A short introduction to machine learning

by **Richard Ngo** 30th Aug 2021

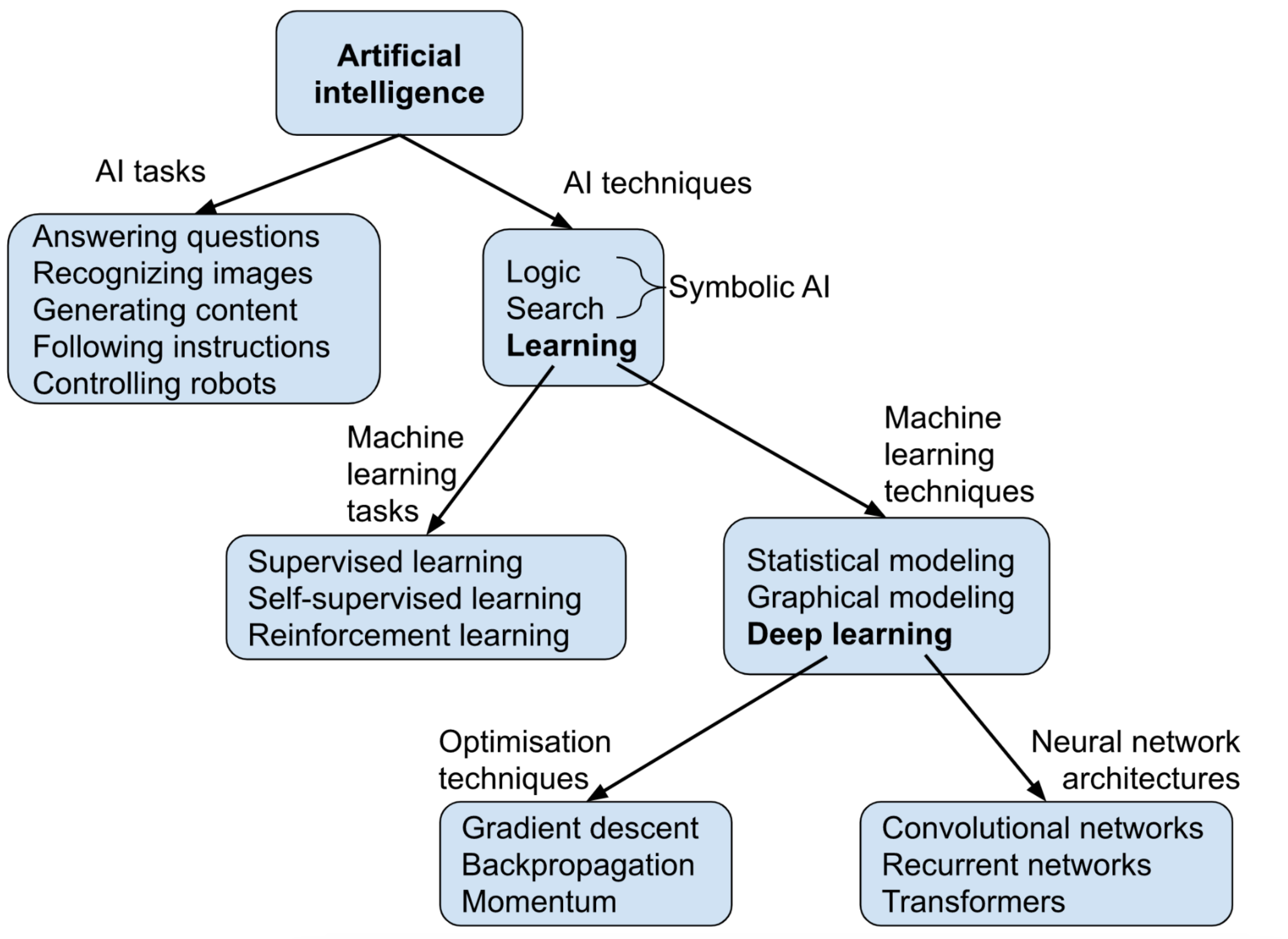
24

AI Frontpage

Despite the current popularity of machine learning, I haven't found any short introductions to it which quite match the way I prefer to introduce people to the field. So here's my own. Compared with other introductions, I've focused less on explaining each concept in detail, and more on explaining how they relate to other important concepts in AI, especially in diagram form. If you're new to machine learning, you shouldn't expect to fully understand most of the concepts explained here just after reading this post - the goal is instead to provide a broad framework which will contextualise more detailed explanations you'll receive from elsewhere.

