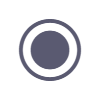
**Information Retrieval (AIMLCZG537\_DSECLZG537)(S2-24)-Sec 3-20250511\_155021-Meeting Recording**

May 11, 2025, 10:20AM

1h 57m 34s

 **V. MAHESWARI** started transcription

 **V. MAHESWARI** 0:20  
Let me share the screen.  
Yeah. Good afternoon, everyone. Let us begin the session. Before that I unmute.  
I love I am allowing Mike for all of you please mute from your side, OK?  
Before moving on into today's lecture right we have some few slides leftover in the previous lecture session. We will finish off that and then we can move on into today's session.  
To give a recap of what we have covered in the very first session, it was an introductory session on information retrieval where we have just seen what do you mean by information retrieval? What are its applications, where it can be used, and then the retrieval process, what actually happens in an IR model or an IR architecture. The various components which are involved in mainly your the indexing, how it converts your physical document into a logical document that is.  
Collection of terms how these terms are generated and in turn and inverted index how it is generated. Right? So we have briefly seen the functional components of the entire retrieval process and what do you mean by an information need? How an information need is to be converted into a query after pre processing this query and your terms right they a search operation is done on this to fetch the relevant documents and.  
Once these relevant documents are pitched, appropriate ranking is done to show the result to the user.  
So this is what the entire scenario of our IR retrieval process. So this is what we have covered in the last session. And then we have seen a very brief intro to the 3 models. That is your Boolean model vector model and the.  
Probabilistic model. We have seen what is a Boolean model and a vector model right Boolean model. Next we will be covering in detail right? Its just based upon your Boolean theory whether you are a term is present or not whereas your vector space model it deals documents and query as vectors and try to assign a weightage for.  
Representing the terms and based upon the similarity between the document and query, the ranking of the result is done and the 3rd one is the probabilistic model which is actually as the.  
Name denotes it. Just try to estimate the probability of whether a particular document is relevant to the user need or not, right? Mainly this is based upon what you called as relevance feedback, right? Explicitly, this model may not be used, but say for example, if you take a Google Scholar there.  
This probabilistic ranking is true. That is, whether that particular article is relevant to the user or not, right? So it is similar to your vector model in ranking and all those things. But The thing is, it measures the weight using this probability theorem, it uses the base theorem. Fundamentally, it uses the base theorem.  
And based upon that, you might be heard of odds ratio, so that is used to estimate this probabilistic probability of classifying that asset, relevant result or not. So usually its a nitrative process, they just begin with some answers set right to start with. Given a you said query which may be from a previous guess or previous history something right guess at the beginning and then.  
Based upon the use of feedback right, it tried to this use of feedback can be implicit or explicit.  
It tried to refine the result until it get the most relevant feedback from the user or get a good relevant score from the displayed result. So this is actually a binary index retrieval model similar to that, but it you the only thing is because since why its called a binary independence model? Is it just classifying either as relevant or nonrelevant? OK so.  
You actually this is not included in your.  
Syllabus we will be covering in detail Boolean and vector model, but someone in the previous section was asking about this probabilistic model, so I will try to press give some idea about this probabilistic model. Also how this algorithm works. The fundamental basic algorithm, how it works right? So then there I will describe in detail about this probabilistic model. OK how your base theorem and the odds ratio are used in identifying a particular document as relevant.  
OK, so based upon that relevancy, the ranking is done, right? So as of now all the 3 models we have given just a line of introduction, right? When I explain in detail, you will get an idea about these 3 models. So this probabilistic model mainly if you see the application its as I said Google Scholar or wherever you need.  
Relevance feedback type of syllabus. They use this probabilistic model and if you might be familiar with.  
A BM 25 algorithm in IR retrieval algorithm or an IR an architecture right BM 25. Have you ever come across this? This. I got the BM 25. Its an okapi generated algorithm. A best match 25 algorithm it it uses fundamentally this. That's a most one of the algorithm used in. If you see any.  
Deep learning or your neural the fundamental algorithm is BM 25 only for search process retrieval process. So this BM 25 best match 25 algorithm. It uses this.  
Basic model is the probabilistic model, but along with that it uses that TFID of scoring also right both put together that BM model works, its expansion is best match 25 algorithm, right? So that's one of the algorithm which uses this. So that's about these 3 models. We'll cover them in detail, right? So if you are not even clear about the difference between Boolean and vector when I explain it will be more clearer to you when we see the examples appropriate examples where how Boolean.  
That's how vector model works, right? So with that, we come to the end of this first session lecture. Here you can see some of the resources it's I have given here. Since you are talking about the IR we already have search engines both open source and commercial search engines. One of the famous mostly.  
Used search engine is Lucy right? You can just go through that Lucy. Its a architect. Its rather than saying it as a search engine. It's actually a true architecture model which is used by.  
Where it these tools are in losing are used by the various search engines, say for example at elastic search or sharp you know they use this disloosing. It provides various libraries, right plugin libraries which can be used. But the only thing is its built on Java. So you need advanced Java.  
Concepts to use these they'll use in models libraries right into your thing, but it accommodates all whatever we have discussed right TFIDF, ranking everything is incorporated in this using.  
You can just go through this the this search engine is an open source search engine and this is elastic search OK which is built on top of using. Actually its its mainly distributed search engine which is used.  
Actually Python, you can build your own thing models. They are not using zapion. If you see this is C plus plus these are all Java, right? But you don't have as such.  
Purely built Python models. We can that we are going to build a models. That's all right.  
Whatever we are going to do, we are going to use only pythag because the Java maybe its not much easy.  
For everyone to do so, all the lab exercise are programs assignments. Any explanation we explain? I try to explain using this Python, OK and this commercial open search is a.  
Play the search engine right, which is fundamentally similar to your elastic, and that's an another search engine. And apart from that you have Microsoft resume, a cognitive search engine is also there which is used for cloud search. So these are all some of the available search engines which has these features which we are discussing. So if we want to develop one it can be easily developed using Python, right? You can put into all the 3 models combined or specifically.  
Vector model or a probabilistic model, right? That can be generated. OK, we see what the end of the course, whether you are able to generate a search engine or not, right? So with that, we have come to the end of this this article, right? Just you can go through it which discuss about the various fact.  
Related to your IR, that is recent advances and beyond, its an interesting article you can go through this. So with that we come to the end of this lecture. One. Is there anything, any doubts, any specific queries you have before proceeding to the next lecture right. If you have any questions.  
Anything you need, no doubt.  
Hope everyone is clear.

 **GAURAV GUPTA** 11:18  
So, ma'am, one question so is this zappy an open search? So all these libraries you told, are we going to use them in subsequent lectures?

 **V. MAHESWARI** 11:20  
Yeah.  
No, no, we are not going to use them, right. It just I am giving you the various resources which are available at which users the concepts which we are discussing right what are all the concepts which are index construction or index generation, right. They use these these search engines are constructed using these concepts which we are going to discuss. OK.

 **GAURAV GUPTA** 11:51  
OK ma'am thanks.

 **V. MAHESWARI** 11:53  
Yeah.  
But shall we proceed to the next?  
Lecture next chapter right.  
Let me take that.  
Check to.  
Yes, OK.  
With this, we are moving on into the next chapter, next unit right, that is Boolean retrieval.  
We will see. What do you mean by an inverted index? How it is generated and how the Boolean queries are applied on that to processed and display the result. So this will be covering in detail. OK in this email and this chapter is actually from your textbook right? As I have already mentioned, the Raghavan.  
Textbook is used here, Manning and Raghavan OK.  
This is the chapter one actually.  
It starts with the this Boolean retrieval as the first chapter. OK, right.  
No.  
You need some data set to start with, right to understand the concept. So the data set which is used here right to explain the the various Boolean model or vector model initially to start with your index we are constructing is a small index. So in order to understand the concept the data set which is example taken here is the place of Shakespeare right we apply the retrieval process on those plays and try to get the.  
Get the results right and then evaluate it so all the plays of Shakespeare is used as an inputting diesel for these algorithm. Say for example, suppose if you want to find all the.  
Ways that contain the words right. You have the list of shade, fair place and I want my. I want to retrieve. OK, I want to retrieve all the place which has these terms with these characters. Brutus and Caesar. They should apply in the play, but not calparina the this character calporina should not be there, but both Brutus and Caesar should apply in the all the plays of Shakespeare, right?  
So how it can be done? What can be a simple process if I want to do this?  
Any answers? So if I want to search for it right, I want to search for all list all the place which.  
Has these 2 characters, Brutus and Caesar, but it should not have calpuria now. So here you have keywords. We have just the place. Its a its a document or something titled scene. Something will be there in the place. Right. So what do you mean by keyword here?  
Is there anything keyword associated? As of now we have you assume that you have all the place the Shakespeare articles place are there, right? Each and everyone.

