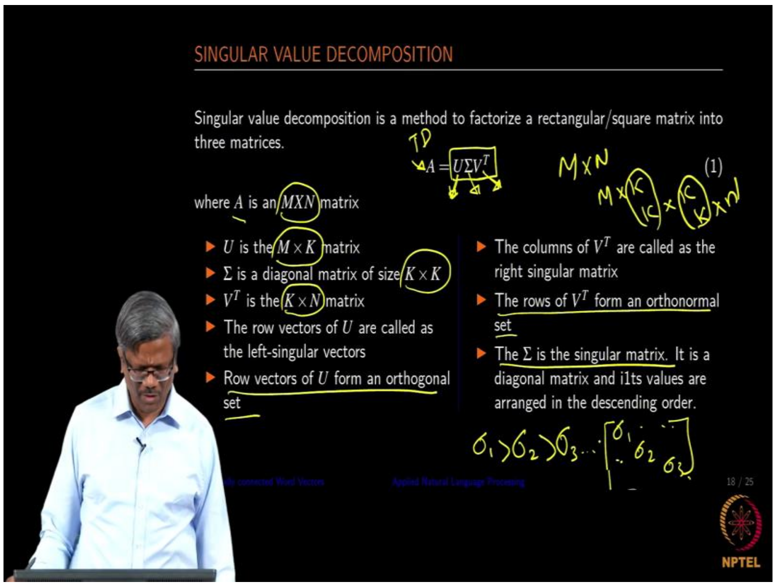
Applied Natural Language Processing
Prof. Ramaseshan Ramachandran
Visiting Professor at Chennai Mathematical Institute
Indian Institute Technology, Madras

Lecture - 17
SVD, Dimensionality reduction, Demo

The first exercise in terms of really converting the standard word term-document matrix into a transform domain, we are going to be using a singular value decomposition.

(Refer Slide Time: 00:29)



SINGULAR VALUE DECOMPOSITION

Singular value decomposition is a method to factorize a rectangular/square matrix into three matrices.

$$\overset{TD}{A} = \boxed{U\Sigma V^T} \quad (1)$$

$M \times N$

$M \times K \quad K \times K \quad K \times N$

where A is an MXN matrix

- U is the $M \times K$ matrix
- Σ is a diagonal matrix of size $K \times K$
- V^T is the $K \times N$ matrix
- The row vectors of U are called as the left-singular vectors
- Row vectors of U form an orthogonal set
- The columns of V^T are called as the right singular matrix
- The rows of V^T form an orthonormal set
- The Σ is the singular matrix. It is a diagonal matrix and i1ts values are arranged in the descending order.

$\sigma_1 > \sigma_2 > \sigma_3 \ldots \begin{bmatrix} \sigma_1 & & \\ & \sigma_2 & \\ & & \sigma_3 \end{bmatrix}$

18 / 25

NPTEL

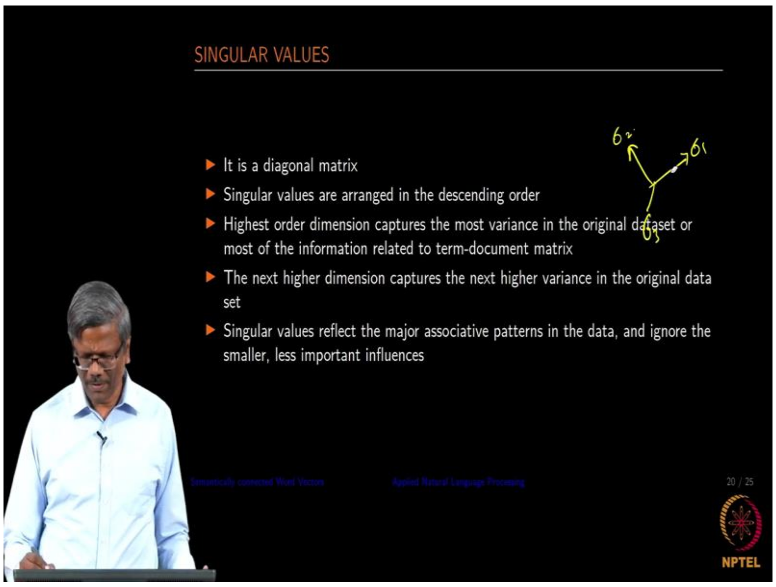
So, this is going to be our first exercise in terms of transforming the term-document matrix into a transform domain. So, we are not going into the details of how this decomposition happens within this singular value decomposition. So, we will only take the outcome of this and then move from there ok. So, in a singular value decomposition we will have some mechanism in terms of factorizing the original matrix. When you factorize that original matrix, it becomes into three different matrices ok.

