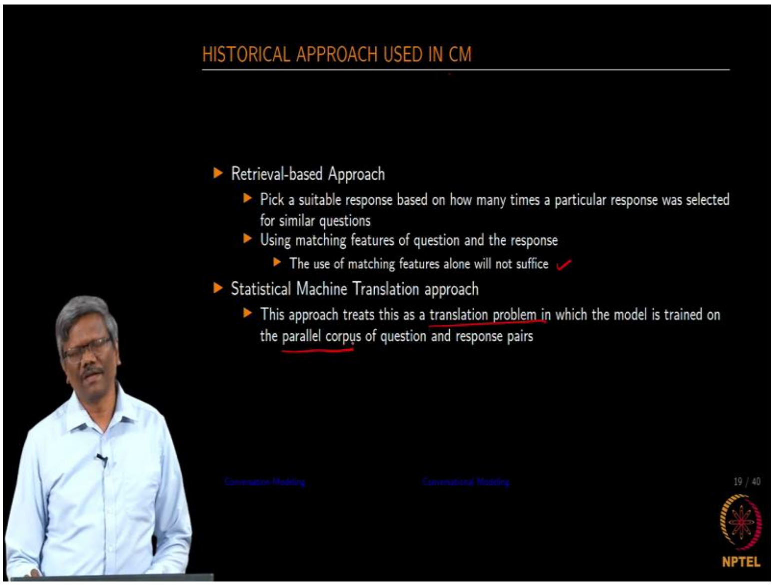
Applied Natural Language Processing
Prof. Ramaseshan Ramachandran
Department of Computer Science and Engineering
Chennai Mathematical Institute, Madras

Lecture - 87
Some ideas to Implement IR-based Conversation Modeling

(Refer Slide Time: 00:15)

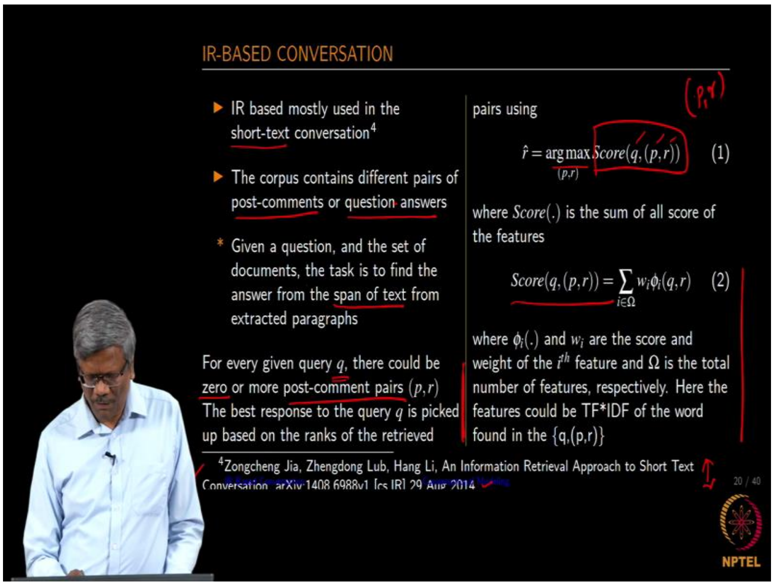


So, we will talk about, a few approaches that are used in Conversation Modeling; one is the retrieval-based approach, the second one is the statistical machine translation approach. So, I am sure you would now appreciate that all conversation modeling would require a retrieval engine as well as correct.

So, what this does is essentially it picks up a suitable response based on how many times a particular response was selected for a similar question. And, then using a matching feature of question and the response it finds the answer. So, we will learn that the use of matching features alone we will not be sufficient for doing the retrieval-based model.

So, let us look at least one example of a retrieval-based model in the subsequent slides. In this statistical model it treats this as a translation problem, in which the model is trained on the parallel corpus of question and response pairs. It is very similar to what we saw in the translation model and then some models are built using the machine translation approach for the conversation model, you would not be discussing this part.

