ANDREI BARBU: All right. To start off with, perception is a very difficult problem. And there's a good reason why perception should be difficult, right? We get a very impoverished stimulus. We get, like a 2D array of values from a 3D scene. Given this impoverished stimulus, we have to understand a huge amount of stuff about the world.

We have to understand the 3D structure of the world. If you look at any one pixel, you have to understand the properties of the surface that produce the elimination that gave you that pixel. You have to understand the color or the texture. You have to see the color of the light that hit that surface, the roughness of the surface, et cetera. So you have a very small channel, a very small window onto the world, and you have to extract a tremendous amount of information so that you can survive and not get killed by cars regularly.

All right. That's exactly the problem we're going to talk about here, which is, how do we use our knowledge of the world to structure our perception? To actually modify what we see in order to be able to solve this problem? How do we take a small impoverished stimulus and extract a huge amount of information about the world around us? So let's start with a few examples where knowledge about the world really, practically changes what we see.

So I'm Canadian. So I'm required to show you a Canadian flag in every talk. Here we are.

You can take this flag and you can give it to any of you. You can give it to any kid and you can ask them, here's a marker. Put big red marks around the regions on this flag or in the regions in this image that are red. And it's pretty clear to all of us that there is a distinction between the red that's in the flag, and the bars, and in the Maple Leaf, versus the red that's actually in the background. And we can all tell those two apart.

Except if you actually look at the pixel values, you open them in Photoshop, or GIMP, or whatever other program you want, you're going to notice that those pixel values are actually not particularly different. There's no threshold that you can choose that will separate the red on the flag from the red in the background. so you're doing a huge amount of inference just to

solve this trivial, little problem of, what color is where? You're using knowledge about regions, knowledges about flags, knowledge about transparency, in order to figure out that the red flag is different from the red in the background. So this is a really practical way that your knowledge about the world really changes your perception. You're not seeing the colors that are really there.

Another nice example comes from a paper by Antonino Torralba. So if you look at the scene, it's pretty blurry. And it has to be blurry because your visual system is so incredibly good, we have to degrade the input for you to see how poor it actually is if we take away some information. So this looks like a scene. And the background looks like a building. In the foreground, you can there's maybe a street. And the thing on the street kind of looks like a car.

Does it look like a car to everybody? Awesome. We can look at a slightly different image. We can look at a very similar scene. Again, same building in the background, same street in the foreground. Now, there's kind of a blob in the foreground. And it looks as if it's a person. It looks like a person to everyone, right? Awesome.

Well, the only problem is the blob on the left is exactly the same as the blob on the right. It's difficult to believe me, but you can find these two images online and in his paper. You can open them up in your favorite image viewer. You can zoom in and you'll see they are pixelized completely identical.

So you're using a tremendous amount of information to put together the fact that buildings and streets-- when you see these horizontal streaks, it means a car. And when you see these vertical streaks, it means people. And this really changes how you see the world. And it changes it to the point where you actually, probably don't believe me that these two blobs are the same. And I couldn't believe it either until I really zoomed in and checked. So you can see lots of interesting effects where your high-level knowledge of the world is structuring your low-level perception, and it is actually overriding it.

You've seen this example with the hammer, where you were unable to recognize what's going on in a small region. But when I give you the rest of the context, you can tell that it's a hammer. And when you see the whole video, you actually don't see the hammer disappear. You're filling in information from context in single images and you're filling in information from context in whole videos.

But if we dig in what's going on here just a little bit more, what's going on is somewhere inside your head, there's something resembling a hammer detector, right? So you run a hammer detector. And you ran that hammer detector over that little region. And it said, I'm not so sure. I'm not very confident about what I see in this little region.

And somewhere inside your head, there's some detector or something that can recognize someone hammering something. So if we look at sort of a more traditional computer vision pipeline, what you would do is you would run your hammer detector. You would take your hammer detector. You would use that knowledge in order to recognize hammering in the scene. And at the end, you would say, I'm really confused because my hammer detector didn't work very well.

The reason why you can actually do this is because you have a feedback. You were able to recognize the hammering event as a whole. And that lets you upgrade the scores of your hammer detector, which is very unreliable in this case. So feedback was really critical in being able to understand this scene.

Unfortunately, pretty much all of computer vision is feed forward, even though most of your visual system has, for the most part, feedback connections. More feedbacks than feed forwards. So in this talk, we're going to talk about that feedback. And we're going to see a way that we're going to build this feedback in a principled way.

That if we choose right detections-- right algorithms and right representations from our low-level perception, we're going to be able to combine it with our high-level perception of the world. So we've seen that perception is very unreliable. The top-down knowledge really affects your perception.

And that, what you're going to see in a moment, is that one integrated representation can be used for many tasks. The advantage of these feedbacks goes beyond just better vision. It lets you solve a lot of different problems that look very, very distinct. But actually, turn out to be very, very similar.

So one problem is recognition. I can give you a picture of a chair, and I can tell you, what is this? And you can tell me it's a chair. Or I can give you a picture and I can give you a sentence, this is a chair. And you can tell me, I believe you. This is true.

There's also a completely different problem, which is retrieval. Related to recognition, right?

How about I give you a library of videos and ask you to find me the video where the person was sitting on the chair. And you can solve that problem.

You can also solve a problem like generation. I can give you a video and I can tell you, I don't know what's here. Please describe it to me. So if you see the scene, you can say, what's on the screen? Well, there's a whole bunch of text on the screen.

You can also do question answering. You can take an image like this. I can ask you a question. What's the color of the font? And you can say, the font is white. So you were able to take some very high-level that's in my head that got transmitted to your head. You were able to understand the purpose of this transmission, connect it to your perception, figure out the knowledge that I wanted extracted from your perception, and give it back to me in a way that's meaningful to me.

Even more than this, you can disambiguate. You can take a sentence that's extremely ambiguous about the world and figure out what I'm referring to. And we do this all the time. That's really what makes human communication possible, right? The fact that most of what I say is extremely ambiguous. That's why programming computers is a real pain, but talking to people is generally easier depending on the person.

You can also acquire knowledge, right? You can look at a whole bunch of videos. If you're a child, you sort of perceive the world around you. Occasionally, an adult comes, drops a sentence here or there for you. But what's important is that no adult ever really points out what the sentence is referring to.

You don't know that approach refers to this particular vector when someone was doing some action. You know that Apple refers to this particular object class. Who knows what it could mean? But you get enough data, and you're able to disentangle this problem of seeing weakly-supervised videos paired with sentences. And we'll see how you can do that.

Pretty much everything I'll talk about is going to be about videos. And I'll tell you a story about how I think we can do images as well. There are a bunch of other problems that you can solve with this approach.

I'm sorry?