So, this is your original matrix, we can call it a term-document matrix. So, when you transform this or decompose that into three matrices that you have, one is U and then the second one is sigma, the third one is V transpose. You let us assume that A is a matrix of size M by N and when you transform U becomes M by K and then a diagonal matrix sigma is of size K by K and then the V transpose is of K by N. So, if you look at this we have M by N is transformed into M by K, K by K and K by N. So, it becomes M by N.

So, when you do the multiplication of these matrices it is again equal to a, which will be in the same size as the original matrix.

So, here the decomposition of a given matrix or a term-document matrix is given in this particular as to size. So, he we call U as the left singular matrix, we call V as the right singular matrix and sigma as our singular values ok. So, the row vectors of U form an orthogonal set ok. The rows of V transpose form an orthogonal set ok, sigma is your singular matrix and it is a diagonal matrix and if you look at the values it will be like where sigma 1 is greater than sigma 2 greater than sigma 3 and so on ok. So, this is what you will see inside this structure of the matrix once decomposition happens.

(Refer Slide Time: 03:47)



SINGULAR VALUES

- It is a diagonal matrix
- Singular values are arranged in the descending order
- Highest order dimension captures the most variance in the original dataset or most of the information related to term-document matrix
- The next higher dimension captures the next higher variance in the original data set
- Singular values reflect the major associative patterns in the data, and ignore the smaller, less important influences

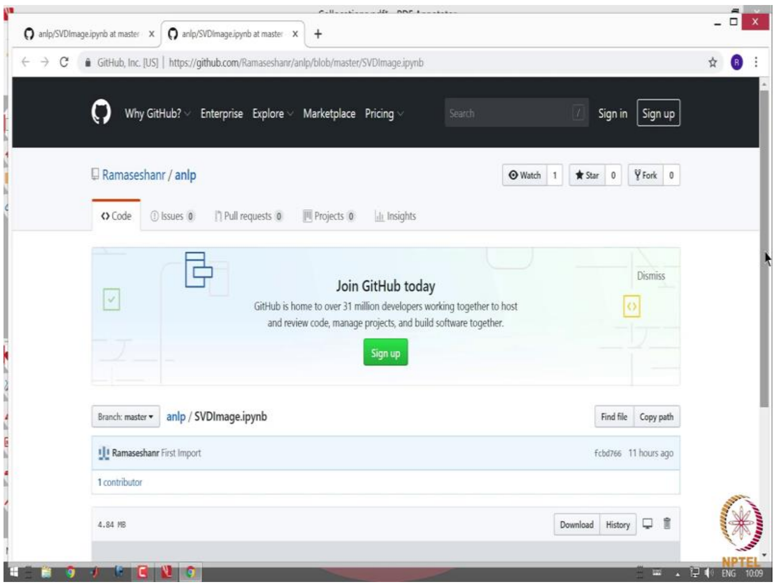
σ_2
σ_1
σ_3
Semantically connected Word Vectors
Applied Natural Language Processing
20 / 25
NPTEL

This slide is about singular values. So, as I mentioned earlier it is our diagonal matrix, singular values are arranged in the descending order, the highest order dimension captures the most variance in the original data set or the most information related to the term-document matrix. The next higher dimension captures the next higher variance in the original data set, singular values reflect the major associative patterns in the data and the ignore smaller and unimportant influences.

