COMPUTATIONAL APPLICATIONS TO POLICY AND STRATEGY (**CAPS**)

Session 5 – Neural Network Models

Leo Klenner, Henry Fung, Cory Combs

Outline

Today's agenda:

1. Perceptrons
2. Basic components of neural networks (NNs)
3. Adapting NNs for all learning approaches
4. How dark is the black box?

Big-picture Goal:

Understand the foundational mechanics and key applications of a set of learning algorithms called neural networks.

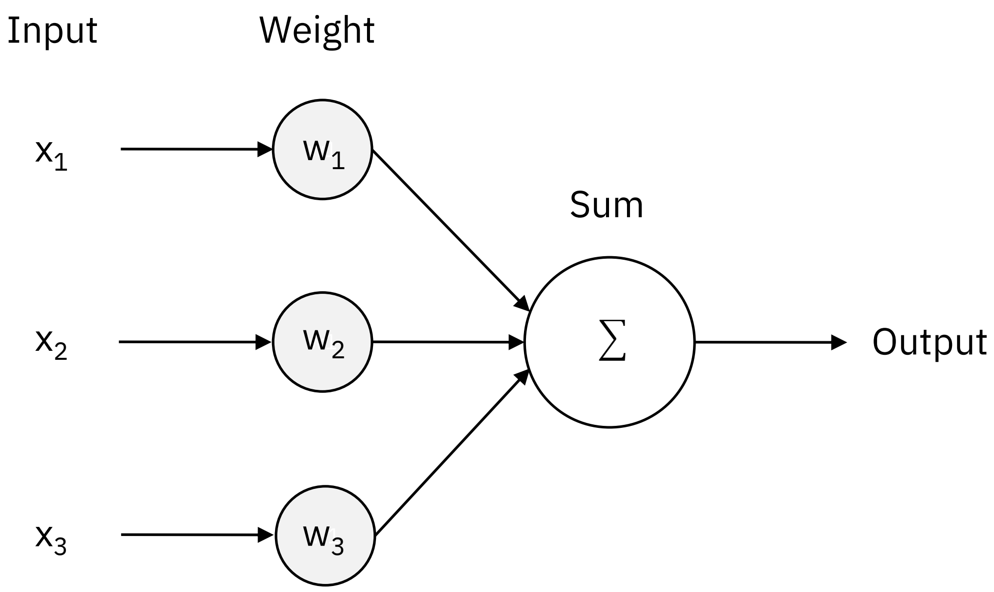
Be an informed observer of and participant in the growth of modern AI systems.

1. Perceptrons

How can a computer begin to “perceive”?

1.1 Perceptrons

- **Perceptrons** are decision-making models that makes decisions by weighting evidence.
- **Weights** are numbers that express the importance of each input (“evidence”).
- Inputs and outputs of perceptrons are binary variables (1 or 0)
- If the weighted sum of the inputs is greater than a **threshold**, then the output is 1, otherwise it is 0.



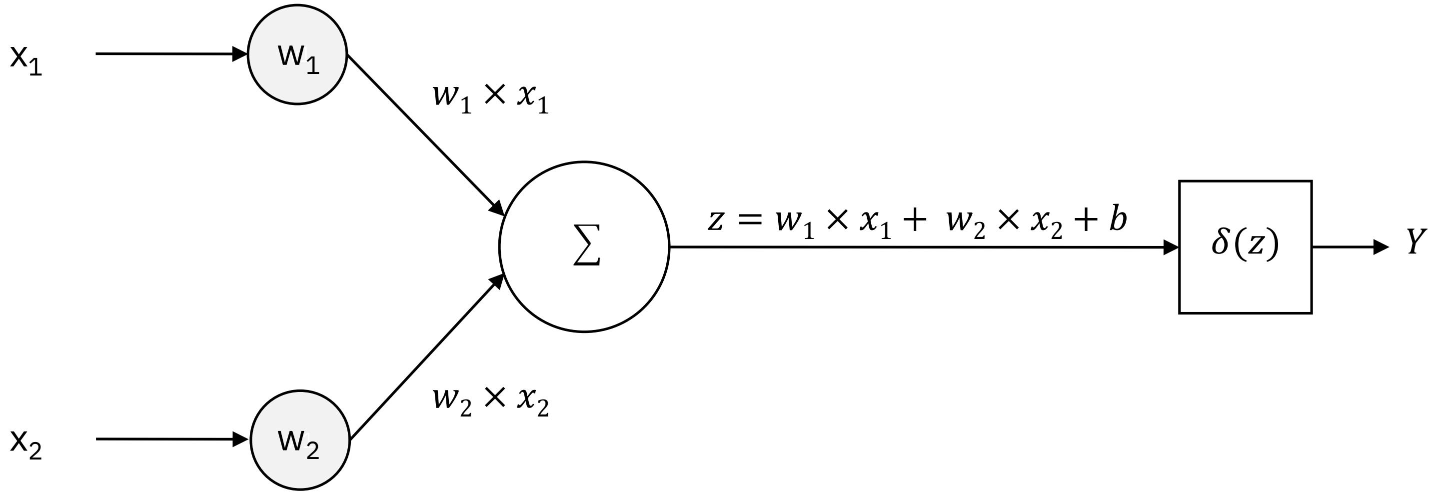
Sketch of a perceptron

1.2 Limitations of Perceptrons

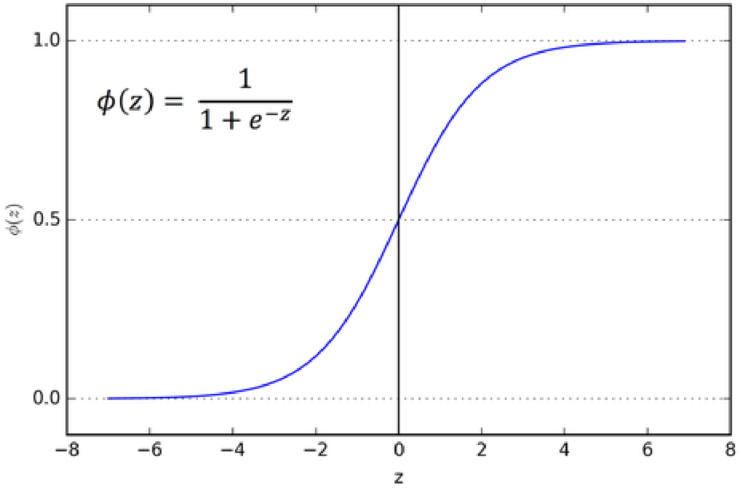
- **Key:** Design model that “learns” weights and biases automatically; progressively the model behaves in the manner we want.
- **Desired specifications:**
  - Continious inputs and outputs
  - Incremental change in i → incremental change in o
- **Problem:** The above specs cannot be achieved with a network of perceptrons. Why? A small change in weights or biases of a perceptron either does nothing or shifts the output from 1 to 0 or v.v. Since the output of one perceptron might be the input of another, a small change in weights or biases will completely change the behavior of the entire network.

1.2 Modified Perceptrons: Sigmoid Neurons

- **Solution:** We need to "smooth out" the output of a perceptron by passing it through an "activation" function called the sigmoid function.



