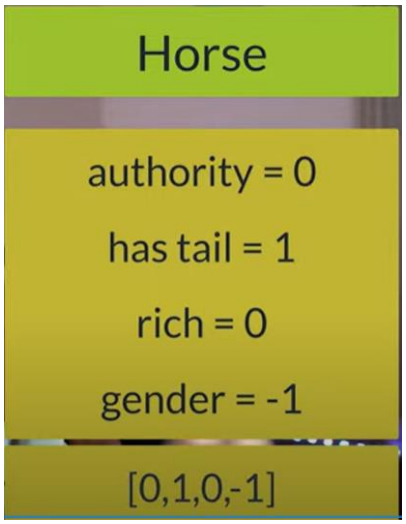
Word2Vec

Word2Vec is a technique that allows u to do mathematics with the word

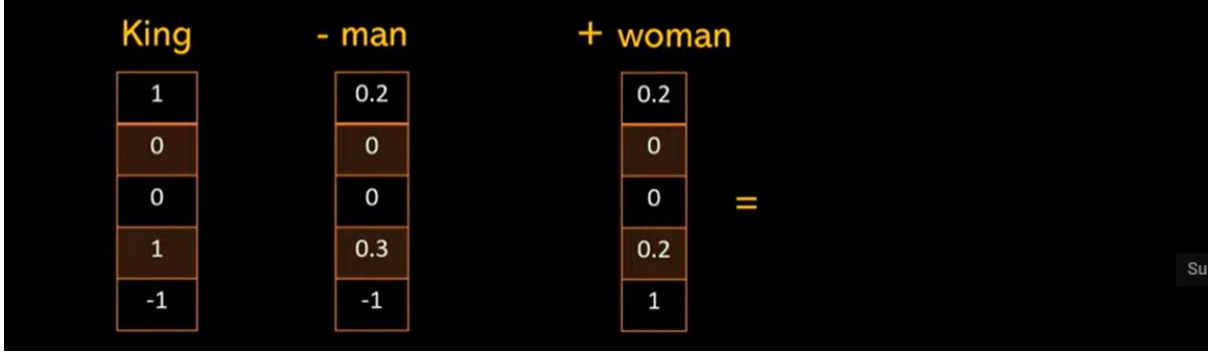


For ex: u can give this equation to computer King – Man + Woman

And computer will tell u the answer is Queen. (King – Man + Woman = Queen)

This is super cool and works really well

	battle	horse	king	man	queen	..	woman
authority	0	0.01	1	0.2	1	...	0.2
event	1	0	0	0	0	...	0
has tail?	0	1	0	0	0	...	0
rich	0	0.1	1	0.3	1	...	0.2
gender	0	1	-1	-1	1	...	1

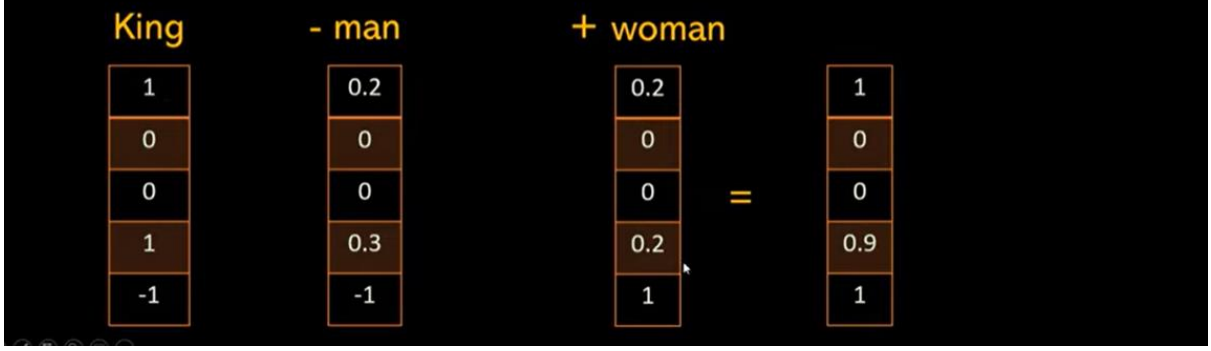


Now when we do king – man + woman

Just a simple math , u take first row 1 – 0.2 + 0.2 = 1

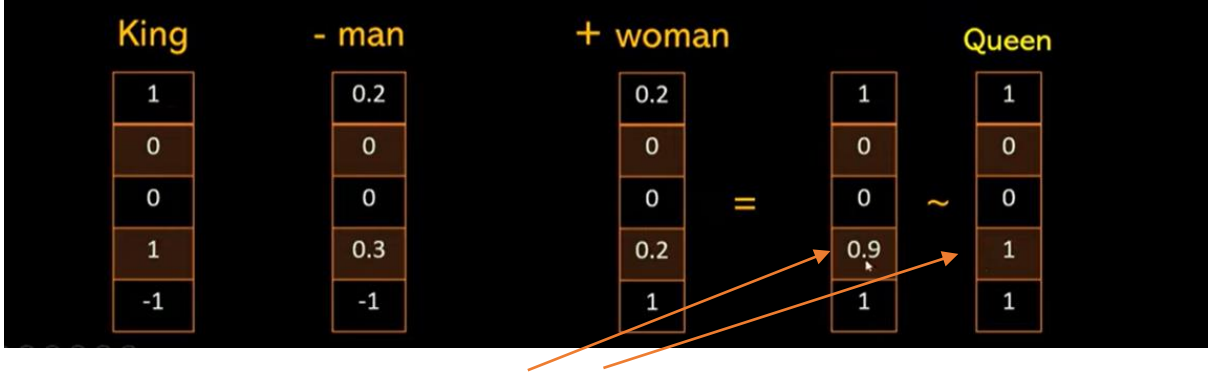
In the same way we can calculate the for all the rows

	battle	horse	king	man	queen	..	woman
authority	0	0.01	1	0.2	1	...	0.2
event	1	0	0	0	0	...	0
has tail?	0	1	0	0	0	...	0
rich	0	0.1	1	0.3	1	...	0.2
gender	0	1	-1	-1	1	...	1



And this resultant vector is similar to a vector of Queen.

	battle	horse	king	man	queen	..	woman
authority	0	0.01	1	0.2	1	...	0.2
event	1	0	0	0	0	...	0
has tail?	0	1	0	0	0	...	0
rich	0	0.1	1	0.3	1	...	0.2
gender	0	1	-1	-1	1	...	1



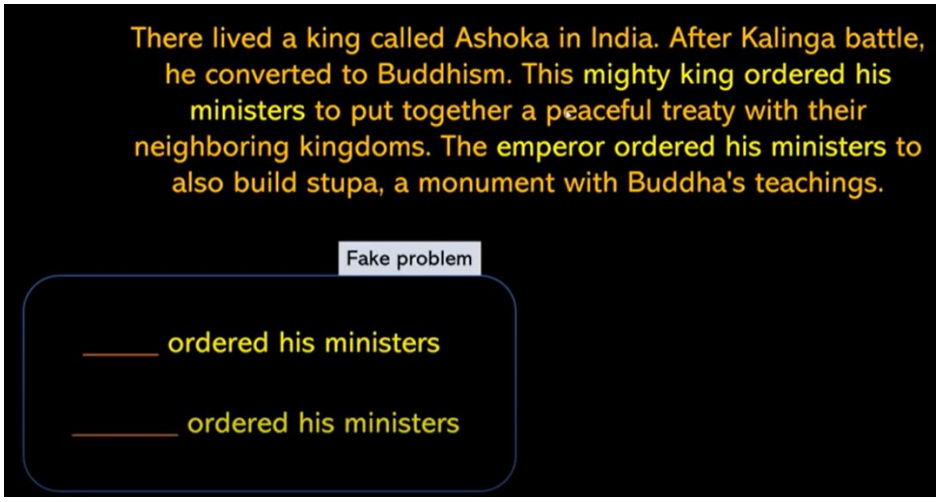
Both vectors are not exactly same but quite similar, 0.9 and 1 that's the only difference rest is same

So when u give this equation to the computer, the computer will be able to tell u that the answer is queen and that is pretty powerful

Embeddings are not hand crafted. Instead, they are learnt during neural network training