 **NUGGAHALLI RAMANUJAM AJAY** 15:14  
We have to go word by word comparison, ma'am read every word from the play.

 **V. MAHESWARI** 15:14  
No, I won't.  
Word by word comparison exactly. Word by word comparison, that is, its a document. If you assume me which one as a document you need to compare whether Brutus is there or not, right. Every line or every sentence, every word you need to do that comparison and then identify Brutus. Similarly we need to compare season and then identify.  
And once you have retrieved it, if it has calporina, you need to drop it out. So this has to be done. But this is a group for search, right? We can call it as a brute force approach searching word by word. That's nothing but your linear scanning the document each and everyone. But this cannot be a solution. We can't do that. Your IR system can't do this. We have already seen this. Your physical document searching is not possible and.  
As I wonders, you might be familiar with the Unix group combat. Similar to that you can.  
Retrieve all the Brutus and Caesar, and then you can strip out or filter out place which contains the term calpurina. So this can be easily done. But this may not be a solution. Why? Because because of the large data collection which we have right, this is not the solution and again we can't have flexible matching operation using this script. What we want to achieve and 3rd one is the ranking cannot be done right. Best documents to return it just search and then.  
Need to sit back? The relevancy may be there, but the ranking may not be there. So now what? How to implement this IRR? How to do this search efficiently, right? The solution is you organise your data or organise your document before doing the search before starting the search bus before applying Reuters and Cesar and not Calparina. Right the query you.  
Organise your input. That is, index the documents. This is what we have already seen. Converting the physical into your logical documents.  
So how to do this indensing how to generate this indexing right? Any idea? Can you tell me so we have a list of documents in your hands, right? Say any place or there Sha Shakespeare place which has various terms. So we have in our hand the terms as after pre processing or something. Right. I have terms like brutal Caesar calporina. Right Antony anything but we have all the terms which appear in the place right so how to.  
Generate this index.  
Any idea?

 **NEERAJ KUMAR TALREJA** 17:42  
Term frequency. So for each term we maintain the links to the document.

 **V. MAHESWARI** 17:48  
You can maintain the link to the document right? So for say for example Pru test means in how many document it appears right? That is that's what you mean, right? Exactly. So what? Yeah.

 **NEERAJ KUMAR TALREJA** 18:00  
Yeah, just similar like similar like a bibliography given in the books.

 **V. MAHESWARI** 18:02  
Yeah.  
Yeah, similar to that, right. So what we will see how it differs from that your bibliography and your inverted index, right? So someone is mentioning about DFIDF, we are not going to use TFIDF yet, right? Be clear your TF or IDF, they are all associated with your vector model only. So now since we are starting with the Boolean model, I am just focusing inverted index which is.  
Connected with your Boolean model. When we move on to the vector model. This is a very basic inverted index. We are going to generate.  
When we move on to the vector model, just we.  
Add up, add up the other features into this inverted index. The complexity of your inverted index increases the add on increases right, so this is a very simple way of representing your.  
Document and your terms is term document incident matrix. Very easy way to represent. Suppose if we have very lesser documents and something you can easily do this by using the term document incident matrix that is here its nothing but the mapping your terms and your documents. OK so this matrix is going to be one for each for a corresponding term and a document it is going to be one if the term appears in T in particular document T otherwise it's zero since.  
Fully and right, we want only present or not whether that document is present. Sorry term is present in that document or not. So you can see here this is a a snapshot of a term document incident matrix wherein you have a series of terms right 1237 terms and 6 documents are here. So these are all the place Anthony cleared. But Tran Julius is right. The name of the place and you have.  
Some terms right which appears in these place, or we can say these documents now we can see. It just indicates whether that particular term is present or not. If you take the 10 pest right, the Antony character is not there. We have. What do you have here is mercy and bursal. OK. All Cleopatra Caesar. Right. They are all missing in this play similar.  
Ly. Its registering the each and every document whether it appears or not.  
Suppose here you just make this as one right we are not bothered how many number of times this particular.  
This particular term appears in this play. Antony and Cleopatra, right? Say for example, in this play and the term Antony may appear, say for example 100 times. OK, but we have not registered there, right? Why?  
Just making this as one or zero. This is what we call it as term frequency, right? The number of times a particular term appear in that document sit clear. So the card, the term Antony may be appearing 100 times in this Antony and Cleo total, but we are not registering or recording that. We just need whether it is present or not. OK, the term frequency is not needed for your boovian model because.  
We are not going to assign any weightage or any sport.  
Just we identify whether it is present and then consider it as a relevant document to us. Sit clear. So this is what you called as a TF term frequency. The number of times a particular term appears in that document here. So nowhere your term frequency is registered. Just either zero or one. OK, so it is a very simple incident. Matrix is very simple. So to search this incident matrix is also very simple now if.  
You if this query right proved us and Caesar and not calporina if I want to apply the query on this right, what should I do?  
It's just a Boolean equation, right? We have bhutas and zisa but not Kalpana. You can take the group Boolean vectors and then easily you can do an and.  
Not operation right? So if you take the Brutus, this is the vector Brutus vector, right? And similar. Here you have this vector.  
Where in which documents it appears so you can take the 2 vectors of Brutus Boolean vectors, right Brutus and C cell, and do an and operation, and also and the calporina take calporina and take the knot of it right. So if you do an end of all these 3, you can easily filter out in which documents what will be the resultant of this query. Right. You will get the result. What actually will be the output of this query.  
According to this incident matrix.  
Which place will be outputted?  
Can you tell me?  
Only one document.

 **DIVYANSHU** 22:58  
Document.

 **MANISH** 22:58  
Only one Antony and Antony and Cleopatra.

 **V. MAHESWARI** 23:01  
Antony, Cleopatra and Hamlet, am I right?  
So 2 documents satisfy this are relevant to this query and answer. So if you can take a simple Boolean query you can see here right? So this is the Brutus vector right where the where it appears in all the 6 documents we have totally 6 documents. So the vector here this C Sir vector is again for the documents variant appears and not of this one. This is for calpuria vector and you take a note of this.  
Just do. This is a simple Boolean operation. If you do that, your result is 100010. That means in clay one and clay 4 satisfy this query. That is Antony and Cleopatra and Hamlet, right?  
So it looks very simple. Your term document instant matrix looks very simple to represent and the search is also very simple. Just apply the query and get your results right. It looks very simple, but what is the disadvantage in moving further right?  
What you feel the disadvantage in using this?  
Document incident macbooks.  
What do you feel as the most prominent disadvantage on seeing this incident matrix?  
Any answers?

 **NEERAJ KUMAR TALREJA** 24:20  
Ranking will not be there.

 **V. MAHESWARI** 24:23  
Usually a boolean model. We are not going to do any ranking. Please understand. So ranking we don't need just leave the ranking. As of now your Boolean model just retrieves the relevant documents. That's all. No ranking.  
Yes, Naveen has given can only tell if something is there or not, but position bear. It appears that's not as mentioned here. That's one thing other thing.  
Performance lag for large data too long time yeah because its not suitable for your huge volume of collection. Why? Why its not suitable if you see this matrix right you can imagine for a huge volume of data OK.  
So in this case you can see a lot of zeros, right? So if your collection volume of data is huge, your matrix is too sparse. You agree that.  
So many zeros will be there, right? Literally it will be a sparse matrix. More zeros will be the percentage of zeros will be more when compared to your one. So why to generate or store such a sparse matrix? Or paste? Your storage is giving sparse matrix. So you need some wig, some other way of overcoming this. Someone has asked right? Why we are taking work vector instead of document vector for calculating so.  
I don't understand what do you mean by work vector. So this is the document vector actually.  
For corresponding the to each and every term we take that documents wherein it appears so. If you want to find the documents, we need to collect what are all the documents in which Brutus appears.

 **BHUPENDRA KUMAR PAL** 26:03  
So ma'am simply.

 **V. MAHESWARI** 26:03  
And then someone has to mute your microphone.  
Right. So yeah, importance of the term, that's it. Boolean model is that's a very simple. OK. So now how to overcome all these disadvantage we have discussed, right? So this just shows how bhupendra it you can see here right. So we are just taking the Antony and Cleopatra the 2 documents, these are the 2 resultant documents based upon the vectors which is given right.  
So on seeing this query right query answer, what actually it employees is we can only find whether the documents make that query. Someone has mentioned we cannot find which parts of the document made the query right where actually it appears position. It's not given here. See it actually comes here in the Act 3, scene 2 or something where you can find these terms. Caesar and protest similarly in Hamlet you can see.  
It just tells him whether the term appears or not. Not beyond that, not any details beyond that.  
Because we have not registered anything in this term document incident matrix. So if you want to retrieve all these things, they inverted index has to be enhanced. So you need to add ONS has to be done to get all these things we cover them later right? How to get these where actually the position of your term can also be retrieved. We will come to that next. Now as of now if you see this term document incident matrix as I said its very sparse right and.  
Term vectors are of fixed length right depending upon the total number of documents. OK so its too large.  
So it becomes too difficult to build when your document carpasses huge. So the solution is simple. Solution is why to register zero. You can register only once, right? Or you can store only once wherever it occurs rather than storing the absence we can store only the presence. That is what your inverted index does right? The solution is to record only the things that occur we.  
Need you leave out the documents where this Brutus is not present. We just record only.  
Take the input where it actually appears clear, so that is what is. The inverted indexes focuses on to reduce that space time. Everything right? Make it more efficient. That concept is used by your inverted index. So next we will see how this inverted index is generated. So this is how an inverted index will look like. You can see here.  
We have the list of terms here, Brutus calporina acid and it is connected to. These are all nothing, but we call it as posting list or inverted list postings its its the name which is given to that and you can see a series of numbers. These are all nothing but the document I DS OK each and every document they give up assign a document ID to them to identify.  
To identify that unique identifier. OK, which makes it easier. So these are all the document IDs wherein.  
This term group does appears similarly Caesar. This is the list of documents where it appears in Kalpana. OK, this is usually both all put together, we call it as an inverted index and this is usually called dictionary and this is called a post English. The the technical name which is given, we call this Dictionary of all the terms, right. And this is the list of postings. Postings are nothing but the document I DS, they usually call it as a posting list, right?  
I can see now when you see this dictionary and your posting list right the the linked list. This is nothing but the data structure which can be easily used. Here is the linked list representation. We'll see that and this dictionary and your.  
Inverted list, right? Your posting list. If you see what actually observe in this anything unique?  
May disorganised right. When you see this inverted list, your dictionary and your thing.  
Anything which strikes to hear you.  
These 2.