So, what this means, if you look at these singular values the topmost one will have the highest value; that means, it a contains all information related to the document matrix and it has captured the entire or the maximum variance or the most variance within the term-document matrix in the transform domain.

And then if you look at the second one, it captures the next set and if you try to find out how these axes are aligned, the sigma 1 and sigma 2 are aligned totally in a different axis set. So, if I consider only one element of that sigma 1 would be if you look at sigma 1, sigma 2 would be here ok. So, the same way sigma 3 is and so on and so forth. So, it is a very crude representation of how you would like to visualize these sigma terms ok. So, we need to really take an example and then showcase this.

(Refer Slide Time: 05:49)



Join GitHub today

GitHub is home to over 31 million developers working together to host and review code, manage projects, and build software together.

anlp / SVDImage.ipynb

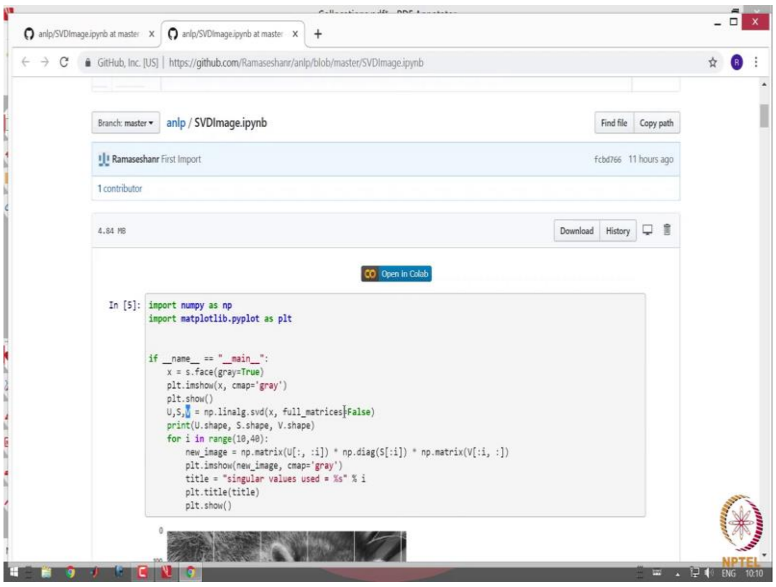
Ramaseshanr First Import fcbd766 11 hours ago

1 contributor

4.84 MB

So, before that you know let me go to one demo and then see how it works, and then we go back to the definitions and meanings of that formula. So, what I have done is I just taken in order for us to understand how this works, I have written a very small program.

(Refer Slide Time: 06:17)

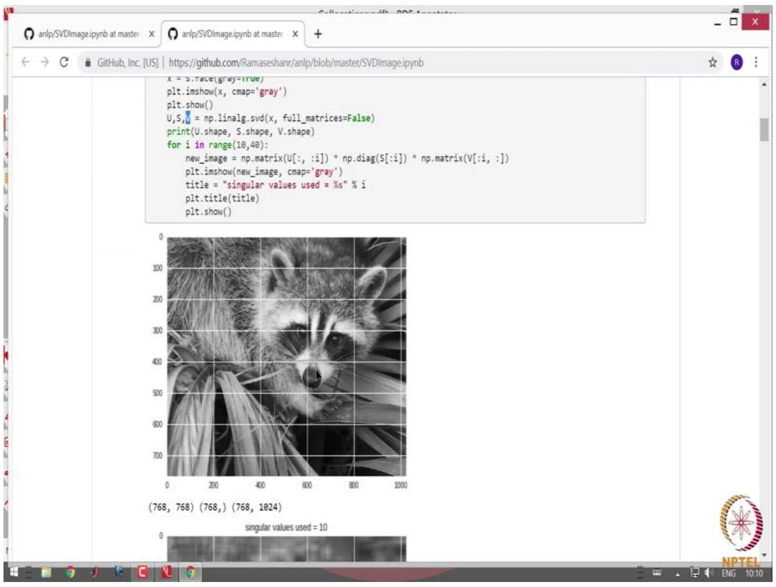


```
In [5]: import numpy as np
        import matplotlib.pyplot as plt


        if __name__ == "__main__":
            x = s.face(gray=True)
            plt.imshow(x, cmap='gray')
            plt.show()
            U,S,V = np.linalg.svd(x, full_matrices=False)
            print(U.shape, S.shape, V.shape)
            for i in range(10,40):
                new_image = np.matrix(U[:, :i]) * np.diag(S[:i]) * np.matrix(V[:i, :])
                plt.imshow(new_image, cmap='gray')
                title = "singular values used = %s" % i
                plt.title(title)
                plt.show()
```

