

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

Module – 10.2
Lecture – 10
Distributed Representations of Words

So, what we saw now was sparse representations; or one hot representation sparse. Because, only one bit is on from there, we will move on to something known as Distributed Representations of Words. And you have already seen that sparse is in theory, but it gives a very simple recipe of converting words to vectors, but it does not serve much purpose.

(Refer Slide Time: 00:36)

A bank is a financial institution that accepts deposits from the public and creates credit.

The idea is to use the accompanying words (financial, deposits, credit) to represent bank

- You shall know a word by the company it keeps - Firth, J. R. 1957:11
- Distributional similarity based representations
- This leads us to the idea of co-occurrence matrix

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

So, let us see what distributed representations of words are? So, this around 1957 J R Firth made this very profound statement that, you shall know a word by the company it keeps and this of course, a play on some other similar code, but what does this actually mean; so it means that, you want to know what does the word bank convey; or what is the essence of the word bank? What this code says is that, if you want to know about bank, you should say, you should see the company that it keeps; that means, what are the other words which appear typically in it is neighborhood? And of course, when you have a large amount of corpus given say the entire Wikipedia.

Of course, at that time Wikipedia did not exist, but any large corpus. And does this led to something known as distributional similarity based representations. So to understand this, we first have to understand the idea of a co-occurrence matrix.

(Refer Slide Time: 01:38)

Corpus:

- Human machine interface for computer applications
- User opinion of computer system response time
- User interface management system
- System engineering for improved response time

	human	machine	system	for	...	user
human	0	1	0	1	...	0
machine	1	0	0	1	...	0
system	0	0	0	1	...	2
for	1	1	1	0	...	0
.
.
.
user	0	0	2	0	...	0

Co-occurrence Matrix

the own α

- A co-occurrence matrix is a **terms** × **terms** matrix which captures the number of times a term appears in the context of another term
- The context is defined as a window of k words around the terms
- Let us build a co-occurrence matrix for this toy corpus with $k = 2$
- This is also known as a **word** × **context** matrix
- You could choose the set of **words** and **contexts** to be same or different

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

So, the basic idea is to use the accompanying words, which in this example happen to be financial; financial deposits credit etcetera to represent bank and to do that we will construct something known as a co-occurrence matrix which looks like this right. So, a co-occurrence matrix is a terms cross terms matrix; that means, every row in the matrix corresponds to a term or a word and every column in the matrix also corresponds to a term or a word.

Can you guess, what how many rows would there be? Size of the vocabulary how many columns would there be size of the vocabulary. Okay, so here is how we construct a co-occurrence matrix. So, we take a word we are interest interested in constructing the row for that word. The number of columns is the same as the size of the vocabulary.

Now for every column we will make an entry, which tells us whether or how many times this this word appeared in the context of the target word right for example. If I look at machine, I am looking at the row for machine; I am trying to construct the entries in that row. I know that the number of columns is equal to the all the unique words in my vocabulary.

So, I look at the first word which is human and in that cell I enter the value, which is the number of times human appeared in a window of k words around machine; is that fine is that straightforward?. And that is how, I will construct this co-occurrence matrix, where I have taken the window size as 2; that means, in any given cell my entry would be the number of times human appeared within a 2-word window of machine is the i th cell clear, the definition of the i th cell clear. So, this tells me that user actually appeared 2 times around the word system in a window of 2 words around it is that clear?

Again gets this how I construct the co-occurrence matrix. Now, you could use the same, so this is known as words and this is known as context; that means, the rows we refer to them as words and the columns they refer to them as context. Now as you said that, the number of rows and the number of columns can be same that we can consider the same words in the context, as well as the same word as the target words right.

But you could also do something different you could say, that I do not want to consider all words that is context words. Because, for example the word for appearing with any other word does not really give me much information because, it is just a stop word right or the word the or and or a these are known as stop words in the language, these do not really give me much information.

So, if you go back to the bank example, financial credit deposit other words which I really care about and these are the financial bank or with deposits, and also these words do not really matter a lot ok, do you get the intuition?

So, you could choose to have fewer columns, which are only the important words that you consider and you ignore the stock words ah. In this discussion, I will alternately switch between considering the columns as the same as the rows and sometimes are restricting the columns to fewer number of entries.