 **NEERAJ KUMAR TALREJA** 30:31  
Its a kind of a relationship like a vector.

 **V. MAHESWARI** 30:31  
Need.  
Relationship, anything here or separately when you look into this dictionary as such, or your inverted list posting list as such sorted exactly ordered right? Can you see this is also in sorted order? Dictionary is sorted order and a document is also in sorted order. So why do we need that in order to make your search process efficiently? They usually inverted index is sorted by both terms as well as the document type.  
OK, so that is how it is represented. That's the fundamental characteristics of your invented index. Always we represent them in the sorted order. Both the terms will be sorted as well as the document ID so that search operation is efficient. Say for example you have a huge list of dictionary I. If I if you want to locate any term right easily you can search if it is in the sorted order rather than a linear search you can go for a binary search right? So.  
That's the reason. Similarly, the If this is also sorted, it makes your search a getting the, retrieving the document process.  
Efficient. So this is your info. An inverted index looks like. Now we will take a small example and see how this inverted index is built. Built. OK so if you see the fundamental, yeah.

 **NEERAJ KUMAR TALREJA** 31:52  
Map.  
One question. So can you go over the backslide?  
So in, yeah, so in this is thing this is one problem that if we use only for not then we don't have number of documents then how will be returned just like I I want to search which document does not have a bonus.

 **V. MAHESWARI** 32:03  
This one.  
Document does not have.

 **NEERAJ KUMAR TALREJA** 32:20  
Or which?  
Rooters.

 **V. MAHESWARI** 32:23  
Protests. You can't retrieve that.

 **NEERAJ KUMAR TALREJA** 32:24  
Yeah.

 **V. MAHESWARI** 32:27  
It.

 **NEERAJ KUMAR TALREJA** 32:28  
Yeah, because in that case.

 **V. MAHESWARI** 32:29  
Your query cannot be not proved us right.  
That.  
Maybe if you want then the resultant will be. You can call. You need to refine your Boolean Curry so to retrieve all the documents and then connect it with that. So just if it is not Brutus it, it may not be possible right? A single word query right? You say like this, am I right?

 **NEERAJ KUMAR TALREJA** 32:54  
Yeah.

 **V. MAHESWARI** 32:56  
Then what it will do is very simple. If it is not proved us what it will do is here you have the document I DS all the document I DS the total collection end is there right all the document I DS are there. So this will be eliminated and the output will you can get it. That gives all the documents. Yeah right. Yeah you can get it.

 **NEERAJ KUMAR TALREJA** 33:13  
OK, so apart from these, yeah, we have to store the all the documents.

 **V. MAHESWARI** 33:20  
Yes, yes, all documents are given a document ID, just you can eliminate this. The resultant will be this minus the interior, right? So that is easily go OK.  
So now we will see how to build a small inverted indexes. A simple example we see what its trying to do is for each term we must store a list of all the documents that contain T that is, you assume that you have done preprocessing and we have the terms. All the terms in your hand. So what do you need to lose? Identify each by the document, Ida, document serial number, a unique document ID is assigned for a.  
A. All the documents and can be any value. Accordingly the document ID has to be assigned. So sort according to the document ID and.  
You need to group them right once you sort it according to the document ID. The same term will be there. Brutus may be appearing in document 1234. You need to group them and split them into what you called as a dictionary and a posting. As I said, right, so this dictionary and posting if you see the data structure most suited data structure right linked list are suitable for posting and.  
Or later is the 2 possibilities, but the most one suddhat is your singly linked list rather than your variable in the rice, because insertion may be a problem in your variable length, there is. So the most optimal if you if you see out of this the singly linked test can be used for what document I later on you see the data structures which we can use for dictionary as well as your posting right in the in the subsequent chapters we will see the index data structures right.  
So this is how it what do you mean by dictionary and a posting? As I said right? So the dictionary will be in its sorted order. Similarly your posting will also be in its sorted order, right?  
So this is sorted by DOC ID and this is sorted by your terms. OK now if you we we assume that if you have a document like this right what actually the preprocessing will do is later on next chapter is in detail we will see the preprocessing step. Just I am briefing out so the collect the documents to be indexed. So each and every document is collected and these special characters are removed as a first step right pre processing So what you get is called.  
Tokens. So these are all the tokens wherein we you want to apply the preprocessing and convert them into the terms right the tokens.  
Converted into terms, if you are not applying any preprocessing techniques, the tokens are equivalent to the terms. That's all right. Directly, it can take all the tokens assumed terms say clear. OK bye now.  
So someone has said right linked list. Can you just? Yeah, other data structure which data structure you feel best for your this.  
Posting list apart from linked list any data structure you have in mind.  
Each one has their own advantages. Yeah. Ray Ray, that's the problem, right. Array should be a variable length array insertion in an array. It's a problem, right? It's very difficult and tree.  
Is suitable for dictionary, but not for linked list.

 **DIVYANSHU** 36:45  
Doubly.

 **V. MAHESWARI** 36:47  
OK. Yeah. Tree or a try is we are we will be using tree or a Tri for dictionary but not your posting list. OK because posting list you cannot store it in a tree. The searching will be a problem.  
OK you its not a first in first order, you are not adding right. Its a number which is added. The document ID has to be inserted. Say for example I have 1279 document ID. If I want to insert 5 it should be inserted into that yeah right so.

 **DIVYANSHU** 37:11  
Doubly doubly linked.

 **V. MAHESWARI** 37:19  
Analysing the various factors they have considered inverted list as the most suited one. OK, by making the storage searching all other process OK.  
So this tokens right they have taken and we do the linguistic preprocessing. We have various steps in doing this converting tokens into into this.  
Is what we will be covering in the next chapter how these detailed preprocessing? Because this preprocessing has a lot of impact on the results, the precision, recall, they all are dependent on this pre processing one. So we will see them later right. As of now we assume that we have in our hand terms. So if you see the previous token after applying some preprocessing its converted into that that is your.  
The small candidate, small cap and you are a singular cleaner is converted into singular. All are normalised into one single OK.  
We assume these are all terms now these terms will be stored in your dictionary and associated with that will be the posting list where these the term friend appears. The documents are in the Roman appears it is it simple right? It appears in the thing. So if you have a linked list it will be easy to search.  
But why, when when we see a solve a Boolean query, you can find out why inverter list is better when compared to the other thing. OK bye.  
Indexes are dependent on a search tokens. Indexes are built on.  
The entire document.  
The query tokens are going to be searched on that OK.  
Yes.  
Whenever you give a query, you are going to search on that index only. Now next we will see. We apply the same rupus and cser query on that right we will build an index sentencing.  
Nowhere you can see.  
Can you see the next screen building an inverted index?  
We can see 2 documents right document one and document 2 just.  
To show you how the inverted index is built on this, they have a sentence right? Something here document one and document 2. So as of now I take each and every token as a toe right, without any preproing. All are listed here, right? I am considering all the tokens as well. You can see starting from I right each and every token in document one followed by each and every token in the.  
Document.  
2 So this is listed here OK.  
No.  
The next step is right. So since this is listed here, what we have to do is we need to merge it right sort according to the document ID. This is already sorted and then we need to sort using the terms. So sort by terms and then document ID. OK so is my screen visible?  
Is sought by terms.  
OK, fine. So now in this sort by terms or document, Idi don't see in my that's why asked.  
So first you sort by terms and then sort by document ID. Here you can see here so.  
This is the output right? So we you see it sorted by the terms.  
And then sorted by the document ID. Also if you take the first term and ambitious right, it appears in document 2 only, whereas be or Brutus. If you see the Brutus, it appears in both the documents one and 2 right similarly killed. You can see so many birds which are terms which appear in both the documents one and 2. So it is repeating twice. So we need to combine this and convert it, convert it into your dictionary.  
And your posting list.  
That is what is done. Finally you can see here you see the car inverted index which is generated. You can see here multiple term entries in a single document are merged.  
Brutus is appearing twice right, so its merged and then split it into dictionary and postings. Here you can see this is your dictionary right? This is your dictionary and this is the posting list OK.  
The clear. So we have all the terms which are in sorted order and if you take Brutus, it appears in both the documents. So you can see here one and 2. Similarly Caesar it appears in the both the documents one and 2. But here you can see one entry is added here that is an information called document frequency right a document frequency information is added.  
What do you mean by document frequency?  
Any answers?  
Count of what?  
Number of time it appears in.