I'm aware that high-level taxonomies can be controversial, and also that it's easy to fall into the illusion of transparency when trying to introduce a field; so suggestions for improvements are very welcome!

The key ideas are contained in this summary diagram:



First, some quick clarifications:

- None of the boxes are meant to be comprehensive; we could add more items to any of them. So you should picture each list ending with "and others".
- The distinction between *tasks* and *techniques* is not a firm or standard categorisation; it's just the best way I've found so far to lay things out.
- The summary is explicitly from an AI-centric perspective. For example, statistical modeling and optimization are fields in their own right; but for our current purposes we can think of them as machine learning techniques.

Let's dig into each part of the diagram now, starting from the top.

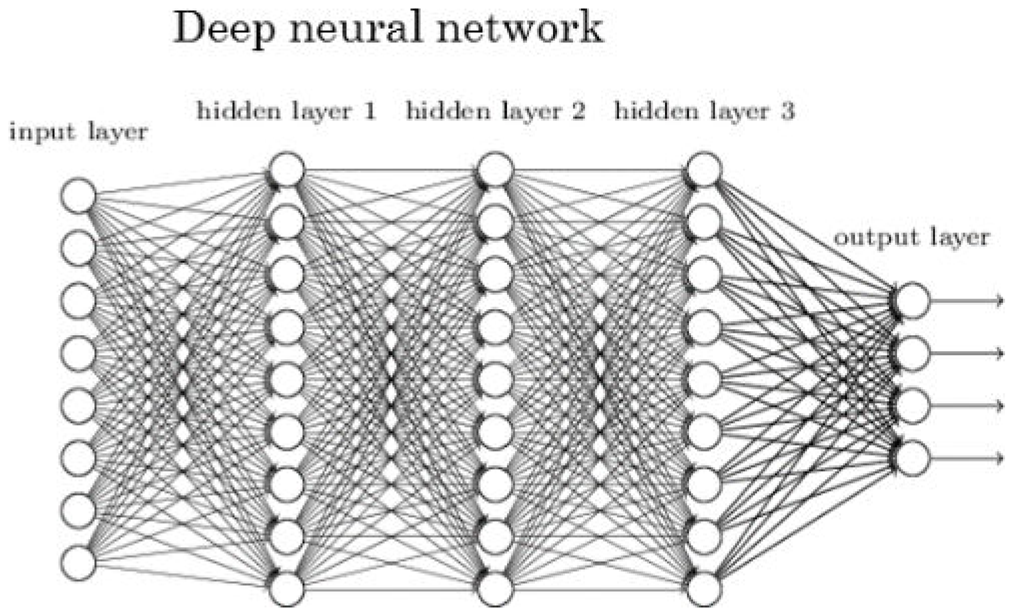
Paradigms of artificial intelligence

The field of **artificial intelligence** aims to develop computer programs that are able to perform useful tasks like answering questions, recognizing images, and so on. It got started around the 1950s. Historically, there have been several different approaches to AI. In the first few decades, the dominant paradigm was **symbolic AI**, which focused on representing problems using statements in formal languages (like logic, or programming languages), and searching for solutions by manipulating those representations according to fixed rules. For example, a symbolic AI can represent a game of chess using a set of statements about where the pieces currently are, and a set of statements about where the pieces are allowed to move (you can only move bishops diagonally, you can't move your king into check, etc). It can then play chess by searching through possible moves which are consistent with all of those statements. The power of symbolic search-based AI was showcased by Deep Blue, the chess AI that beat Kasparov in 1997.