So, in this program what I am doing is I am just taking an image this image is available as part of the Python libraries ok. So, I am this is the original image that I am looking at and then I am just using a NumPy to do my SVD and then I am taking these sigma values from the first 10 to the first 40 and then see how really they are contributing to the structure of the image during the reconstruction process. So, I have done the SVD here and the values are available in the U, the left singular matrix and this is singular values and then right singular vector right.

(Refer Slide Time: 07:17)


anlp/SVDImage.ipynb at master
https://github.com/Ramaseshanr/anlp/blob/master/SVDImage.ipynb
singular values used = 10

```
    x = s.face(gray=True)
    plt.imshow(x, cmap='gray')
    plt.show()
    U,S,V = np.linalg.svd(x, full_matrices=False)
    print(U.shape, S.shape, V.shape)
    for i in range(10,40):
        new_image = np.matrix(U[:, :i]) * np.diag(S[:i]) * np.matrix(V[:i, :])
        plt.imshow(new_image, cmap='gray')
        title = "singular values used = %s" % i
        plt.title(title)
        plt.show()
```

```
(768, 768) (768,) (768, 1024)
```

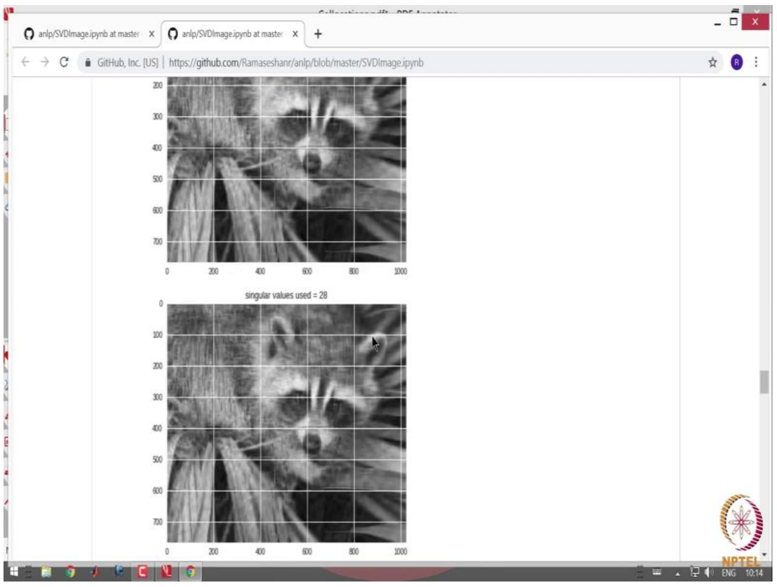
And this is my original image. The reason why I am taking the images is very easy to visualize and then see how when you reduce the sigma size, how we are able to still reconstruct the original image you remember the first element of the singular matrix captures the most variance.

So, is it really possible to just use only one so that I can reconstruct the image it is not possible? So, I have just taken the first ten and so, in this case I need to tell you the size of the image, the size of the image is 768 pixels or rows and then 1024 columns. So, this is my resolution to the image that I have here. So, the first one is our original image and then I take that image it is a gray level image and then I do an SVD and then reconstruct the SVD after truncating the sigma values let us assume that I am taking only the first 10 sigma values; that means, I have to take only the 10 rows of the left singular vector and then 10 columns of the right singular vector ok.