Perceptron with sigmoid activation function



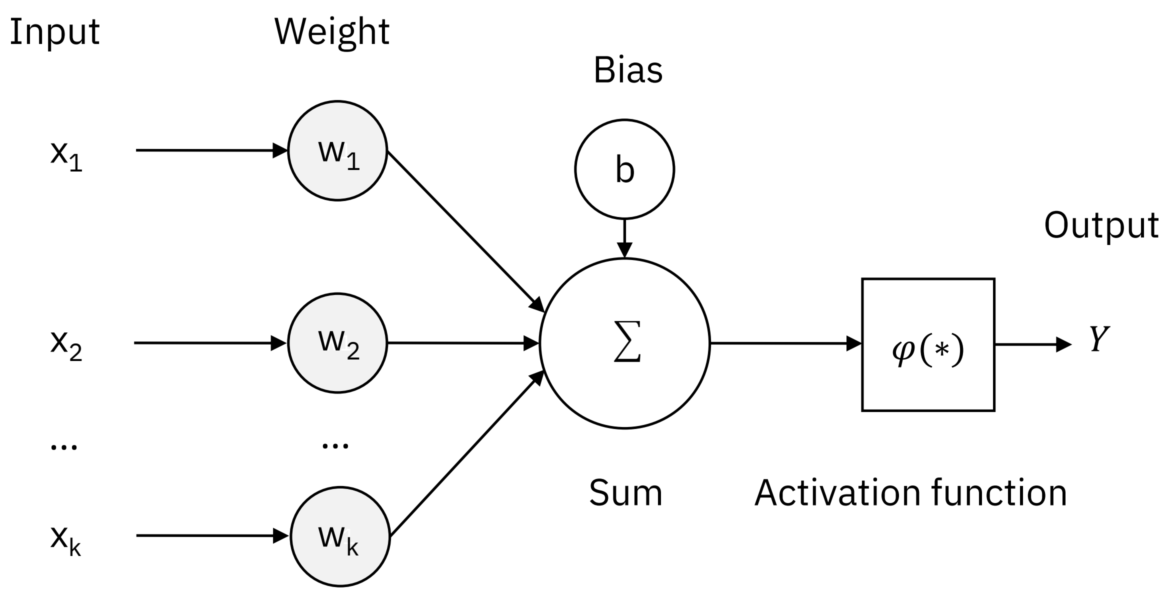
Sigmoid function

2. Neural Networks

Stacking neurons: an attempt at human cognition

2.1 The Basic Neural Network

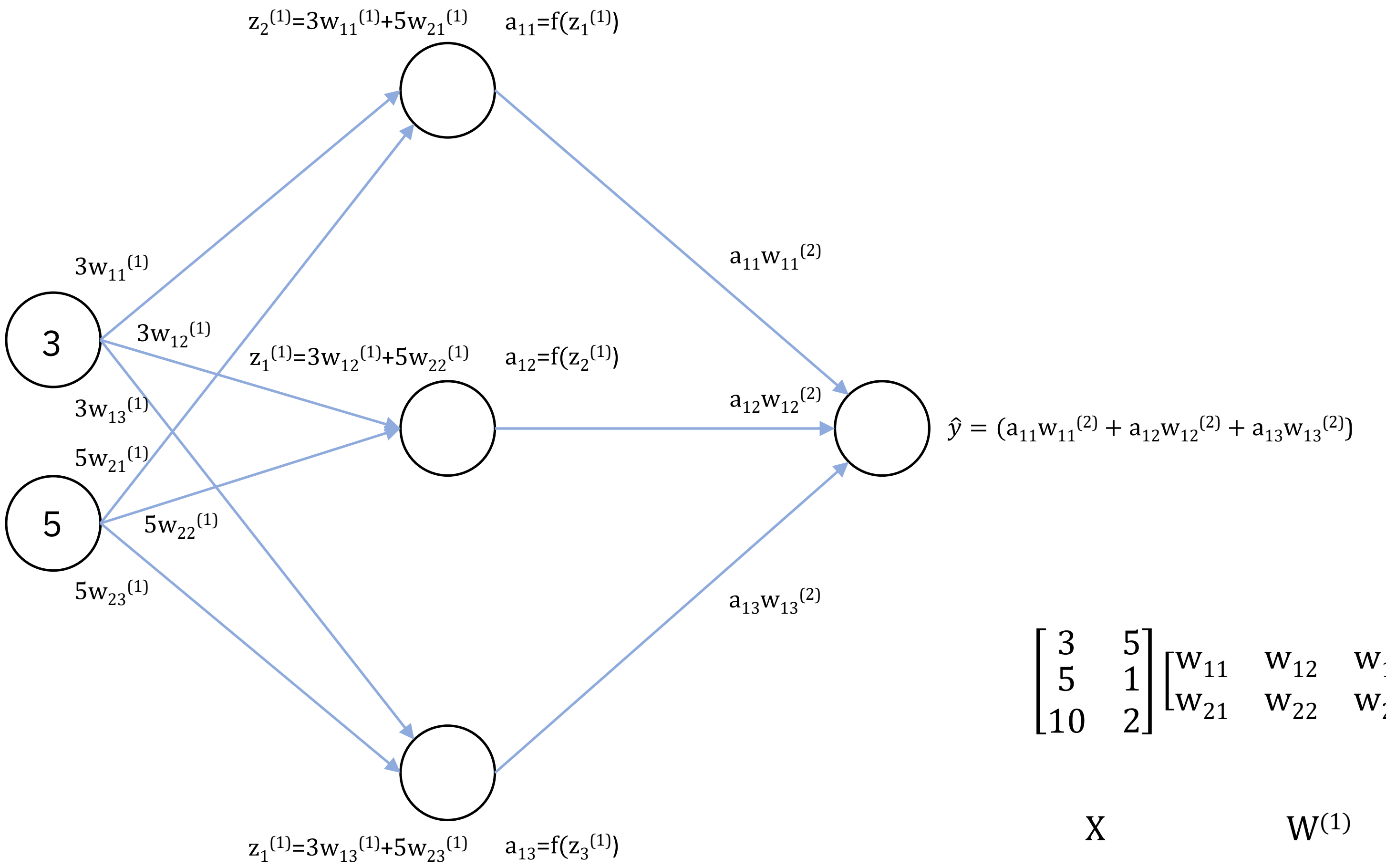
- **Key components:**
  - Inputs
  - Weights (learning)
  - Bias (correction)
  - Activation function
  - Outputs



Sketch of a basic neural net

- Different types of inputs and outputs can be used to achieve any type of learning.

