Introduction to Large Language Models (LLMs)

Prof. Tanmoy Chakraborty, Prof. Soumen Chakraborti
Department of Computer Science & Engineering
Indian Institute of Technology, Delhi

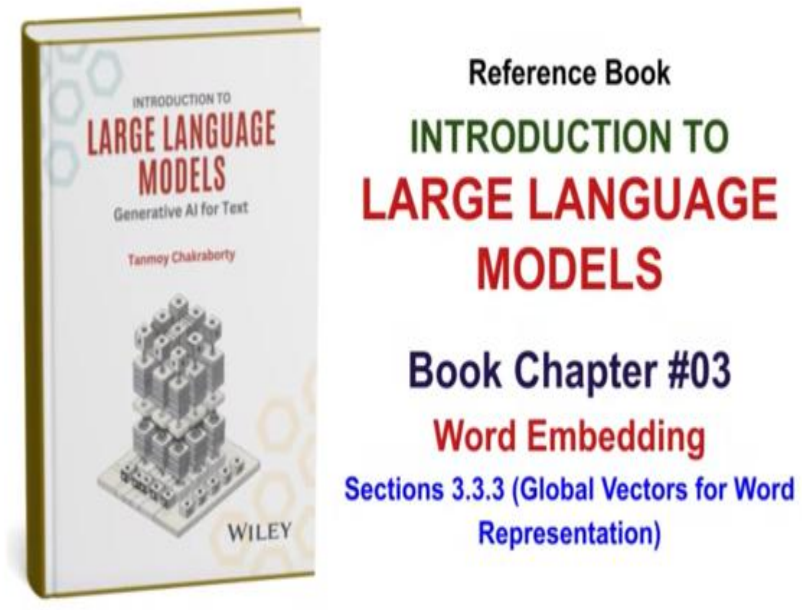
Lecture 8

Word Representation: GloVe

Hello everyone, let us get started. So, in today's class, we will continue our discussion on word representation. So in the last lecture, we started discussing the old school method of representing a word. We looked at count-based methods, simple term context matrix. From term context matrix, we moved to TF-IDF based matrix. In fact, there are other ways to measure the co-occurrence like PMI, PPMI, point-wise mutual information, positive point-wise mutual information and so on.

So I did not cover those parts because those essentially come under statistical NLP part of it. And then we also discussed what I'm doing, specifically what to make. And we saw that how a prediction-based method like what to make addresses some of the problems of count-based methods. What are the major problems of count-based methods? Major problem is that you have a larger matrix, many entries are zero and the good part about the count-based method is that you can capture the statistics properly.

For example, the number of times two words appear together that count, it's a very statistically important count and that count you can capture properly in a count-based method. Whereas if you look at the prediction-based methods, there are also some problems we will discuss here.



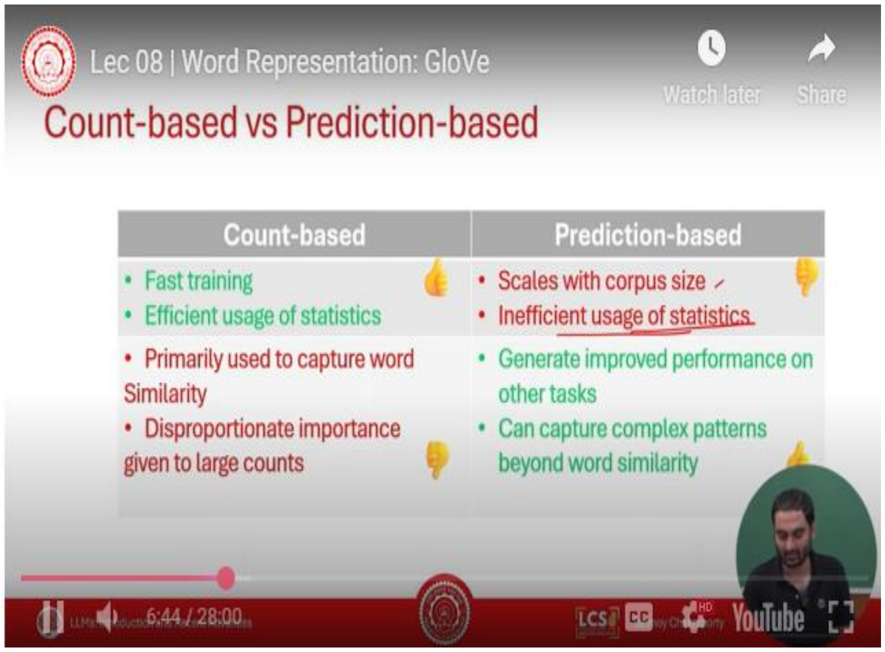
Reference Book

INTRODUCTION TO LARGE LANGUAGE MODELS

Book Chapter #03

Word Embedding

Sections 3.3.3 (Global Vectors for Word Representation)



Lec 08 | Word Representation: GloVe

Count-based vs Prediction-based

Count-based	Prediction-based
• Fast training • Efficient usage of statistics	• Scales with corpus size • Inefficient usage of statistics
• Primarily used to capture word Similarity • Disproportionate importance given to large counts	• Generate improved performance on other tasks • Can capture complex patterns beyond word similarity

So in today's class, the idea is that we will finish this chapter, we will talk about another embedding method which is called the GloVe embedding method, global vectors. And then we will see some of the important properties of these embedding methods and then we close this chapter. Okay, so let's now proceed.

We will look at the advantage of count based method and prediction based method and disadvantage of count based method and prediction based method. Count based methods include a TF-IDF based methods, PMI, PPMI based methods. Those methods are fast in training. Why? Because those methods essentially, what do you need to run these methods? You need to just scan the entire document, right? And prepare this matrix, right? So once you scan the entire document and prepare this matrix, co-occurrence matrix, you are done, right? And then you need to run TF-IDM and so on. It's a very simple computation.

