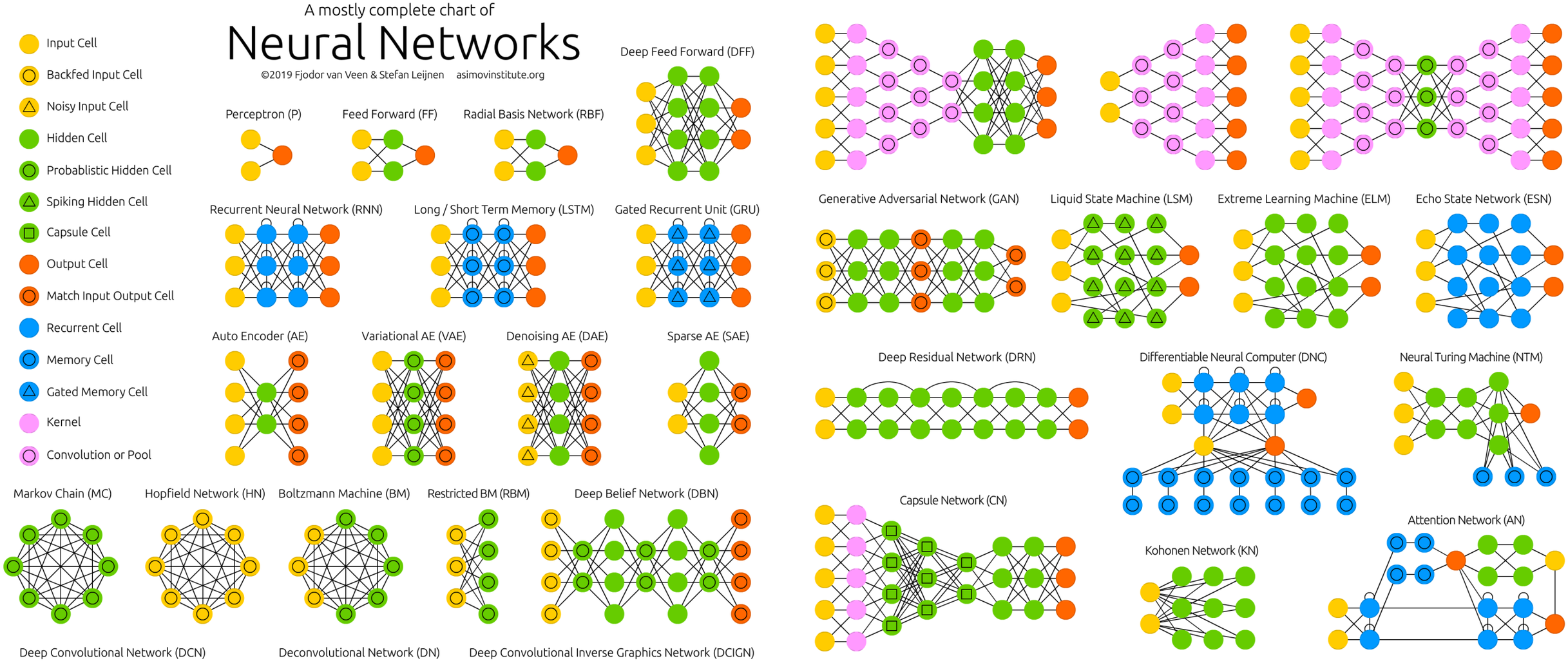
An overview of Network Architectures and Innovations