AUDIENCE: So those images go with the video?

ANDREI BARBU: Yes. So rather than doing videos, we're going to do images. So one thing that you can do is you can try to do translation. We haven't done this. We're going to be doing this in the fall. We have two students. And I'll tell you at the end what the story is for how you're going to do a task that sounds as if it's from language to language, but you're going to do it in a grounded way that involves vision.

Even more than that, you can do planning. And I'll tell you about that at the end a little bit. And finally, you can also incorporate some theory of mind. And that's actually the project that the students are doing as part of summer school. And I'll say a few words about that.

What's important about this is the parts at the top, we understand better. We've published papers about them. The parts at the bottom are sort of more future work, and I'll say less about them.

Well, one important part about this is I've shown you all these tasks. But you really have to believe me that humans perform these tasks all the time. Every time you're sitting at a table and you ask someone, give me a cup. That's a really hard vision language task. There may be 10 different cups inside of you on the table if you're sitting on one of the big, round tables. And you have to figure out, what object am I talking about? What kind of cup am I talking about? Which cup would I be interested in?

If I drank out of the cup, I would expect that you give me my cup, not your cup. Otherwise, let me know. I will sit at different tables from now on.

If I ask you, which chair should I sit in? Again, you have to solve a pretty difficult problem where you look at chairs. You figure out what I mean by which chair should I sit in. Is it that there's a chair that's reserved for someone. Is it that a chair is for a child and I'm an adult, that kind of thing.

You can say something like this is an apple. And when you say that to a child, you're saying it for a particular reason. To convey some idea. You have to coordinate your gaze with the other person's gaze to make sure you're drawing their attention to the real object.

Even more than that, you can say very abstract things, like to win this game, you have to make a straight line out of these pieces. That means that we both agree on what a piece is. That I've drawn your attention to the right idea of a piece. That we agree on what a straight line means on this particular board. There's a lot of knowledge that goes into each of these. But the

important part is that they're each grounded in perception.

We have to agree on what we're seeing in front of each other in order to be able to exchange this information. And pretty much everything that we do in daily communication is a language-vision problem on some level.

All right. So if we believe these problems are important, we can make one other observation, which is none of you got training in most of these problems. No adult ever sat you down and said, OK. Now, you're four. Now I'm going to teach you how to ask questions about the real world.

Or no one sat you down and said, OK. Now, let's talk about language acquisition. You're supposed to do gradient descent. So what's important is you have some core ability that's shared across all of these tasks? And you're able to acquire knowledge, maybe in one of these tasks or across all of these tasks. You're able to put it together. And as soon as you have this knowledge, you can use this for all these other tasks without having to learn anything else. And that's what we're going to see.

And the core of this that we're going to focus on is recognition. So we're going to build one component. This is that engineering portion of the talk.

We're going to build one scoring function that takes a sentence in a video and gives you a score. How well does this sentence match this video? If the score is 1, it means the system really believes the sentence is depicted by the video. If the score is 0, it means the system really believes the sentence does not occur anywhere in this video. And this is the basic thing that's going to allow us to connect our top-down knowledge about what's going on in the world with our low-level perception.

And after we have this, we're going to see how we reformulate everything in terms of this one function, so we don't have to learn anything else about the world. All right. So we said we need this one function, scoring function between sentences and videos. So let's look at what we would need to have inside this function in the first place.

If I give you a video like this, it's just a person riding a skateboard. And I give you a sentence. The person rode the skateboard leftward.

Well, I can ask you, is the sentence true of this video? Indeed, it is true. But let's think about what you had to do in order to be able to answer this question.

Well, you had to at some level decide there's a person there. I'm not saying that you're doing this in this order in your brain. I'm not saying that they have to be individual stages. I'm not saying you have to have to object detectors. But at some point, you had to decide there really is a person there somehow.

You also had to decide there is a skateboard there. You had to look at these objects over time, or at least-- in at least one or two frames decide that they have a particular relationship, so that the person isn't flying in the air and the skateboard continues onwards.

And you had to look at this relationship and decide, yeah. OK, this is writing. And it's happening leftward. So you have to have these components on some level.

You've got to see the objects. You've got to see the relationships, the static and the changing relationship between the objects. And you have to have some way of combining those together to form some kind of sentence you can represent that knowledge. And that's what we're going to do.

Everything I described to you is this feed-forward system, right? We had objects. We have tracks. We take tracks and we build events. Events like ride. And we take those events together and we form sentences out of them. And there's this hard separation, right?

It's easy to understand a system where what you do is you have objects, tracks, events, and sentences. And you use tracks in order to see if your events happened and your events in order to see if a particular sentence occurred. So that's what we're going to describe first, and then we're going to see how, because we're going to choose the right representations for each of these, these feedbacks become completely trivial and very natural to implement. All right.

We need to start with some object detections. Otherwise, we're just going to hallucinate objects all the time.

Any off-the-shelf object detector that you choose will sometimes work. Here, we ran a person detector in red and a bag detector in blue. It will sometimes give you false positives.

Trees are often confused for people. I guess we're both two vertical-- two long, vertical lines. And sometimes, you get false negatives. Sometimes, a bag is so deformable that you think the person's knee is the bag.

Lest you think that object detection is solved, it actually isn't. So if you look at something like the image net challenge, mostly people talk about the image classification. The stuff in light blue. And they're saying that there's 10% error. These days, there's 5% error on this. But that's really not what you're doing in the real world. You're not classifying whole images.

When you see an image in the real world, what you're doing is you're trying to figure out what objects are where. And that's the red part. That's the part where you have an average precision of 50%.

In other words, the object detector really, really, really sucks. Most of the time, it's going to be pretty wrong. It's very, very, very far away from how accurate you are.

If your object detector was that bad, you would die every time you crossed the street. All right. So we believe that object detection doesn't work well. In order to fix this, because somehow we have to be able to extract some knowledge about the video that's pretty robust for us to be able to track these objects and recognize these sentences, we need to modify object detectors a little bit.

We're going to go into our object detector. And normally, they have a threshold. At some point, they learn that if the score of this detection is above this level, I should have confidence in it. And if the score of this detection is below this level, I shouldn't have confidence in it.

And what we're going to do is we're going to remove that threshold. We're going to tell the object detector, give me thousands or millions of detections in every frame. We're going to take those detections and we're going to figure out how to filter them later on. All right.

The way we're going to do this-- and this is the only slide that's going to have any equations. And it's just going to be a linear combination. All we're going to do is we're going to take every detection in every frame of this video. We're going to arrange them in the lattice. In every column of this lattice, we're going to have the detections for one particular frame. And in essence, what we want is one detection for every object for every frame.

In other words, we want the path through this lattice. We want to select one detection in every column. But we want tracks that have a particular property, right?

