanted similar words to come close to each other, but now I have the side effect that dog is coming close to eats, but that is bad how can I live with that.

So, the I mean the key thing that you should notice is that, you have one matrix of words; the other matrix is of context words. So, the representation of dog in the word matrix is coming close to the representation of eats, sleeps etc in the context representation on the context matrix. The representation of cat is also coming close to these words in the context representation and transitively because of this dog and cat in the word matrix are coming close to each other and this is the matrix that we care about.

In this matrix dog and eats, dog and sleeps are not close to each other right is that fine everyone gets this now. So, this is only an intuition and this becomes very tricky when I will blow this up; what do I mean by blow this up? Right now what am I trying to do what is the size of  $n$ ? 2 right I am taking one word and outputting the other word, hence you get all these neat interpretations that you are moving close to that vector and so on. The moment I add more words to  $n$  these interpretations become more and more hard right, but this again I mean this is good to understand that this is what happens at least in the best case. So, this is only an intuition which is reasonable in my opinion, I have not come across a formal proof, which says that this is what actually happens and that is one criticism forward to a grade. It works very well, but there is no formal proof which tells you why exactly it works right.

As opposed to SVD right there we know there is a principle behind it, here that is not very clear right, but it works very well based on this intuition. So, everyone gets the whole set up how we started with a classification problem of predicting the  $n$ th word given the  $n$  minus 1 words, which had nothing to do with word representations that is a simple language modeling problem which has existed forever. We smartly modeled it or someone smartly modeled it using a neural network such that the, parameters of the neural network end up giving you the word representations. And this network is end to

end trainable using an objective function the training data comes for free, for popular languages you have like tons of training data the entire Wikipedia entire web whatever you can scrape, that is why with more and more training data you can learn even better and better representations. So, for popular languages the representations are really good.

And then we saw an intuitive explanation for why this works because of this movement of things closer to each other and the key thing to notice there are 2 different representation matrices one for the words one for the context. And this is not surprising the same thing happened for SVD also, u sigma was W word and v was W context right. So, it is all in the same spirit right.

(Refer Slide Time: 28:38)

- In practice, instead of window size of 1 it is common to use a window size of  $d$
- So now,
 
$$h = \sum_{i=1}^{d-1} u_{c_i}$$
- $[W_{context}, W_{context}]$  just means that we are stacking 2 copies of  $W_{context}$  matrix
 
$$\begin{bmatrix} -1 & 0.5 & 2 & -1 & 0.5 & 2 \\ 3 & -1.0 & -2 & 3 & -1.0 & -2 \\ -2 & 1.7 & 3 & -2 & 1.7 & 3 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \text{sat}$$

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \text{he}$$

$$= \begin{bmatrix} 2.5 \\ -3.0 \\ 4.7 \end{bmatrix}$$
- The resultant product would simply be the sum of the columns corresponding to 'sat' and 'he'

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10 35/70

Now, in practice instead of window size of 1 it is common to use a window size of  $d$ ; either  $d$  could be 4 or 7 I have even say and seen 11 actually, but not beyond that. Now let us see what happens if you have 2 and here itself it should become clear that, now those interpretations are not very neat. So, what I will use suppose I want to take a context of 2 words, then I have he sat and now I want to predict the next word right.

So, what is my input now? He and sat right. So, I will take the one hot representations of he and sat, I will just concatenate them sorry I just concatenate them is that fine and my input now belongs to  $R^{2|V}|$ , in general it will belong to  $R^{d|V}|$  and now what is the next step? Do you see something funny here? I have just created 2 copies of this I am telling you an inefficient way of doing this, later on it will be a very simple