Generative Adversarial Networks

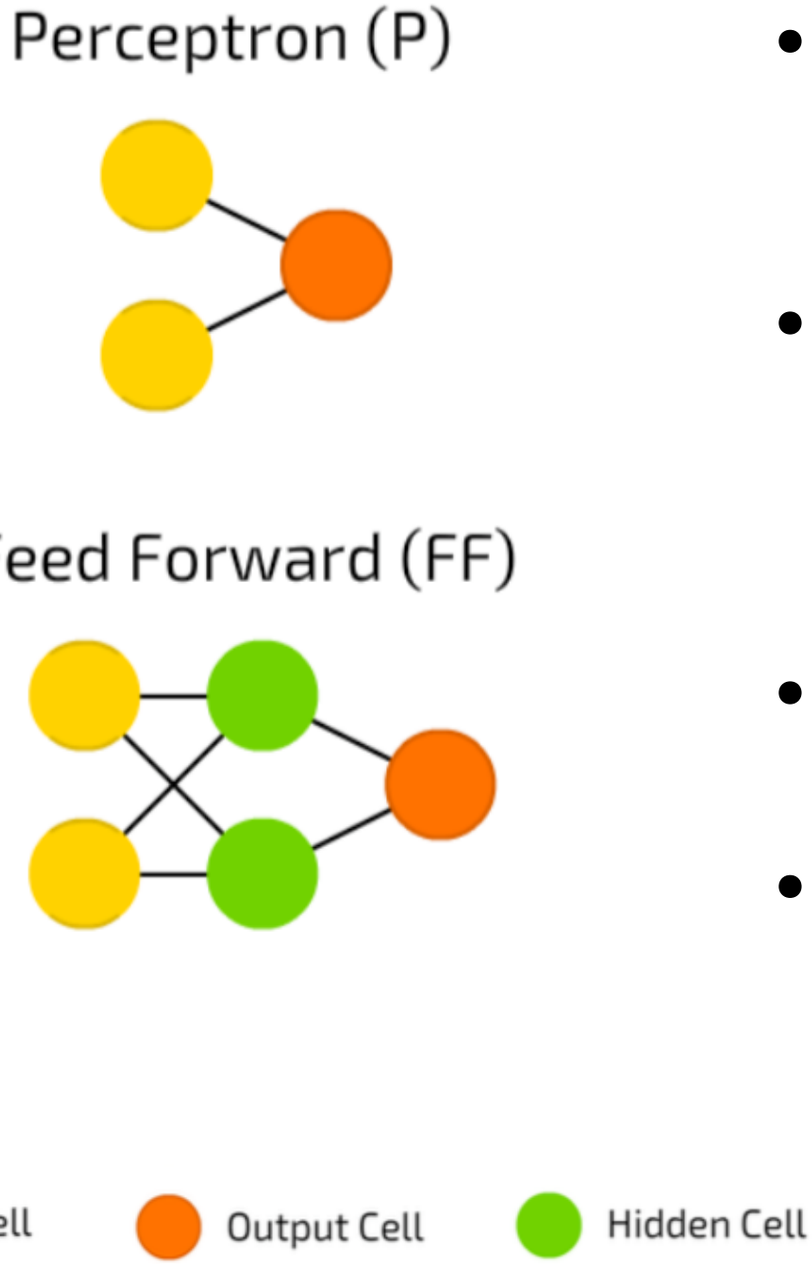
Francis Steen and Xinyu You
Red Hen Lab
August 2019