2.2 Visual vs. Mathematical Representations of a NN



Annotated visual representation of a simple NN

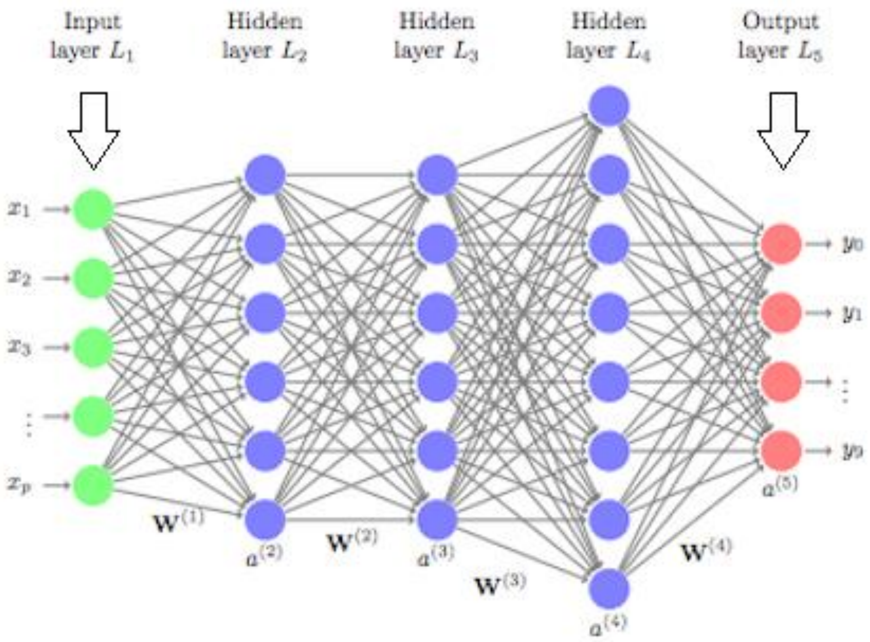
$$\begin{bmatrix}3 & 5\\ 5 & 1\\ 10 & 2\end{bmatrix}\begin{bmatrix}w_{11} & w_{12} & w_{13}\\ w_{21} & w_{22} & w_{23}\end{bmatrix} = \begin{bmatrix}3w_{11}^{(1)}+5w_{21}^{(1)} & 3w_{12}^{(1)}+5w_{22}^{(1)} & 3w_{13}^{(1)}+5w_{23}^{(1)}\\ 5w_{11}^{(1)}+1w_{21}^{(1)} & 5w_{12}^{(1)}+1w_{22}^{(1)} & 5w_{13}^{(1)}+1w_{23}^{(1)}\\ 10w_{11}^{(1)}+2w_{21}^{(1)} & 10w_{12}^{(1)}+2w_{22}^{(1)} & 10w_{13}^{(1)}+2w_{23}^{(1)}\end{bmatrix}$$

X $\quad$ $W^{(1)}$ $\quad$ Z^1

The math behind the model

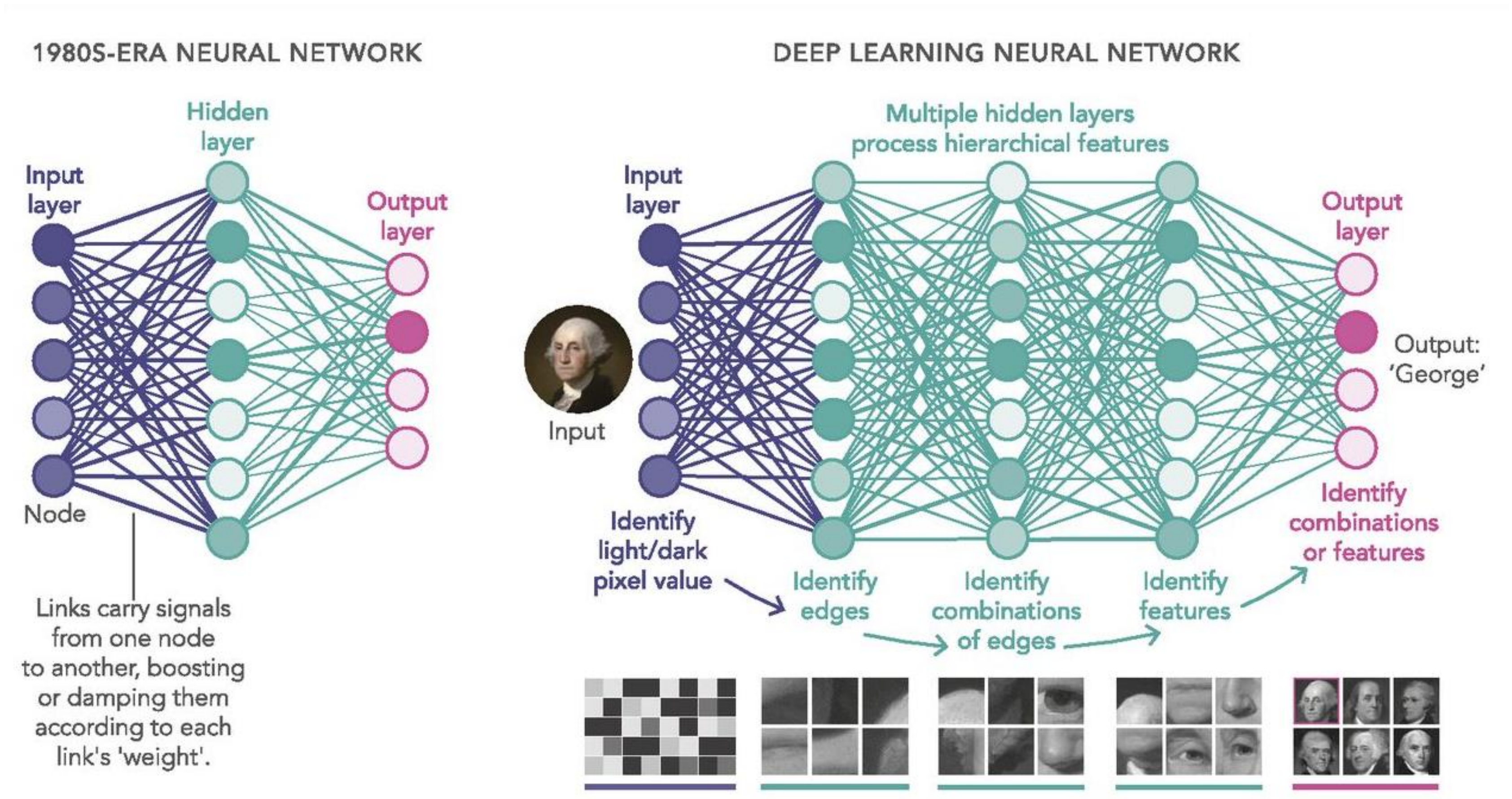
2.3 Going "Deeper": Deep Learning

- **The big picture**
  - Neurons are grouped into **layers**
  - Layers between the input layer and the output layer are called **hidden layers**
  - A system that has multiple hidden layers is called **deep**
- **Differentiated learning**
  - Each hidden layer conducts a new processing of weighting, which is how the system "learns"
  - However, the system can achieve very different aims with each layer