(Refer Slide Time: 01:44)



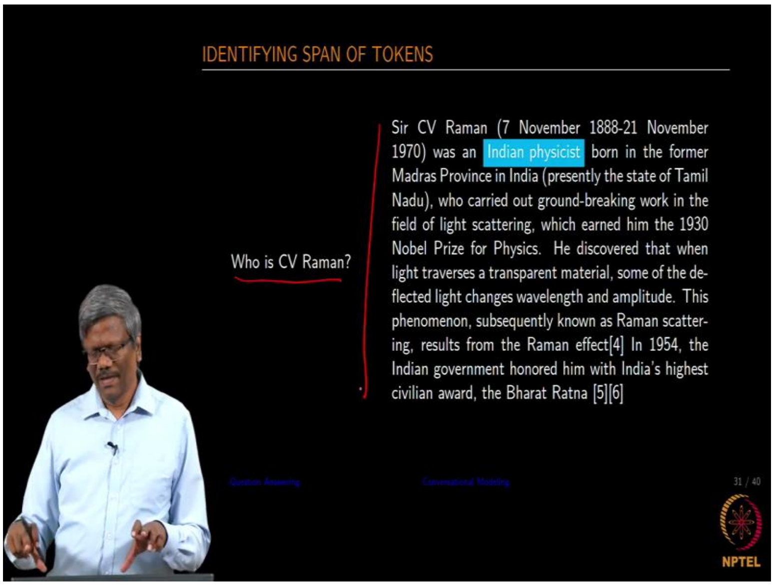
So, in the IR based conversation modeling, we require definitely a retrieval engine and then we are going to be restricting our self to a short text conversation. And, this particular example is taken from this paper published by Jia, Lub, and Li and the title is an information retrieval approach to short text conversation ok. This is this was published in 2014 again this is a research topic.

So we are going to be having a short text for the conversation modeling, which is about the 2 or 3 steps involved in it not beyond that. The corpus contains different pairs of post comments or question answers. Given a question on the set of documents, where you would find the post comments pair or question answers pair, the task is to find the answer from the span of text from the extracted paragraph ok.

So, if you look at the IR model, so what it does is based on the query that you have using many of the approaches followed in the information retrieval, you get the list of documents correct. So, you have the query with certain keywords and then you start matching the query and the documents which are there in the corpus, and then based on the certain ranking mechanism we list those documents.

So, in this fashion we are going to be listing certain post comments based on the query given ok. So, once the paragraphs where the query matched with the responses, we want to extract only a portion of the text not the entire paragraph ok.

(Refer Slide Time: 04:01)



So, the portion of the text could be the span of text for example, assuming that this is your let me take an example from ok. So, let us take a small example here to understand what the span of text is and then will go back to the information retrieval model for conversation.

So, if you search for who is CV Raman ok you get lot of a documents related to CV Raman right. So, what is that you require, you do not require the entire document that is coming in as links and you want to read they read through the whole thing, you just want to get only this an Indian physicist right. So, the span up text is when the document is retrieved through the retrieval engine you get the whole thing assuming that this is a document. I am simplifying it could be a document containing various paragraphs and one of the paragraphs would contain the answer to the question.

So, in this case let us assume that this is one document, this is the whole thing is one document and then the span of text is defined here. So, and this is the answer that we want to get this pair of text here is starting from this word ending here. So, these 2 words that you want to pick. And, then there is a start and there is an end. Let us assume that this is in the 10th word. So, the 10th word to 11th word is my span of text, which contains the answer to this question. I am just making this number up.