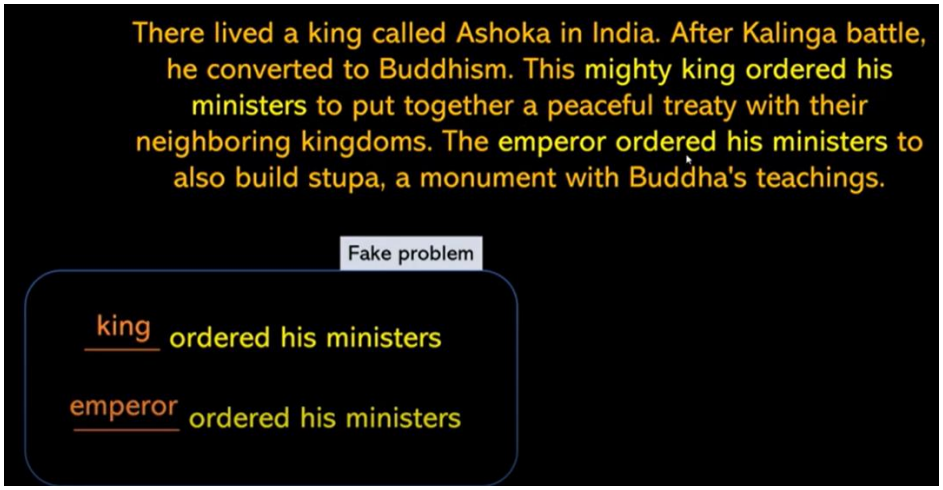
1. Take a fake problem
2. Solve it using neural network
3. You get word embeddings as a side effect

fake problem: fill in a missing word in a sentence

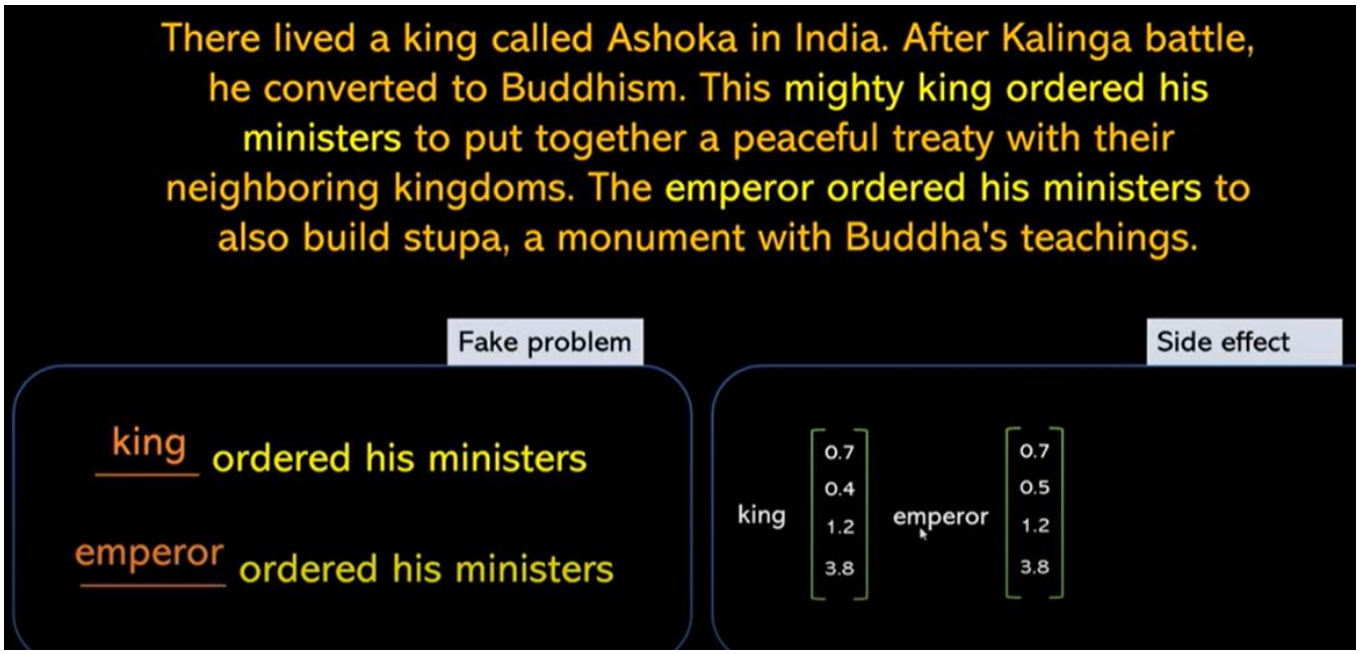


So the fake problem is complete this sentence

So based on the above text we can say the missing word can be king or emperor



And when we give the task of filling in the missing word to a computer as a side effect it will learn the vectors for king and emperor



And once we have vectors we can do math , we can say the king is almost equal to the emperor.

So see u will be able to derive the synonyms, antonyms

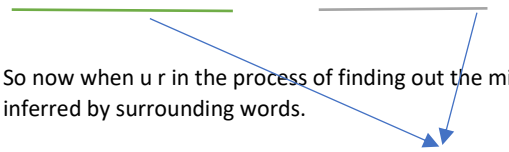
U can do maths such as King - Man + Woman = Queen

For example u have this above sentence and if I ask u to fill in the missing word

Well most likely we will say apple and walnut bcz that's food and that's healthy

Pizza is also food but that's not healthy.

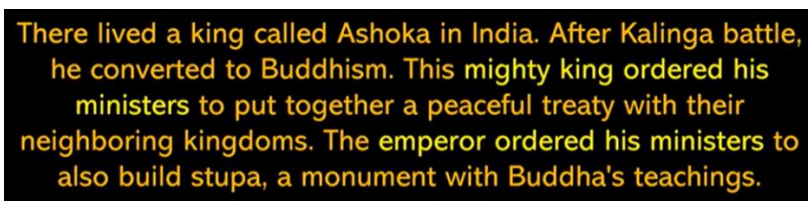
Similarly when we have this sentence the likely missing word will be rocket.



So now when u r in the process of finding out the missing words, u realize one fact that the meaning of a word can be inferred by surrounding words.

So these surrounding words are also called context.

So based on the context u can figure out what that missing word is.

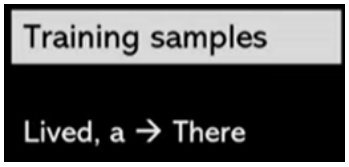


Let's take this paragraph

We will try to autocomplete missing words & auto-completing missing words is really not the area of our interest

Its our fake problem, our interest is to learn the word embeddings.

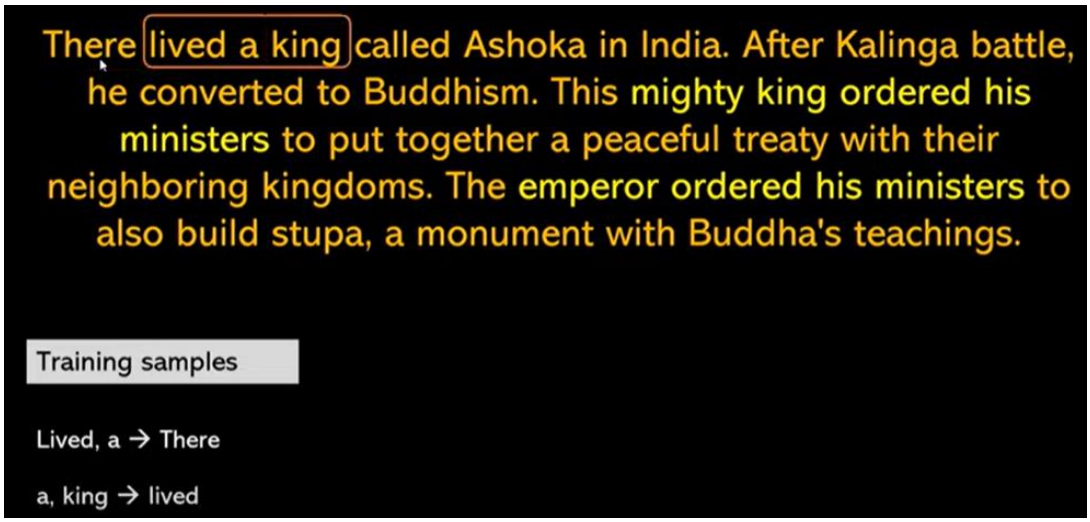
So we will parse this paragraph and we will take a window of three words