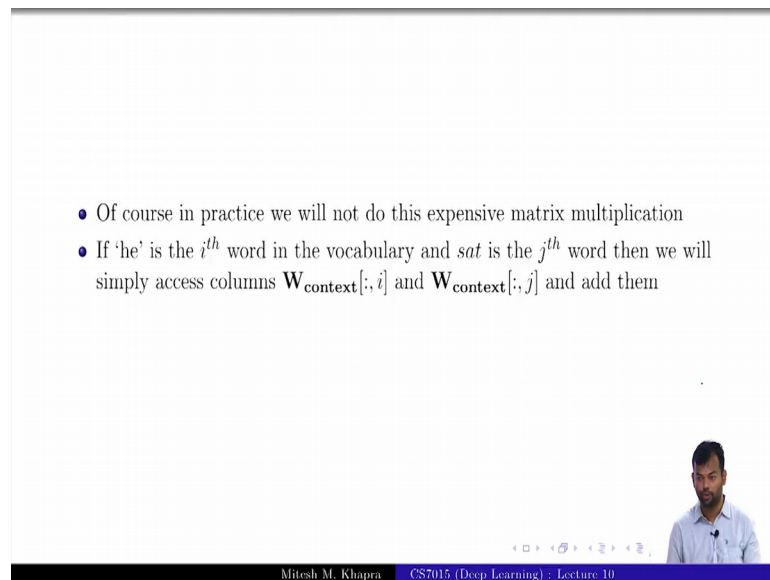
thing to do a very efficient way of doing this right, but first just to get the matt around I will just do inefficient way of doing it.

Why have I staged it twice 2 words right. So, now, my  $h$  is actually going to be the sum of all the columns of  $w$  which correspond to my input words is that fine? I have to earlier I had just one word as the input. So, my  $h$  was just equal to that column of  $W$ ; now my  $h$  is going to be equal to the sum of all the columns of  $w$  corresponding to the words that I have and I will tell you why. So, I have taken  $w$  contexts comma  $W$  context which is just the  $W$  context matrix staged twice back to back. So, this was my  $W$  matrix, this is my 2 hot vector because I have 2 inputs now right. So, my vocabulary size is 3. So, the first one hot vector followed by the next one hot vector and now I am going to repeat  $W W$ , now what is the product of this? Is the sum of the 2 columns that you see highlighted right and exactly that is what I have written here.

So, if I do it this way then I can just do this very expensive matrix multiplication and to do a something very trivial which is just taking the sum of 2 columns right. But at least you get the operation and I will just on this next slide or something I will tell you an obvious simple way of doing that. So, I just get the sum of the 2 columns. So, that is the input to my network. If I had  $k$  words as input if I had my window size 4 what would it be? I would have these 4 copies of  $W$  context I will have these 4 one hot vectors and it will just give me the sum of those 4 columns that is going to be the input and the rest of the story remains the same right. Once you have this edge the rest of it from there remains the same and this is the formula for  $h$  in general.

In the special case it was just the  $i$  th column, in the general case is the sum of all the columns that are there in your input.

(Refer Slide Time: 31:35)

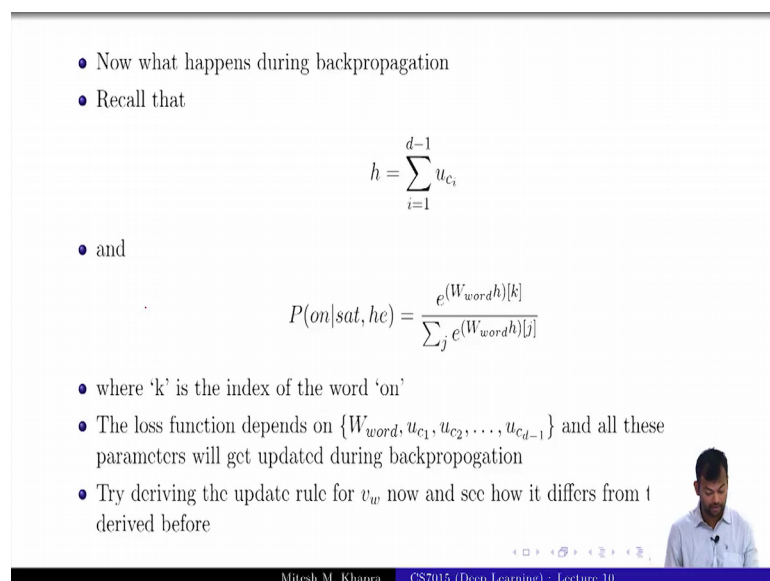


- Of course in practice we will not do this expensive matrix multiplication
- If 'he' is the  $i^{th}$  word in the vocabulary and *sat* is the  $j^{th}$  word then we will simply access columns  $\mathbf{W}_{\text{context}}[:, i]$  and  $\mathbf{W}_{\text{context}}[:, j]$  and add them

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

Now, in practice of course, this is a very mate expensive matrix multiplication, it is stupid to do it that way; what you will do is you will just slice of those columns from  $\mathbf{W}_{\text{context}}$  right and then just add them up. So, you do not really need to do that stupid mate matrix multiplication because you know that the matrix multiplication is essentially just selecting these columns and adding them. So, just select those columns and add them up. So, you do not do that bad matrix multiplication operation that fine.