 **BHUPENDRA KUMAR PAL** 43:06  
How? How? Yeah.  
How many times about club words appear and and?

 **V. MAHESWARI** 43:13  
Appears in.

 **BHUPENDRA KUMAR PAL** 43:14  
Document document. Suppose a word. Yeah. Words or Brutus come and 2 documents. That that is. That is. Yeah, that is document frequency.

 **V. MAHESWARI** 43:16  
Document.  
2 documents, yeah.  
Document treatment. So now we have.  
Introduced an another term earlier, we have seen what is term frequency.  
Here term frequency is the number of times a particular term appear in a document in that document, right? So term frequency is measured for each and every document. This factor is document frequency is a particular term right among the collection and how many document it appears right. The number of document it appears in the whole collection. So here we have only.  
2 documents just we are registering that recording the number of documents where in it up.  
Why do we need this? Thus Boolean model actually require this.  
Why we are recording this document frequency? Any idea? Anyway we will see later. Next we will see we are not recording term frequency because term frequency is of no use to your Boolean queries. But now you can see a new added entry document frequency. So depending upon your total document collection say for example this term appears OK then this will be thin right? OK.  
Here these 2 are not in sorted order, maybe yeah.  
Actually this has to come first. Exactly sharp, I said. OK, fine. So now we will see later why this document frequency is recorded here. OK, right now this is will be your inverted index.  
One minute.  
And check with my.  
So this is your final inverted index which is generated. So how to?  
Implementation. If you move on into the implementation, how do we index them efficiently, right? How do we index them efficiently both storage wise and as well as retrieval wise search wise right? Both the common storage to be made. So what are all the compression techniques that can be applied so that you can reduce the space as well as efficiency retrieve the search should also be efficient right? So these.  
Will be covering later on. OK, these parts. Like, that's a separate chapter. We will be covering. We will cover that.  
OK, efficient storage efficient retrievals. All those things will come. So as of now, this is your inverted index which will be generated. OK. I hope you are able to follow this. Now your next thing is once we have generated this inverted index, how to apply that Boolean query on this inverted index and retrieve your results. OK.  
So now we have these dictionary along with the posting list. So now it becomes very simple. You consider the query Brutus and Calculina right? What you can do. The first step is you need to get all the postings wherein Brutus appears.  
In order to get this, you need to search your dictionary again. As I said, it can be you know since its sorted it can be a binary search to locate that term in your dictionary. Then locate Kalpurina in the dictionary, get the postings. Then you need to intersect the 2 posting list. So here is where the complexity or the efficiency of your algorithm lies. You need to intersect.  
Both the posting list. Am I right? So here you can see after retrieving these are all the document I DS where the routers appears and these are all the documents where calparina appears. Finally you need to merge these 2. OK, the same merge operation of 2 sorted arrays or 2 sorted list right? The way the same process has to be applied on this right? But how efficiently this can be done right? What are all the other added on?  
Strategist applied to this so that you can reduce the time complexity of this merge operation because the majority lies in this comparison and then getting the common documents right. So in this simple example, how actually it will proceed, just a comparison right, in order to retrieve these 2 outputs, 2 and 31 actually.  
What actually to do is one and 2 will be compared, right? So the first one and 2 are compared. Since these are all not equal right the.  
Lowest one right, the lowest document ID will be moved on. The pointer will be moved to the next document in the list, so this will be pointing to the next document where it remains the same. And if you compare these 2212 its a match so you get an output. Here it will be listed as your output. Then both the pointers has to be moved to the next document ID. Now 4 and 31. Since this is smaller, it will be moved on 11 again 11 and 31.  
Again, move it 31 and 31. It is the match. You can output it and then both the pointers are moved so it will be 5445.  
Since this is lesser, it will be 173 again, 173 in 54 are compared again, OneNote one OneNote one and 173. This is lesser and since this is the end of your list, it stops and the.  
These are the 2 matching documents for the corresponding query group. Does and calorie? So this is how your intersection or merge operation of this Boolean query is done. Actually, right? Suppose if it is an hour, right? Routers are calculated. That's very simple. You need to just merge these 2, eliminating the duplicate ones. You can just merge it and give all the documents. That's very simple. And again no.  
Comparisons are nothing is needed here.  
Except for the why bhaji eliminating the duplicate ones, but here in the case of and right, you need to find a match and then retrieve the document current in comparison. Sometimes you know, so the complexity of the algorithm or the time taken by the algorithm. Even though we say that it is a linear right algorithm, it is the complexity is linear, but again it depends upon the number of comparisons which is done between these 2 so.  
What are all the ways in which these comparisons can be reduced? The strategies which can be used to reduce this comparison?  
So this is the reason why the dog your document I DS or posting list are also in the sorted order to make a searching easier to it is merge. Merge operations can be done easily because in the sorted order right. So this is how it works. The and operation simple and operation without applying any strategies right? It just compare and then retrieve the documents. Is it clear?  
Nowhere. This is the pseudo code for that right. Every time the pointer is moved to the next one, until if the whichever document ID is lesser, it will be moved on to the next pointer until you reach the nail pointer.  
Now my question is, suppose you assume that the proof does right, we have taken an example prudence and calparina Brutus the size of the link posting list of Brutus is X. We assume and the size of the.  
Hosting list for Calpuria is why, right? So what will be the total number of worst case scenario right? What will be the number of comparisons?  
Between these 2 list postings, total number of comparisons may be in the worst case scenario.  
Plus y, right? So you say its as X plus y.  
What is you? I don't get it.  
Give me.  
Union of X Plus one. Yeah. X&YX Union y. You need to Add all these things. So that's the.  
Maximum number of comparisons that can be done right, worst case scenario.  
Always we want the best case, right? So how to get reduce these number of comparisons? Any any suggestions?  
Based on these hosting list, right suppose if you want to reduce the number of viruses. Yeah, last class. Also someone has given win of X comma y no min comma XY is not possible. Why you just say check it outright and tell me if you are not able to get it. I will explain why min of XY is not applicable here. The maximum it has to extend because it depends upon the value also so.  
Worst case I am asking the worst case it will be X plus y only. OK.  
So how to reduce this merges number of merges? Any any idea what can be done?  
Pollution. Any possible solutions?  
Already here in wasted is in the embarrassing.  
Which is sort.

 **NEERAJ KUMAR TALREJA** 53:12  
So start with the smallest.

 **V. MAHESWARI** 53:13  
What sort is just starting thing? Start with the.

 **NEERAJ KUMAR TALREJA** 53:19  
Smallest list and then which one?

 **V. MAHESWARI** 53:19  
Not with the.

 **NEERAJ KUMAR TALREJA** 53:22  
Which one has a smallest list like a like a 2 term there and one has a document of 3 and other document has a 3 frequency of 8. So we start with the only 3 and then.

 **V. MAHESWARI** 53:34  
You mean the smallest one? That is, when you have a series of and. Yeah, right. Say for example, if you have an another hand operation to a series of index that you can do. That's a query optimization which we can do that's on the query part, right?

 **NAVEEN KUMAR** 53:34  
Number of comparisons will be same.

 **V. MAHESWARI** 53:51  
Anything can be done on this linked list.  
Can modify this linked list so that the comparison can be reduced.  
Anything any implement any implementation part can be done on this linked list. I am asking so then we can really say for example here in this here after 2 we have 31 here right? So in between you have so many comparisons we shall dash here.  
Which can be avoided, right? So what can be done to avoid these comparisons? We I I come back to that, what are all the strategies which can be applied that as you said starting with the smallest frequency is the query query optimization we are trying to optimise the query so that we reduce the number of comparison.  
We come to that next sit clear. So this is just applying the Boolean query on your model and retrieving the result. Now this is just a small example.  
Can you try this? Yeah.

 **NEERAJ KUMAR TALREJA** 54:50  
So if we use the node list, if we use inverted list so then we know that OK from where to?

 **V. MAHESWARI** 54:56  
If we use.

 **NEERAJ KUMAR TALREJA** 54:58  
Inverted list, so instead of ascending and we you the descending list.

 **V. MAHESWARI** 55:04  
What what makes the difference?  
I think right, can we reduce it?  
Even whether its reverse order or in the increasing order or decreasing order right, will it will not have any impact right? I I hope so.

 **NEERAJ KUMAR TALREJA** 55:21  
Comparison.

 **V. MAHESWARI** 55:21  
I think that won't be a solution. Yeah. Anyway, we need to retrieve the documents matching documents, right? Whether it is in this or visiting order or assigning order, right.  
OK, just can you try this? We will have a break for 5 minutes.  
Meantime, can you if its a problem right? I am just given which users this Boolean query right compute the hit list for this Boolean query. Just can you try it?