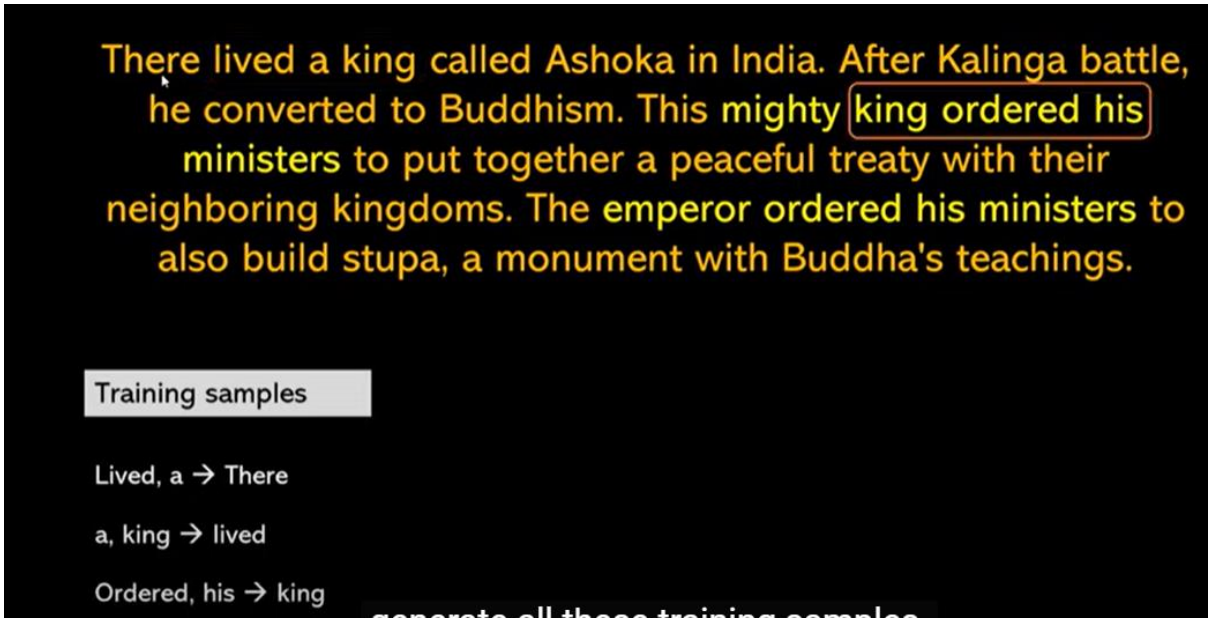


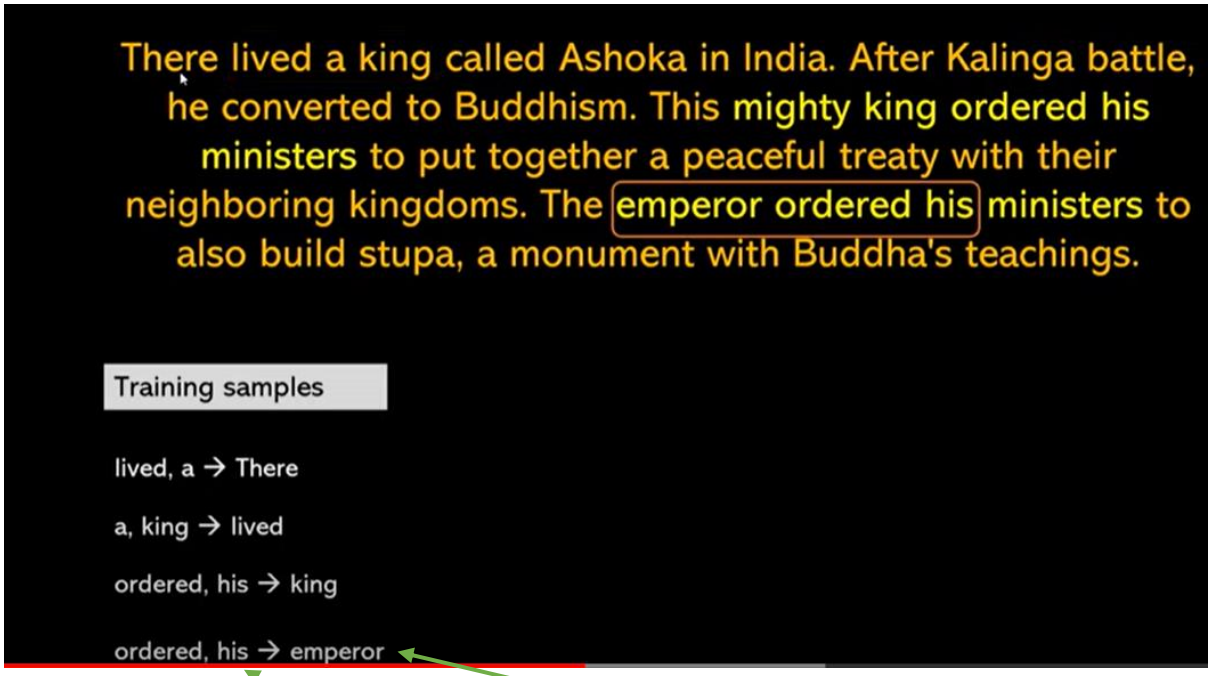
If I have a word 'lived' , and 'a' then I can predict that there is a word 'There'.

I am taking the 2nd and 3rd word and trying to predict the 1st word

So I can move that window of three words throughout the paragraph







And these are our training samples, so now this becomes our training set for neural network

The words on the left-hand side are my x, and the words on the right hand side are y.

<u>This problem is called *self-supervised* bcz all u had was this paragraph.</u>

<u>U did not have like x and y training, u just had a paragraph</u>

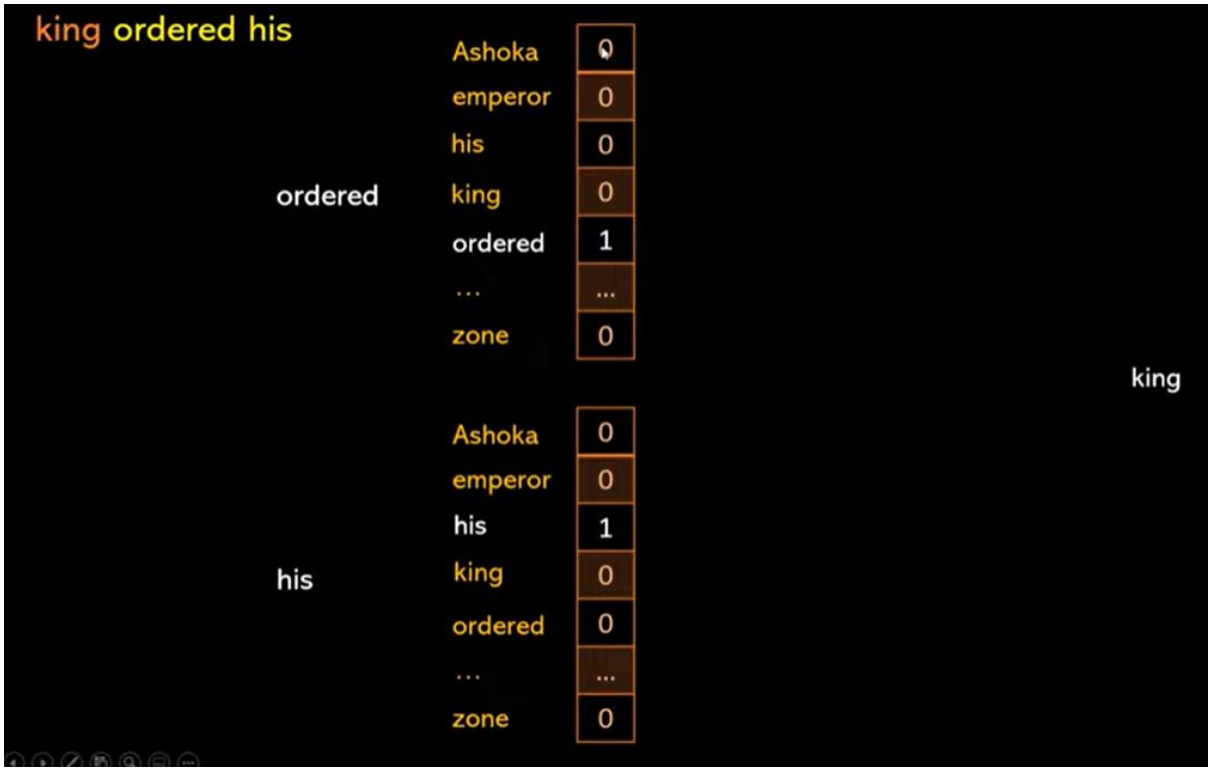
<u>From that paragraph we generated these training samples.</u>

Now lets train the neural network using these training samples.

So lets my first sample is this

So 'ordered his' is an is an input.