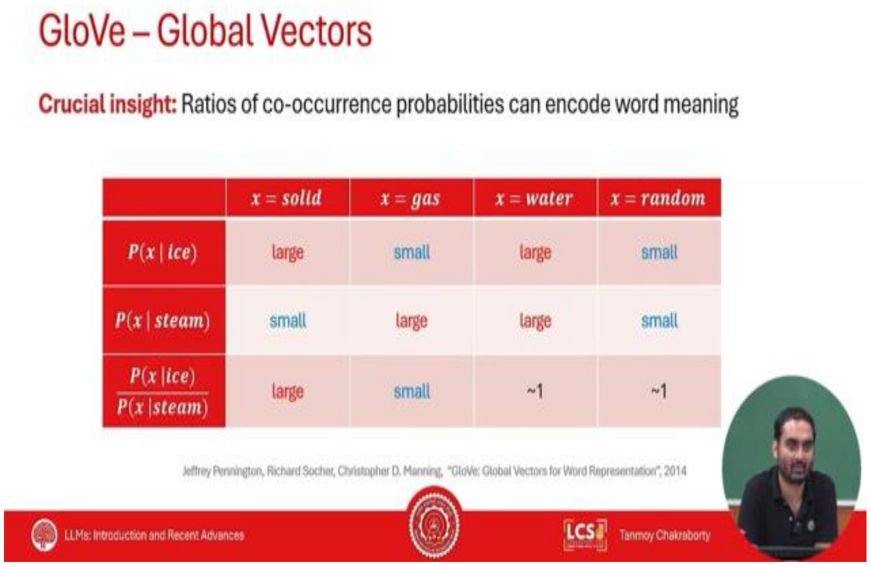
So this is fast in training and it also uses the statistics of co-occurrence very carefully. Because if two words co-appear several times, you see the numbers would be quite high and you would give more importance to those pairs of words and so on and so forth. What are the problems in count-based methods? The first problem is that it is primarily used to capture word similarity. If you see that two words co-appear with multiple other words again and again, that means that those two words are similar. But it is not a good way, it is not a good measure or it is not a good algorithm to obtain the embedding of a robot because I mentioned last day embedding is generally a dense vector, right, small size vector.

It gives disproportionate importance to highly frequent co-occurrences. If two words co-appear so many times, let's say two stop words co-appear so many times within the same window, it will not be able to kind of give them low wattage. We know that stop words co-appear so many times, so we need some ways to ignore those co-occurrences but count based methods don't do that. If you look at prediction based methods like what to wait, the bad part about these methods are as follows. It scales with corpus size.

What does it mean? Let's say if you have a fixed set of vocabulary. Number of vocabulary words is the same, fixed, right? And you keep on increasing the size of the documents, right? Without adding additional vocabulary words. So in case of count-based approach, the metrics will not change. Because count-based approach, this is basically vocabulary words times vocabulary words, the matrix, right? Whereas in prediction-based approach, you need to scan through all these documents, right? All those newly added words and for every word, you need to run, for every context, every context window, you need to run this logistic regression again and again. So let's say there were 1000 watts initially, now 100 watts came in.

You have to move this context window through this 100 watts 99 times because let's say the context window size is 7 and for every context window you need to run the logistic regression again and again. So with the increasing size of the corpus, the effective time to run this the entire algorithm will increase, whereas in case of count based approach you just need to scan and that's all. Okay, and of course, the other problem is that it doesn't consider the statistics properly because you know it is not based on the fact that how many times two words co-appear right. Therefore it is not that efficient in capturing the statistics. What are the plus points of prediction-based methods? The first plus point is that it generates improved performance on other tasks right let us say even you are interested in sentiment analysis or you are interested in right let us say name entity recognition.

You can use these embeddings and you can feed those embeddings to a task and that can be useful right. Given the fact that this is a dense vector right it is less likely that the model will over feed right. Sometimes it also captures complex patterns. We have seen last day this analogy test right, king minus man, man plus woman equals to queen right. So you know king, man, woman these words have complex patterns, right? And these complex patterns are so smoothly integrated using linear operations, right? That's very interesting property of prediction based approach which you cannot do with a count based approach, right?



GloVe – Global Vectors

Crucial insight: Ratios of co-occurrence probabilities can encode word meaning

	$x = solid$	$x = gas$	$x = water$	$x = random$
$P(x \mid ice)$	large	small	large	small
$P(x \mid steam)$	small	large	large	small
$\frac{P(x \mid ice)}{P(x \mid steam)}$	large	small	~1	~1

Jeffrey Pennington, Richard Socher, Christopher D. Manning, "GloVe: Global Vectors for Word Representation", 2014

LLMs: Introduction and Recent Advances

LCS Tanmoy Chakraborty

So, let's try to utilize the best part of both the words.

Let's try to utilize the best part of count-based approach and the best part of prediction-based approach. And that's the idea behind GloVe. This was proposed in 2014, right after the year when the Word2Vec was proposed. This was by Chris Manning's group, his postdoc, Jeff Pennington, and the PhD, that time's PhD, Richard Saucer. So let's look at the idea here.

The idea here is, we will again build this co-occurrence matrix, like count-based approach. Words cross words, and we have this window size, a specific window, and if two words co-appear multiple times within different windows, you can just increase the count. But here, we will not use the raw count. We'll use something else. Okay, let's see, let's assume that we are interested in a concept of thermodynamics We are understanding that we are trying to understand different words related to thermodynamics, right And let's say the words are something like ice and steam, right Let's look at these two words, ice and steam, okay so, what we will see? We will see what are the other words which co-appear within the context of ice and steam.

And how do we get this information from this co-occurrence matrix? So we have all those tokens which are present in the vocabulary. All those tokens will essentially act as context. And we'll see how many of them essentially appeared with these words in the context. And for this example, let's assume that there are four context words that we are interested in. solid gas water and let's say some random word like fashion right, so if ice is the target word, it is highly likely that, the word solid will appear as a context word right, so you see that this count.

So this count indicates that if ice is the target word what is the probability that the contest word will be solid. So this will be high if ice is a target word the contest word gas will be less likely to So this should be small, right? The word water may appear with ice multiple times. So this can also be high, right? Whereas the word fashion, which is a random word, it will not appear as a context word, right? So this will be small. Clear? Now, let's look at another target word, steam. So solid being the context word of steam will be less.

