Large language models. They are everywhere.
They get some things amazingly right
and other things very interestingly wrong.
My name is Marina Danilevsky.
I am a Senior Research Scientist here at IBM Research.
And I want to tell you about a framework to help large language models
be more accurate and more up to date:

Retrieval-Augmented Generation, or RAG.
Let's just talk about the "Generation" part for a minute.
So forget the "Retrieval-Augmented".
So the generation, this refers to large language models, or LLMs,
that generate text in response to a user query, referred to as a prompt.
These models can have some undesirable behavior.
I want to tell you an anecdote to illustrate this.
So my kids, they recently asked me this question:

"In our solar system, what planet has the most moons?"
And my response was, “Oh, that's really great that you're asking this question. I loved space when I was your age.”
Of course, that was like 30 years ago.
But I know this! I read an article
and the article said that it was Jupiter and 88 moons. So that's the answer.
Now, actually, there's a couple of things wrong with my answer.
First of all, I have no source to support what I'm saying.
So even though I confidently said “I read an article, I know the answer!”, I'm not sourcing it.
I'm giving the answer off the top of my head.
And also, I actually haven't kept up with this for awhile, and my answer is out of date.

So we have two problems here. One is no source. And the second problem is that I am out of date.
And these, in fact, are two behaviors that are often observed as problematic
when interacting with large language models. They’re LLM challenges.
Now, what would have happened if I'd taken a beat and first gone
and looked up the answer on a reputable source like NASA?
Well, then I would have been able to say, “Ah, okay! So the answer is Saturn with 146 moons.”
And in fact, this keeps changing because scientists keep on discovering more and more moons.
So I have now grounded my answer in something more
believable.
I have not hallucinated or made up an answer.
Oh, by the way, I didn't leak personal information about how long ago it's been since I was obsessed with space.

All right, so what does this have to do with large language models?
Well, how would a large language model have answered this question?

So let's say that I have a user asking this question about moons.
A large language model would confidently say,
OK, I have been trained and from what I know in my parameters during my training, the answer is Jupiter.
The answer is wrong. But, you know, we don't know.
The large language model is very confident in what it answered.
Now, what happens when you add this retrieval augmented part here?
What does that mean?
That means that now, instead of just relying on what the LLM knows,
we are adding a content store.
This could be open like the internet.
This can be closed like some collection of documents, collection of policies, whatever.
The point, though, now is that the LLM first goes and talks
to the content store and says, “Hey, can you retrieve for me
information that is relevant to what the user's query was?”
And now, with this retrieval-augmented answer, it's not Jupiter anymore.
We know that it is Saturn. What does this look like?
Well, first user prompts the LLM with their question.
They say, this is what my question was.
And originally, if we're just talking to a generative model,
the generative model says, “Oh, okay, I know the response. Here it is. Here's my response.”
But now in the RAG framework,
the generative model actually has an instruction that says, "No, no, no."
"First, go and retrieve relevant content."
"Combine that with the user's question and only then generate the answer."
So the prompt now has three parts:
the instruction to pay attention to, the retrieved content, together with the user's question.

Now give a response. And in fact, now you can give evidence for why your response was what it was.
So now hopefully you can see, how does RAG help the two LLM challenges that I had mentioned before?
So first of all, I'll start with the out of date part.
Now, instead of having to retrain your model, if new information comes up, like,
hey, we found some more moons-- now to Jupiter again, maybe it'll be Saturn again in the future.
All you have to do is you augment your data store with new information, update information.
So now the next time that a user comes and asks the question, we're ready.
We just go ahead and retrieve the most up to date information.
The second problem, source.
Well, the large language model is now being instructed to pay attention
to primary source data before giving its response.
And in fact, now being able to give evidence.
This makes it less likely to hallucinate or to leak data
because it is less likely to rely only on information that it learned during training.
It also allows us to get the model to have a behavior that can be very positive,
which is knowing when to say, “I don't know.”
If the user's question cannot be reliably answered based on your data store,

the model should say, "I don't know," instead of making up something that is believable and may mislead the user.
This can have a negative effect as well though, because if the retriever is not sufficiently good
to give the large language model the best, most high-quality grounding information,
then maybe the user's query that is answerable doesn't get an answer.
So this is actually why lots of folks, including many of us here at IBM,
are working the problem on both sides.
We are both working to improve the retriever
to give the large language model the best quality data on which to ground its response,
and also the generative part so that the LLM can give the richest, best response finally to the user
when it generates the answer.
Thank you for learning more about RAG and like and subscribe to the channel.
Thank you.