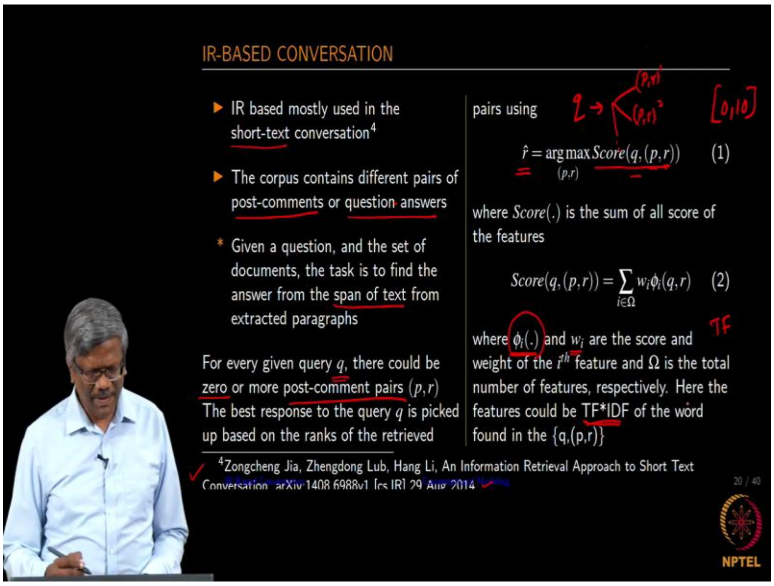
So, now, we know what that span of text means right? So, given the question and the set of documents, the task is to find the span of text from the extracted paragraph. For every given query q, there could be 0 or more post common pairs ok, this is given right. The best response to the query q is picked up based on the ranks of the retrieved pairs using some ranking mechanism.

Let us assume that the score is obtained using this. And, there could be various scores you would obtain based on the query post and response pairs. And, then you have some kind of a mechanism to score the values, and then once those values are obtained you pick up the maximum of that score and then say that that is the answer to the query that you have just posted.

I am sure you will be able to understand this I know it, I would like you to go and then read this paper I am not going to be covering the entire paper, I just want to mention that this is one of the ideas that is followed in the information retrieval based a conversation model. Where you have the query, you have the paragraph, and the rather the post and the responses for these short text conversation, you try to match the query and the post and the response kit set of the post and the response pairs. And, then try to find the score for each of those right, for the query and the post and response pair.

So, if you have more of p r you will have more of this based on this retrieval model, you will have more of this and then you try to find the score for each of this using some scoring mechanism. And, then the scoring mechanism is defined as follows.

(Refer Slide Time: 08:17)

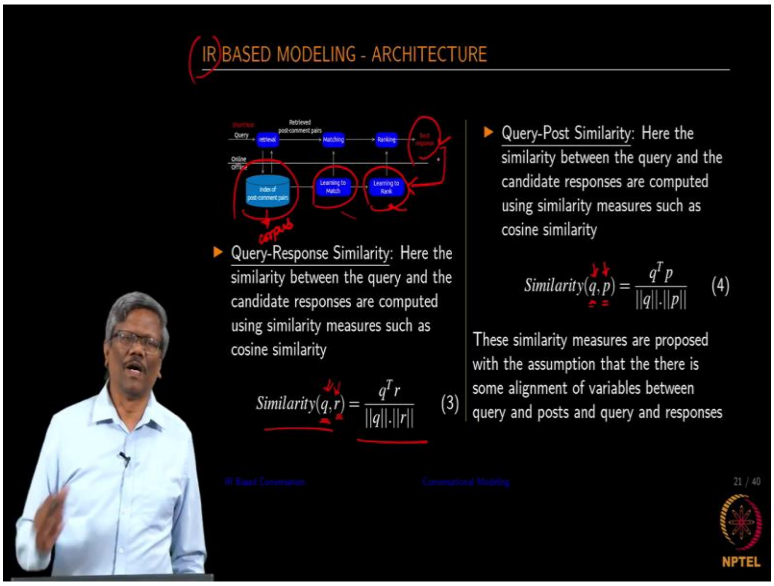


So, what we are going to be doing is we are going to be finding the score, based on the features that we have extracted from both query posts and the responses. So, let us assume that wi is the weight, and phi i is the score.