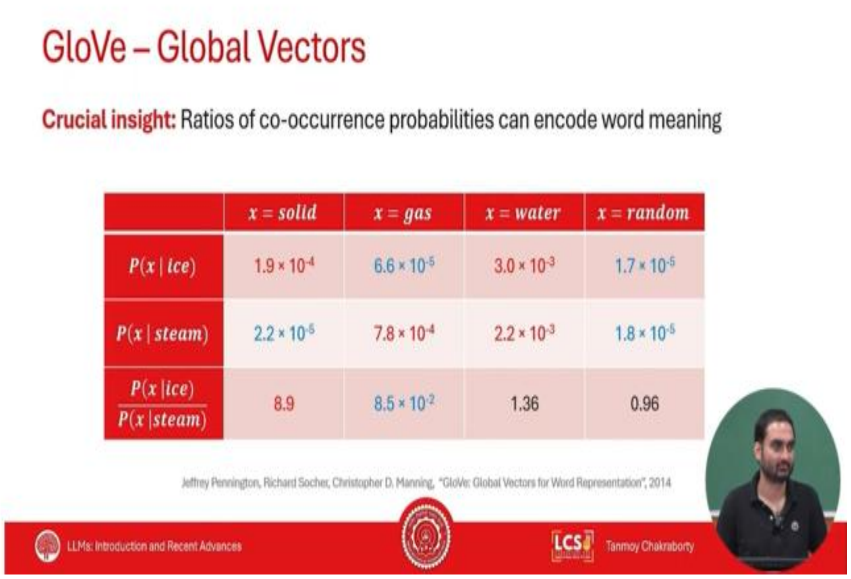
Gas with the context word of steam will be high. Right? Water as a context word of steam will be high. Right? And fashion as a context word of steam will be small. Right? Now what we do, now these context words are called pivot word.

They are pivot. How many such pivots are there? Vocabulary number. The size of the vocabulary. Right? Then what we do, we take the ratio of these two. this by this, this by this, this by this, this by this, okay. So if we take the ratio of large and small, it will produce a large value.

If we take a ratio of small and large, it will produce a small value. Large and large number tends to 1, right. And small and small also the number tends to 1 ratio. These numbers are very important. So now let's look at these numbers.

So using these numbers, let's say if you look at the last row, you can differentiate relevant context from the non-relevant context. You can differentiate relevant context words like solid and gas from the non-relevant context words like water and random because based on these numbers. The numbers tend to 1, that means that context word is not relevant to differentiate ice and steam. If a word is non-relevant, it may be completely non-relevant like fashion or it may be so close that it appears with both ice and steam like water. So water is not a good context word to differentiate ice and steam.

Clear? So this ratio tells you a lot of things. So it differentiates important context words from not so important context words. It also differentiates which context word is important for ICE and which context word is important for steam.



GloVe – Global Vectors

Crucial insight: Ratios of co-occurrence probabilities can encode word meaning

	$x = solid$	$x = gas$	$x = water$	$x = random$
$P(x \mid ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(x \mid steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$\frac{P(x \mid ice)}{P(x \mid steam)}$	8.9	8.5×10^{-2}	1.36	0.96

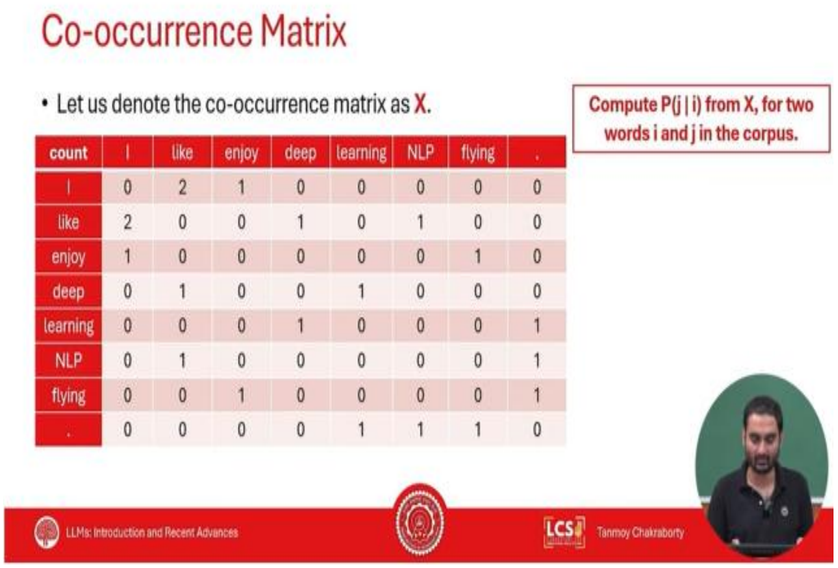
Jeffrey Pennington, Richard Socher, Christopher D. Manning, "GloVe: Global Vectors for Word Representation", 2014

LLMs: Introduction and Recent Advances LCS Tanmoy Chakraborty

So let's look at the number here. So I just gave an intuition here.

Let's look at some of the numbers. and so these are some random numbers and this is the ratio. You see this is very high, this is very low and these numbers are quite same as 1. I will utilize this philosophy to derive the GloVe embedding objective.

This is just an example.



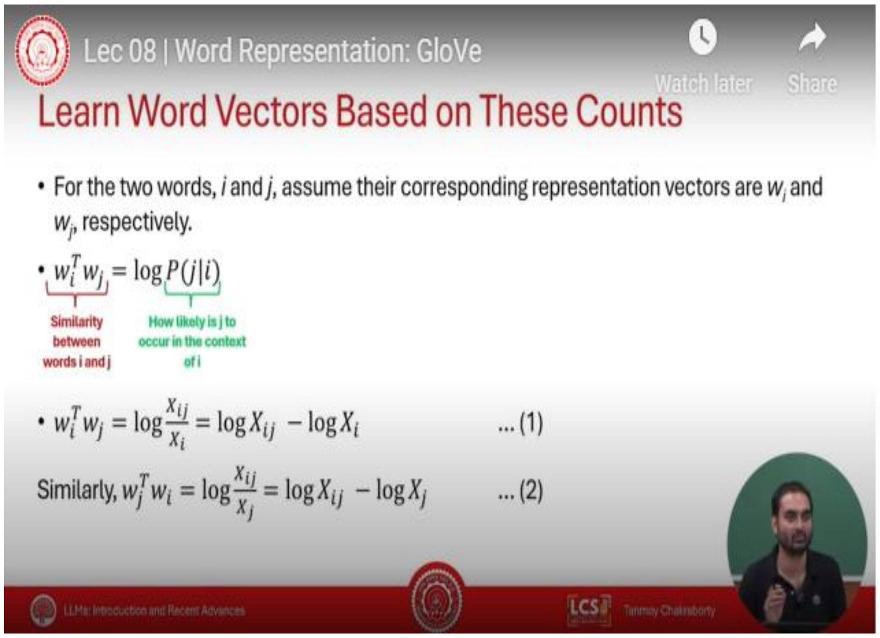
Co-occurrence Matrix

- Let us denote the co-occurrence matrix as **X**.