Source: University of Cincinnati

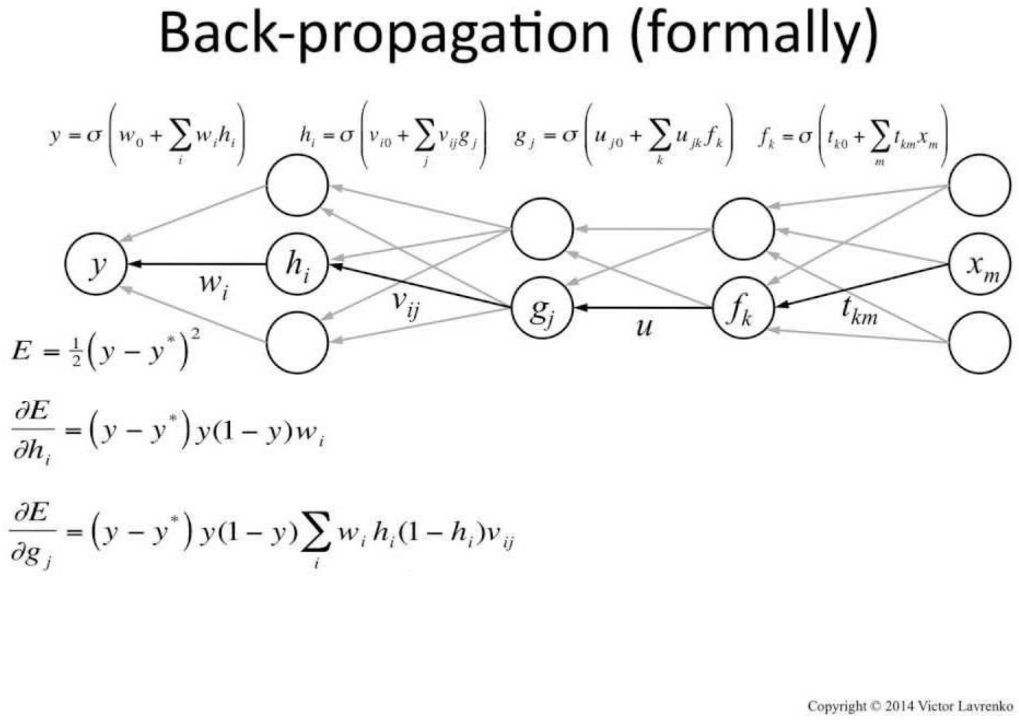
2.4 Neural Networks Over Time



Source: [PNAS]()

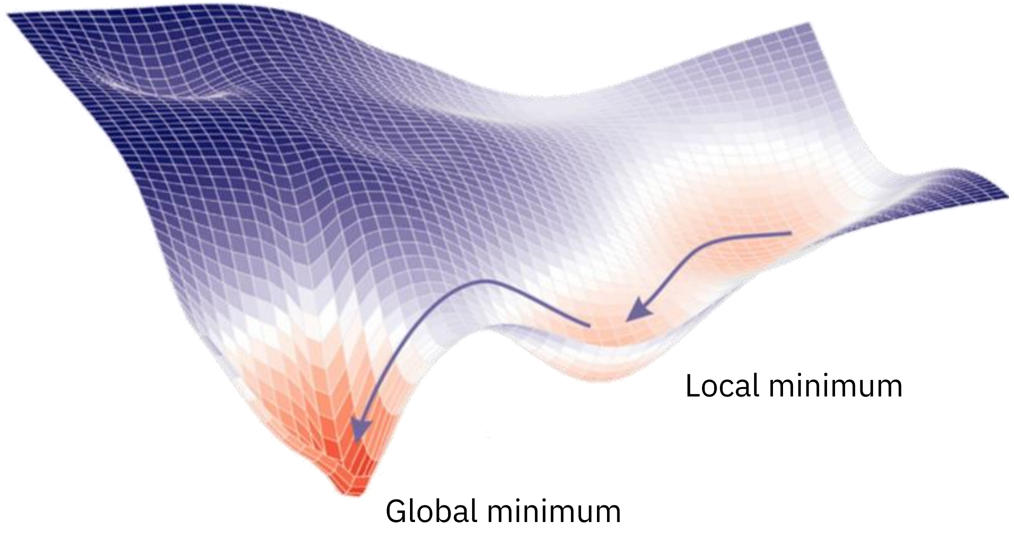
2.5 Learning from Mistakes (Given Labels), Part 1: Backpropagation

- **Problem: how do we improve the model?**
  - What we have seen is "forward propagation".
  - Our goal is to minimize loss (error) – but how?
- **Solution: "backward propagation"**
  - Once we have outputs, we can examine their error. We can then modify the weights of the previous layer to *lessen* that error.
  - But that layer depends on the previous layer… so then *that* layer gets modified. And so on. Backward all the way to the start.



2.6 Learning from Mistakes (Given Labels), Part 2: Gradient Descent

- **But how? Introducing the gradient**
  - Mathematically, we want the partial derivative of the error with respect to each weight.
  - In other words, we want the partial derivative of the *dependent variable* with respect to each *independent* variable. This group of partial derivatives is called **the gradient**.
  - The gradient of the final hidden layer is calculated first; this gradient is used in the gradient of the second to last hidden layer; and so on, until it reaches the first layer.
  - With all weights related, the model can **optimize the weights** by **minimizing the gradient**.
  - Hence, **backprop** and **gradient descent** work together in optimizing neural networks.



Visualization of moving down the gradient to reach a global optimum; hence the name gradient descent.

2.7 The Black Box

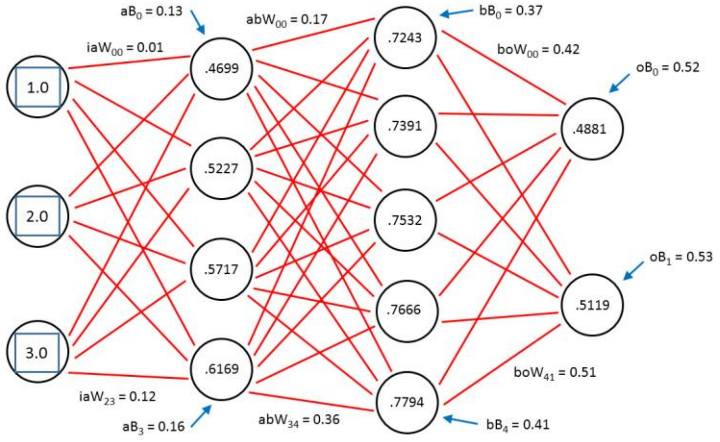
- **Seeing vs. understanding**
  - We can **see** what is happening inside a neural network at any stage. It’s usually just printing some tensors.
  - But what does that mean for our *understanding* of it?
- **Policy consequences**
  - It is usually difficult to explain *why* the model does what it does in any useful way.
  - Sure, the basic answer is “to optimize a loss function”; but that does not explain why a certain NN thought a white van was a cloud or inidcate a solution for how to fix this mistake.

```
inp/(1-p)

tensor([[1.0808, 0.5191, 0.2710, 0.1703],
        [1.2254, 0.7723, 1.3676, 0.7046],
        [0.0537, 1.5665, 1.5272, 0.6948],
        [0.4290, 0.0778, 0.3689, 1.0284],
        [0.6909, 0.3813, 0.0646, 1.2920]])

outp

tensor([[1.0808, 0.5191, 0.2710, 0.0000],
        [0.0000, 0.7723, 0.0000, 0.0000],
        [0.0000, 1.5665, 1.5272, 0.6948],
        [0.4290, 0.0778, 0.3689, 1.0284],
        [0.6909, 0.0000, 0.0646, 0.0000]])
```