If I'm approaching this microphone, you know that you expect to see me kind of far away. Then, getting closer. Then, eventually I'm close to the microphone. You don't expect me to be over there, and then to appear over here as if I've teleported. So we want to build this intuition

that objects move smoothly. And that objects move according to how they previously moved, right?

It's not like someone moves from one frame 10 pixels over. Then, the next frame, they move 10 pixels to the left. And they keep oscillating between the two. And that's what we're going to do. I'm not going to talk about how we compute this. It's really trivial. If you know about optical flow, you can do it.

But basically, what we want is a track where we don't hallucinate the objects. So every node in our resulting detections should be strong. If we ignore the strength of the object detector, we're just going to pretend that there are a whole bunch of people in front of us.

And every edge should also be strong. In other words, when we look at two detections from adjacent frames, if I have a person over here in one frame and a person over here in another frame, I shouldn't really think that's a very good person track. But if I have a person over here that kind of moved to the right the previous frame and I have a new detection that's just slightly to the right of that one, I should expect that it's a much better track. So that's all we do.

And encoding this intuition is very, very straightforward. It's just a linear combination. So the score of one path, the score of the track of an object, is just the sum of your confidence in the detections. So in every detection and every frame, along with the confidence that the object track was actually coherent. All right.

So is the only equation we're going to see in this talk. And it's just a linear combination. But it will come back to haunt us several times before the end. All right.

So we use dynamic programming. We find the path through this lattice. And this is a tracker.

And actually, Viterbi did this in 1967 for radar. This is not a new idea.

Here, we ran it for just a computer vision task where we just wanted to track objects. We ran a person detector and a motorcycle detector, but we don't have a person standing up and a person sitting down detector. So the tracker is good enough that it can keep the two people separate from each other, despite the fact that they're actually pretty close in the video.

So you see we do a pretty decent job of tracking all the objects until they get pretty small in the field of view and the object detector doesn't work well anymore. All right.

So now what we have are the tracks of objects. We can see object motion over time and from a video. And somehow, we have to look at these tracks and determine what happened to them. Was someone riding? Was someone running? Was someone bouncing up and down?

In order to do this, we're going to get some features from our tracks. You can look at the track in every frame and you can extract out a lot of information. You can extract out the average color. You can extract out the position, the velocity, acceleration, aspect ratio. Anything that you want to get out of this frame knowing that this bounding box is there, you can compute and the algorithm doesn't care. All right.

There's one small problem, though. Most of the time, we need more complicated feature vectors. So for example for ride, it's not enough to have a feature vector that only includes the person. You needed to look at the relative position between the person and the skateboard to determine-- that are actually going together and one isn't going right and the other one is going left.

So for that, what we're going to do is we're going to build a feature vector for the agent of the action in the case of ride and a feature vector for the instrument-- the skateboard. We're going to concatenate the two together, so we get a bigger feature vector. And then we're going to have some extra features that tell us about the relationships between these two.

So we can include things like the distance, the relative velocity, the angle, overlap. Anything that you want to compute between these two bounding boxes in this frame, you're welcome to compute. All right.

And if you build this feature vector between this person and the skateboard, you could recognize the person rode the skateboard in this video. If you build a different feature vector, for example between these two people, you could recognize the person was approaching the other person, or the person was leaving the other person.

If you build a feature vector between the skateboard and the other person, you could recognize the skateboard is approaching the person, et cetera. So depending on which feature vector you build, you can recognize different kinds of actions.

So when we have our tracks, we know how the objects moved in these videos. We get out some feature vectors from our tracks. And what we need to do is decide what these feature vectors are actually doing. Is the person riding that skateboard?

The way we're going to do this is using hidden Markov models. Hidden Markov models are really simple. All they assume is that there is a model of the world that follows a particular kind of dynamics.

In this case, imagine that we have an action like approach. I'm far away from the object. I get closer to the object. Eventually, I'm next to the object. So this action, for example, has three states. One where I was far, one as I was getting nearer, one when I was very close. And we have a particular transition between these states, right?

We already said that I don't teleport. So I shouldn't be able to go from being far away to being near. So you should expect me to go from the first state to the second state and to the third state without going from the first to the last. In each state, you have something that you want to observe about me, right? You want to really see that I'm far away in the first state, that I'm getting closer in the second, and I'm actually there in the third. So we have some model for what we expect to see in every state and we can connect this with our feature vectors.

So the idea is there's some hidden information behind the motion of these objects. And we're going to assume that hidden information is represented within HMM. And what we need to recover is the real state of these objects.

So if you see a video of me moving towards this microphone, you have to recover some hidden information of, which frames was far away in? Which frames was getting nearer? And which frames was I actually next to the object?

For now what we're going to do is we're going to assume that we have one of these hidden Markov models for every different word. So for every verb, we have a different hidden Markov model. There's one for approach. There's one for pickup. There's one for ride, et cetera. And if you want to tell me what's going on in this video, you just have a big library of hidden Markov models. You apply every one to every video. You have some threshold. And anything above that threshold, you say happened. And you produce a sentence for it. OK.

If we look at how you actually figure out what this hidden information is, what state am I in when I'm approaching this object, it looks a lot like the tracker. What you have is you have to make a choice in every frame. Your choice is, which state is my action in? Is it state 1 through 3, or some other state? In the same way that in tracker, you have to make a choice. You have to choose, which detection is the system in for each frame?

And here, you also have edges. Edges tell you, how likely am I to transition between different states in my action? And every node also has a score. It's the score of, did you actually observe me doing what you're supposed to observe me doing in every action?

So if you're saying I'm in the first state, did you actually see me stationary and far away from that object? And what you want is a path through this lattice in the same way that we had a path before. And a path just means you made a decision that I'm in state 1 in the first frame or in state 1 in the third frame, et cetera. And that's just the linear combination of the scores. So it's the same equation we saw before.

So here's an example of this sort of feed-forward pipeline in action. We ran it over a few thousand videos. It produces output like the person carried something, the person went away, the person walked, the person had the bag. It's pretty limited in its vocabulary. It has 48 verbs, about 30 different objects, a few different prepositions.

And it even works when the camera moves. So the person chased the car rightward, the person slowly ran rightward to the car. And it should also probably say the person had a really bad day, but that's for the future.

So we've seen this feed-forward pipeline. We've seen that we can get objects. We can get tracks. We can look at our tracks, get some features, run event detectors, take those event detectors, and produce some sentences.

And now, all we're going to do is we're going to break down the barriers between these and show you how you can have feedback in a really, really simple way. All right. So first, let's combine our event detector and our tracker. Because what that's going to say is, if you're looking for someone riding something, well, you should be biased towards seeing people that are riding something.