And, then what is the kind of feature say that we would use we probably would have used term frequency or TF IDF, or we can probably use some combinations of you know NER or we can use POS, or the part of speech as a set of features and so on. And, then using the combinations of all this, we can achieve a score using this model ok. And, then get various courses for the responses that we have picked up based on the IR. And, then finally, pick up the one which has the highest score right.

So, this query gives you 10 different and so on ok. And, then use some mechanism using TF IDF for some scoring mechanism to find out, how close the query is with respect to the post and the response ok. And, then use that to get the rank, and then the rank is defined by your score ok. So, this is one very simple approach using which you can find the response to a given query.

(Refer Slide Time: 10:10)



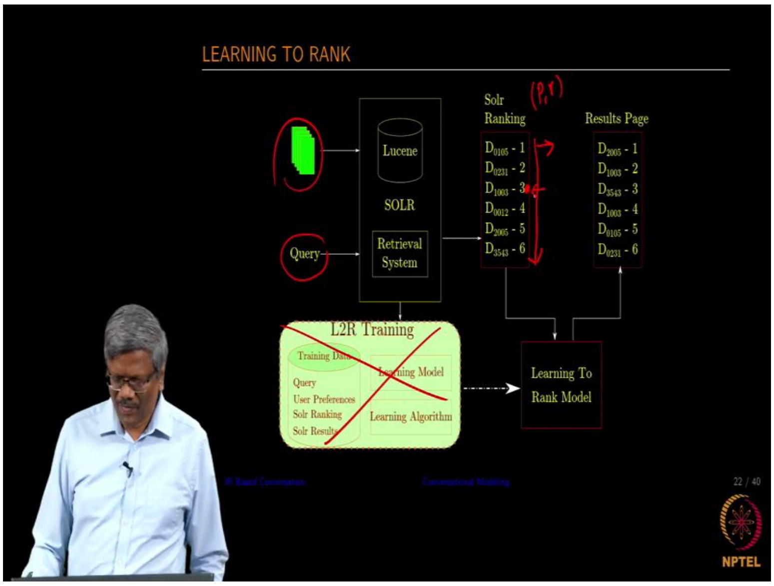
So, if you look at the architecture part of that as I mentioned there is an index of a post comment pairs that is coming from some corpus right. And, then you have a retrieval mechanism that retrieves this set of post and responses based on the query. And, then use a matching mechanism to find out whether how close that query is with respect to the retrieved post comment pairs and then provide some kind of a ranking as we have done earlier. And, finally, pick up the best response ok. The similarity could be found using a cosine similarity in this fashion ok. So, you can find the similarity between the query and the response and query and the post.

So, the idea is if the query and the response if they contain is words that are similar, then we are going to have a similarity score attached to the query and their response. In the same fashion, if you have some words which are common to both query and the post you are going to have some similarity scores, which we would be computing. And, finally, figure out how close the final result is with respect to the query that is given and that would be chosen as the best response ok.

So, it is a very simple model where you require the IR engine to get you the top 10 matching documents and then using some scoring mechanism. So, get the ranks for all those retrieved documents and then finally, get the best response out of that. So, again there is a learn to match and there is learning to rank that is also part of the system. So, how do you learn this? So, every time when the system picks up the response and then finally, the best response is chosen based on the score, the user looks at the option that was provided earlier not just looking at and only at the best response.

And, if the user picks up the second response as the best response then the learning to match and the learning to rank will start learning that ok, for this type of query user as responded with this score ok. So, meaning that I need to change the way I compute the rank. In this fashion the ranks are changed as well ok. So, this is a feedback mechanism that comes from the user. So, when the user ranks are provided that would get a higher ranking than this system generated ranks ok.

(Refer Slide Time: 13:46)



So, using that approach, then the system would slowly start to learn, and maybe after several responses from the user for this similar query, the user selected response would move up in the rank ok. So, this is one model that is followed.