Compute P(j | i) from X, for two words i and j in the corpus.

count	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

LLMs: Introduction and Recent Advances LCS Tanmoy Chakraborty

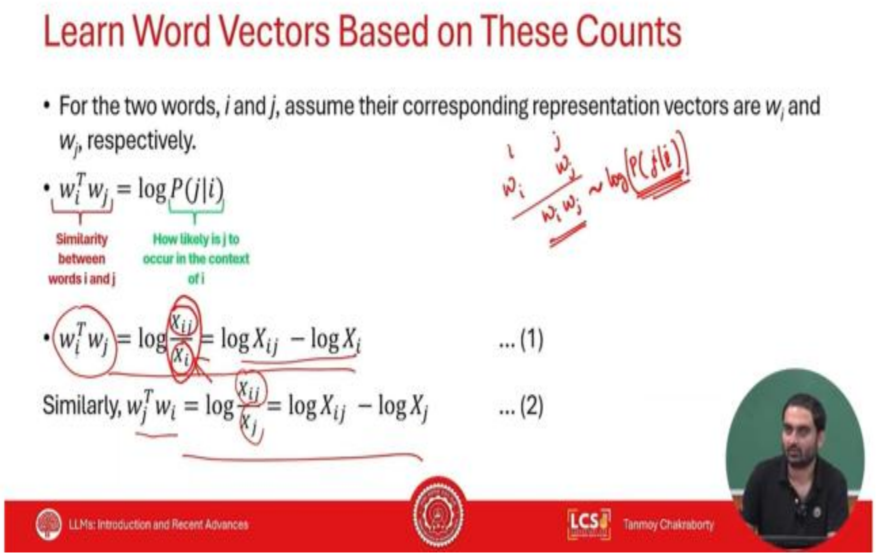

Lec 08 | Word Representation: GloVe
Watch later
Share

Learn Word Vectors Based on These Counts

- For the two words, *i* and *j*, assume their corresponding representation vectors are w_i and w_j, respectively.
- $\underbrace{w_i^T w_j}_{\text{Similarity between words i and j}} = \log \underbrace{P(j|i)}_{\text{How likely is j to occur in the context of i}}$
- $w_i^T w_j = \log \frac{X_{ij}}{X_i} = \log X_{ij} - \log X_i$ ... (1)

Similarly, $w_j^T w_i = \log \frac{X_{ij}}{X_j} = \log X_{ij} - \log X_j$ ... (2)

LLMs: Introduction and Recent Advances
LCS
Tanmoy Chakraborty



Learn Word Vectors Based on These Counts

- For the two words, *i* and *j*, assume their corresponding representation vectors are w_i and w_j, respectively.
- $\underbrace{w_i^T w_j}_{\text{Similarity between words i and j}} = \log \underbrace{P(j|i)}_{\text{How likely is j to occur in the context of i}}$
- $w_i^T w_j = \log \frac{X_{ij}}{X_i} = \log X_{ij} - \log X_i$ ... (1)

Similarly, $w_j^T w_i = \log \frac{X_{ij}}{X_j} = \log X_{ij} - \log X_j$ ... (2)

$\overset{i}{w_i} \quad \overset{j}{w_j}$

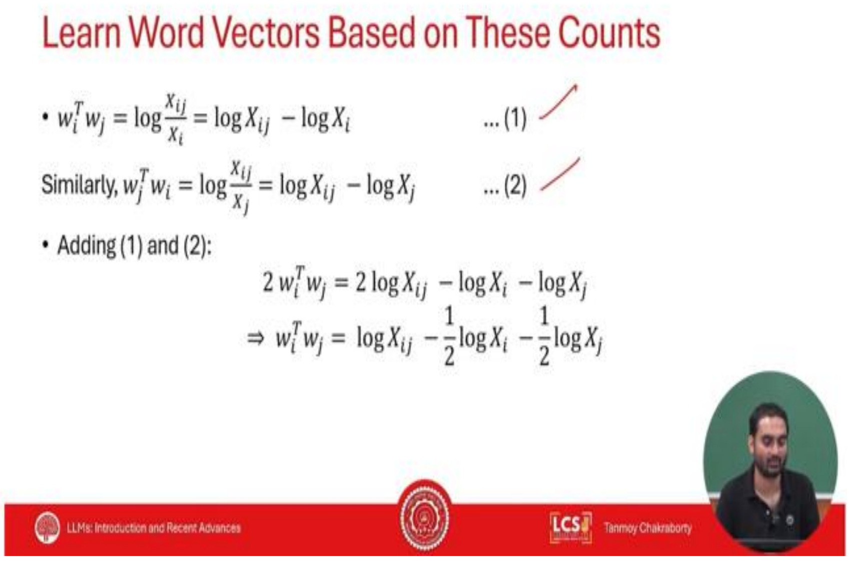
$w_i w_j \sim \log(P(j|i))$

Let us say this is the co-occurrence matrix. You can easily compute P of j given I; these are I's and these are J's. I's are target words and J's are context words and so on and so forth. Now, let's see how we can model this, how you can utilize this. Given a pair of words I and J and their corresponding embeddings, WI and WJ, that we want to learn.

How do we learn this? We say that the similarity between these two embeddings, which is captured using the dot product wi dot product wj, this should be modeled using the probability that I just measured. So this probability I just measured from the count-based approach. I try to model this dot product similarity using this probability. instead of taking the log probability, I take a log probability.

Clear? Okay. So this is asymmetric by the way because when J appears as a context word and I appears as a target word, the ratio will be different from when J appears as a target word and I appears as a context word. So asymmetric. So what I do here? WI WJ, this is the dot product, right? And what is XIJ? XIJ is a count of number of times I and J co-appear, right? What is XI? XI is the count of the word I. So this is essentially the probability. So given that WI is the target word, how many times WI and WJ co-appears? This is the probability.

I want to come up with such two vectors wi and wj which kind of mimics this count based approach. We can unfold it further and this is going to be my formula. Similarly, if j is the target word and i is the context word, you have wj times wi and how do we model this xij? The numerator will remain the same and the denominator is going to be xj. Okay the same formula. Okay but the left hand part is same dot product.