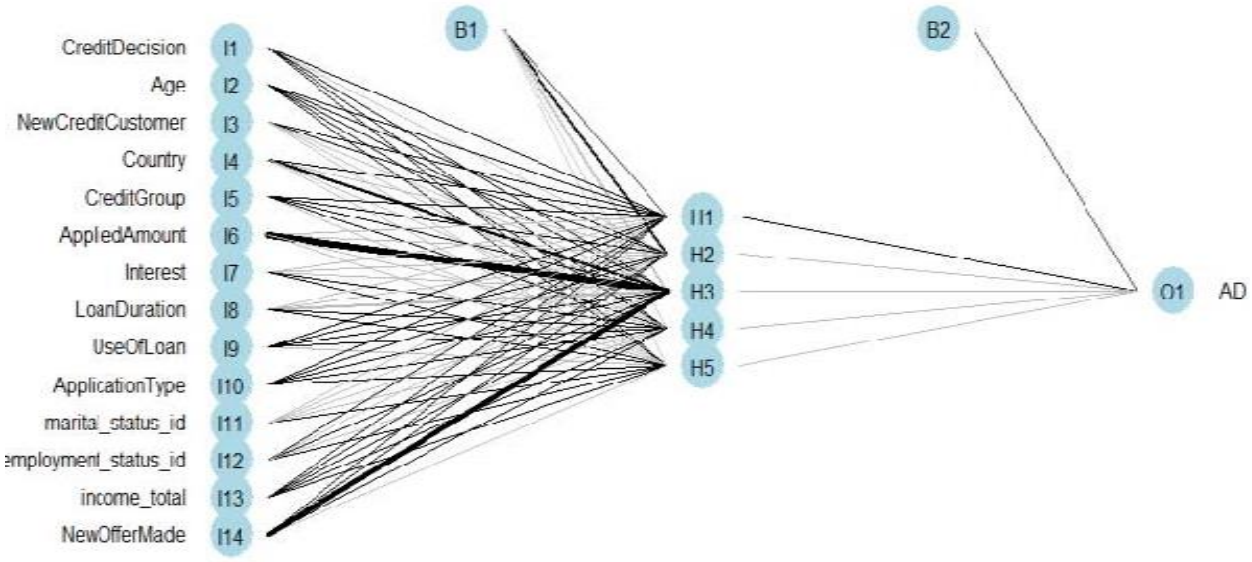
3. The Learning Landscape

How can NNs be applied across every field of ML?

3.1 Supervised Learning

- **Supervised learning requires labeled data**
  - This is a natural setup for a neural network…
  - …so let's setup a NN in Google Colab!

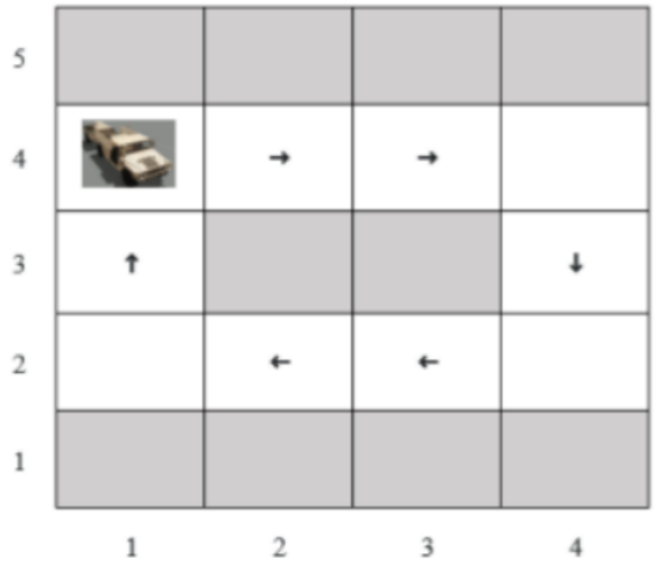
- **Always ask…**
  - What type of **input** do we need?
  - What type of **output** do we expect?
  - What types of learning should happen in the hidden layers?



Source: [Ajay Byanjankar et al. 2015]()

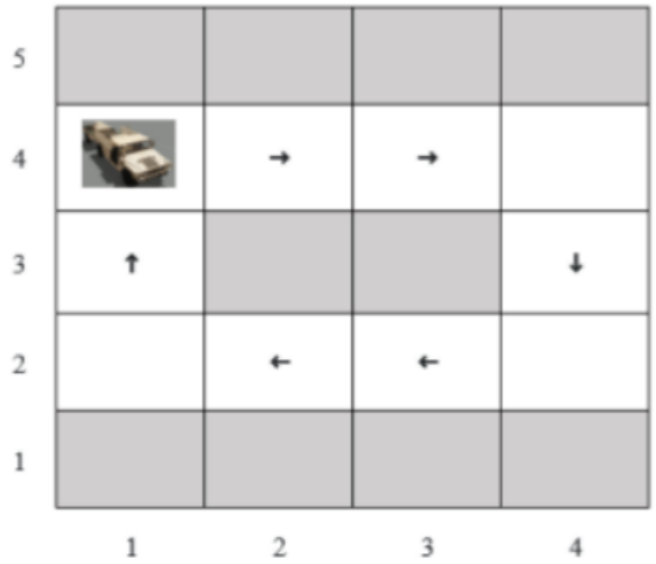
3.2 Deep Reinforcement Learning

- What **inputs** would you give an NN doing RL? What **outputs** would you get?
- Let's review the autonomous Humvee case…



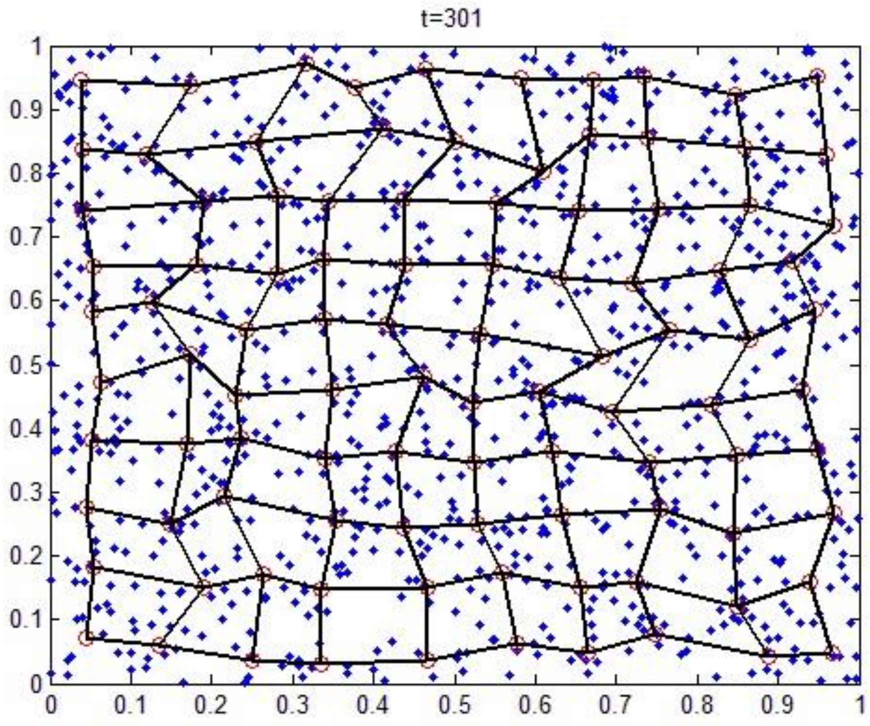
3.2.1 Deep RL Takeaways

- Neural networks **build functions**
- Hence, they can be used to estimate an appropriate **policy** or **value function** for an RL problem
- This allows us to build a function by **sampling** states and possible actions, making RL feasible where we do not know the entire environment