(Refer Slide Time: 04:38)

Corpus:

- Human machine interface for computer applications
- User opinion of computer system response time
- User interface management system
- System engineering for improved response time

	human	machine	system	for	...	user
human	0	1	0	1	...	0
machine	1	0	0	1	...	0
system	0	0	0	1	...	2
for	1	1	1	0	...	0
.
.
.
user	0	0	2	0	...	0

Co-occurrence Matrix

- A co-occurrence matrix is a **terms** \times **terms** matrix which captures the number of times a term appears in the context of another term
- The context is defined as a window of k words around the terms
- Let us build a co-occurrence matrix for this toy corpus with $k = 2$
- This is also known as a **word** \times **context** matrix
- You could choose the set of **words** and **contexts** to be same or different
- Each row (column) of the co-occurrence matrix gives a vectorial representation of the corresponding word (**context**)

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10 9/70

So now, each row gives us the vectorial representation of the word. So, we have seen how we have moved from sparse representations to distributed representations. So now take a guess now would this vector be sparse.


So, we saw the extreme sparse right which was one hot now, the vectors which you get here are they going to be dense or still sparse, sparse right? Because every word does not appear with every other word right; you still have these V dimensional vector and there are some words which will appear with very few words right. So, you expect to have non 0 entries in very few columns right; so, these representations are also sparse.

(Refer Slide Time: 05:19)

Some (fixable) problems

- Stop words (a, the, for, etc.) are very frequent → these counts will be very high

	human	machine	system	for	...	user
human	0	1	0	1	...	0
machine	1	0	0	1	...	0
system	0	0	0	1	...	2
for	1	1	1	0	...	0
.
.
.
user	0	0	2	0	...	0



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

So, there are some problems some of which are fixable. So, we look at the fixable problems first. The first thing as the stop words are very frequent so, these counts should be very large.

So, if you take the entire Wikipedia corpus and you take the word machine or system. Then, the words and for and so on would have appeared like more than 1000 times in the context of the word machine right. And as compared to the other words like system or user they would have appeared much fewer times. So, this kind of skews your counts right, it is like highly biased in the favor of stop words. So, how do you deal with this?

(Refer Slide Time: 05:55)


Some (fixable) problems

- Stop words (a, the, for, etc.) are very frequent → these counts will be very high
- Solution 1: Ignore very frequent words
- Solution 2: Use a threshold t (say, $t = 100$)

$$X_{ij} = \min(\text{count}(w_i, c_j), t),$$

where w is word and c is context.

	human	machine	system	for	...	user
human	0	1	0	x	...	0
machine	1	0	0	x	...	0
system	0	0	0	x	...	2
for	x	x	x	x	...	x
.
.
.
user	0	0	2	x	...	0



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

So there are 2 ways, one is ignore frequent words right. So, that is the solution which I suggested earlier; that your number of columns would be less than the number of words that fine. So, you do not actually consider frequent words at all.

The other is user threshold t so; that means, in these columns like for and with and so on. Whatever be the entry, if that crosses a certain threshold then, I will just replace it by that threshold is it clear?

So, I am just saying that, this means that the word has appeared more than 100 times. And I am not interested in keeping the actual count which was more than 100. I just saying that, more than 100 is enough for me right. Because, I know that all the other entries are going to be much less than this. So, just like replacing it by a very large number instead of actually counting that number.

(Refer Slide Time: 06:43)

Some (fixable) problems

- Solution 3: Instead of $count(w, c)$ use $PMI(w, c)$

$$PMI(w, c) = \log \frac{p(c|w)}{p(c)}$$

$$= \log \frac{count(w, c) * N}{count(c) * count(w)}$$

N is the total number of words

- If $count(w, c) = 0$, $PMI(w, c) = -\infty$

Instead use,

$$PMI_0(w, c) = \begin{cases} PMI(w, c) & \text{if } count(w, c) > 0 \\ 0 & \text{otherwise} \end{cases}$$

or

$$PPMI(w, c) = \begin{cases} PMI(w, c) & \text{if } PMI(w, c) > 0 \\ 0 & \text{otherwise} \end{cases}$$

	human	machine	system	for	...	user
human	0	2.944	0	2.25	...	0
machine	2.944	0	0	2.25	...	0
system	0	0	0	1.15	...	1.84
for	2.25	2.25	1.15	0	...	0
.
.
user	0	0	1.84	0	...	0