So, when you take those and appropriately shape them and reconstruct the image back. So, when you reconstruct the image as I mentioned earlier the multiplication of the matrices U, S and V give you about the original image. So, in this case, I am taking the so, if you look at the U shape, S shape and so on, it is 716 plus 768 that is your U and the diagonal elements are given by the S which is 768 and then our right singular vector the vector sectors 768 by 1024 ok. So, in this case what I am doing is I am just taking only 10 values ok; that means this is 10. So, this is 10 and this will be 10.

So, in the end, I will be reconstructing as 768 by 1024, but my sigma sizes are reduced and my column sizes are reduced this becomes 10, this becomes 10 I return the rest of the shape so that at the end I will have 768 by 724. So, I have just taken the first 10 sigma values.

(Refer Slide Time: 09:59)


anlp/SVDImage.ipynb at master
GitHub, Inc. [US] | https://github.com/Ramaseshanr/anlp/blob/master/SVDImage.ipynb
singular values used = 28
NPTEL

So, look at the image you know some important features of the image is captured correctly you can see the nose, some shapes in the eyes, the important part of the between the two eyes are also captured in some fashion, these are not very clear even though we can make out a bit. So, I increase the size of the singular values to 11, you can see there is a small change happening every time and increase the singular value by one more number ok.

So, slowly I will move them slowly so we can appreciate the change that is happening. See the ears have come out, the eyes are now coming out little clearer than before, the noses have come out well slowly I am just increasing the singular value size from 10 to 21 now, 22 let me go down to 30, see we almost have a good image reconstructed with just about 28 values of our singular matrix correct. So, instead of having a 768, I have restricted that size to 28 and I am still able to reconstruct the good image of this type.

So, as I increase the size, you can see the clarity increases; that means the singular values that you are seeing after 35 are making smaller improvements in the image structure so, how is it really working ok? So, this is the last one the 39th one, you can see the image is almost reconstructed to a reasonable clarity even those you know the grass is the grass that you see here, they are clearly the hair patterns are somewhat clear, the eyes or clear even the lights on the eyes or very clearly shown so on.

So, what is happening inside this ok? So, if you look at this image, it is made up of a gray level picture and then there are certain patterns inside the image that we do not visually see, but the system has captured those patterns.

So, it captured a lot of patterns and rearranged them in the transform domain and the patterns with maximum variance are kept at the first element of the diagonal matrix. And then as you go down the path of the singular value matrix, more and more patterns emerge, and then since they are orthogonal to each other, they are superimposed on top of whatever was done in the previous reconstruction and so on.

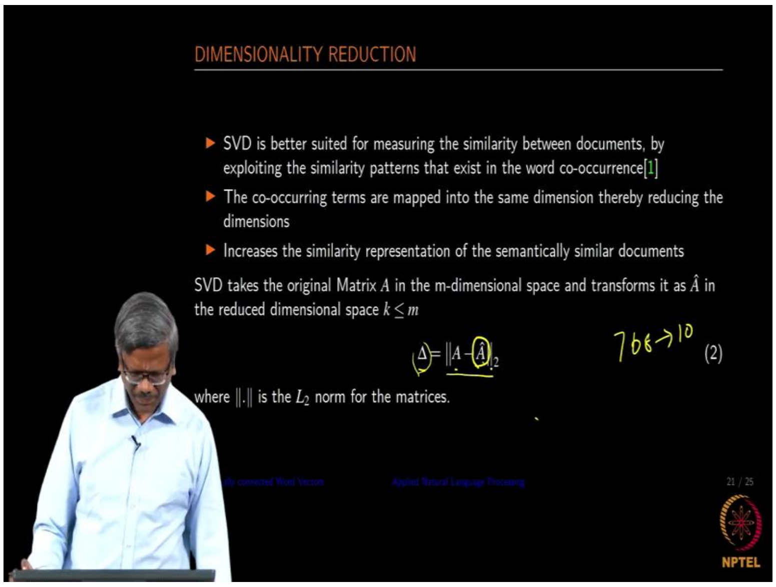
So, every time when you increase this singular value, more patterns appear which did not appear in the previous set of singular values and so on. So, in this fashion if you go back and then look at what has happened in the transform domain, the patterns are really captured and those with the maximum variants are made available in the top 10 or 20 singular values and as you go down the path after 100, do you not see any change in the picture; that means, you do not really require the entire size of these singular values to really reconstruct the image if I go back to the previous one or the original size ok.