Learn Word Vectors Based on These Counts

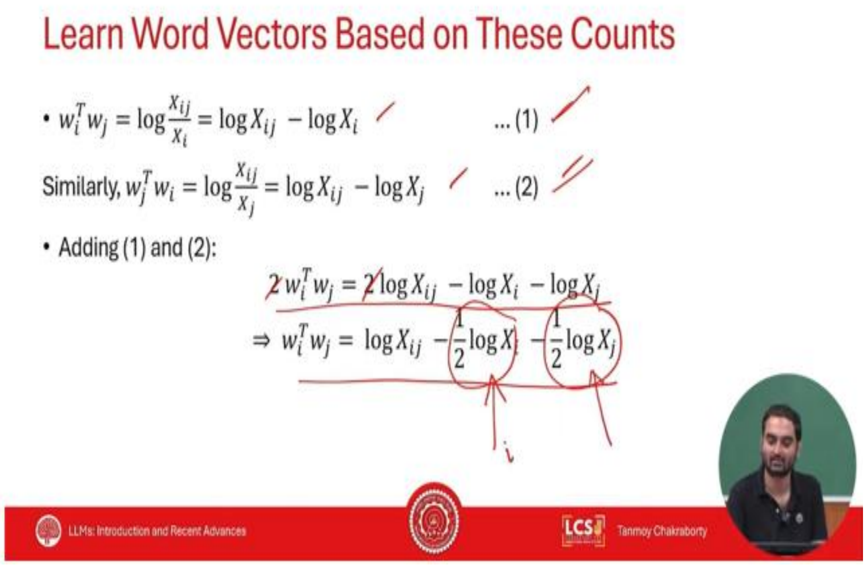
- $w_i^T w_j = \log \frac{X_{ij}}{X_i} = \log X_{ij} - \log X_i$ ... (1)

Similarly, $w_j^T w_i = \log \frac{X_{ij}}{X_j} = \log X_{ij} - \log X_j$ ... (2)

- Adding (1) and (2):

$$2\, w_i^T w_j = 2 \log X_{ij} - \log X_i - \log X_j$$
$$\Rightarrow w_i^T w_j = \log X_{ij} - \frac{1}{2}\log X_i - \frac{1}{2}\log X_j$$

LLMs: Introduction and Recent Advances
LCS
Tanmoy Chakraborty



Learn Word Vectors Based on These Counts

- $w_i^T w_j = \log \frac{X_{ij}}{X_i} = \log X_{ij} - \log X_i$ ... (1)

Similarly, $w_j^T w_i = \log \frac{X_{ij}}{X_j} = \log X_{ij} - \log X_j$ ... (2)

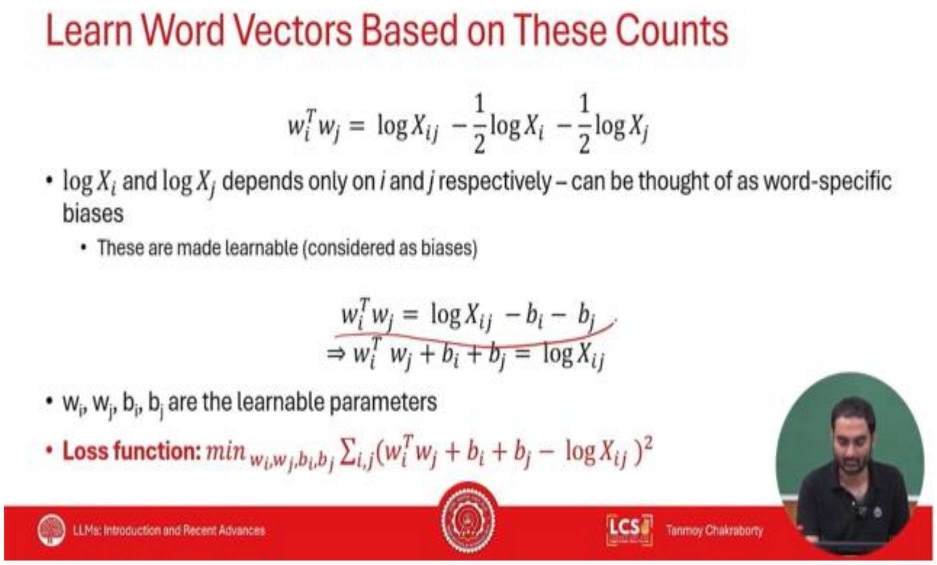
- Adding (1) and (2):

$$\cancel{2} w_i^T w_j = \cancel{2} \log X_{ij} - \log X_i - \log X_j$$

$$\Rightarrow w_i^T w_j = \log X_{ij} - \frac{1}{2}\log X_i - \frac{1}{2}\log X_j$$

So these are the two formulas. So what is xij? xij is the count of i and j appearing together and what is xi? xi is the count of i meaning the sum of this row values. So these are two formulas now. What we do? We add them. If we add them, this is going to be the resultant formula, right. We cancel this two and we move this two here. Now this is my resultant formula, w ij equals to log xij minus half log xi minus half log xj, okay. Now if you look at this term and this term carefully, these are the terms which are not dependent on the co-occurrence. These two terms are dependent on the word, specific word. Right? So I can say that, you know, I can basically use these two components.

I can use these two components as basically a property of that word. Right? And when I model this using neural network, I will assume that this is a biased term or some sort of term which is associated with the word I. And I will again consider this as a biased term which is associated with the term j.



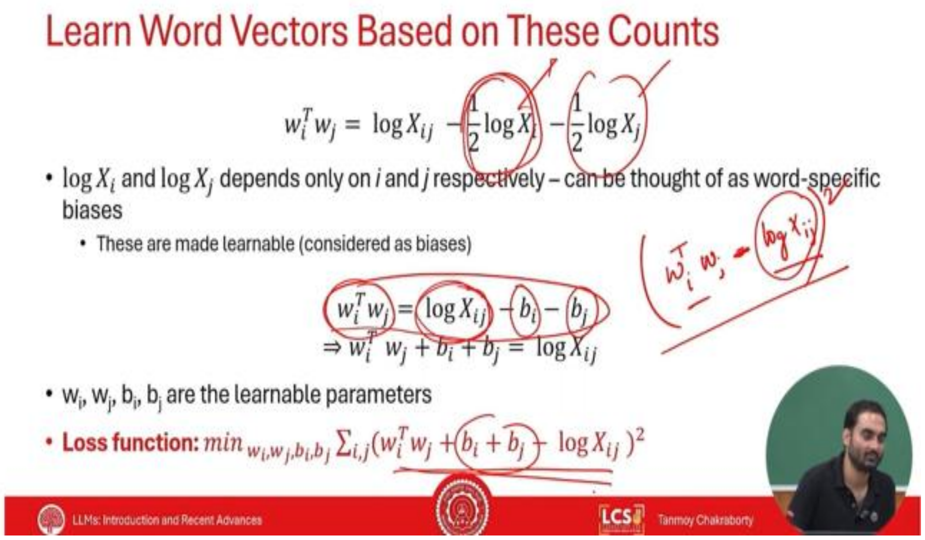
Learn Word Vectors Based on These Counts

$$w_i^T w_j = \log X_{ij} - \frac{1}{2}\log X_i - \frac{1}{2}\log X_j$$

- $\log X_i$ and $\log X_j$ depends only on *i* and *j* respectively – can be thought of as word-specific biases
  - These are made learnable (considered as biases)

$$w_i^T w_j = \log X_{ij} - b_i - b_j$$
$$\Rightarrow w_i^T w_j + b_i + b_j = \log X_{ij}$$