 **MANISH** 58:15  
Sunana mere Ko Siddharth, late Mein de do bas.  
Amini, Abhi, dukhi.  
Abhi Kuchh nahi suni.  
Tomorrow.

 **DIVYANSHU** 58:58  
He's mute.

 **V. MAHESWARI** 1:01:28  
So moving on.  
Yeah, you take the features in mean size.  
Most work case scenario, but you have the best case options.  
In the worst case, takes maximum of twice. OK, your video is usually your worst case scenario. That's what the mean of XY. You are taking the mini homes X. We see query optimization part where you can reduce this the number of comparisons. OK, right now coming back to this question, the actors which are listed here.  
Not actually 610121561015. What is the exactly answer?  
Paris and France.  
Its already the paramount.  
Someone has given 6101215.  
And sells all the 15 records so Paris and not France.  
This will be the answer, am I right?  
Yes, all the 610.  
6 and 10 will be the output of this proprietors and not France.  
Yeah.  
Senthil, OK so the this one, your result will be 6 comma 10.  
Or layer right? So it will be 6101250 right?  
Answer is 610 12:50. AM I right? OK, fine. But Rohit, you have given 61050. I think you have left out 12. OK, just a simple Boolean query to understand how it it works with your list. Your in your linked list. Inverted posting list. OK, right.  
That means.  
Actually according to this.  
Indian parade, right?  
First up, since this is given within the parenthesis, right?  
We are taking this and then the output of this is combined with layer OK.  
Maybe if you use this broad mass.  
We refined actual query. Will you answer as you said, yes.  
According to this order, right, the inner bracket is sold and then if you just an hour operation is done, it will be because it doesn't know the what is the output of the previous one where actually it is. So it will be literally it will be like the 6101215.  
Suppose if it is not and not France earlier, right? Anyway we can say 1650, right? Some tricky question.  
OK, fine. Right.  
Literally, according to this execution, you will get only this 16101215, which may be logically not a correct answer, right? Am I right?  
So what do you want to search is you want to search for all the.  
Documents which has Paris but not France.  
Or it should have been air, but this layer appears in both the documents 1215 which where in your Francis also there. So it overcomes this. That's the reason we are getting a 12 and 15 OK.  
In that case, both are not there 12 or 15, so that's why someone has given 6 and 10 as there Sir. Right. OK.  
So this around the redundancy is wherein your queries has to be refined.  
No, these going moving on into the next slide, right, your Boolean Paris, what we have seen is the exact match whether it appears its giving a precise result whether your document matches it or not, OK, there is no partial match or something it is present you will get that document otherwise it will not be listed here. OK. But it is very simple model to build the on the IR system. OK and this has been the.  
Not has been till now. Its a commercial retrieval tool for more than 3 decades and if you see the applications where in this Boolean model almost you can see very simple example is your email search.  
So where we need this Boolean model actually where you you don't go for ranking like we are not most prominent about ranking in that case we can go for this email search so your sorry Boolean model search email is one thing just you give a keyword to finish all the emails sent by someone or where that keyword appears right. So in that case these are all best suited the Boolean model because it's easy to implement.  
Boolean model is one thing and then the library catalogues wherein you want to search for a particular book or an offer or something.  
You don't. You don't need any ranking, right? You are just fetching all the books in that case. So these are all some of the application simple application there in this Boolean models we can do that OK where it can be used. You don't need any ranking or something here only The thing is it will be retrieving the relevant.  
Documents and if you see some commercial search engines where they use this Boolean model not not a simple Boolean model, they want to extend this Boolean model.  
OK, add something to make it the they want to use the same boolen model but added features right? So that is what we call it as an extended Boolean retrieval model which is used in some of the commercial search engines in rather than search engines retrieval systems. You can say right? So one such retrieval system commercially thing retrieval system available is the vessel vessel is a legally searching engine. We can say right, the most use for the legal purpose they.  
Used to search this here. They don't want any vector model or vector ranking or something.  
They want to record the retrieving all the legal documents corresponding to a particular query. Particular information need that.  
They use only the Boolean model till now, right for retrieving the documents, but refining the Boolean queries. That's all right. Rather than using just a simple Boolean query with and or operator, they're reframing the Boolean query so that it can be more precise in retrieving the results. So that is how these search engines are using, and one such example is West Club.  
You can just go through your textbook. Also, as I said, your textbook has a lot of information in that. If you see the chapter one, it will be very small chapter right? But the content each and every sentence carry lot of information in that textbook. So please go through your textbook after this lecture, right. It will be very interesting to read that chapter in your textbook so they have lot of examples, lot of exercise which you can go through that so this.  
Is a vessel an example which is discussed in your this book, right, which deals with this extended Boolean retrieval model, right?  
So what do you mean by this extended Boolean retrieval is rather than using the same query, right? Just a simple query. Proximity operators are added into that so that you can refine your results, refine your query, and so you can see this long precise queries with appropriate approximate proximity operators. The proximity in the sense how much that is, we measure the distance wherein that each and every term appears. That's all so.  
They you can see these are all the proximity operators which are used in generalised. If you say it will be slash K slash P.  
Slash s these are all some of the proximity operators which are used as far as this vessel information retrieval system is comes slash P in the sense which it appears in a particular paragraph s means, which appears that, but terms which are pair in a particular sentence K is in a proximity of K words, say for example.  
And also the vessel, right? The user can give the queries like not as a query as an information made. So just a question is posted by the user, right, the statutory of limitations in cases involving the federal taught clients act. So this is a question which has been given by our information need.  
They want to retrieve all that illegal documents relating to that right, so internally its converted into this query, right? This is what we have been talking about and information need which is converted into a represented asset way. So this cannot be given as an input to your IR system. We need something has to be done. So this is a extended Boolean query which has been generated. You can see here this is the.  
First one limit right limit followed by exclamation mark.  
Simple your language, you know operators which are used. This is it. That means it starts with the word limit. All the verbs will starts with the limit, say for example limited limitations, right? All those can be covered by this word limit and then this is a proximity operator slash 3. Between these 2 words limited statutory that is.  
These 2 terms appear within a span of 3 words in middle. In between we can have 3 words. That's it, right? That's the meaning of this one right 3 words distance.  
Of 3 words this way or that way, anything similarly here. If you say Slash 2 right it it. Obviously the span of 2 words between them these 2.  
Hot and climb. So they are trying to retrieve all the documents wherein they appear in the centers. Similarly, if it is sentence, that is where reaction and federal appear in the same sentence. So all put together it is it will try to retrieve the documents wherein it updates. That's what is given here. Proximity operators within 3 words within a sentence and within a paragraph. And here there is no.  
Boolean operators in between this. If you see this query right, just its given as a long precise period without any operator. So what actually will be the Boolean operator here?  
So these are all the terms 123456 terms are here, right?  
So what actually the Boolean term is used here. Boolean operator, sorry.  
Which is used here.  
Alright, by default you can see here then this sentence space is disjunction and not conjunction. So someone was asking about DNF last class if I remember right. In your class only talking we are talking about the Boolean border, the DNF dejective normal form right which is of sum of.  
That is separated by us.  
This is nothing but each and every term is limit our statute or action or federal or talk along with that added proximity operators are there to refine your results, right? So its not an and operator its an R operator. So what is the difference between using an R and an and here what will happen instead of using R if an and is used OK.  
What will be the difference? What makes the difference?  
In using an R or disjunction and a conjunction.  
Can you tell me?  
In terms of results, I am asking displayed results.  
Number of matches, things exactly. Number of matches will be very less if you use an and the result will be. May be very few because since you are putting Anand right including all in a single sentence may be few. So that's the reason it uses this disjunction. Now in this scenario if you Google search engine right, if I give a query right anything right, any word right or I say machine learning algorithms right. I give a sentence for searching machine.  
Learning and problems. Will it use a distinction or conjunction?  
Google what they can do.  
Internally, all these terms are considered as disjunction or conjunction error and R only, yeah.  
By default its an or operator only. So either machine or learning or algorithms. But the if all the 3 appears.  
Will be considered as the most relevant one, and the ranked as the top most one. Its not an ad, its an or operator which is internally it is converted to all the queries which you give. So this is just a simple example to show you how an extended Boolean model is used right.  
This is an another example query, right? And this an information need how it is converted into this and when you give within quotes exactly as such it should appears the trade secret together OK.  
The exact phrase should appear OK, that's the now this is.  
Another exercise which is given in your textbook only write a query using Whistler syntax which would find any of the words professor, teacher or lecturer in the same sentence as a form of the verb verb explain. This is just a information need this one.  
Yeah, this is the information which which is given information on the legal theories involved in preventing the disclosure of trade secrets by employees formally employed by a competing company, right. So this is an information deed which has been converted into a query, right? You can see here.  
Trade secret, since this is the keyword they have, it's given in the double quotes. That means it should appear as such as a phrase.  
Secret should follow trade right should not change it you this should appear as such in a sentence.  
Then disclose may be disclosure disclosement anything it starts with disclose prevent employee all should appear in a single sentence, right? Just its converting and then trying to search it.  
This is just a interesting question. Any answer for this? Write a query you for a wesla syntax which should find any of the words professor, teacher or lecturer in the same sentence as a form of the verb explained. Any answer for this?  
List.  
We need to search all the words. Professor, teacher, lecture right in the same room. You can continuously give it professor. Teacher.  
A lecturer right, I am just writing it.  
May be in a same sentence all put in a same sentence and you.  
Need the verb explain, right? Yeah, put slashes everywhere, right? All in the same sentence. Put a slashes here and finally explain the verb explain and of the form of the verb explained, right. So it can be any of the word can start with an explain may be explanation explains whatever it is right. That's why.  
Explain explanation mark.  
This one possible and again I have missed the slashes here so we can put a S since they all come in the same sentence, right? They can put it right just.  
OK, so this is the deviation of your Boolean simple Boolean model which can be extended. This is what many of the that the IR retrieval system right, they don't use the simple, but just to refine it so that you can get your more relevant and precise answers for your query. That's it, right. So with that.  
You know, finished off and move on to the next topic. That is query, yeah.  
Yes.

