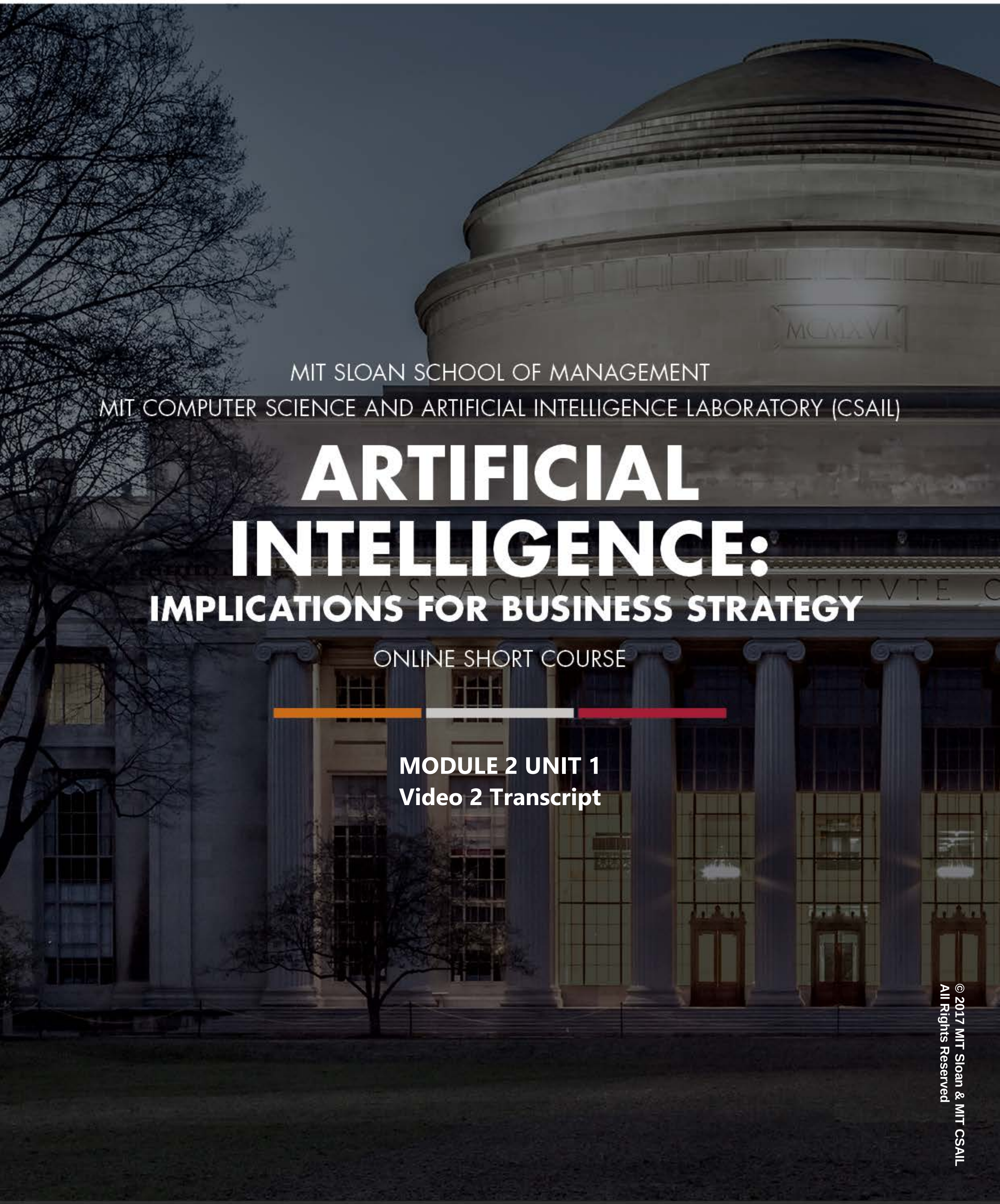

MIT SLOAN SCHOOL OF MANAGEMENT

MIT COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE LABORATORY (CSAIL)

ARTIFICIAL INTELLIGENCE: IMPLICATIONS FOR BUSINESS STRATEGY

ONLINE SHORT COURSE

MODULE 2 UNIT 1
Video 2 Transcript

Module 2 Unit 1 Video 2 Transcript

TOMMI JAAKKOLA: Now the question that you might have is why is this all happening now? Why didn't it happen, say, a decade ago? Why are all these sort of major advances seeming to happen right around now? And I'm going to give you four reasons for that, some are obvious and some not so much, and we will focus a little bit on those more subtle reasons.

So, the first reason is that now people are accumulating lots of data, so methods that learn from examples can really learn very refined solutions as they are given access to lots of examples. Reason number two is that we have lots of compute power, so we can make the models and methods much more complex, and learn them from lots of data. Reason number three is a little bit more subtle, which is that it might seem that these much more complex models, that have tens of millions of parameters, would be very hard to learn from data, but it turns out the opposite is true. In fact, it is easier to find a good solution once you have these very complicated architectures. So, there is a happy coincidence in that regard. And the last reason is that these deep learning architectures allow you to put very different types of objects on a common footing. You can easily learn common representations for things and move them from one representation to another. And you can also easily combine these from little Lego pieces into much more complex solutions.