(Refer Slide Time: 32:01)



- Now what happens during backpropagation
- Recall that

$$h = \sum_{i=1}^{d-1} u_{c_i}$$

- and

$$P(\text{on} | \text{sat}, \text{he}) = \frac{e^{(W_{\text{word}h})[k]}}{\sum_j e^{(W_{\text{word}h})[j]}}$$

- where 'k' is the index of the word 'on'
- The loss function depends on  $\{W_{\text{word}}, u_{c_1}, u_{c_2}, \dots, u_{c_{d-1}}\}$  and all these parameters will get updated during backpropagation
- Try deriving the update rule for  $v_w$  now and see how it differs from  $t$  derived before

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10

Now, what happens during back propagation in this case in the generic case? The ordering does not matter is what you have seen yes it does not matter yeah there is some assumption of the model. So, it is that is why the name of bag of words, you are not relying on the sequence. So, this comes from NLP that if you rely the sequence you call it sequence, if you just going to take the words in the sequence, you just call it a bag of words. Because once you put them in a bag there is no ordering there right that is why the word name bag of words.

So, and again  $P$  on given sat is given by this softmax formula, now tell me during back propagation and if you give me a right answer to this I really feel happy that you have understood everything right from the beginning of the course so no pressure. So, which are the parameters which are going to get updated during back propagation, which are the 2 large matrices  $W$  word and  $W$  context? So obviously, the answer is not  $W$  word and  $W$  context otherwise I would not have asked you. The answer is some dash of these two some subset of these two, which subset let us start with  $W$  context which is the input do we are we going to update the entire  $W$  context, did it participate the entire  $W$  context participate in the computation, only those columns corresponding to the words. So, only those parameters will get updated right.

So, how many columns will get updated?  $D$  columns right.  $W$  word till all the columns of  $W$  word participate in the computation how many of you say yes, how many if you say no? The others do not care; can you just focus on this circle did all the columns of  $W$  word participate in the computation? You see the summation at the bottom, it is over all the columns of  $W$  word all of them participated. So, the parameters which will get updated are  $W$  word and all the columns of the input words and same back propagation will work again is that fine.

So, remember that and this is I cannot emphasize it enough; whatever I have explained is only for an intuitive explanation, you will never ever do this matrix multiplication right and that is why what you are going to do is you are just going to select those columns add them up and feed them. And the network will take care or rather you will take care that you update those parameters only and you do not update the entire  $W$  context matrix because anyways there is no gradients flowing to the other components. So, remember that in the practical implementation of  $W$  of word 2 vec do not search for this matrix



multiplication at the input, or if you are writing the code on your own which is highly unlikely do not do it that way right.

So, if you whatever code that you look at did not have this complex matrix multiplication typically. They will just pick up the columns and add them and feed them right and I think the tensor flow way of doing is you have this word embeddings matrix and you can slice columns from there and so, this is fine. So, everyone understands this so far. Now what are these problems with this, why is this not as simple in some sense as the MNIST data set. Again focus on the circle, this softmax computation is a very expensive operation right you have a v cross k sized matrix somewhere there, and unlike at the input here you will have to do this matrix multiplication right.

So, we have a v cross k matrix multiplied by a k cross 1 vector, and there is no simplification of this you have to do this multiplication what are the sizes of v that we saw in practice? 50 k, 100 k and if you had Googled 13 million or something right. So, this is not feasible we cannot do this expensive matrix multiplication right.

(Refer Slide Time: 35:56)

Some problems:

- Notice that the softmax function at the output is computationally very expensive

$$\hat{y}_w = \frac{\exp(u_c \cdot v_w)}{\sum_{w' \in V} \exp(u_c \cdot v_{w'})}$$

- The denominator requires a summation over all words in the vocabulary

NPTEL Mitesh M. Khurana CS7015 (Deep Learning) : Lecture 10

So, although all of this works very fine we need to think of ways to simplify this softmax computation, where the denominator requires the summation over all the words in the vocabulary. So, you have to do that many matrix multiplications.