However, the symbolic representations designed by AI researchers turned out to be far too simple: there are very few real-world phenomena easily describable using formal languages (despite valiant efforts). Since the 1990s, the dominant paradigm in AI has instead been **machine learning**. In machine learning, instead of manually hard-coding all the details of AIs ourselves, we specify models with free parameters that are learned automatically from the data they're given. For example, in the case of chess, instead of using a fixed algorithm like Deep Blue does, a ML chess player would choose moves using parameters that start off random, and gradually improve those parameters based on feedback on its moves: this is known as the *learning, training* or *optimization* process.* In theory, statistical models (including simple models like linear regressions) also fit parameters to the data they're given. However, the two fields are distinguished by the scales at which they operate: the biggest successes of machine learning have come from training models with billions of parameters on huge amounts of data. This is done using **deep learning**, which involves training *neural networks* with many layers using powerful *optimization techniques* like gradient descent and backpropagation. Neural networks have been around since the beginning of AI, but they only became the dominant paradigm in the early 2010s, after increases in compute availability allowed us to train much bigger networks. Let's explore the components of deep learning in more detail now.

Deep learning: neural networks and optimization

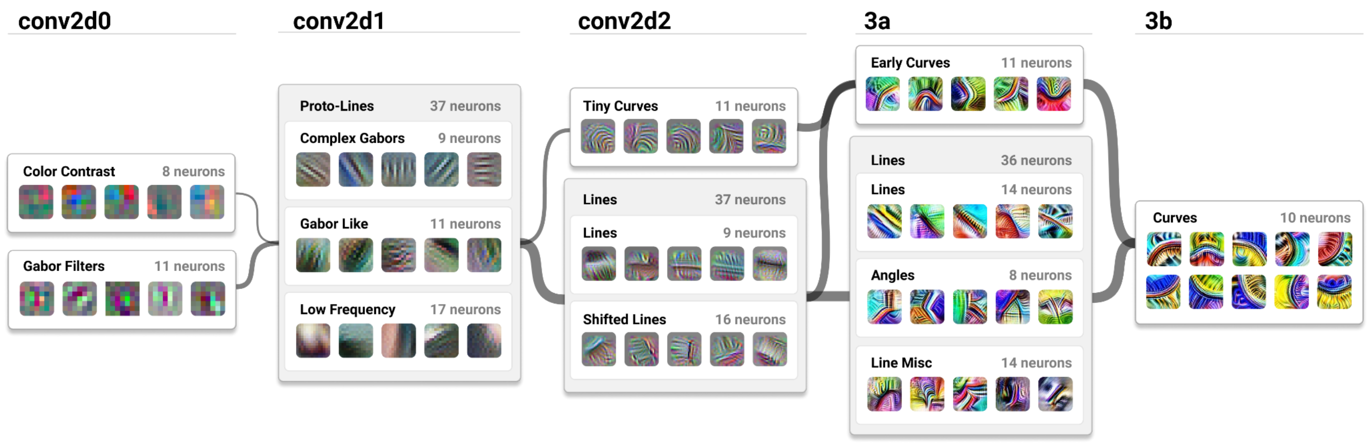
Neural networks are a type of machine learning model inspired by the brain. As with all machine learning models, they take in input data and produce corresponding output data, in a way which depends on the values of their parameters. The interesting part is *how* they do so: by passing that data through several layers of simple calculations, analogous to how brains process data by passing it through layers of interconnected neurons. In the diagram below, each circle represents an "artificial neuron"; networks with more than one layer of neurons between the input and the output layers are known as *deep* neural networks. These days, almost all neural networks are deep, and some have hundreds of layers.



Each artificial neuron receives signals from neurons in the previous layer, combines them together into a single value (known as its *activation*), and then passes that value on to neurons in the next layer. As in biological brains, the signal that is passed between a pair of artificial neurons is affected by the strength of the connection between them - so for each of the lines in the diagram we need to store a single number representing the strength of the connection, known as a *weight*. The weights of a neuron's connections to the previous layer determines how strongly it activates for any given input. (Compared with biological brains, artificial neural networks tend to be much more strictly organised into layers.)

These weights are not manually specified, but instead they are learned via a process of **optimization**, which finds weights that make the network score highly on whatever metric we're using. (This metric is known as an *objective function* or *loss function*; it's evaluated over whatever dataset we're using during training.) By far the most common optimization algorithm is **gradient descent**, which initially sets weights to arbitrary values, and then at each step changes them so that the network does slightly better on its objective function (in more technical terms, it updates each weight in the direction of its gradient with respect to the objective function). Gradient descent is a very general optimization algorithm, but it's particularly efficient when applied to neural networks because at each step the gradients of the weights can be calculated layer-by-layer, starting from the last layer and working backwards, using the **backpropagation** algorithm. This allows us to train networks which contain billions of weights, each of which is updated billions of times.