So, I do not require 768 to really reconstruct the entire image; that means I have reduced the size of my singular values and also reduce the computational complexity so that I do not do a lot of matrix multiplication beyond a certain number. So, by looking at this you know from above, we see that the dimensionality of the original domain has been reduced in the transform domain and the patterns which are very similar or captured and probably put in the same axis so that you do not have too many dimensions like 768 axes for each one of the pixels or if you look at it, it is 1024 into 768 axes you should have had, but instead, we are able to cut short that number of dimension to a very small number and at the same time the patterns are also captured as part of that.

So, if you apply the same model to the natural language, singular value decomposition will capture similar patterns and capture the most variants in the topmost singular values and the least ones will be kept at the bottom of the topmost. So, in the same fashion if you have a huge matrix of size 10 billion documents by 1 million words, you probably do not have to have those many values to reconstruct the original term-document matrix, you probably you require only a smaller number equal to 100 to 200 sigma values in order for you to reconstruct the original term-document matrix ok.

Let me go back now, a singular value slide one more time. So, we know that it is a diagonal matrix we saw that, they are arranged in this descending order. The highest dimension captures the most variant we saw that right.

(Refer Slide Time: 16:37)



DIMENSIONALITY REDUCTION

- SVD is better suited for measuring the similarity between documents, by exploiting the similarity patterns that exist in the word co-occurrence[1]
- The co-occurring terms are mapped into the same dimension thereby reducing the dimensions
- Increases the similarity representation of the semantically similar documents

SVD takes the original Matrix A in the m-dimensional space and transforms it as $\hat{A}$ in the reduced dimensional space $k \leq m$

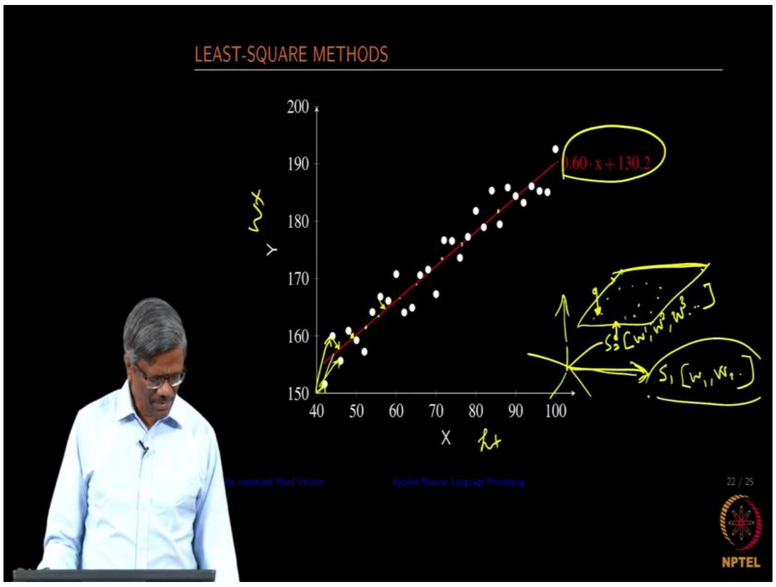
$$\Delta = \underline{\|A - \hat{A}\|_2} \qquad 768 \to 10 \qquad (2)$$

where $\|.\|$ is the L_2 norm for the matrices.