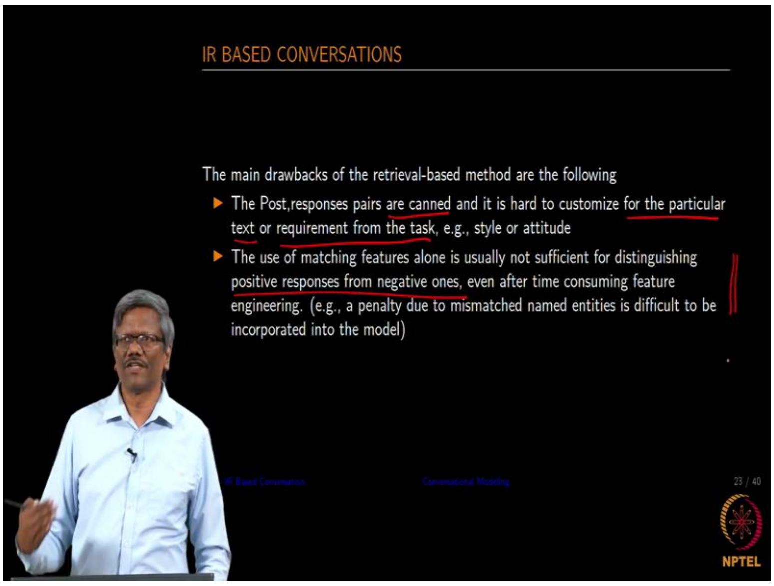
I spoke about the learning rank. So, and this is the very simple architecture of the learning to rank models, where you have the query coming in and you have the set of documents right. And, the documents are indexed using engines called solar or other indexing engines. And, then you have a retrieval system as part of this actually Lucene is the indexing engine and solar is the complete retrieval engine which provides all kinds of ranking mechanism and so on ok. Initially when you do not have the LTR a the solar provides you a set of documents with it is own ranking scheme ok.

And, then let us assume that this is the post response a document that is coming in from the system and then we will pick up document 105 as the top response for the given query. Assuming that now we have asked the user to rate the responses coming from the system user picks up that ok. One is not the right response for me 3 is the best response as far as I am concerned. And, then there are so, many users who have given let us say a similar query or the same query, and then they also start to respond that 3 is the best answer and not 105 document is the best answer we need to let the system learn that part.

So, that is where they learn into rank comes in ok. So, when the system retrieves that and then user preferences or chosen based on the clip through or some kind of a rating mechanism that you provide as part of the list. You store the query that is coming in, you store the user preference, there is a solar ranking or the engine ranking that comes in and there is a set of result that is coming in. The learning model based on the combinations of all these parameters, we will start to realize that document 1003 should be on top and not something else ok.

So, in this case what I have given is document 2005, I am sorry maybe we should have picked up this the beginning itself. So, based on the user preference document 25 should be coming on top. So, the learning mechanism starts to learn user preference as well and start to rank the documents in a different fashion. So, users later would not see this rank and start seeing this rank you know the documents are ordered in this fashion ok. So, this is at a very high level what is learning to rank ok.

(Refer Slide Time: 17:09)



So, what are the major drawbacks of the retrieval-based model? The post response pairs are canned, it is very hard to customize for a particular text or a requirement from the task example style and attitude. So, maybe I will rephrase this, you know this is fine-tuned only for the given task that is given right. So, you cannot take the same model out and then use it for some other tasks, it is not going to work we need to retrain the whole thing one more time ok.

The use of matching features alone is usually not sufficient for distinguishing positive responses from the negative ones. Even after time-consuming feature engineering ok. So, we say it what it says is the matching features are not sufficient we need to be able to bring something else as part of this. So, it is again as I mentioned right, it is based on the information retrieval query and ranking engine, which is based only on the keywords that are picked up from the query as well as from the post in responses.

So, the IR based conversation does not have any other mechanism to capture other features. So, we need to find a different mechanism in order to find you on this, when we look at this model, it is only based on the keywords that are provided not beyond that right. So, that is why it is is not sufficient. So, we need to look for some more additional features that could be fed as part of the system.