 **DIVYANSHU** 1:18:44  
It is any any world, there is no significance of any.  
Previous question.

 **V. MAHESWARI** 1:18:50  
This one.  
Here.

 **DIVYANSHU** 1:18:54  
I'm.

 **V. MAHESWARI** 1:18:55  
To find any of the words.  
Whether it up anyone appears. Oh, you mean this? This its an hour only. No right. Its a distinction. If any one of them appears or all put together appears it will be right.  
By default its an hour either professor or teacher or lecturer, lecturer. So it will retrieve all the.

 **DIVYANSHU** 1:19:18  
Find all of the old professor, teacher, and lecture. What do I need to do?

 **V. MAHESWARI** 1:19:24  
If you want to, you need to explicitly specify and you need an and operate that has to be specified in the way. OK, I want all the words to be up here.  
Professor and a teacher and a lecture here instead of any. And OK, all should be in the same sentence if you make it as all, then explicitly an and has to be given.  
If by default its an hour. If you need an and.  
Refined baby and operator will be there, right, professor and teacher. And election. You need to add the bull in operator and OK.  
So in Besla will is using that and operator along with all these things.  
Or its not there but and is there right? I hope like OK.  
Right.

 **DIVYANSHU** 1:20:15  
Simply I need to write and P and slash St and slash s.

 **V. MAHESWARI** 1:20:16  
OK.  
And tea. Yeah. Yes, exactly. That's all that Weslan uses explicitly and has to be given, but R may not be given, right.  
OK. Next we will move on into the query optimization. Naveen, does the book cover the query generation part? You mean this vesla?  
Its the very generation, yes its there see as far as this textbook is concerned, it will be very precise and very little info will be given. We have to elaborate it. Its described there you can see as a one single paragraph everything is described or whatever I have discussed. Just read it right like very how its converted into a query is not given actually.  
Right.  
In order to understand that, we need to move on into the besla. Actually, yeah. You can go into that Vesla website and you can see you can ww.wesley.com fine. So now we are moving on into the query optimization as sometime back. Someone arnab, I hope right, you were talking about the minimum of X&Y, right? Retrieve that so that we can apply in this query optimization. That's what we have been discussing was the worst case scenario now.  
If you want to reduce the time complexity, how it can be done so?  
As someone has said query of take the minimum frequency right, consider a query that is a series of and connected with and terms for each of these n terms, get the postings and then and and them together. OK, so if you have Brutus and seizure and Calpurnia.  
Now in this example actually.  
It will be executed in the same order unless it you have a parenthesis. It will be order in executed in this order right first Brutus and Cesar will be evaluated and the resultant document list will be compared with your cadparina and finally your result will be executed. So in that case.  
Brutus and Cesar, right, the normal. Same comparisons will be done and it will retrieve the matching documents from this and that will be compared with your calpoli.  
That's the usual scenario. Now if you want to optimise this query. If you see this, your final result right, final document result will not be going to exceed 2 because Calculina has got only 2 document. So definitely it will not exceed this.  
13 and 16 will be any option, or it may be less than that. OK.  
So in order to do that right, what you can do is instead of comparing Brutus so you can rate cut the candidate terms which are you are going to compare which are you are going to merge. So in order to reduce the number of merge operations or comparisons or intersections rather than combining these 2 right, these 2, Brutus and Caesar you can start with the one which has the minimum number of documents.  
Wherein we are posting list is minimum. So in this case Calparina is the first one. You can sort it in the order of the document.  
Inverted list size of your inverted list. So if you sort it calparina will be the first one grouped us next followed by C cell. So you can do the and operation like this calporina and Brutus and then you can apply this result on your seizure.  
So that it can reduce the number of comparisons it it it is here, it need not be told be a maximum case X plus y. Definitely it will be dependent on the minimum value right? Can reduce it. But if you have 2 things.  
Right. When you have only 2 terms in your query previously, what we were discussing is only 2 operator, 2 terms and an ad operator there. Your doc like size of your inverter list does not matter right? You need to compare.  
Even though you take the smallest one and do the comparison and the the the document ID value also matters. So in this case, in order to cut short that that candidate items, we can do this calporina and Brutus followed by the seas. So definitely the number of comparisons will be lesser, right? So that is one kind of query optimization which can be done Sir to reduce this. OK.  
Right. I hope you are able to follow this rather than doing this.  
Processing the order of increasing frequency. Start with the smallest and then keep cutting it further. OK. So in this example you can first to do the calparina process followed by Seaside. So definitely the number of comparisons would be reduced. OK.  
But this optimization is applicable only for a series of N terms, not for R right R as such, I have already told you just you are going to merge it, add it together. So no optimization technique will work on it, right? Just merge to both the list. So in that case, so not much optimization racket for R queries. But if it is a series of and you can apply this to reduce the comparisons and.  
This composite queries which has R and an in this type right of.  
Ind OR Fire, Thunder or lightning. Right. These type of example. Also you can use this query optimization.  
When you have a series of Ant, here again, get document frequency of all the terms. You can estimate the size of this R query right. Estimate the size by just adding. Adding that document frequencies but here.  
Your duplicate IDs is not taken into account, just we are adding the document. Yes, in the increasing order of our sizes. Now in this here when you do this right I am getting the size of the process in the order of the frequency right? We need the size of the inverted list.  
Total number of documents which appear in the inverted list, so that is where you need this document frequency. You got it earlier in the inverted index we have stored the document frequency. You remember right the number of documents it appears so nowhere it it it is already stored bruptus it appears in him.  
7 document SE, Sir, it appears in 8 documents and similarly calporina it is in 2 documents, so this is nothing but your document frequency. So this document frequency can be used.  
OK, to do the comparison this you can sort it according to this document frequency and do the operation.  
It.  
This.  
Generally, for your our cases also hope you are able to follow this any queries.  
We can build the comparison logic based on the based on what.  
I think your question is not completed based on the number of docs. Yeah, that's what this is. What we have already recorded. You remember in the inverted index we have stored the document frequency. So this is where your document frequency plays a role. OK, in the query optimization.  
Whenever you have a optimised queries means document request right?  
You remember we have stored the recorded the document frequency earlier, right?  
Now these are all some of the problems right? Again given in your textbook. Only example problem nowhere. You will see some 4 documents are given here. Document 1234 and patient given here is.  
And draw the term document incident matrix for this document collection.  
Draw the inverted index representation for this collection and do that and he nothing. Nothing is mentioned about preprocess or nothing right? So in that case you can consider each token a sector, right? It can be considered. So what will be the size of the term document incident matrix?  
Many rows and how many columns?  
Of T comma d right? Or it will be the size of this incident matrix term document incident matrix for this example.  
The answers.  
You can do that right? Just I am asking.  
Size of this matrix T comma D.  
That's what we have started with, right? We we started constructing a term document incident matrix so.  
4 documents are there so D is equal to 4. That means 4 columns.  
Ended rose.  
What is the value of T?  
Need to take the unique terms right, so unique terms are 12.  
34.  
You say this is 5.  
678.  
Me.  
OK. Yeah. So we have 10 unique terms. The size of the matrix is 10 by 4, right. So the total entries will be 40 in your term incident matrix wherein you record for each and every entry the document were in present. So first we will if you again if it is organised in your acceling model, maybe it starts with the.  
A, B is there breakthrough, so breakthrough in which document it up is document one only. I think 1000 will be the term vector for that particular one. So that is how this term incident matrix is generated. If you want to construct the inverter index same same 10 documents but only thing is we need a list like this we will say document ID document one or D1 or something document one similarly.  
For the next thing is limit right? So just a simple example to understand the concept which we have discussed. OK then.  
This is another problem. Recommend a query processing order for so you have a combination of R operator and an and operator. So as I said, whenever you have R just you will be adding up both the posting list right? So we will neglect we assume omit the dummy piece dummy duplicate elements so we just take assets. So what you have to do is you need to multiply add.  
All the values and process them in the increasing order. That's it for this query you need to add up all the values.  
And process them in the increasing mode. I think so. These are huge numbers. It has to be added and accordingly it will be it has to be done.  
Maybe you can check it outright. So that's it. That is how this problem has to be solved and.  
So that's all about Ms I I moving on into another part interesting part so that any worries, any questions you have till now.