As a result of optimization, the weights end up storing information which allows different neurons to recognise different features of the input. As an example, consider a neural network known as *Inception*, which was trained to classify images. Each neuron in Inception's input layer was assigned to a single pixel of the input image. Neurons in each successive layer then learned to activate in response to increasingly high-level features of the input image. The diagram shows some of the patterns recognised by neurons in five consecutive layers from the Inception model, in each case by combining patterns from the previous layer - from colours to (Gabor filters for) textures to lines to angles to curves. This goes on until the last layer, which represents the network's final output - in this case the probabilities of the input image containing cats, dogs, and various other types of object.



One last point about neural networks: in our earlier neural network diagram, every neuron in a given layer was connected to every neuron in the layers next to it. This is known as a fully-connected network, the most basic type of neural network. In practice, fully-connected networks are seldom used; instead there are a whole range of different neural network architectures which connect neurons in different ways. Three of the most prominent (convolutional networks, recurrent networks, and transfomers) are listed in the original summary diagram; however, I won't cover any of the details here.

Machine learning tasks

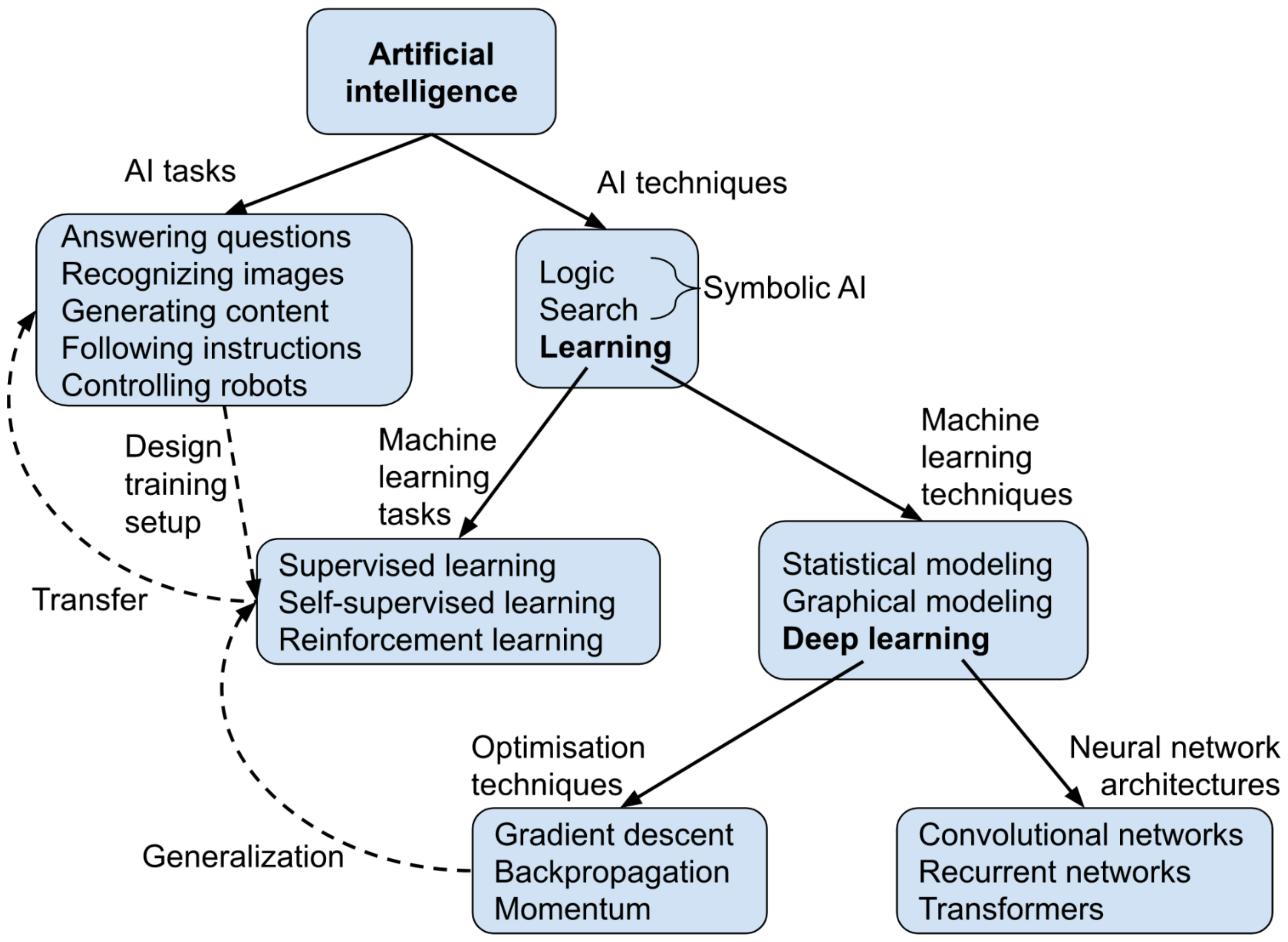
I've described how neural networks (and other machine learning models) can be trained to perform different tasks. The three most prominent categories of tasks are supervised, self-supervised, and reinforcement learning, which each involve different types of data and objective functions. **Supervised learning** requires a dataset where each datapoint has a corresponding label. The objective in supervised learning is for a model to predict the labels which correspond to each datapoint. For example, the image classification network we discussed above was trained on a dataset of images, each labeled with the type of object it contained. Alternatively, if the labels had been ratings of how beautiful each image was, we could have used supervised learning to produce a network that rated image beauty. These two examples showcase different types of supervised learning: the former is a *classification problem* (requiring the prediction of discrete categories) and the latter is a *regression problem* (requiring the prediction of continuous values). Historically, supervised learning has been the most studied task in machine learning, and techniques devised to solve it have been extensively used as parts of the solutions to the other two.