- w_i, w_j, b_i, b_j are the learnable parameters
- **Loss function:** $min_{w_i,w_j,b_i,b_j} \sum_{i,j}(w_i^T w_j + b_i + b_j - \log X_{ij})^2$



Learn Word Vectors Based on These Counts

$$w_i^T w_j = \log X_{ij} - \frac{1}{2}\log X_i - \frac{1}{2}\log X_j$$

- $\log X_i$ and $\log X_j$ depends only on *i* and *j* respectively – can be thought of as word-specific biases
  - These are made learnable (considered as biases)

$$w_i^T w_j = \log X_{ij} - b_i - b_j$$
$$\Rightarrow w_i^T w_j + b_i + b_j = \log X_{ij}$$

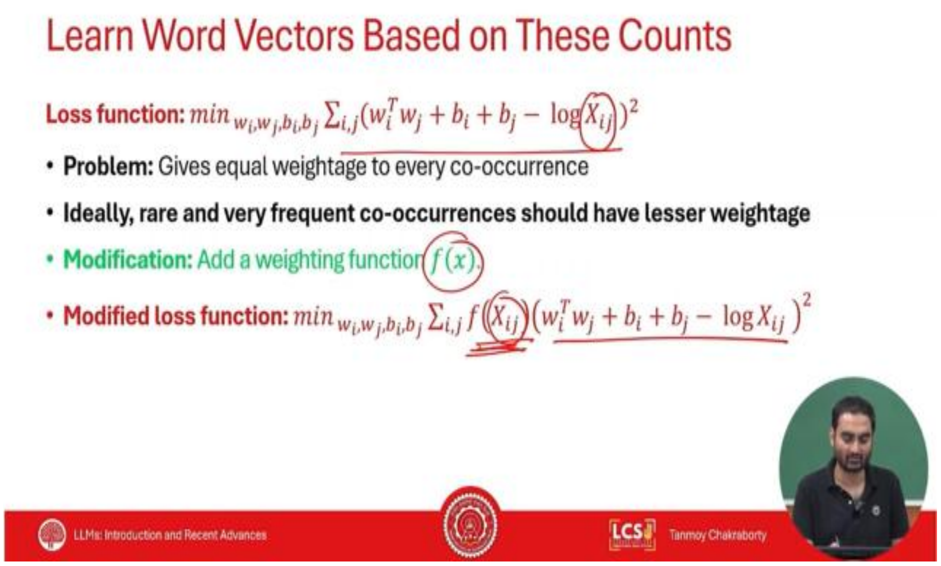
$$(w_i^T w_j - \log X_{ij})^2$$

- w_i, w_j, b_i, b_j are the learnable parameters
- **Loss function:** $min_{w_i,w_j,b_i,b_j} \sum_{i,j}(w_i^T w_j + b_i + b_j - \log X_{ij})^2$

So now this is my resultant formula. WIWJ log XIJ and what I am saying is that I will essentially create a neural network which will try to model this using a bias term BI and try to model this using another bias term BJ.

Now if you think that this is not possible, you can also simply ignore these two terms. That is also possible. So what I have at this moment I have this one and this one ok and so WIWJ and log of XIJ ok and what I do, these two things are same, what I do? I try to minimize these and this. because I will try to, what is the aim? The aim is that I'll try to come up with two such vectors, w i w j, which will mimic this. So I'll try to minimize the difference between these two, right? I'll basically take a square loss.

So this is my resultant formula. I mean, if you consider b i, b j, this will be there. Otherwise you can also ignore this part. This I want to minimize.

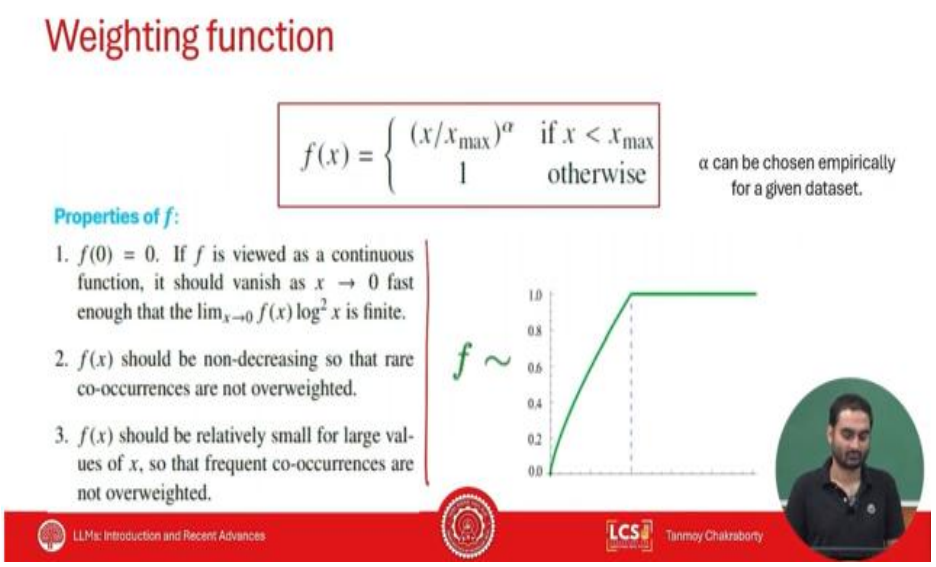


Learn Word Vectors Based on These Counts

Loss function: $\min_{w_i, w_j, b_i, b_j} \sum_{i,j} (w_i^T w_j + b_i + b_j - \log X_{ij})^2$

- **Problem:** Gives equal weightage to every co-occurrence
- **Ideally, rare and very frequent co-occurrences should have lesser weightage**
- **Modification:** Add a weighting function $f(x)$
- **Modified loss function:** $\min_{w_i, w_j, b_i, b_j} \sum_{i,j} f(X_{ij}) (w_i^T w_j + b_i + b_j - \log X_{ij})^2$

LLMs: Introduction and Recent Advances LCS Tanmoy Chakraborty



Weighting function

$$f(x) = \begin{cases} (x/x_{\max})^\alpha & \text{if } x < x_{\max} \\ 1 & \text{otherwise} \end{cases}$$

α can be chosen empirically for a given dataset.