 **KRISHANU CHAKRABORTY** 1:32:02  
Ma'am, just one question. During this indexing process.  
I mean in these examples like we are considering all the terms, but in real life, are we actually considering all the terms? Then it would be huge, right? I mean.

 **V. MAHESWARI** 1:32:16  
Yes, yes, exactly. But in order to maintain your precision and recall, right, the if you take Google, they stop.  
List actually, but now they are not even removing the spotless stop list, but they are trying to normalise it may be say for example you have various forms of verbs, right? As we have seen, now limited limitations or something so they are trying to normalise it and then convert it into terms. But the inverted index will be very huge.  
You need to store it and you need it to search it. If you take the Google search engine right it its the index itself is a distributed index generation only. You cannot generate it at 1:00 place its a distributed index and second thing is what we are discussing all are static thing. Imagine when it comes to a dynamic inverted index right? So you need to generate inverted index dynamically and then you have to put everything together right? So the complexity is there.

 **KRISHANU CHAKRABORTY** 1:32:58  
Yeah.  
Rupaya.

 **V. MAHESWARI** 1:33:17  
Imagine the mission we are supercomputers are used by Googles for doing this, generating the inverted index itself.

 **KRISHANU CHAKRABORTY** 1:33:18  
Hello.

 **V. MAHESWARI** 1:33:23  
Right.

 **KRISHANU CHAKRABORTY** 1:33:25  
SM.

 **V. MAHESWARI** 1:33:25  
Definitely your invented index size will be very huge.  
That's where they use various compression techniques so that the searching can be isolated. OK, fine. So yeah, with that we move on into the last part of this, that is evaluation evaluating a Boolean model. So what can be done for an evaluating a Boolean needle? What do you think?

 **KRISHANU CHAKRABORTY** 1:33:37  
Customer thanks.

 **V. MAHESWARI** 1:33:50  
What has to be evaluated in a Boolean model? Because we don't have a ranking, right? So how do what factor can be evaluated you consider?  
The Boolean model.  
Any answers?  
We are not having any ranking in your Boolean model. You agree that. So in that scenario.  
What is there to evaluate in your Boolean mode?  
Free speed. OK, that is why we are using this optimization technique. All those things, right?  
Evaluating your results how your Boolean model works right?  
Number of comparisons. OK, fine then. So these are all your internal systems.  
Accuracy.  
So what do you mean by accuracy here?  
Not actually accuracy you.  
Means. Can you tell me?  
If not ranking right, what is the other feature Boolean model is associated with that when we are retrieving results you we know the definition of an information retrieval, right? Yes.  
Exactly.  
There we have received the same document. That means document whatever is retrieved is relevant or not. So when I say relevant right, you may think because you are recording only the presence and absent of a particular term. So your document Boolean model is going to retrieve the documents where that particular term appears.  
Definitely you may think its a relevant one, right? How the irrelevant documents are going to be retrieved definitely.  
You agree that.  
Retrieved documents may be relevant both.  
Relevant and irrelevant it will be. Will it be a combination?  
Or not.  
So you see here documents retrieved or relevant to the information need or not. That's what we want to measure. OK, what do you mean by this relevancy? You your say for example your.  
Input has a set of relevant documents or they are retrieved.  
Are all the document relevant? Documents are retrieved for the information need say we can take the same vessel itself. If a query is given, are all the relevant documents related to that query is retrieved or not? OK, that's the first question. All relevant documents are retrieved or not. That is here you can see you have a set of relevant documents. If all are retrieved then its 100%.  
Right, you are getting the results. Suppose if the retrieved documents you have a set of retrieved. This is. This is the percentage which is retrieved. That means these are all left out. OK your query or your IR system is not optimised to retrieve all your relevant records. So how to either you can refine your query or you have to. Maybe you are indexing it also matters you are preprocessing all those things. Timing lemonization removing stopwatch all those.  
Matters in the relevancy. Retrieving the relevant documents also. So this is one scenario.  
All the because 100% relevant record Ru may not be possible, but you can increase this the percentage of retrieved records are relevant. OK, this is first scenario. All relevant records are retrieved.  
Second one is.  
And all the documents retrieved are relevant. So here previous case only the relevant documents are retired. Sometimes irrelevant documents are also retrieved, right? So here you can see we have a set of documents which are retrieved out of which only this percentage is relevant. OK. Whereas you can you have a doctor.  
Documents which are not relevant to our query, right? So this is also the. So we need to measure these 2 and appropriately tune your model to reimprove this. OK, So what the evaluation metric which is used for this?  
Might be familiar with that, right?  
What are all the evaluation metrics which can be used for evaluating this?  
How to measure whether your document retrieves are relevant or or all the relevant documents are retrieved.  
So I have already familiar with these metrics, right?  
You have. If you use the input as confusion matrix, right? What you are going to estimate exactly as Kapil has given precision and recall. So you might be familiar right? F score, precision, recall, right? All these are all.  
Associated metrics.  
Exactly. So the 2 evaluation metrics which are used which can harbour model evaluation is precision and recop OK?  
So the precision is usually how it is defined is out of the returned result, right? Whatever result. What fraction of the returned result are relevant to the information neighbor. So measuring this is what you called as. Precision and recall is from the entire relevant records, right? How many are retrieved by your system? What fraction of the relevant records in the collection was written biosystem so.  
Both precision and recall matters. Whenever given a query, right? Both precision and recall matters because it varies.  
But your model has to try to improve both, so this precision recall where it gets affected is based on your query based on your preprocessing techniques, there are various matters where this precision and drain call gets affected, right? So before moving on into this, your search engine right when does they focus on only position, only recall or both? What do you think?  
Both actually earlier they were not bothered about the recall initially recall was not a matter because they don't know the relevant documents. It cannot be measured, right? How how will they explain relevant records? So recall was not a matter, only the patient. They tried to estimate whether the result relevant results are retrieved or not. OK, but now they.  
Current scenario, the search engines they focus on.  
And break off right?  
Now coming back to this, if a is the total number of relevant records in the collection, right, we have a a number A and C is the total documents retrieved for a given query test, retrieve C documents out of which we are relevant. OK, so A is the total relevant documents, C is the document retrieved as a resultant of query out of which B are relevant. So now what is P and what is recall in terms of a BC?  
What will be this fraction?  
You define this precision and recall in terms of these 3.  
Parameters a, B&C.  
B is b by a.  
Recall us.  
What about recall?  
You say precision is B by a.  
C by CC by A, B by a by C.  
Decision is actually.  
Out of the total results, how many are relevant? Am I right?  
So precision is.  
Total document retrieved is C out of which we are relevant. Measuring that is your.  
Precision, whereas recall is out of a relevant documents. How many retrieved or relevant. OK, so measuring these 2 is you. So your denominator is going to be the same. Numerator is going to be the same. P OK, both the cases BB is the relevant documents which is retrieved right, but only the denominator is meant to change. You see.  
Your precision is B by C and recall is b by a. Hope your inductive follow it because the fraction is.  
For me, the relevant documents are retrieved out of the total and out of the relevant available. So pre call is B by A and precision is b by C which is more difficult to achieve out of these 2 right which is more difficult to optimise.  
Definitely its recall right, you cannot retrieve all the relevant documents, but you can see to that right out of the retrieved documents. You can at least increase the relevant documents, right? So these 2 measures are used to tune your model. Improve your boolean model. The main thing is your query can be refined and.  
Your model can be tuned based upon these relevancy score, OK.  
Say for example, can you give any scenario where your Boolean model lacks this relevancy, right? Or it retrieves an irrelevant records?  
May give any query any any example query how a Boolean model will retrieve a nonrelevant documents for a given query.  
Any example?  
Which comes to your mind, right? Since we are talking about.

 **NUGGAHALLI RAMANUJAM AJAY** 1:44:20  
Word that is having a word that is having multiple meanings, ma'am. Yeah, yeah.

 **V. MAHESWARI** 1:44:22  
Yeah.  
Multiple meanings, OK any example?

 **NUGGAHALLI RAMANUJAM AJAY** 1:44:29  
Apples, fruit and company.

 **V. MAHESWARI** 1:44:31  
OK, yeah, apple. So as as discussed earlier, right, when we type Apple its giving rather than the real apple its giving on the Apple Company right? So Apple incorporation it will be giving. So that is based upon the information need of the user am I right so that can be refined here the question is not properly given just when you give Apple.  
Your IR system does not know what user has in mind. OK, whether the user is trying to retrieve details related to your real apple or your Apple incorporation. OK, so in that case, what it will do is.  
It will show both results OK.  
Associated with Apple Incorporation and Apple with food. OK, that's one thing which it can be refined using your feedback. The based on the clear links which the user clicks, that relevancy can be improved. OK.  
Any other example based on the query right?  
The redundant information can also be retrieved ambiguous statements redundants Jaguar. Now when I say the same example as we have been discussing mission OK I will give you a question like this right? I give the query as machine language algorithms or something.  
OK, ML algorithms, this is my query. I want to retrieve all the documents wherein which are relevant to machine language algorithms. Now I give a query like this to a Boolean model right? I have a set of documents. I want to retrieve all the documents matching this way. So what you think as irrelevant words.  
That will be filtered for this.  
Retrieved for this query.  
Leave the relevant documents right. What about the irrelevant documents?  
Moments which actually does not, it does not suit my need, but that may be retrieved for this query considering it as a match.  
Any example?  
Anything which comes into your mind when I give this question wishing language algorithm I say right?