Based on that u want to predict the word king which is an output.



now u can build a neural network that looks like this

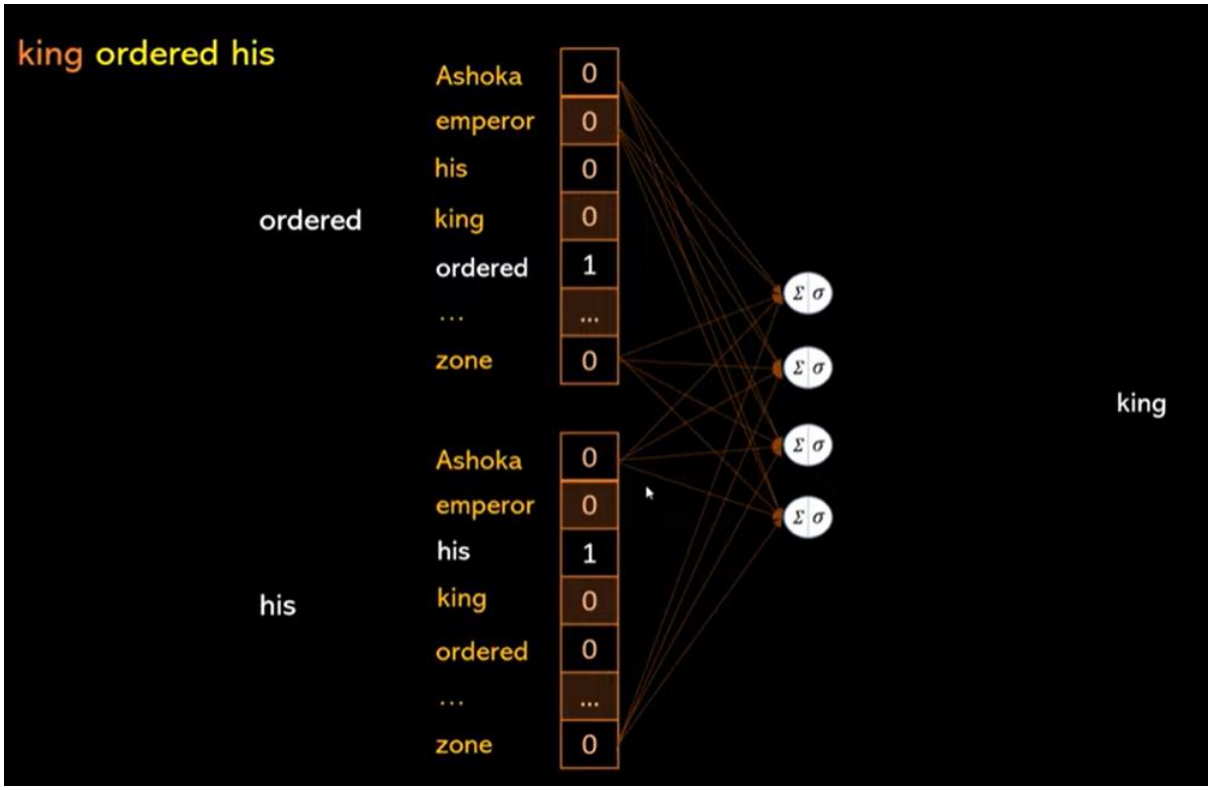
The input layer will have a one-hot encoded vector

So lets say there are 5000 words in my vocabulary.

Then there will be a vector of size 5000 and only one of them will be 1

If the word is ‘ordered’ only ‘ordered’ will be 1, remaining will be 0 & same thing for the word ‘his’.

Vocabulary means unique words in ur text corpus or in the text problem u r trying to solve



& in the hidden layer here we have put 4 neurons and these 4 neurons are the size of my embedding vector.

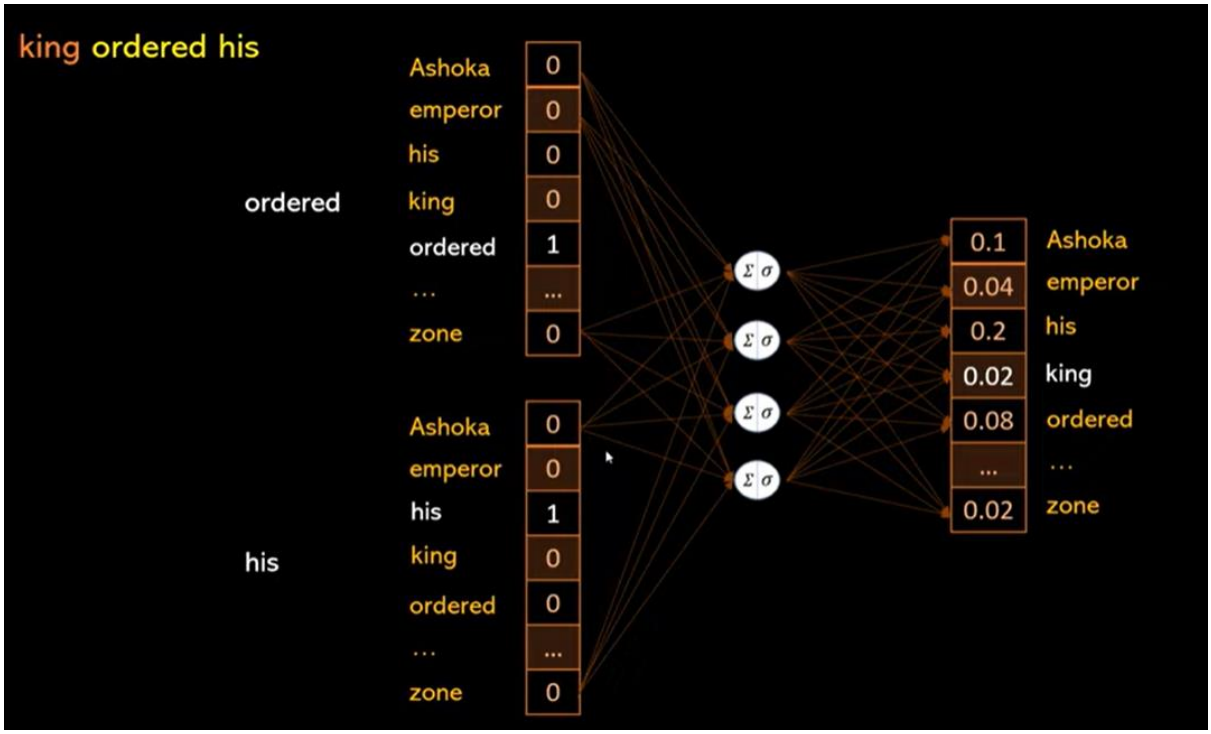
The size of embedding vector could be anything , there is no golden rule

We just selected 4, but it could be 5, 101 500 anything.

In the output layer we will have 5000 size vector.

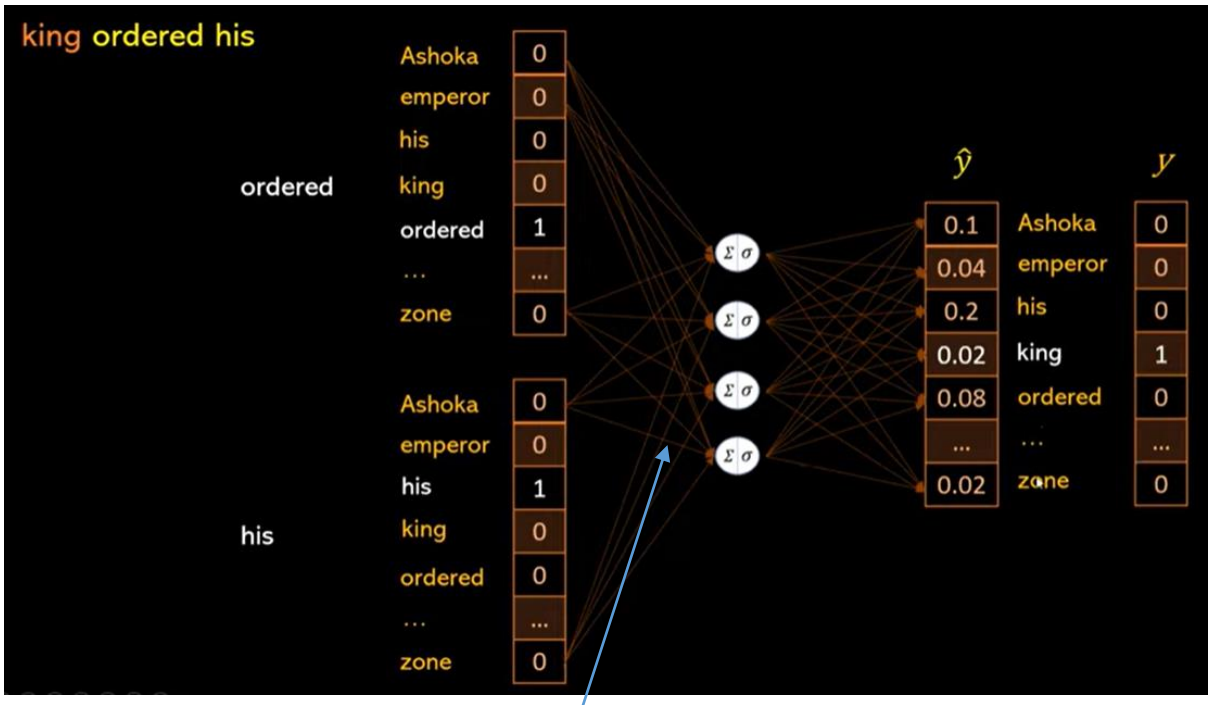
And when I feed this training sample into my neural network

This weights or edges will have random weights so using random weights , it will predict some output which will be wrong most likely.