3.3 Unsupervised Learning and Beyond

- **Tougher, but still possible…**
  - NNs can learn to group and classify data using **self-organizing maps**, built on biological models and morphogenesis models and… to be frank, things get incredibly complicated. But also powerful.
- **General summary**
  - Both the power and the (general) opacity of NNs are enabled by having interacting **layers**, making them not merely *complicated*, but *complex*.



A self-organizing map

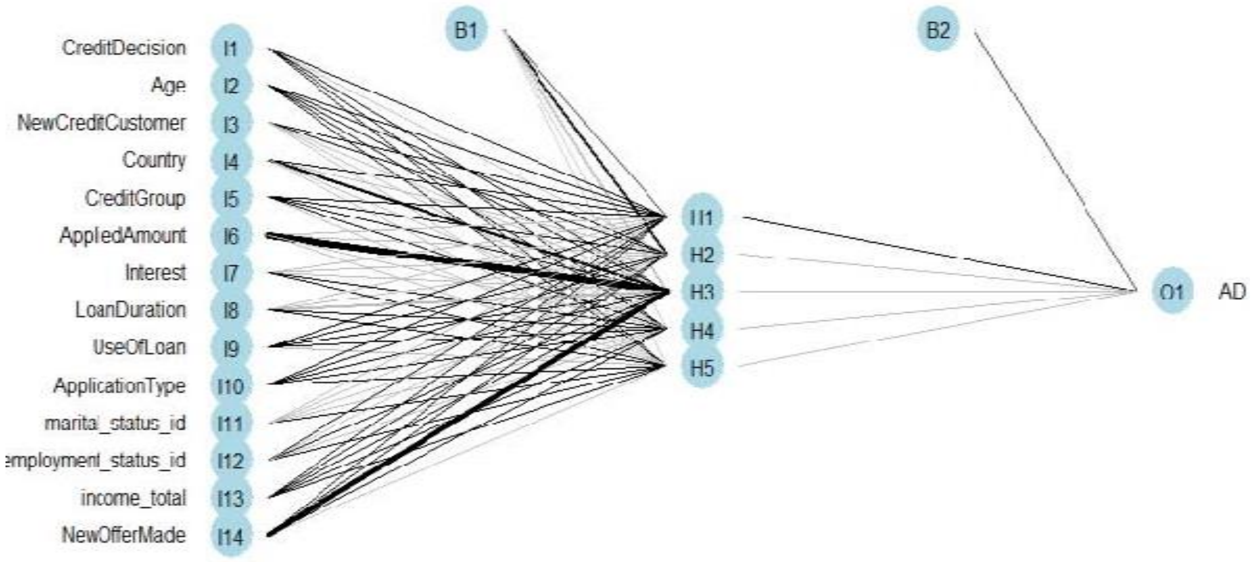
4. Neural Networks in Practice

Case studies on deployed neural networks

4.1 Credit Modeling Demo

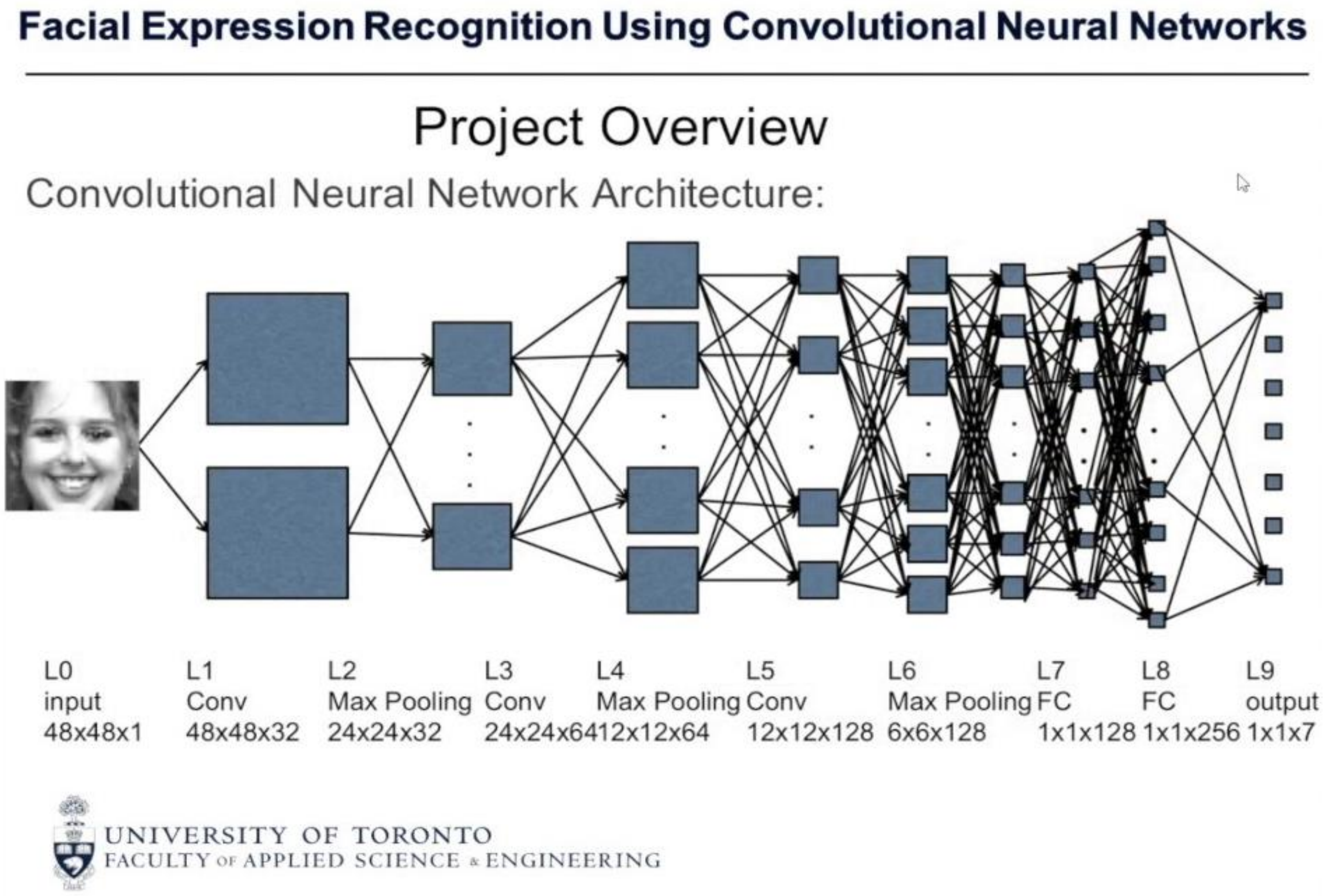
- **Supervised learning requires labeled data**
  - This is a natural setup for a neural network…
  - …so let’s setup a NN in Google Colab!