 **MUKESH KUMAR YADAV** 1:47:04  
Documents which are not containing all the 3 words, just one word may not be relevant document.

 **V. MAHESWARI** 1:47:10  
Morning bird. That can be OK. That's the least possible, right? OK, right. Even then, I suppose if I give.  
We assume that I give Anand operator right. I give machine and language and algorithms. I am just I am still refining my query right? Instead of RI am making it as machine and language and and further still there are possibility of retrieving irrelevant records agree if so.  
Any example which strikes your mind?  
It.  
So The thing is how this irrelevant records here come here is may be you are thinking about a ML model right? But in your document a machine can be a word appearing as such somewhere and language can be something right? It it may be related to some terms. A language as a term it appears and algorithm somewhere it appears. But all put together may be does not correspond to a machine language algorithm, right? It is not at all relevant in machine language.  
But since all these 3 terms appear in your record in your document, it will be considered as a match and it will be given to you why its appearing here is because the context is not at all semantic or context. Match is not evaluated by your Boolean model. OK its not checking for ML model. It doesn't know the context for which the user is searching. Actually that is the reason why we get this irrelevant.  
So that we cannot avoid in your Boolean model, but we can be improvised only if you go for a phrase search or a semantic search, right when it comes like a machine and a language, it corresponds to an ML. It has to understand your model has to understand this and then accordingly, I believe the words because its considering each one as an independent term. He does not know the the context contextual relationship between each and every term. That's the reason we get.  
Irrelevant records, but that we need to fine tune your model, right? That's the reason. Yeah, similarity semantic similarity should be there. Contextual similarity should be there. All these things has to be taken care. But what we are discussing is a simple model. So definitely we will have this precision and recall we cannot go beyond much thing it unless you do these all these.  
Enhanced versions correct. OK.  
So just to understand this, I am giving this example. That's all. OK. So with that this chapter is over and one question example. Problem 3 is given right. You can go through this and try this problem and come out with an answer. We will discuss in the next class see consider 4 terms. Sorry. Consider the 4 terms in order Park Mountain trails and difficult. We have the 4 terms.  
Assume that the query is in dejective normal form, that is it, DNF. Is it connected with ours, where R is the logical and. Here you have the conjunction. This is the conjunction query. Can you give the description of the query first? One second thing you have 2 documents. Document one and document 2. What will be the result of the query when applied? That is, if this query is applied.  
On these documents, what will be the resultant documents whether document one matches 2 matches or both matches but none of them match? OK, just you give it. But what is the implication of this query also right its its an interesting query you just go through this and write we will discuss the result in the next class if you OK you try this example problem.  
So that's the end of this session. I have given some papers here. You can go through this.  
Is there any queries any doubts in this?  
Today's lecture.  
1110 OK, which document will satisfy?  
Anything any doubts, specific doubts or specific queries you have.

 **KRISHANU CHAKRABORTY** 1:51:46  
Ma'am, regarding the query optimization. So like you know, I mean if we look into high level then in that case you know query will be provided by the user. So I mean.  
User might not be, you know, avoid about the optimization technique. So once the query is being provided then you know its the responsibility of the IR system to optimise it. OK OK, understood.

 **V. MAHESWARI** 1:52:11  
The IR system. Yeah, yes, yes.  
Very optimised that very expansion, very feedback all are there, right? That's a separate component which takes care of everything based upon the feedback, the thing and when you type any query.  
Mistake in your Google right you spelling mistake. If you make right what did you ask? Did you mean its asking right? You get a sentence. Did you mean? So that's a spell checker will which how it identifies that you have made a mistake right. And its refining the query and giving you a suggestion that spell check model which is include later on we will see right how that.

 **KRISHANU CHAKRABORTY** 1:52:36  
Yeah, right. Yeah.

 **V. MAHESWARI** 1:52:52  
Tolerance retrieval is done in this search engines. Your IR system OK, so its a part of your query only as of now as we have seen in the vessel, right the information need is just a natural language sentence to be refined and converted into an appropriate tool. Enquiries taken care by your IR system, so you go Google, you can type whatever you want, right? So its appropriately converted into a refined query internal.

 **KRISHANU CHAKRABORTY** 1:53:18  
Understood, ma'am.

 **V. MAHESWARI** 1:53:19  
Yeah. So you try this example problem 3 and give me the result we will discuss in the next class, right? What about? And Naveen, Naveen, you have any answer?  
Just tell me right. You can work it out. And what? Actually, we come out of this problem. I want to give you your feedback. Right. OK. If if any. Any other questions, any other doubts. Hope you are able to follow this right. I am not too fast or something. Right. The speed is OK fine.  
Yes, OK.  
Means no many not much queries are there, right? If no, no questions are there right? Its a Boolean either its very clear or totally not clear for see in between fussy mains you will get lot of patience. So what should I assume its totally clear not clear right. You have a negative options. If not much questions are there so I will assume that its very clear right.  
OK, fine. We will see in the next class, right? Yeah, yeah.

 **NA.R. YANGYESWAR** 1:54:22  
Yeah.

 **RAGU RAMA CHANDRAN. M.** 1:54:23  
Madam.  
Yeah.

 **NA.R. YANGYESWAR** 1:54:25  
So for Python, what kind of libraries we will be using for this kind of?

 **V. MAHESWARI** 1:54:26  
Yes.  
At least yes, NLTK, right? Natural language. Because next we will be doing the preprocessing. I thought of talking about all the libraries there. Natural Language Toolkit is there, right? You can import that it. It has all the libraries by that. OK, no problem. All these will be discussed in detail in your webinar. Right. They will. They will explain you the libraries everything to start with. Your first webinar is planned on. Actually it all the dates has to be gone. May be next week so she.  
We should discuss in detail about your invented index generating inverted index bullet model. There all the libraries which are used will be discussed. OK may be given.

 **NA.R. YANGYESWAR** 1:55:10  
Also, when will be our assignment and request dates? Are there any announcement will be?

 **V. MAHESWARI** 1:55:16  
It you know, they have to finalise. It may be your first quiz I have planned after 4 lectures.  
Similarly followed by that your assignments also there.

 **NA.R. YANGYESWAR** 1:55:22  
OK.

 **V. MAHESWARI** 1:55:26  
They they are yet to finalise and release it, I think next week you may get it. You may be not dates. Also you will be getting.

 **NA.R. YANGYESWAR** 1:55:31  
OK, OK. OK. Thanks.

 **RAGU RAMA CHANDRAN. M.** 1:55:38  
Pranav, Raghu here. So here for this question its problem 3. So you are asking about the description of the query, right?

 **V. MAHESWARI** 1:55:46  
Yes, yes.

 **RAGU RAMA CHANDRAN. M.** 1:55:48  
OK. Is there any coding involved?

 **V. MAHESWARI** 1:55:48  
Description of the query.

 **RAGU RAMA CHANDRAN. M.** 1:55:52  
Just only that line that slash years and all you have put.

 **V. MAHESWARI** 1:55:55  
Yeah. 1010, right? Yeah. What do you mean by that? One zero and 0010, actually. What you are get trying to get out of this query.

 **RAGU RAMA CHANDRAN. M.** 1:56:04  
OK.

 **V. MAHESWARI** 1:56:04  
That's the description. OK, leave that description. What will be the result of the query when applied to this document you have any answers for that? That's what I want to find out. So how will you find out that what will be the result of the query when applied to these documents as we have done earlier, we need a mapping and then you need to find find it out because the 4 terms are given, they are path, mountain trails difficult are there, right? So you.

 **RAGU RAMA CHANDRAN. M.** 1:56:11  
OK, OK.  
OK.

 **V. MAHESWARI** 1:56:29  
Seeing that, yeah, you need to find out.

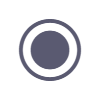
 **RAGU RAMA CHANDRAN. M.** 1:56:33  
OK.

 **V. MAHESWARI** 1:56:35  
You you try make it in the next class, right? OK.

 **RAGU RAMA CHANDRAN. M.** 1:56:41  
OK.

 **V. MAHESWARI** 1:56:41  
Thank you. All right, if not any doubt.  
Shall we wind up? Yeah. Thank you.

 **RAGU RAMA CHANDRAN. M.** 1:56:48  
Thank you.

 **V. MAHESWARI** stopped transcription