So in the occlusion example, if you see someone go behind some large pillar, well, you might lose them. But you have a bias that you should reacquire someone riding a skateboard after they leave the pillar, which you don't have if you just run the tracker independently from the event detector.

So the way we're going to put them together is very, very easy. There's a reason why these two look completely identical and why the inference algorithm between them is identical.

Right now, what we're doing is we have a tracker on the left, or on your left. And we have an event recognizer on the right.

Right now, we're running one, and then we're feeding the output of one into the other. Basically, we run one maximization, and then we run another maximization. And all we're going to do is move the max on the right to the left. And you get the exact same inference algorithm.

The intuition behind this is you have two lattices. And you can take the cross-product of the lattices. Basically, for every tracker node, you just look at all the event recognizer's nodes and you make one big node for each of those. And every node represents the fact that the tracker was in some state and the event recognizer was in some other state. So we have a node that says the tracker chose the first detection. The event recognizer was in the first state.

We have another node that says the tracker chose the second detection. The event recognizer was still in the first state. And you do this for every detection. Then, you do the same thing for the event recognizer being the second state, et cetera. So you're just taking a cross-product between all of the states.

Does that make sense? Another way to say it is that we have two Markov chains. One that's observing the output from the object detector and another one that's observing the output of the middle Markov chain. And you do joint influence over them. And the way you can do joint inference is by taking the cross-product.

Basically, you have two hidden Markov models. One that does tracking and one that does event recognition. And all we're going to do is joint difference in both of them. So rather than trying to choose the best detection, and then the best state for my event, I'm going to jointly figure out, what's the best detection if I assume I'm in this state? What's the best detection if I assume I'm in this other state? And at the end, I'll pick the best combination.

Make sense? So this is a way for your event recognizer to influence your tracker, because now you're jointly choosing the best detection for both the tracker and the event recognizer. So that was really, really simple.

We put in a tremendous amount of feedback by just taking a cross product. So we can see this in action. I'm going to show you the same video twice.

The person is not going to move in this video at all. What we told the system is that a ball will

approach a person. That's it we didn't tell them which person. We didn't tell the system which particular ball, which direction it's going to come from, or anything like that.

The top detection in this frame happens to be the window. It's a little hard to see. It's quite a bit stronger than the person. But because neither the window nor the person ever move in this scenario, the tracker can't possibly help you. You have no motion information. The only way to override that window detection is to know something else about the world.

So we told it that the ball will approach. And you can see that for the combined tracker and event recognizer. Indeed, when the ball comes into view, it will make more sense. So the reason why we actually-- coming back to the question that you asked. Why we don't run it over small windows is because we want this effect of knowledge that's much, much later on in the video. Like the fact that the ball will enter or approach that person as opposed to that window to actually help you much earlier in the video. If you run it over small windows, you lose that effect.

So here, you track the person correctly from the very first frame despite the fact that the ball only comes into view halfway through the video. There are many more examples of this.

In this case, it's a person carrying something. Here, we told the system one person's carrying something. And you'll see when the person moves, we can detect the person and the bag. The object detector fails much, much earlier because the person was deformable. So we've seen how we can combine together trackers and events recognizers. And now, we need to add sentences.

And the trick for adding sentences is going to do more of the same. What we're going to do is we're going to take a tracker. It's just exactly what we saw before. And what we just did a moment ago is we combined it with an event recognizer.

Well, there's no reason why we can't add more trackers. We actually kind of did that, right? We were tracking both a person and a ball a moment ago. So we can take an even bigger cross-product, have multiple trackers, and have multiple words.

So all we're saying is, I have, say, five trackers that are running. I have five words that I want to detect, or 10 words that I want to detect. And I want to make the choice for all of these 5 trackers jointly, so that they match all of these 10 words.

In this picture, basically our words are kind of-- our sentences are kind of like bags of words, right? Every word is combined with every tracker. But we know if you look at the structure of a sentence like the tall person quickly rode the horse, not every word refers to every object in the sentence.

So you can run your object detectors over your video. And you can look at your sentence. And you can look at the nouns and say, OK. So I have people and horses inside the sentence.

And you can say, OK. Well, if I have people and horses, I need two trackers. But you can look a little bit more at your sentence and see that, oh, well, it's the other horse. So you analyze your sentence and you can determine there are three participants in the event described by the sentence. There's a person and two horses. One's the agent. One's the patient-- the thing that's being ridden-- and one's source-- the thing that is being left.

Does that make sense? Awesome. So now, given a sentence, we know that we need n trackers. And for every word, we can have a hidden Markov model. We can have a hidden Markov model for ride. It's just another verb. And we just have to be careful how we build a feature vector for ride.

Because if we build it in one way, we're going to detect the person rode the horse. And if we build it in the opposite way by concatenating the vectors the other way around, we're going to detect the horse rode the person, which is not what we want.

We can also detect tall. Tall is kind of a weird hidden Markov model, right? It has only a single state, but it's still a hidden Markov model. It just wants to see that this object is tall. So maybe its aspect ratio is more than the mean aspect ratio of objects of this class. But nonetheless, it still fits into this paradigm.

We can do the same thing for quickly. We can have an HMR for that. We can do leftward. We can do away from. Away from looks a lot like leave. It's the same meaning. And basically, we end up with this bipartite graph.

At the top, we have lattices that represent words. Each word has a hidden Markov model. And in the middle, we have lattices that represent trackers. We can combine them together according to the links. And you can get these links from your favorite dependency parser. You can get them from Boris's START system. Any language analysis system will give you this.

So this is actually all the heavy lifting that we have to do. Everything from now on is kind of eye

candy.

One thing that we really wanted to make sure that system was doing is that we could distinguish different sentences. So we tried to come up with an experiment that is, in some way, maximally difficult where events are going to happen at the same time. So you can't use time in order to distinguish them. And the sentences only differ in one word or one lexical item.

So in this case, we have a sentence like the person picked up an object and person put down an object. There are two systems that are running. One is running on one sentence. One is running on the other sentence. You're going to see the same video played twice side by side. And you can already see that one system, when we primed it to look for pickup, it detected me picking up my backpack. And then, the other one it detected one of my lab mates picking up a bin.

So the only way you could focus its attention on the right object is if it understood the distinction between these two sentences, or if it was able to represent them. So we can play this game many, many times over.

We can have it pay attention to the subject. Is a backpack approaching something or is a chair approaching something? We can have it pay attention to the color of an object. Is the red object approaching something or a blue object approaching something?

We can have it pay attention to a preposition. Is someone picking up an object to the left of something or to the right of something? And we have many, many dozens or hundreds of these. And I won't bore you with all of them.

But the important part is we can handle lots and lots of different parts of speech. And we can still represent them and we can still be sensitive to these subtle distinctions in the meanings of the sentences. All right.