- **Always ask…**
  - What type of **input** do we need?
  - What type of **output** do we expect?
  - What types of learning should happen in the hidden layers?

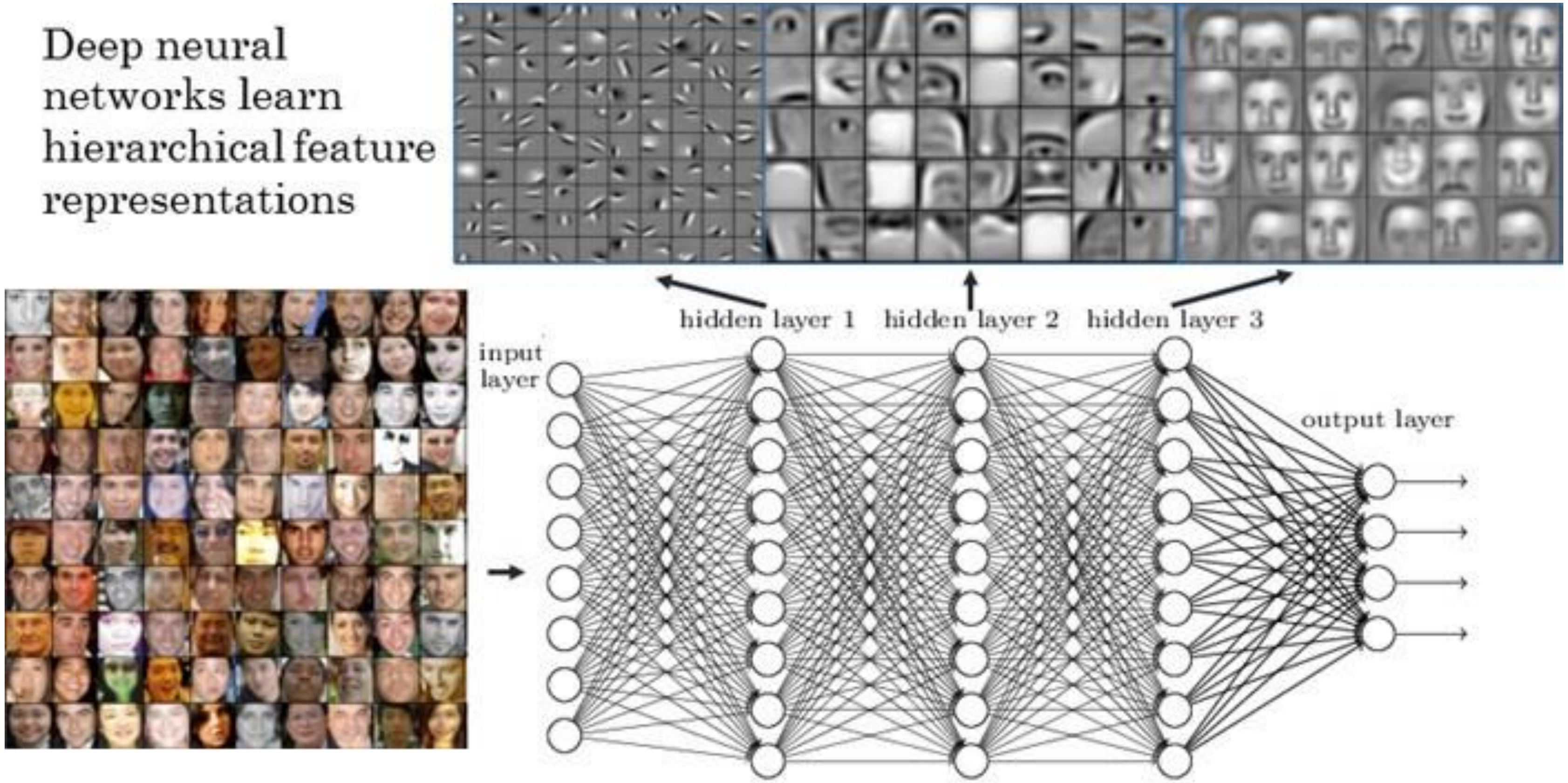


Source: *[Ajay Byanjankar et al. 2015]*

4.2 Facial Recognition Part 1



4.3 Facial Recognition Part 2

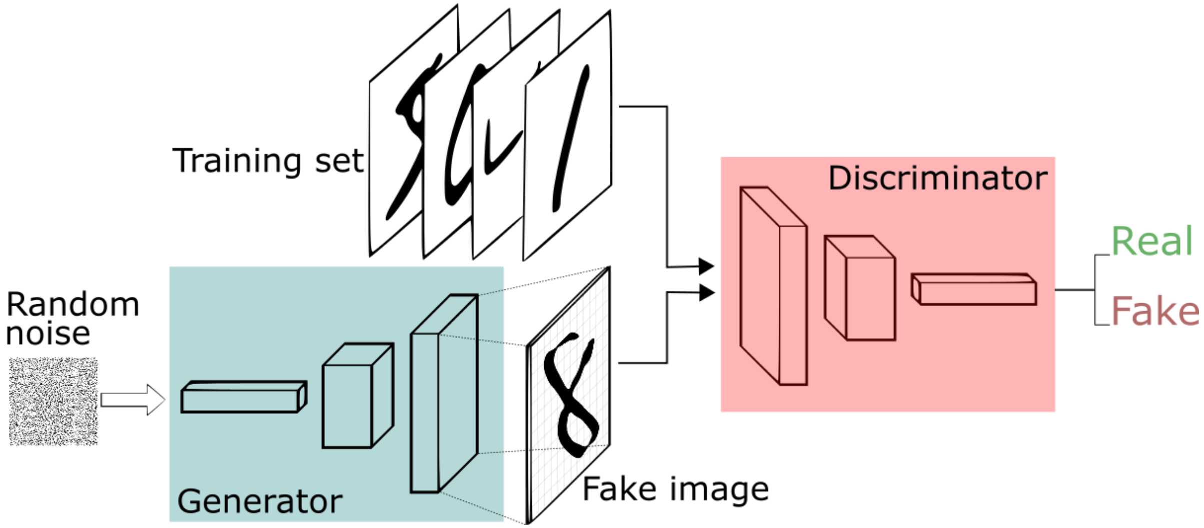


Source: Youssef Fenjiro

4.3 Heliogen's Superpowered Solar Array

4.4.1 Generative Adversarial Networks (GAN)

- Rather than limit ourselves to a single network for a model, we can use *multiple networks* in combination to achieve extraordinary results.
- A GAN is a model in which one network works to *produce* outputs, while another attempts to *classify* its outputs. Using this setup, we can have one learn how to “fool” the other, while the latter simultaneously begins to learn how to “catch” the fooler!