King is the right output, so u all know how neural network works

Back propagation error works



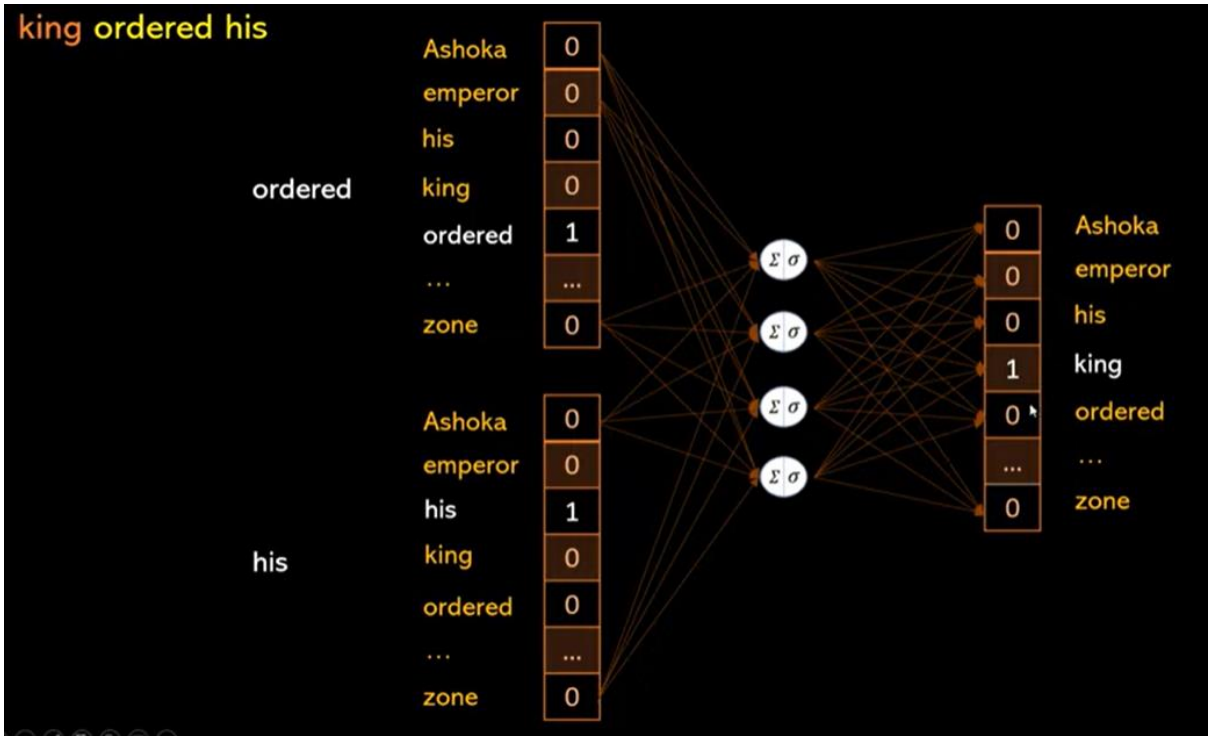
U compare ur actual output y with ur predicted output y^

U calculate loss, so loss is the difference between ur predicted output and actual output and u again back-propagate

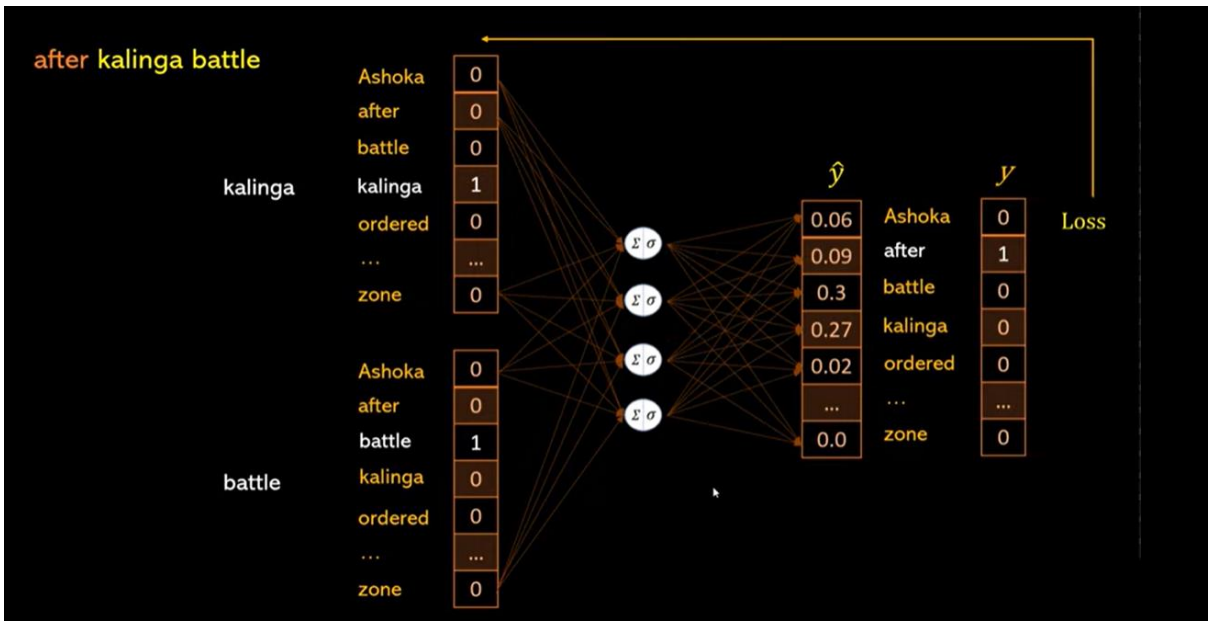
And when u back propagate u adjust all these weights

And then u take 2nd sample, 3rd sample, u take all 10000 or 1 million samples and ur goal is to train ur network in such a way that

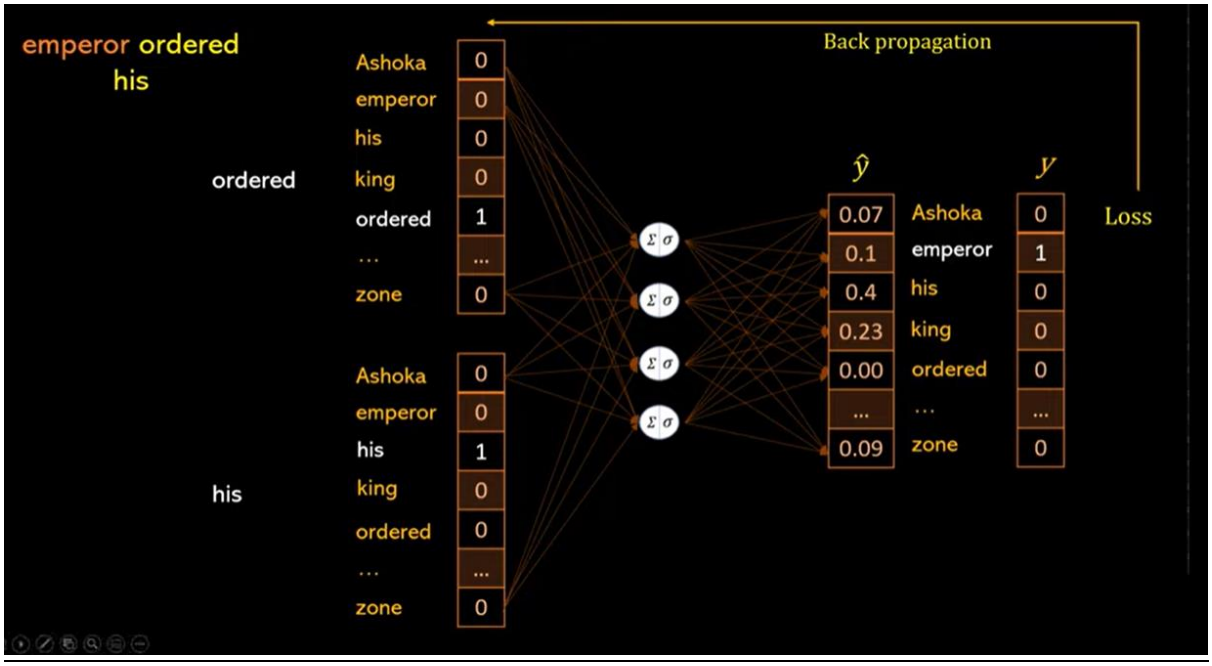
When u input ordered his the network should accurately find out that it's a king.