Properties of f:

1. $f(0) = 0$. If f is viewed as a continuous function, it should vanish as $x \to 0$ fast enough that the $\lim_{x \to 0} f(x) \log^2 x$ is finite.
2. $f(x)$ should be non-decreasing so that rare co-occurrences are not overweighted.
3. $f(x)$ should be relatively small for large values of x, so that frequent co-occurrences are not overweighted.

$f \sim$

LLMs: Introduction and Recent Advances LCS Tanmoy Chakraborty

This is my square loss formula and I want to minimize.

So this is my objective function now. There is a problem here. What is the problem? The problem is it still gives disproportionate advantage to those counts which is significantly high. Right? xij will be high. So what we do to address this? I will essentially add a function fx before this.

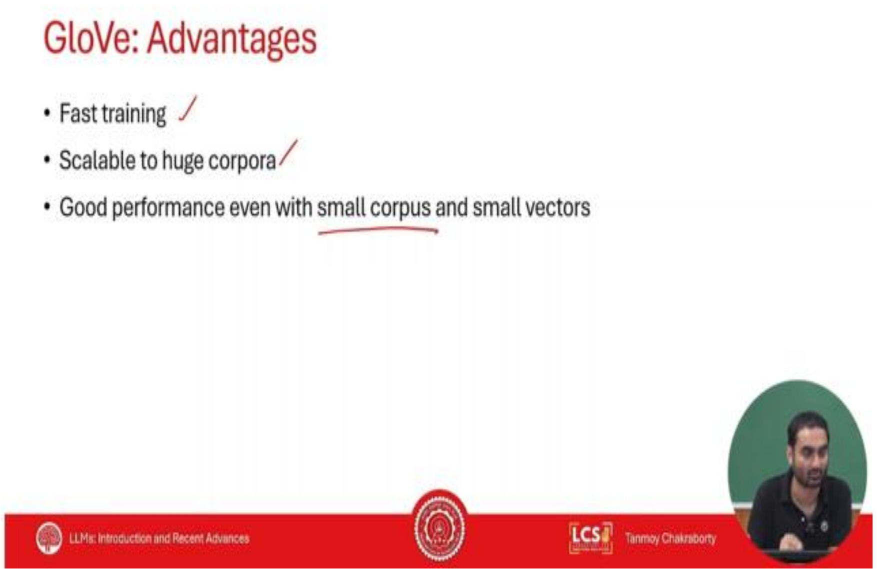
So fx is a function. which is a property of this co-occurrence xij. If the co-occurrence value is too high, I will penalize. So I will basically penalize those pairs which are, let's say, stopwatch pairs or something like that. So this is just a weight before this. And how do we choose this f? You can choose this f many ways such that these three properties hold.

So what are these three properties? It says that you should choose an f, forget about this graph okay, forget about this function. You choose an f such that f is viewed as a continuous function right and as you move towards 0 right this would be finite. property 1. Property 2 is that it should be non-decreasing.

It will never decrease with the increase of co-occurrence. And the third property says that the value of this function should be relatively small for larger value of co-occurrence. If the co-occurrence is so large you should not give proportionate weight right. So this is the formula that they came up with and they also said that if I mean you can also come up with new formula, new function that can satisfy these three properties right and this is the function that they came up with and the function looks like this. So this is x and this is f.

In our context, this is xij and this is f. So you see here that there is a cutoff here. This cutoff is called xmax. So after xmax, whatever co-occurrence you have, you have the same weight for that and the weight is 1. So it will not give additional weight or extra weight to those co-occurrences which are highly frequent.

Whereas, before this xmax part, this is almost linear.

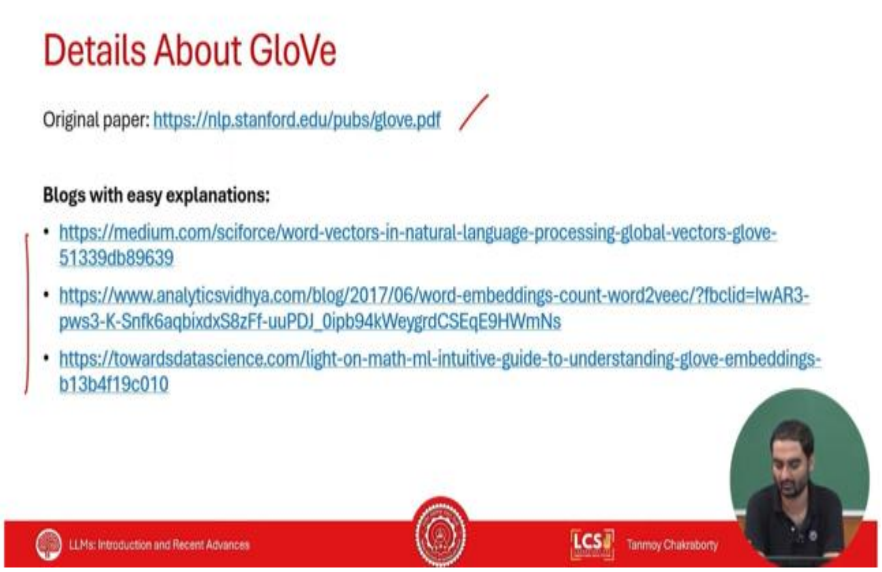


GloVe: Advantages

- Fast training
- Scalable to huge corpora
- Good performance even with small corpus and small vectors

So, this is club embedding. Now, this is my objective function. What I do now? I will minimize this and when I do the gradient descent, I will take the derivative of this with respect to wi and with respect to wj. Here also, remember similar to word2vec, here also there are two embeddings that we are learning. One is wi, other is wj, meaning one is acting as a target, other is acting as a context.

okay. So this essentially captures both the walls you know appropriately right because you have this count-based method, right, this component log of xij which comes from the count-based approach whereas this one is very similar to the prediction based approach that we have seen. Okay so this is glove embedding. This is also static embedding and they showed the efficacy of this method across multiple tasks and so on and so forth Glove embedding is fast in terms of training scalable to huge corpora and it also showed good performance when you have small corpus. This is very important right unlike what to make okay.