So we did all the hard work. And we actually built this recognizer-- the score of a sentence given a video. And now, it turns out that we can reformulate all of these other tasks in terms of this one score. And it's going to do all the heavy lifting for us.

So when we tune the parameters of whatever goes into the scoring function, we're going to get the ability to do all these other tasks. So let's look at retrieval. It's the most straightforward kind of task, right? It's what YouTube does for you.

You go to YouTube. You type in a query, and YouTube comes back with some answers. So let's see what YouTube actually does.

If you look at YouTube. And if you look at something like pickup, you get men picking up women. If you look at approach, you get men picking up women. If you look at put down, once upon a time you did get men picking up women, but rap is now more popular.

If you ask something more interesting-- the person approached the other person-- you don't get videos where people approach each other. You get videos about how you should approach women.

I didn't select these. I typed them in and this is just what happened.

If you type in, like the person approached the cat, you get lots of people playing with cats, but no one approaching cats, including a link that's kind of scary and an Airbus landing. And I have no idea what that means.

So what we did is we built a video retrieval system that actually understands what's going on in the videos as opposed to just looking at the tags that the people apply to these videos. People don't describe what's going on. People describe some high-level concept.

So we took a whole bunch of object detectors that are completely of the shelf for people and for horses. And we took 10 Hollywood movies.

Nominally, they're all Westerns. They involve people on horses. And the reason why we chose people on horses was because people on horses tend to be fairly larger in the field of view. And given that object detectors suck so much, we thought we should kind of help the system along s best we could.

So we build a system. It's a system that knows about three verbs. It knows about two nouns, person and horse. It knows about some adverbs, quickly and slowly. It knows about some prepositions, leftwards, rightwards, towards, away from. And given this template, you can generate about 200, 300 different sentences.

So we can type in something like the person rode the horse. And we can get a bunch of results. So you can see, we were in 90% accurate in the top 10 results. You can see these are really videos of people riding horses.

The way this works is we took one of these long videos. We chopped it up into many small segments and we ran over each individual segment.

You could run it over the whole video, but then it would just classify the whole video because it's an HMM and would sort of adapt to the length of the video.

We can also ask for other kinds of queries, like the person rode the horse quickly. You can see we get videos that really are quicker.

We can ask for something more ambitious, like the person rode the horse quickly rightward. And we get videos where people are riding horses rightward. All right.

So we did the hard work of building this recognition system. And we saw we can use it for another task, which is retrieval. But let's do something else. Let's do generation. Someone asked about generation earlier.

Generation is very similar to retrieval. In retrieval, what we had was we had a fixed sentence and we searched over all our videos to see which ones were the best match.

Here, we have a fixed video. And we're going to search over all our sentences. The only trick is you have a language model, so it can generate a huge number of sentences. But we're going to see that's OK.

So we have a language model. It's very, very small model by Boris' standards, or the standard of NLP. We have only four verbs, two adjectives, only four nouns, some adverbs, et cetera. But the important part is even if we ignore recursion, we have a tremendous number of sentences. And this model is recursive, so we can really generate an infinite number of sentences from it.

But nonetheless, it turns out that you can search the space of sentences very, very efficiently and actually find the global optimum. And the intuition for why that's true is pretty straightforward.

You can think of your sentence as a constraint on what you can see in the world. The longer your sentence, the more constrains you have. So the lower the overall score is.

So every time you add a word, the score can't possibly increase, right? The score has to always decrease. So basically, you have this monotonically-decreasing function over a lattice

of sentences.

And if you ignore the fact that you only have to search sentences, you can start off with individual words, aggregate words together. So you look at all one-word phrases. You can a two-word phrases, three-word phrases. Eventually, get out to real sentences.

But because this is a monotonically-decreasing function, this is a very quick search. So you can start off with an empty set. You can add a word. For example, you can add carried. You can look at all the ways that you can extend carried with another word or two. So you get a phrase like the person carried. And you can keep adding words to it until you get to the global optimum.

So given a video like this, where you see me doing something, you can produce a sentence like the person to the right of the bin picked up the backpack. And that's pretty straightforward. We built a generator in just a few lines of code as long as we had our recognition system.

So you have this problem in question answering that you have to connect two sentences with a video. And instead of doing that, what we're going to do is we're going to make some connection between two sentences. So we're going to take our question. We're going to give it to something like Boris' system. And it's going to tell us this question, like, what did the person put on top of the red car?

If you wanted to answer it, you would produce an answer like, the person put some noun phrase on top of the red car. So you can run the generation system exactly as was suggested. You seed it with this. You give it a constraint that what it has to produce next inside this empty gap is a noun phrase. And you're going to get out the answer.

Another way to think about this is you have sort of a partial detector. You look inside the video to see where it matches. You choose the best region where it matches, and then you complete your sentence. And you get an answer like the person put the pair on top of the red car.

There's one small problem with question answering, and it differs from generation in one way. So imagine that we're in a parking lot and there are a hundred white cars inside this parking lot.

And you come to me desperate and you say, I lost my keys. And I say, don't worry. I know exactly where your keys are. And you look at me and I say, they're in the white car. And then you think I'm a complete asshole, because that was totally worthless information, right? I told

you something that's basically true. It's a parking lot full of white cars, but isn't actually giving you anything useful.

So to handle this-- in the same way that in generation, we had this one parameter that we could tune to get, more or less, for both sentences. We're going to add only one parameter to question answering, which is kind of a truthfulness parameter. Which basically is going to say, this sentence, the person put an object on top of the red car in this video, is very ambiguous, right? It could either be Danny that did it or it could be me that put something on top of the red car.

So what we're going to do is we're going to take this candidate's answer. We're going to run it over the video. And we're going to see how many times it has really close matches in the video.

And depending on this one parameter, we're going to say you are allowed to say more things about the video to become more specific about what you're referring to. But potentially, slightly less true because the score will be lower. In the same way that you were saying slightly more in the generation case at the risk of saying potentially something that's slightly less true.

So this way, you can ignore the sentence, which is unhelpful. And you can end up saying something like, the person on the left of the car put an object on top of the red car. So we can actually do that and the system produces that output.

We built one recognition approach. And we did retrieval, generation, and question answering with it. We can also do disambiguation with it.

In disambiguation, we take a sentence, like Danny approached the chair with a bag. And you can imagine that this sentence can mean multiple things.

It could mean Danny was actually carrying a bag and approaching a chair. Or it could mean there was a bag on a chair and Danny was approaching it.

And there's the question of, how do you decide which interpretation for the sentence corresponds to which video? Basically, you can take your sentences and you can look at their parse trees. And you're going to see that they're different.