So u expect 1 to be here.



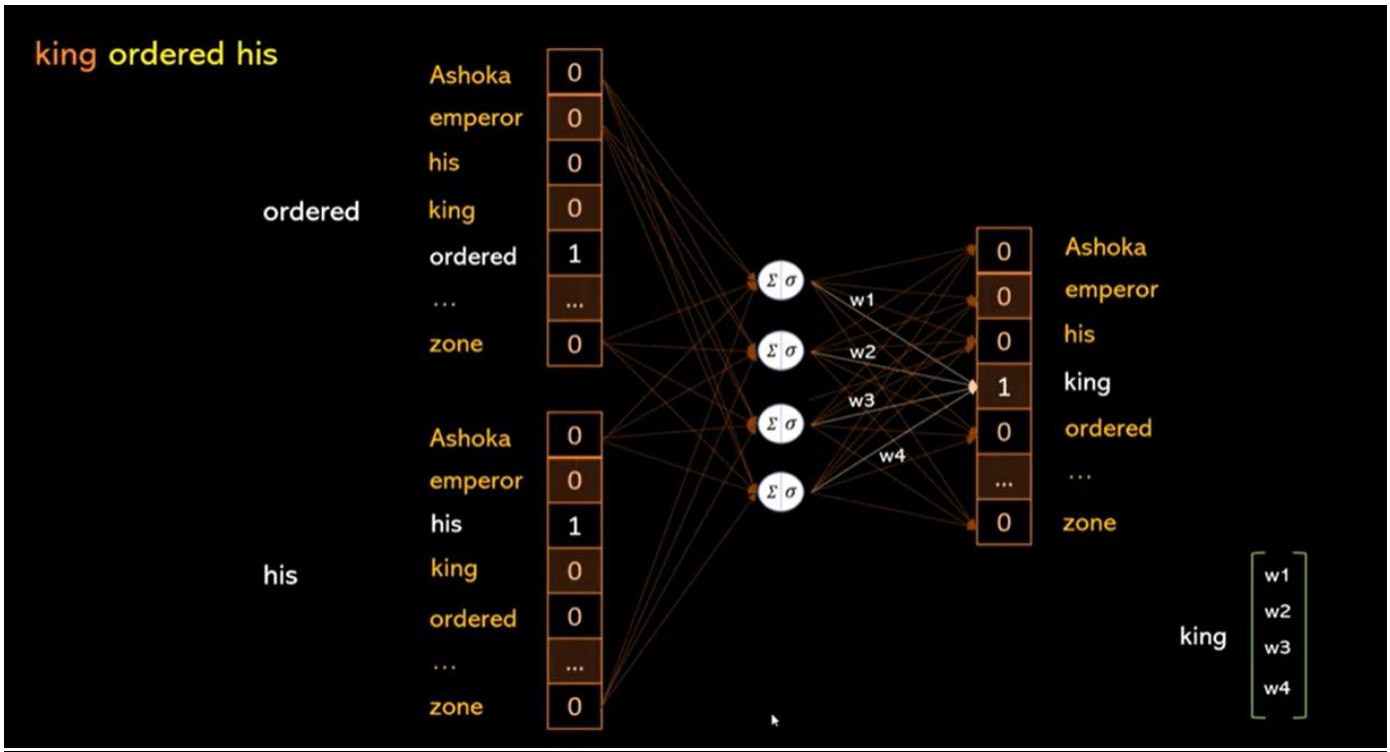
Similarly for 2nd sample



Same with 3rd sample

And when u r done feeding ur 1 million elements, and lets say u run 50 epochs

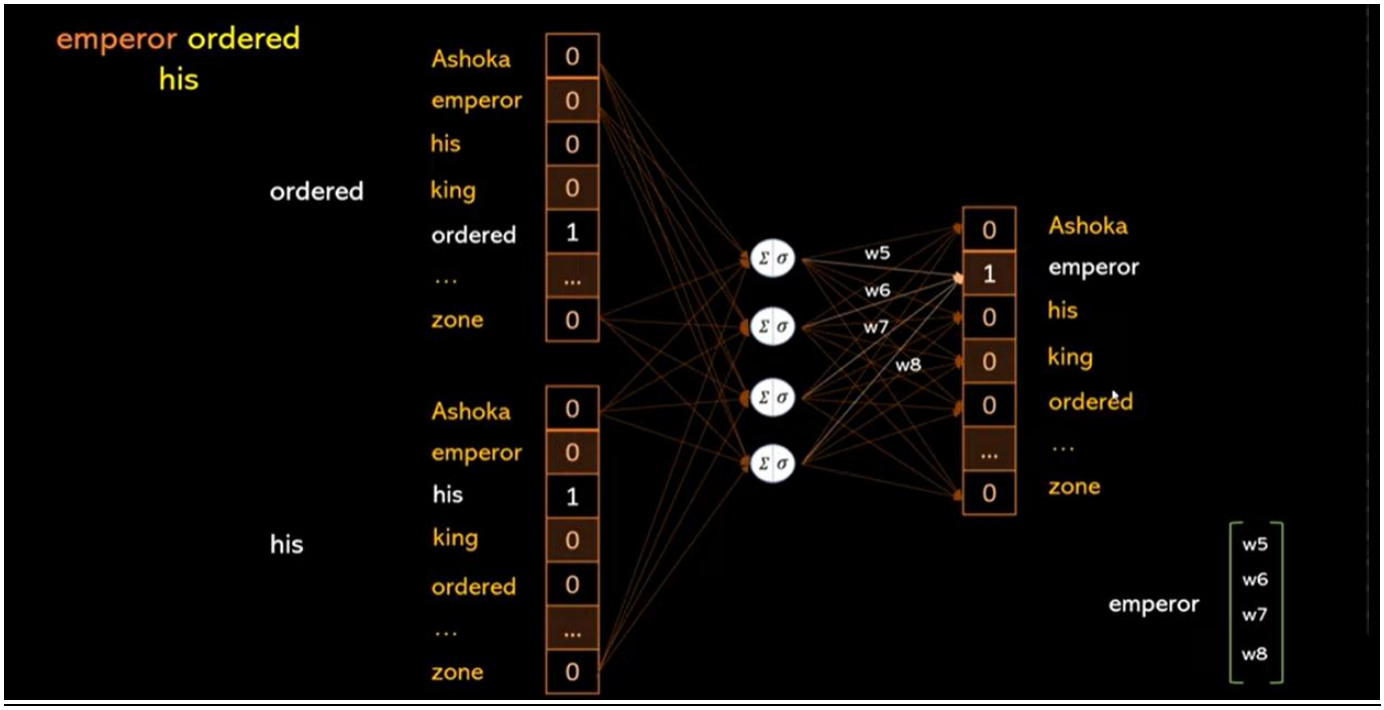
And ur neural network is trained



And at that point the word vector for the king would be these weights.

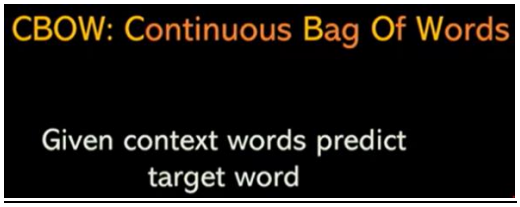
So the weights are nothing but a trained word vector

And this vector would be very similar to vector of emperor.