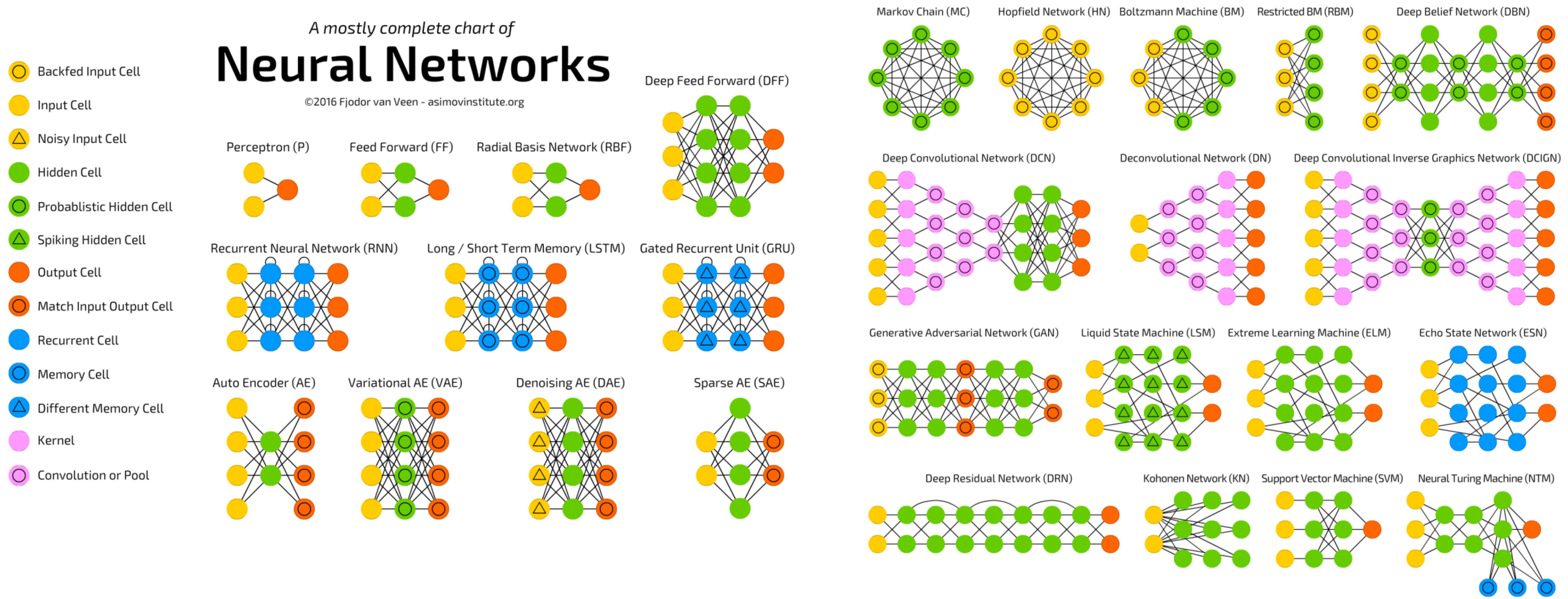
Source: [skymind.ai]()

4.4.2 GANs and Deepfakes

Figure 5: 1024×1024 images generated using the CELEBA-HQ dataset. See Appendix F for a larger set of results, and the accompanying video for latent space interpolations.

Source: [Karras et al. 2018]()

4.5 The Many Paths of Deep Learning



Source: [Karras et al. 2018]()

4.6 Final Thoughts

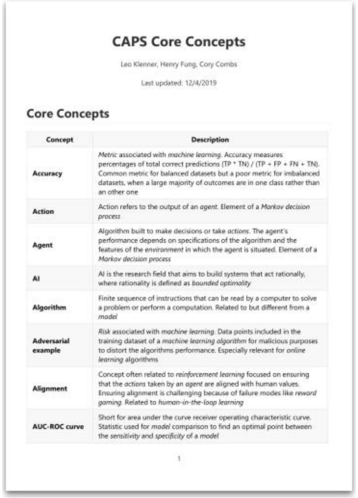
- **Powerful method:** Neural networks provide a powerful means to pursue *almost all* current forms of machine learning
- **Limitations:** NNs are not always a preferable means:
  - They require *vast* amounts of data and quickly become highly expensive
  - More importantly, they are *highly opaque*; even seeing what's inside does not often help us understand why a "choice" was made
- **Probabilistic:** NNs are purely statistical constructs, yielding *probabilistic outcomes* limited by known statistics: they will never be "correct given X circumstances", but rather "correct X out of Y times under Z circumstances"
- **Next frontiers:** Deep learning gave a new energy to the AI space in recent years – as have other areas of machine learning and intelligence research in the past. But NNs are not the final frontier of AI. Keep looking for what's next!

5. Moving Forward

Resources for your professional engagement with AI

5.1 Recommended Resources

Reference docs on our website



CAPS Core Concepts

Leo Klenner, Henry Fung, Cory Combs

Last updated: 12/4/2019

Core Concepts

Concept	Description
Accuracy	*Metric* associated with *machine learning*. Accuracy measures percentages of total correct predictions (TP * TN) / (TP + FP + FN + TN). Common metric for balanced datasets but a poor metric for imbalanced datasets, when a large majority of outcomes are in one class rather than an other one
Action	Action refers to the output of an *agent*. Element of a *Markov decision process*
Agent	Algorithm built to make decisions or take *actions*. The agent's performance depends on specifications of the algorithm and the features of the *environment* in which the agent is situated. Element of a *Markov decision process*
AI	AI is the research field that aims to build systems that act rationally, where rationality is defined as *bounded optimality*
Algorithm	Finite sequence of instructions that can be read by a computer to solve a problem or perform a computation. Related to but different from a *model*
Adversarial example	*Risk* associated with *machine learning*. Data points included in the training dataset of a *machine learning algorithm* for malicious purposes to distort the algorithms performance. Especially relevant for *online learning* algorithms
Alignment	Concept often related to *reinforcement learning* focused on ensuring that the *actions* taken by an *agent* are aligned with human values. Ensuring alignment is challenging because of failure modes like *reward gaming*. Related to *human-in-the-loop learning*
AUC-ROC curve	Short for area under the curve receiver operating characteristic curve. Statistic used for *model* comparison to find an optimal point between the *sensitivity* and *specificity* of a *model*

1

...

90+ definitions

Social

Twitter

https://twitter.com/jackclarksf

https://twitter.com/Miles_Brundage

https://twitter.com/chipro

DeepMind, OpenAI, Google Brain, Oxford FHI

Blogs, reports, papers

https://medium.com/@deepmindsafetyresearch

https://arxiv.org/ftp/arxiv/papers/1802/1802.07228.pdf

Medium, ArXiv in general

Newsletters

https://jack-clark.net/

Other courses

Reinforcement Learning for Policymakers

Our second course, on YouTube end of Jan

Various courses on Coursera, Edx...