Essentially, your language system is going to give you a slightly different internal representation for each of these. And we already know that when we build our detectors for

the sentence, we take these kinds of relationships between the words as inputs. So even though there's one sentence in English that described both of these scenarios, when we build detectors we're going to end up with two different detectors. One for one meaning, one for the other meaning. And then we can just run the detectors and figure out which meaning corresponds to which video. And indeed, that's what we did.

Except that there are lots and lots of different potential ambiguities. There are different kinds of attachment. In the same case-- I won't go through all of them. But for example, you might not know where the bag is. You might not know who's performing the action. You might not be sure if both people are performing the action or only one person is performing the action.

There may be some problems with references. So this is a very simple example, like Danny picked up the bag in the chair. It is yellow. But this is the kind of thing that you would see if you had a long paragraph. You would have some reference later on or earlier on to some person. And you wouldn't be sure who was the referent.

And it turns out that if you have sentences like this, you can disambiguate them pretty reliably. So what's important is it's not just a case of parse trees. We need a more interesting internal representation.

And an example of how we do this is we take a sentence and we make some first-order logic formula out of it. So you have some variables. The chair is something like x. You have Danny, who moved it, and I moved it. Or in the other case, you have two separate chairs. And I moved one and Danny moved the other. And they're distinct chairs.

What we do is we first ignore the people. So we just say there are two people. And in both cases, we're distinct from each other. But we don't have person recognizers, face recognition, or anything like that.

Then for each of these variables, we build a tracker. And for every constraint, we have a word model. And essentially, you can go from this first-order logic formula to one of our detectors. So it's exactly the same thing as the case where we had a sentence and a video. And we just wanted to see, is the sentence true of the video? Except that now we have a sentence interpretation and the video.

So we've seen that if all you have are multiple interpretation of a sentence, you can figure out which one belongs to which video. And we'll come back to this in a moment, because it's

actually quite useful.

So you can imagine a scenario where you want to talk to a robot. And you want to give it a command. You don't want to play 20 questions with it, right? You want to tell it something. It should look at the environment. And it should figure out, you're referring to this chair and this is what I'm supposed to do.

So the other reason for disambiguation is going to be because you get a lot of ambiguities while you're acquiring language. So we're going to break down language acquisition into two parts.

One part is we want to learn the meanings of each of our words. And another one is we want to learn how we take a sentence and we transform it into this internal representation that we use to actually build these detectors.

So if you look at the first one, let's say you have a whole bunch of videos. And every video comes with a sentence. You don't know what the sentence is actually referring to in the video.

When children are born, nobody gives mothers bounding boxes and tells them, put this around the Teddy bear so your child knows what you're referring to. So we don't get those. We have this more weakly-supervised system.

But what's important is we get this data set and there are certain correlations in this data set, right? We know the chair occurs in some videos. We know that backpack occurs in others just by looking at the sentence. We know pickup occurs in others.

So basically, this is the same thing as training one, big hidden Markov model. Except that now we have multiple hidden Markov models that have a small amount of dependency between them. And I won't talk about this. You'll have to take my word for it. You can look at the paper. But it's identical to the Baum-Welch algorithm.

Essentially, all you do is you take the gradient through all the parameters of these words and you can acquire their meanings. There are lots of technical issues with this, but that's the general idea.

So we can also look at learning syntax. And this is something that we haven't done, but we really want to do. And this is where disambiguation work really comes into play.

So if I give you a sentence, like Danny approached the chair with a bag, you feed it into a parser. Something like Boris' start system. And you get potentially two parse trees, right? One for one interpretation and one for the other interpretation.

You take the video and you can select one of these parse trees. That's the game we just played a moment ago. But imagine that we take Boris' system and we brain damage it a little bit. Or we take some deep network that does parsing and we just randomize a few of the parameters.

So now, rather than getting a single or two parse trees for our two interpretations, we get 100 or 1,000 different parse trees. We can take each one of those and we can see, how well does this match our video? And we get some distribution over them. Maybe we won't get a single one that matches the best. Maybe we'll get a few that match well and a bunch that match really, really poorly. So this provides a signal to actually train the parser.

Essentially, you have a parser that produces a distribution over parse trees. You use the vision system to decide which of these parse trees are better than others. And you feed this information back into the parser and retrain it.

We haven't done this, but it's in the pipeline. And eventually, the idea is that we're going to be able to close the loop and learn the meanings of the words while we end up learning the parser. But that's further down the line.

So lest you think that there's something remarkable about language learning in humans, actually lots of animals learn language, not just humans. And here's a cute example of a dog that does something that our system can't do. And actually, no language system out there can do.

So there's this paper, but this is from PBS. And what ended up happening is this dog knows the meaning of about 1,000 different words because there are labels that have been attached to different toys.

So it has 1,000 different toys. Each one has a unique name. And if you tell the dog, give me Blinky, it knows exactly which toy Blinky is. And it has 100% accuracy getting you Blinky from it's big, big pile of toys.

So what they did is they took 10 toys. They put them behind the sofa. And they added one additional toy that the dog has never seen before. They tested the dog many times to make

sure that it doesn't have a novelty preference or anything like that. And then they asked the dog, bring the Blinky. And you can see the dog was asked. It goes behind. It quickly finds Blinky. It brings it back. And there we go. And now, the dog is really happy.

So now, the dog is going to be asked, bring me this new toy. Bring me the professor, or whatever the toy is called. It's a little less certain. OK.

So it's going to go behind and it's going to look at all the objects. The toy with the beard is the new one that it hasn't seen before. And it was there in the previous trial.

So it looks around and it's a little uncertain. It doesn't quite want to come back. We're going to see that we're going to have to give it another instruction in a moment.

He's going to call it back and ask the dog to do exactly the same task again. Isn't telling it anything new. It's just to give it some encouragement.

So looking around for some toy. And it picks the-- you'll see in a moment. It picks the toy that it hasn't seen before, because it's a new word. And the dog is really happy. And I think the human is even happier that this actually worked.

But the important part is, there's this dog that we normally don't associate with having a huge amount of linguistic ability. But it's learning language in a way that is far more advanced than anything that we have. And it's learning it in a grounded way, like it hard to connect its knowledge about what it sees with these toys to this new object that it's never seen before and understand this new label.

And dogs are not the only animal that can do this. There are many other animals that can do this. All right. And of course, children do this as well.

So there was a question about the fact that we're constantly using videos here. And we're very focused on motion. But of course, in many of these sentences, we were referring to objects that were static. So we're not only sensitive to objects that are moving.

So for example, when I said something like it was the person to the left of the car, neither the person nor the car were moving in that question. It was the pair that was moving.

But there's an interesting question, what if you want to recognize actions in still images? After all, we can do it. It probably didn't involve looking at photos. You know, 200 million years ago

when our visual system was being formed. So somehow, we take our video ability and we apply it to images.