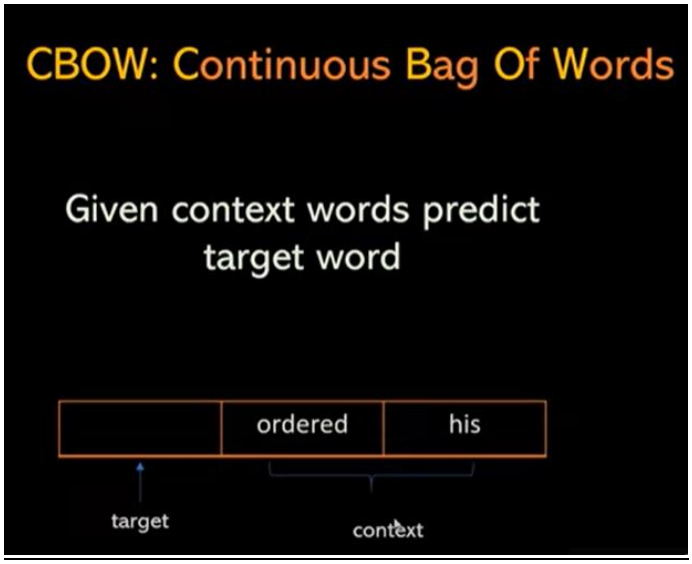


Both will get similar vectors bcz the input is same for both and when input is same u expect that these weights will also be similar.

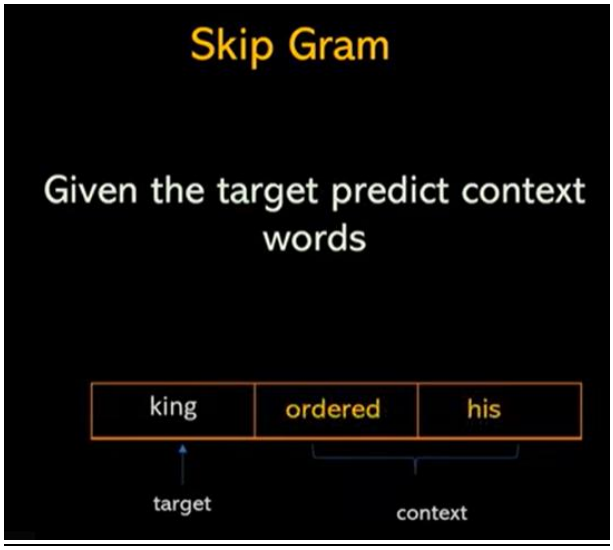
And hence the vector for king and emperor will be similar using this approach.



This approach is called Continuous Bag Of Words.



So here u have context which is 'ordered his' and based on that context u r trying to predict target which is 'King'.

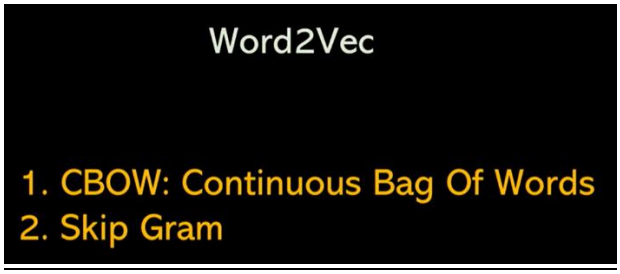


There is a second methodology called Skip Gram. In skip gram we do reverse

We have a target word 'king' and based on that we try to predict 'order his'.

Again predicting target from context and context from target these are fake problems.

We are not interested in solving these problems but while solving those problems we get word embeddings as a side effect.

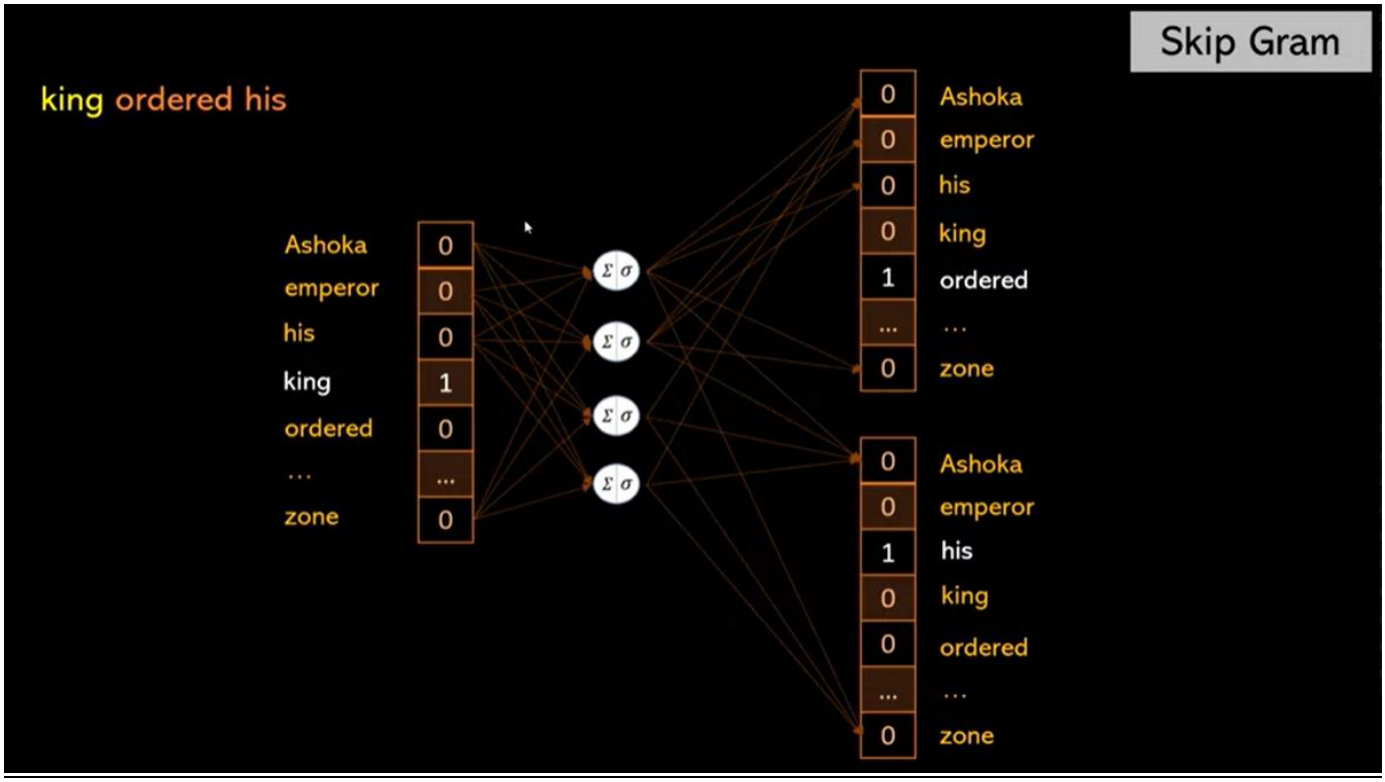


Word2Vec is not a single method , it could be using one of the two techniques which is either continuous bag of words or Skip Gram to learn Word Embeddings.

So Word2Vec is a revolutionary invention in the field of computer science

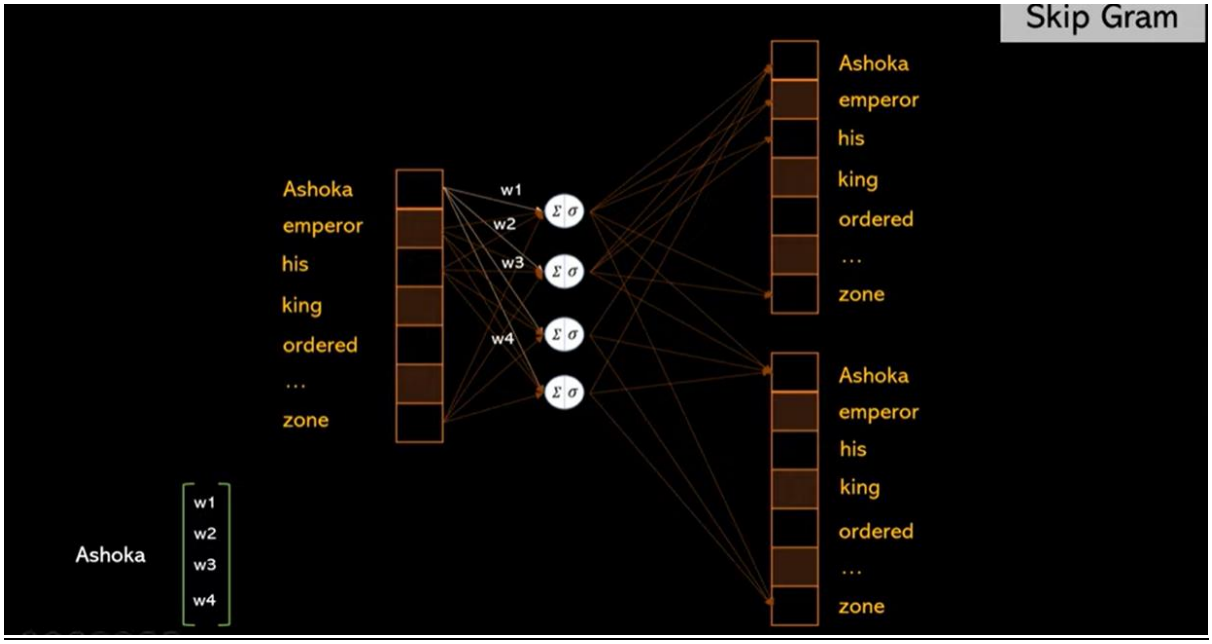
Which allows u to represent words in a vector in a very accurate way so that u can do mathematics with it.