One downside of supervised learning is that labeling a dataset usually needs to be done manually by humans, which is expensive and time-consuming. Learning from an unlabeled dataset is known as **unsupervised learning**. In practice, this is typically done by finding automatic ways to convert an unlabeled dataset into a labeled dataset, which is known as **self-supervised learning**. The standard example of self-supervised learning is next-word prediction: training a model to predict, from any given text sequence in an unlabeled dataset, which word follows that sequence. Some impressive applications of self-supervised learning are GPT-2 and GPT-3 for language, and Dall-E for images.

Finally, in **reinforcement learning**, the data source is not a fixed dataset, but rather an *environment* in which the AI takes actions and receives observations - essentially as if it's playing a video game. After each action, the agent also receives a reward (similar to the score in a video game), which is used to reinforce the behaviour that leads to high rewards, and reduce the behaviour that leads to low rewards. Since actions can have long-lasting consequences, the key difficulty in reinforcement learning is determining which actions are responsible for which rewards - a problem known as *credit assignment*. So far the most impressive demonstrations of reinforcement learning have been in training agents to play board games and esports - most notably AlphaGo, AlphaStar and OpenAI Five.**

Solving real-world tasks

We're almost done! But I don't think that even a brief summary of AI and machine learning can be complete without adding three more concepts. They don't quite fit into the taxonomy I've been using so far, so I've modified the original summary diagram to fit them in:



Let's think of these three dotted lines I've added as ways to connect the different levels. The ultimate goal of the field of AI is to create systems that can perform valuable tasks in the real world. In order to apply machine learning techniques to achieve this, we need to design and implement a supervised/self-supervised/reinforcement **training setup** which allows systems to learn the necessary abilities. A key element is designing datasets or training environments which are as similar as possible to the real-world task. In reinforcement learning, this also requires designing a reward function to specify the desired behaviour, which is often more difficult than we expect.