And the way we're going to do that is by taking an image and predicting a video from it. We haven't done this, but we've done the part where you can actually get predicting motion from single frames. So the intuition about why this works is, if you look at this image and I ask you, how quickly is this baseball moving? You can give me an answer.

AUDIENCE: Not very quickly.

ANDREI BARBU: Not very quickly. Right. And if you look at this baseball, you can decide that it's moving very quickly, right?

So the other story in this talk is I'm becoming more and more American. I started with the Canadian flag and now I ended up with baseball. All right.

So you can clearly do this task. There is good neuroscience evidence that people are doing this fairly regularly. Kids can do this, et cetera. All right.

So now, what we did is we went to YouTube and we got a whole bunch of videos. Videos that contain cars or different kinds of objects. We had eight different object classes. And we ran a standard optical flow algorithm just off the shelf. And this gives us an idea of how the motion actually happens inside this video.

Then, we discard the video. And we only keep one of the frames. And we train a deep network. This is the only time deep networks appear in this talk. That takes as input the image and predicts the optical flow. It looks a lot like an auto-encoder, except the input and the output are different from each other. And it turns out this works pretty well. It has similar performance to actually doing optical flow on the video with sort of a crappier, earlier optical flow algorithm.

So up until now are things that we've done. At the end I'll talk briefly about what we're doing in the future. So one thing that you can do is translation. And you can cast translation as a visual language task, even though it sounds like it has nothing to do with vision.

So if I give you a sentence in Chinese, you can imagine scenarios for that sentence, and then try to describe them with another language that you know. This is very different from the way people do translation right now.

So right now, the way it works is you have a sentence, like Sam was happy. And you have a parallel corpus. If you want to translate into French, you go off you. Get the Hansard corpus and you get a whole bunch of French and English sentences that are aligned with each other and you learn the correspondence between them.

Here, I translated into [AUDIO OUT] Russian. The important part is in English, there's no assumption about the gender of Sam. Sam is both a male name and a female name. But the problem is Romanian, Russian, French, et cetera, they really force you to specify the gender of the people that are involved in these actions. And you have to go through a certain amount of [AUDIO OUT] really want to avoid specifying their gender.

So here, we specify the gender as male. Here, we specify the gender as female. And if all you have is statistical machine translation system, you may get an arbitrary one of these two. And you may not know that you've got an arbitrary one of these two. And there may be a terrible faux pas at some point.

So this problem is not restricted to gender. And it occurs all the time. For example, in Thai, you specify your siblings by age, not by their gender. So if you have an English sentence like my brother did x, translating that is quite difficult.

In English, you specify relative time through the tense system, but Mandarin doesn't have the same kind of tense system. In this language that I never tried to pronounce after the first time that I tried, you don't use relative direction. So you don't say the bottle to the left of the laptop. We all agree on a common reference frame like a hill or something. Or we agree on cardinal directions. And you say, the bottle to the north or something. And these people are really, really good at wayfinding because they constantly have to know where north is.

Many languages don't distinguish blue and green. Historically, this is not something that languages have done. It's pretty new. For example, Japanese didn't until a hundred years ago. They only started distinguishing the two fairly recently when they started interacting with the West more.

And many languages don't set that boundary at exactly the same place. So one language, you may say blue. In another language, you may have to say green.

In Swahili, you specify the color of everything as the color of x. So like in English, we have orange. But in Swahili, I could say the color of the back of my cell phone. And I expect you to

know that's blue as long as you can see it.

In Turkish, there is a relatively complicated evidentiality system. So you have to fairly often tell me why you know something.

So if you saw somebody do something, you have to mark that in the sentence as opposed to hearing it from someone else. So if it's hearsay, you have to let me know.

There are much more complicated evidentiality systems where you have to tell me, did you hear it, did you see it, did you feel it? It can get pretty hairy. So there are a lot of reasons why just doing the straightforward sentence alignment can really fail on you. And you can make some pretty terrible mistakes.

And more importantly, you just won't know that that made these mistakes. So instead, what we've been thinking is sort of translation by imagination.

So you take a sentence. And it's a generative model that we have that connects sentences and videos.

And what you do is you sample. You sample a whole bunch of videos. So basically, you imagine what scenarios the sentence could be true of.

You get your collection from the generator. You search over sentences that describe these videos and you output a sentence that describes them well in aggregate. So basically, you just combine your ability to sample, which comes from your recognizer, and your ability to generate. And you get a translation system. So you do a language-to-language task mediated by your understanding of the real world.

Something else that you can do is planning, which I'll just say two words about. All you do is--

[PHONE RINGING]

--in a planning task, what you have is you have a planning language. I'm glad that wasn't my cellphone. You have a planning language, right? So you have a fairly constrained vocabulary that you can use to describe your plans. And this allows you to have efficient inference.

Instead, you can imagine that I have two frames of a video, real or imagined, where I have the first world. I am far away from the microphone. I have the last world where I'm near the microphone. And I have an unobserved video between the two. People have work and I've

done some work on filling in partially observed videos. So it's a very similar idea, except that here we have a partially-observed video. And we know that this partially-observed video should be described by one or more sentences. So we're going to do the same kind of sampling process, where we sample from this partially-observed video and we try to describe what the sentence is.

And now you're doing planning. You're coming up with a description of what had happened in this missing chunk of the video. But your planning language is English, so you get to take advantage of things like ambiguity, which you couldn't take advantage of in many languages.

Theory of mind. The idea here is relatively straightforward as well. So what we have right now, basically are two hidden Markov models. Or two kinds of hidden Markov models, right?

There's a video. We have some hidden Markov models that are tracks and we have some hidden Markov model that look at the tracks and they do some inference about what's going on with the events in these videos.

So now imagine that I had a third kind. A third kind of hidden Markov model that only looks at the trackers. Doesn't look at the words directly.

And what it does is it makes another assumption about the videos. So first, we assume the objects were moving in a coherent way. Then, we assumed that the objects were moving according to the dynamics of some hidden Markov models.

Now, we're going to assume that people move according to some dynamics of what's going on inside our heads. So you can assume that I have a planner inside my head that tells me what I want to do and what I should do in the future to accomplish my goals.

And you can look at a sequence of my actions and try to infer. If you believe this planner is running in my head, what do you think I should do next?

Now, the nice part about many of these planners is that they look a lot like this hidden Markov models. And the inference algorithms look a lot like these models.

So basically, you can do the same kind of trick by assuming that HMM-like things are going on inside people's heads. So you can do things like predict actions, figure out what people want to do in the future, what they did in the past. That's what the project is.

I want to show you another example of vision and language, but in a totally different domain that I won't talk about, which is in the case of robots. This is something that we built several years ago.