Mitesh M. Khapra CS7015 (Deep Learning) : Lecture 10 11/70

The other solution is instead of count you can use something known as PMI . So, this is how you compute PMI. Even if you do not know, it does not made a lot of difference because, you know that it will always be there on the slide that is why you guys do not read anything. So, PMI is computed like this. So, intuitively tell me what does PMI capture? Look I would say focus on this formula rather than the upper one; when would it be high? The easier question to answer is when would it be low; remember you are dealing with a fraction.

So, if independently the 2 words appear a lot of times. But together, they appear very rarely then the PMI is going to be low is that clear? Now if both the words appear 100 times and together also they appear 100 times, that is the best case scenario; that means these are very tightly tight words right, they always tend to appear together right.

So, the PMI would be high for words which are very frequently co-occurring. ok. Now, so this is what would happen, if you replace the counts; the co-occurrence counts by the corresponding PMI. Now, if the count of 2 words is 0 we have a problem because, then the PMI tends to be minus infinity right. So, how do you deal with this situation? Epsilon or some we will use some hack right as usual.

So, instead of PMI use something known as PMI 0, which works like this. If the count is greater than 0, then you use PMI; if the count is not equal to 0, then you just put the entry 0 in the set, make sense. There is also something known as positive PMI, which is

slightly more extreme; it says that use the PMI only if the PMI is greater than 0 otherwise use 0.

Can anyone tell me the rationale for doing this? You see the subtle difference between the 3 things right. One is of course doomed because, you cannot handle 0 counts. The other one is saying that, if the count is 0 then, I will just substitute 0, the last one is saying that, if the PMI is negative right. Then I will replace it by 0; that means, in the last case all the cells in your or in your PMI matrix would be positive right, non negative rather.

So, can you tell me the rough intuition for using this and there is only a rough intuition, but can you tell me, so the very rough intuition for this is, that what does it even mean to say that 2 words are negatively correlated. I mean, either they occur together right; which means, there is some relation between them, but a negative relation between words does not make sense that is the intuition behind this. Now of course, I could argue that what about antonyms and things like that, but that is also not the same right, because you could have good and bad in the same sentence right.

But that is the roughly the intuition that negative values do not mean much. So, just replace them by 0s. There is no again a formal reasoning behind this, but just the intuition. So, we have looked at the co-occurrence matrix where, we started with counts. These counts were very sparse and there are also some other problems with counts in the terms of some frequent words taking a lot of limelight and so on.



So, we have fixed although then, we have done some very minor and simple fixes. And I just very rush quickly rush through them because they are very simple. But these were all fixable problems, what a non-fixable severe problem with this? What is the problem with the one hot representations large?

(Refer Slide Time: 10:27)

Some (severe) problems

- Very high dimensional ($|V|$)
- Very sparse
- Grows with the size of the vocabulary
- **Solution:** Use dimensionality reduction (SVD)

	human	machine	system	for	...	user
human	0	2.944	0	2.25	...	0
machine	2.944	0	0	2.25	...	0
system	0	0	0	1.15	...	1.84
for	2.25	2.25	1.15	0	...	0
.
.
.
user	0	0	1.84	0	...	0



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

What about these representations, still large it is still of size V ? It is still very high dimensional, still very sparse not as sparse as the one hot encoding, but still sparse, and it grows with the size of the vocabulary. So now, remember that penn treebank at 50 k words Google 1 t corpus at the 13 million yeah. So, it keeps going with the size of the corpus.

So now how do you know? How do you fix this? I wish I had that Harry Potter thing. Anyone remembers that spell to wipe out your memory? How would you deal with it? So you now see how it connects? So now, again you have ended up in a situation where, you have a very high dimensional matrix right. And you are looking for ways to reduce the dimensions; so we will go back and rely on things that you have learned and one of those was SVD right. So, you can use singular value decomposition.

Why did I say SVD and not PCA? Because, this is not necessarily a square matrix A ; this could be a rectangular matrix and for all practical purposes SVD is just a generalization of PCA right.