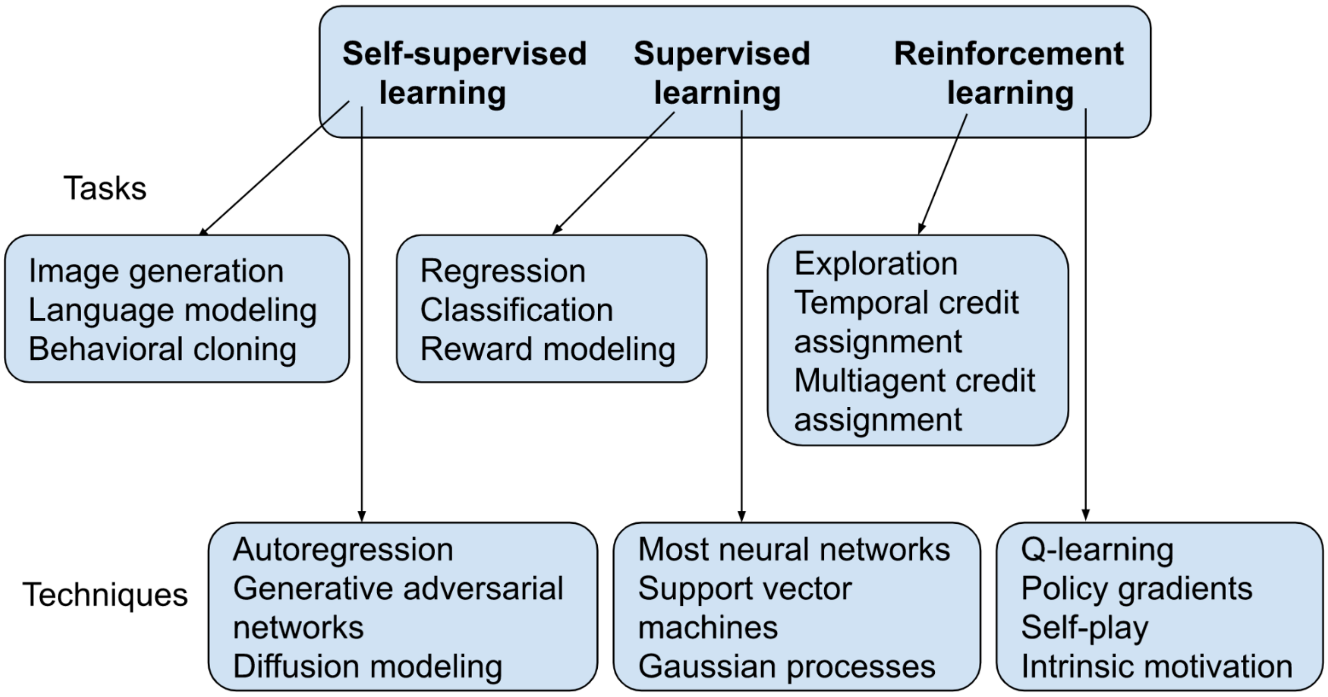
But no matter how good our training setup, we will face two problems. Firstly, we can only ever train our models on a finite amount of data. For example, when training an AI to play chess, there are many possible board positions that it will never experience. So our optimization algorithms could in theory produce chess AIs that can only play well on positions that they already experienced during training. In practice this doesn't happen: instead deep learning tends to **generalise** incredibly well to examples it hasn't seen already. How and why it does so is, however, still poorly-understood.

Secondly, due to the immense complexity of the real world, there will be ways in which our training setups are incomplete or biased representations of the real-world tasks we really care about. For example, consider an AI which has been trained to play chess against itself, and which now starts to play against a human who has very different strengths and weaknesses. Playing well against the human requires it to **transfer** its original experience to this new task (although the line between generalisation to different examples of "the same task" versus transfer to "a new task" is very blurry). We're also beginning to see neural networks whose skills transfer to new tasks which differ significantly from the ones on which they were trained - most notably the GPT-3 language model, which can perform a very wide range of tasks°. As we develop increasingly powerful AIs that perform increasingly important real-world tasks, ensuring their safe behaviour will require a much better understanding of how their skills and motivations will transfer from their training environments to the wider world.

Footnotes

* *Learning*, *training* and *optimization* have slightly different connotations, but they all refer to the process by which a machine learning system updates its parameters based on data.

** Here's a more detailed breakdown of some of the tasks and techniques corresponding to these three types of learning. I've only mentioned a few of these terms so far; I've included the others to help you classify them in case you've seen them before, but don't worry if many of them are unfamiliar.



AI 2 Frontpage

Mentioned in

33 AGI Safety Fundamentals curriculum and application

1 comment, sorted by top scoring

[-] **Kerrigan** 3mo 1

How was Dall-E based on self-supervised learning? The datasets of images weren't labeled by humans? If not, how does it get form text to image?

Moderation Log