LETS talk about Skip Gram



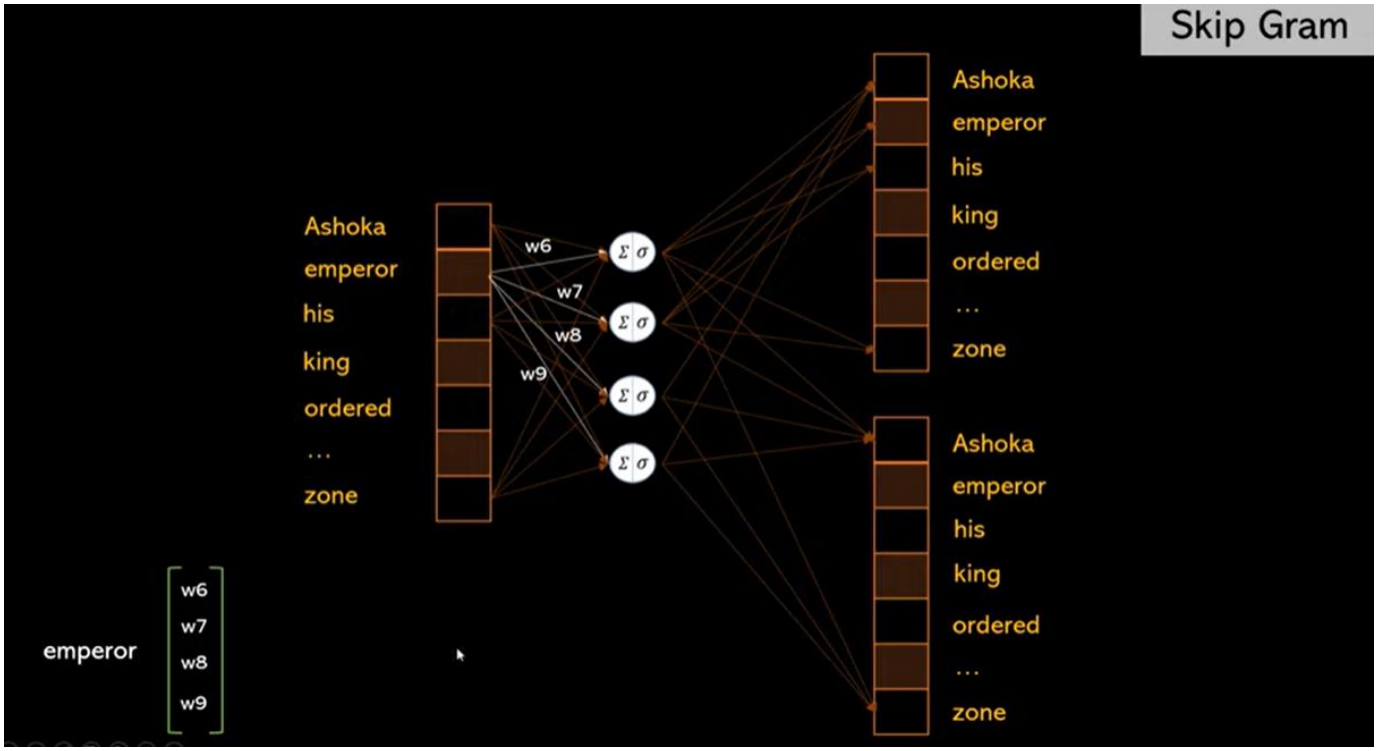
Here u have the word 'king' based on that u r trying to predict 'ordered his'.

And we will do the same thing. U will feed one sentence, calculate the expected output. Compare it with actual output, do back propagation and so on.



And in the process u will learn the word embeddings for each of these words.

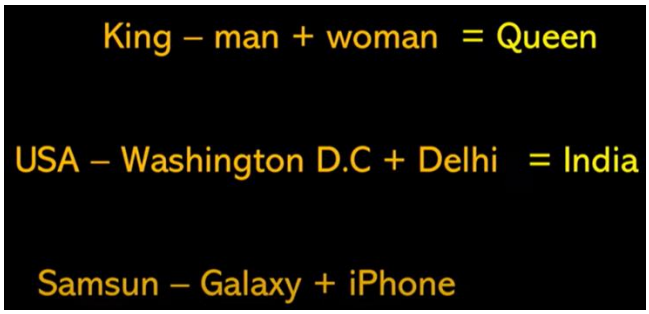
There are 5000 words in our vocabulary



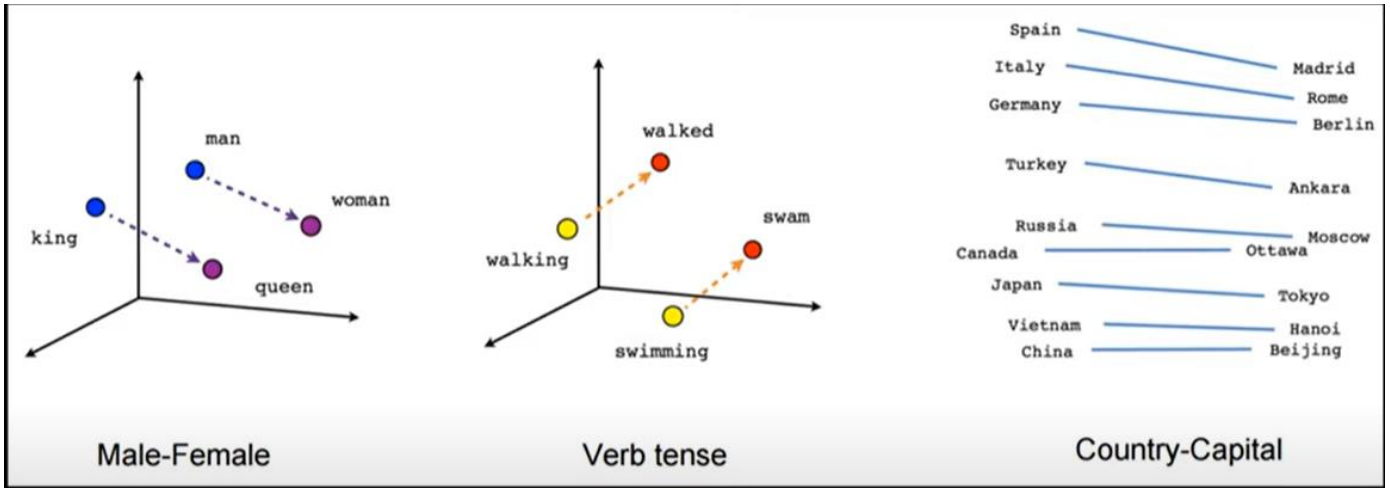
So when u r using skip gram the word embedding is a layer between the input layer and the hidden layer.

In the CBOW (continuous bag of words) it was the weights between hidden layer and output layer

U can do wonderful things with word2vec



So computers can do this kind of maths, this works really well



U can represent these vectors in a vector space

So here I am showing u 3-dimensional vector space, there could be n-dimensional vector space

The relationship between walking and walked will be similar to swimming and swam

So once u have learnt this relationship, when u give a word ‘swimming’ to computer

It will give output ‘swam’.

It can also figure out a relationship , it will say whatever Tokyo is to Japan, the same thing is Moscow to Russia

So it can learn these kind of complex relationships