21 / 25

So, as I mentioned that it is a better method to capture this similarity between the documents or the similarity of patterns within an image ok. So, as I mentioned earlier, when you do the reconstruction of the matrix you know we you know remember we reduce the dimension from 768 to 10 ok. So, the difference between the original matrix and the reconstructed matrix should be very minimum, I think we saw in that case as well right when you do the L 2 norm of the matrices, the delta is very small, we are able to visually see that there is not a lot of errors in the reconstructed image. So, the same thing we expect that we expect in the term-document matrix as well ok.

(Refer Slide Time: 17:39)



It is very similar to fitting this; I think some of you are very familiar with the regression methods where you want to fit various points to a line. So, in this case, I think it is a height versus weight graph ok. So, and then the values are taken from various people and then your line has to be drawn that fits all the points in this ok. So, when you do that what happens is the distance between this point and this point is always minimized so that you know this line is formed. So, whenever you want to form the line, all the distances of the point these are the points to the line should have a minimum value when you consider all of the points in this particular set.

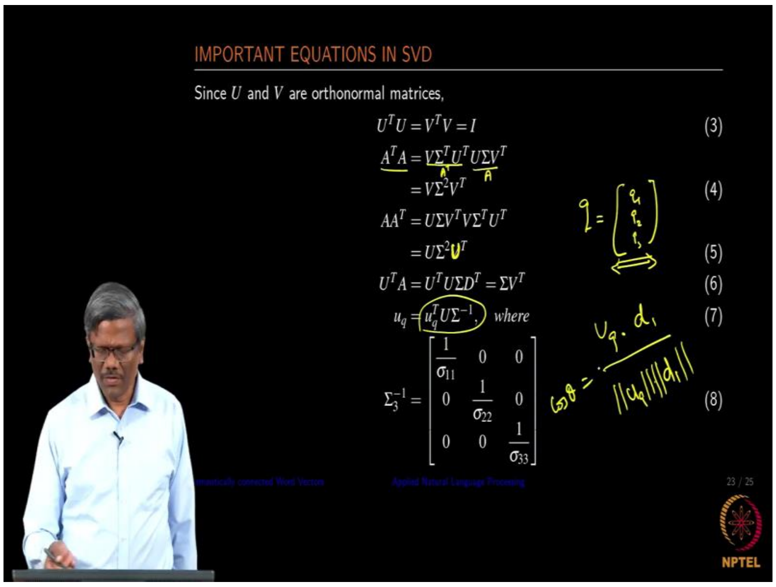
So, we are trying to fit a line which gives you an approximate fitment for any point in this given space. So, in the same fashion or what you can say is once this is found, we say that all these points lie in this line right, we approximated those points; that means, now we there is no separate representation for each of these points in different axis correct. So, now, they are all represented in one single dimension and all those could be accommodated in that particular line. In the same fashion when you look at SVD is doing something very similar to this.

The documents which are very similar would be brought into one place since the size is huge their going to be talking about the hyperplane. So, when you look at the hyperplane which is a very crude representation of that. So, all these documents would be brought into this even though you might see some documents would be here, here and so on. So, we approximate it and then put them and call those documents as similar documents, they occur in the same different hyperplane ok.

So, SVD is doing something very similar to that so it is trying to bring in all the documents which are similar in some hyperplane of this type and the similar words that you find in the vocabulary would also be brought in to the same axis that we have so; that means, in the case of SVD, the number of axes with respect to the word that I have used in the original matrix would be reduced in this fashion.

So, meaning similar words let us say similar set 1, word 1 2 would be in this axis, and so on ok. So, not only SVD would help you in terms of reducing the dimensions, but it also tries to find similar words and similar documents and put them together in the respective places ok.