So, let's look at just the last bit here, how that works, in the form of an example, just to appreciate how simple this actually is. So here is an illustration where I have a sentence, I have mapped each word into a vector representation as we did before but now the vector representations are all learned. I can take representations for words, learn to map those into representations for sentences. I can do the same for images, events, and so on. So now everything is in the same common vector representation. And what that allows me to do is I can take a sentence and I can generate a corresponding image. I can take an image and then learn to generate a verbal description of the image, and so on. It allows me to transfer from one domain to another in a very simple way.

How do I generate these representations for a complex object? So, let's take a very simple example in the context of sentences. So here I'm giving you now something called recurring neural network, which I will represent just as a little box that takes information that I already have and a new piece of information, both represented as vectors, and it transforms it into a summary of having incorporated the new information. That box has parameters that allows me to change what this transformation does. This box can be very simple or it can be more complicated, but it essentially works the same way – it transforms the data that I have, incorporates new pieces of information, and outputs, then, a summary of it in the same form.

Now, I don't know what the parameters should be, but I'm going to entertain a large number of possibilities here, and then select one that actually works better for the task that I'm using this box for. So, let's see how I can take this box with a particular set of parameters and use it to get a vector representation for a sentence. As a result, once I have that, I could then learn to classify sentences, just as I did movie descriptions, images, and so on.

So here I have the sentence below, now I can take just a null starting point. I don't know anything initially. I have a vector representation for the first word. I can just apply the box, having incorporated the first word. Now I can take the vector representation for the second word, apply the box again, and so on and so forth. As a result of following these steps, I will have a vector representation for a sentence.

Now the form of that representation depends on those parameters that I did not know. Okay? But I can now adjust those parameters based on how this vector functions for the classification task that I wish to use it for, or if I wish to take that vector and unravel it as a sentence in another language. And I can adjust that vector based on adjusting the parameters in that little box. And this is how I'm going to learn to generate very complex representations for complex objects and adjust them so that they work for the task that I wish to use them for.

So hopefully it is clear now how these things turn into Lego pieces. Now I have a little procedure here that takes any sentence and maps it into a vector, and we saw what actually underlies what's in that box and how it's represented. The Google machine translation system operates in a very similar way – it has a few additional bells and whistles, but operationally it is very similar to what we just described.

Another point that I would want to make about this advantage is that they seem to require lots of data to give the problem in terms of examples of correct behavior. Now that's not entirely true, so you can get by with very little data. So, the question of how much data do I need to solve a particular problem is, I would argue, the wrong question. The right question is what data I have prior to trying to solve that problem. So, for example, in terms of a machine translation, I could have already solved that for one language pair, and then the question is what additional things do I need in order to translate sentences into new language? Machines can learn from very few examples, even from a single example. It all depends on how much data I already have, that I have been able to learn from, prior to solving that new task.

There are lots of different types of machine learning problems, and the only one that we've discussed so far is a supervised learning problem where I've given you explicitly an input example and the corresponding target label that I wish to predict for that instance. Unsupervised learning is a problem where I try to model the data that I see without having any explicit feedback about what I need to do with that data. You can combine the two into a semi-supervised learning. You can actively formulate the method in such a way that you can actively query new data so as to learn better. You can learn to transfer from having solved one problem to better solve another without requiring lots of data. And we've already seen some hints about how to learn to control machine learning methods.

The last problem that I would wish to highlight is one that comes with incorporating these very complex models, and that is to make them interpretable. We've talked about deep learning architectures that perform multiple layers of transformation of those examples, and those transformations are learned for the task that you wish to use them for. But, in doing so, while they solve the problem, they remain quite opaque about how they actually solve the problem. So, there is a question, then, how we can learn to interpret and trust the predictions in those methods? So, we need to make them interpretable. We want the methods to also highlight the rationale behind their predictions. This is a two-way problem – once the methods going to articulate better how and why they make their predictions, it also gives us access to guide them in times of our intuition about how they should try to solve the problem better.

So, with that, I would just highlight a few key concepts that we covered, which is posing supervised machine learning problems, learning from examples. That I have a training set, that's the examples given, but what I wish to actually solve is the problem on the test set, for examples that I don't yet have. The key problem here, fundamentally, is a problem of generalization – how do I learn from the training set so that I can generalize and do well on the task that I have not yet seen? And there is a theory and methods for understanding that problem as well. We also discussed how these complex deep learning architectures are taking over machine learning problems and enabling us to solve really challenging problems, and that trend is rapidly accelerating.

THOMAS MALONE: Did you understand all the concepts covered in this video? If you'd like to go over any of the sections again, please click on the relevant button.