This is a robot that looks at a 3D structure. It's built out of Lincoln Logs. They're big. They're easier for the robot to manipulate than LEGOs. The downside is they're all brown, so it's very difficult to do vision on this. But it actually will, in a moment, reconstruct the 3D structure of what it sees. And we annotated and read what errors it made. We didn't tell it this. What it does is it measures its own confidence and it figures out what parts are occluded. So it has too little information. And it plans another view. It goes, it acquires it by measuring its own confidence.

This view is actually worse than the previous view, but it's complementary. So it will actually gain the information that it's missing. And all of this comes from the same kind of generative model trick that I showed you a moment ago. A similar model, it just makes different assumptions about what's built into it.

So now, because we have a nice generative model, we can integrate the two views together. You're going to see in a moment. It'll still make some mistakes. It won't be completely confident because there are some regions that it can't see, even from both views. And then what we told it is, OK, fine. For now, ignore the second view, take just the first few. Here's a sentence. Or in this case, a sentence fragment. The fragment is something like, there's a window to the left and perpendicular to this door. It'll just appear a moment.

And integrating this one view that it saw that it was uncertain about with that one sentence, that's also very generic and applied to many structures, determine that these two completely disambiguate the structure. And now, it's perfectly confident in what's going on. And it can go and it can disassemble the structure for you.

And we can play this game in many directions. We can have the robot describe structures to us. We can give it a description and it can build the structure. One robot can describe the structure in English to another robot who can build it for it. And it's exactly the same kind of idea.

You connect your vision and your language model to something in the real world, and then you can play many, many different tricks with one internal representation without modifying it at all. But I realized yesterday that I was the last speaker before the weekend, so I want to end

by leaving you as depressed as I possibly can, and tell you all the wonderful things that don't work. And how far away we are from understanding anything.

So first of all, we can't generate the kind of coherent stories that Patrick looks at. Really, if you look at a long video, what we can do is we can search or we can describe small events. A person picks something up. They put it down.

What we can say is the thief entered the room and rummaged around and ran away with the gold. That's the kind of thing that you want to generate. It's the kind of thing that kids generate, but we're not there yet. Not even close.

We also only reason in 2D. There's no 3D reasoning here. And that significantly hurts us. Although, we have some ideas for how we might do 3D.

Another important aspect is we don't know forces and contact relationships. Now, that's fine as long as pickup means this kind of action where you see me standing next to an object and the object moving up. But sometimes, pickup means something totally different. So you're going to see this cat is going to pickup that kitten in just a moment.

And you're going to see if you pay attention to the motion of the cat, that it doesn't look like it's picking something up. It's not very good at picking up the kitten, mind you. I think this may be its first try.

I think it's having a good day. It's OK. Struggling a little bit. But see? So definitely, picked it up. But it didn't look anything like any of the other pickup examples I showed you.

But conceptually, you should totally recognize this if you've seen those other examples. And kids can do this. So the important part is you have to change how you reason you can't just reason about the relative motions of the objects. You have to assume that there are some hidden forces going on. And you have to reason about the contact relationships and the forces that the objects are undergoing.

What happens if you try to recognize a helicopter picking something up. It looks totally different from a human doing it, but no one has any problems recognizing this.

Segmentation is also a huge problem. For many of these problems, you have to pay attention to the fine boundaries of the objects in order to understand that that kitten was being rotated and then slightly lifted.

There's also a more philosophical problem about what is a part and what it means for something to be an object. We arbitrarily say that the cat is an object, but I could refer to its paws. I could refer to its ears. I could refer to one small patch on its back. As long as we all know what we're talking about, that can be our object. And that's a problem throughout computer vision.

It also occurs in a totally different problem. So if you've ever seen Bongard problems, there are these problems where you have these weird patches, and you have to figure out what's in common between them. And that's the case where you have to dig deep into your visual system to extract a completely different kind of information. And this is an example that I prefer.

So in this task, you can try to find the real dog. And we can all spot it after you look for a little while. Right? Does everyone see it? OK. So you can all see it

The interesting part is-- I mean, I doubt you have ever had training detecting real dogs amongst masses of fake dogs. But somehow, you were able to adapt and extract a completely different kind of information from your visual system. Information that isn't captured by our feature vector, as I talk about the color, location, velocity, et cetera. So you have this ability to extract out task-specific information.

You can do things like theory of mind, but you can do far more than assume people are writing a planner. You can detect if I'm sad. If I'm happy. You can reason about whether two people are having a particular kind of interaction. Who's more powerful than the other person.

You also have a very strong physics model inside your head that underlies much of this. And even more than that, there's the concept of modification. So walking quickly looks very different from running quickly. And the way you model these is quite complicated. And the system that I presented doesn't do a good job of it.

But one of my favorite examples from my childhood long ago is this one, which is a kind of modification. So coyote is going to draw this. You're going to see the Roadrunner try to run through it. And he makes it. And you can imagine what's about to happen next. Coyote is not going to have a good day.

So this looks silly, right? And you would think to yourself, how could we possibly apply this to

the real world? But actually, this happens in the real world all the time. A cage can be open for a mouse, but closed for an elephant.

So if you're going to represent something like, is something closed or not, you have to be able to handle situations like this. And that's why kids can understand really weird scenarios like this because they're not so outlandish.

There's also the problem of the vast majority of English verbs-- things like absolve, admire, anger, approve, bark, et cetera. All of them require far more knowledge. They require many of the things I've talked about before. And actually, far more than them.

And what's even worse is we also use language in pretty bizarre ways. So there are some kinds of idioms in English, like the market [AUDIO OUT] bullish, that you have to have seen before to understand, right? There's no reason to assume that bears are better or worse than bulls when you apply them to the stock market.

On the other hand, there are certain things that are very systematic. I can have an up day or a down day, because we've both kind of as a culture agreed that up is good and down is bad. Some cultures have made the opposite choice. But usually, it's up is good, down is bad.

So an idea can be grand or it can be small. Because we've decided big things are better than small things. Someone's mood can be dark or light. And these are very systematic variations that underlie all of language. And we constantly use metaphoric extension in order to describe what's going on around us and to talk about abstract things. It really seems as if this is kind of built-in to our model of the world. And modeling this is kind of over the horizon. And there are many, many, many other things that we're missing here.

So I just want to thank all my wonderful collaborators, like Boris, Max, Candace, and people at MIT, and people elsewhere. But to recap, what we saw is that we can get a little bit of traction on these problems. We can build one system that does one simple problem just connects our perception with our high-level knowledge, takes a video and a sentence and gives us a score. And once we have this interesting connection, this interesting feedback between these two very different-looking systems, it turns out that we can do many different and sometimes surprising things.