(Refer Slide Time: 21:25)



IMPORTANT EQUATIONS IN SVD

Since U and V are orthonormal matrices,

$$U^T U = V^T V = I \tag{3}$$

$$\underline{A^T A} = \underbrace{V\Sigma^T U^T}_{A^T} \underbrace{U\Sigma V^T}_{A} = V\Sigma^2 V^T \tag{4}$$

$$AA^T = U\Sigma V^T V \Sigma^T U^T = U\Sigma^2 U^T \tag{5}$$

$$U^T A = U^T U \Sigma D^T = \Sigma V^T \tag{6}$$

$$u_q = \boxed{u_q^T U \Sigma^{-1},} \text{ where} \tag{7}$$

$$\Sigma_3^{-1} = \begin{bmatrix} \frac{1}{\sigma_{11}} & 0 & 0 \\ 0 & \frac{1}{\sigma_{22}} & 0 \\ 0 & 0 & \frac{1}{\sigma_{33}} \end{bmatrix} \tag{8}$$

$$q = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} \Longleftrightarrow$$

$$\cos\theta = \frac{u_q \cdot d_1}{\|u_q\| \|d_1\|}$$

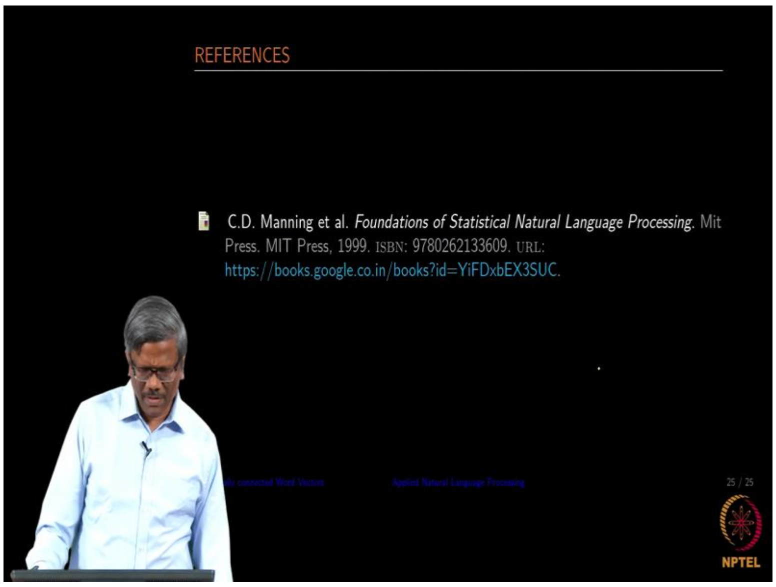
Semantically connected Word Vectors
Applied Natural Language Processing
23 / 25
NPTEL

And there is some more important formula that you have to remember. I have listed some of them which would we would be using, if you use the A transpose A-ok. So, we know that this is our A, this is our A, this is our A transpose. So, it becomes V sigma square V transpose ok. So, in the same fashion when you do A transpose, it becomes U sigma square V transpose I am sorry this should be U and then when you do a U transpose A, you get a sigma V transpose, you see how the transformed domain really reacts to various element ok.

So, what is the most important thing that we are looking at when you do the term-document matrix, we want to find out the similarity using a cosine distance or we want to find out how the word sorts how similar words are coming together in the same axis and so on. So, let us look at one example where we want to find out a query that is given ok, let us say some values are there for the query.

I just want to find out how this query can be used and then later we can find out how this particular one could be close to the document that we have in the transform domain. Since the query exists in the domain of the term-document matrix, we have to convert it into the transform domain using a formula like this and then applying that particular query using our these our cos theta right. So, in this fashion, you can find out the distance in the transform domain as well ok.

(Refer Slide Time: 23:53)



REFERENCES

C.D. Manning et al. *Foundations of Statistical Natural Language Processing*. Mit Press. MIT Press, 1999. ISBN: 9780262133609. URL: https://books.google.co.in/books?id=YiFDxbEX3SUC.

So, with this, I conclude this lecture, what I would do next is I will actually take a term-document matrix in the next class and then really explain you the important steps that we go through when we do the SVD and then how a query can be constructed in the transform domain and how it can be applied to find similar documents and then later we find out how the same technique can be used for topic modeling and so on so forth.