Details About GloVe

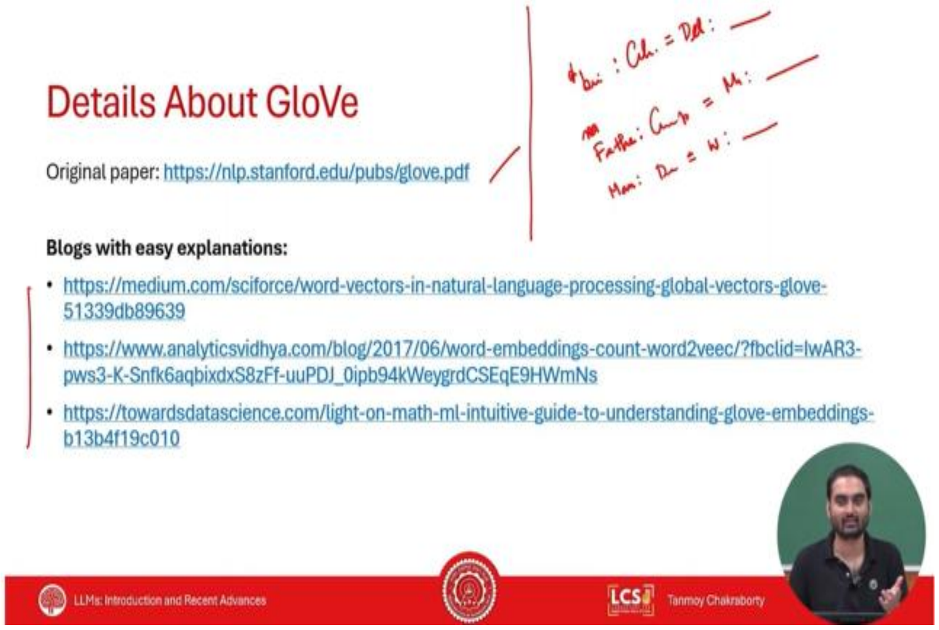
Original paper: https://nlp.stanford.edu/pubs/glove.pdf

Blogs with easy explanations:

- https://medium.com/sciforce/word-vectors-in-natural-language-processing-global-vectors-glove-51339db89639
- https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/?fbclid=IwAR3-pws3-K-Snfk6aqbixdxS8zFf-uuPDJ_0ipb94kWeygrdCSEqE9HWmNs
- https://towardsdatascience.com/light-on-math-ml-intuitive-guide-to-understanding-glove-embeddings-b13b4f19c010

LLMs: Introduction and Recent Advances | LCS | Tanmoy Chakraborty

So if you want to know more about glove this is the original paper and these are the very interesting blogs that you can go through where things are explained easily and easier way.



Dli : Ch. = Dd : ___

Father : Cmp = M : ___

Man : D = W : ___

Details About GloVe

Original paper: https://nlp.stanford.edu/pubs/glove.pdf

Blogs with easy explanations:

- https://medium.com/sciforce/word-vectors-in-natural-language-processing-global-vectors-glove-51339db89639
- https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/?fbclid=IwAR3-pws3-K-Snfk6aqbixdxS8zFf-uuPDJ_0ipb94kWeygrdCSEqE9HWmNs
- https://towardsdatascience.com/light-on-math-ml-intuitive-guide-to-understanding-glove-embeddings-b13b4f19c010

LLMs: Introduction and Recent Advances | LCS | Tanmoy Chakraborty

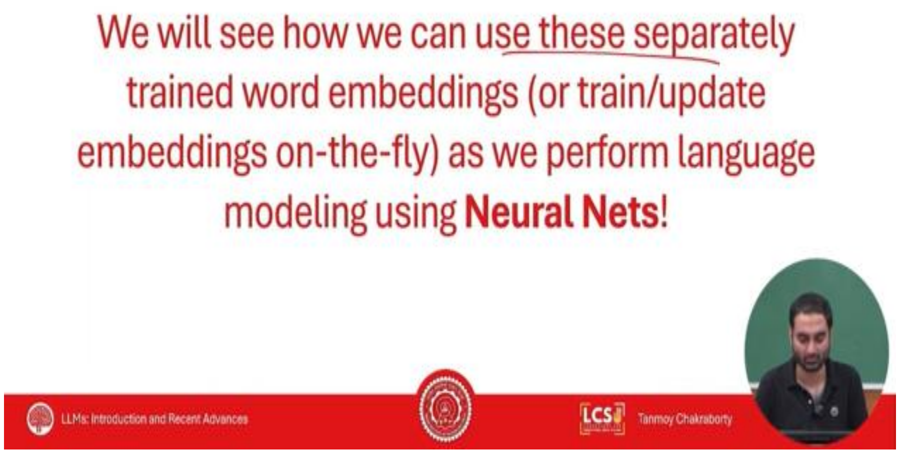
So there are very interesting properties which I am not explaining in this lecture but with this WORD embeddings right you can not only use this WORD embeddings for I mean as an input to different downstream task but you can also do very interesting studies for example What analogy test I already mentioned, if you want to understand like this, let us say Beijing is to China equals to Delhi is to Dash, you will get India, something like that. Similarly, these kind of analogies, these are linear analogies. It was also shown that these kind of methods are prone to bias, right. For example father is to a computer programmer equals to mother is to dash it will return home-maker okay Man is to doctor equals to woman is to dash, it will produce a nurse.

So these are essentially some of the biased examples. You can also do very funny thing for some of the funny experiments using what to wake or using glove. So what you do, let us say you have historical data from 1950s to 2010. Let us say you have New York Times news articles from 1950 to 2010. So what you do, for every decade, 1950 to 60, you collect all the documents and you run a glove embedding.

Again, 61 to 70, take all the documents and run glove embedding. And then if you look at all these decades corresponding vector space, you see the words, the meaning of these words are changing over time. How do you know that? You look at the neighbors. You identify one word and look at their neighbors. So look at the neighbors in 1950, 1960.

Look at the neighbors 60 to 70, 70 to 80. You see that the meaning is changing. For example, the word broadcast. Early days, the word broadcast was used to basically indicate spreading of seeds. right in cultivation, agriculture and these days broadcast is used in radios and frequency related stuff and so on and so forth.

You can do a lot of things using this kind of embedding methods.



We will see how we can use these separately trained word embeddings (or train/update embeddings on-the-fly) as we perform language modeling using **Neural Nets**!

LLMs: Introduction and Recent Advances

LCS

Tanmoy Chakraborty

In fact, these embedding methods are going to be the input for the transformer model that we will discuss in the next two class, maybe, right. So, we will see how this words embeddings are used to essentially to perform different types of neural networks in the next chapter which is going to be on neural language model, okay. Thank you.