The Neural Network Zoo



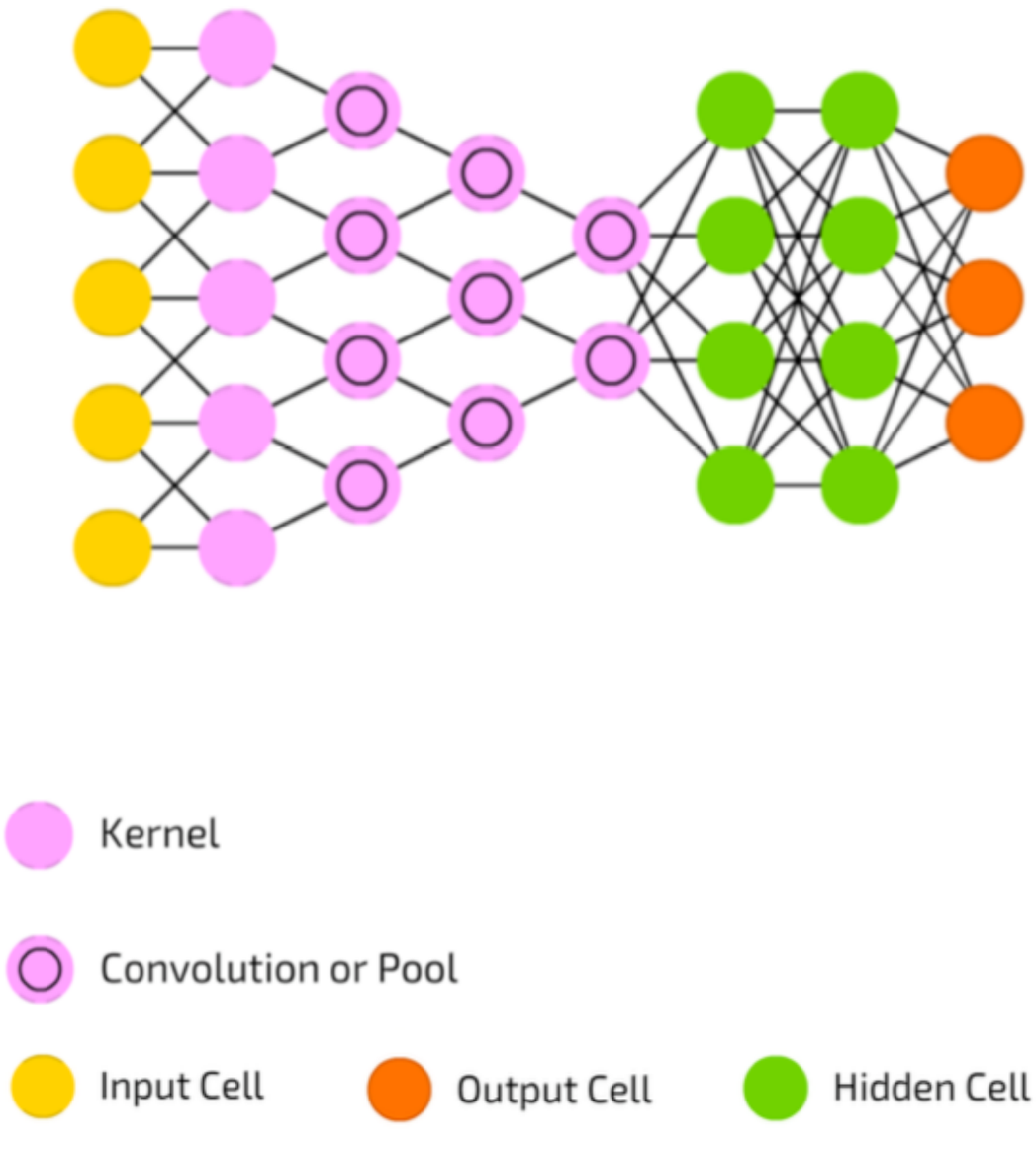
http://www.asimovinstitute.org/wp-content/uploads/2019/04/NeuralNetworkZo19High.png

Perceptrons and Feed forward neural networks



- Feed information from the front to the back
- Two adjacent layers are fully connected (every neuron form one layer to every neuron to another layer)
- Trains FFNNs through back-propagation
- The error being back-propagated is often some variation of the difference between the input and the output (like MSE or just the linear difference)

Convolutional neural networks (CNNs)



- CNNs are quite different from most other networks.
- It primarily used for image processing but can also be used for other types of input such as as audio.
- A typical use case is where you feed the network images and the network classifies the data, e.g. it outputs “cat” if you give it a cat picture and “dog” when you give it a dog picture.
- CNNs often glue an FFNN to the end to further process the data, which allows for highly non-linear abstractions.

What is the problem with CNNs?

- If the images have rotation, tilt or any other different orientation then CNNs have poor performance.

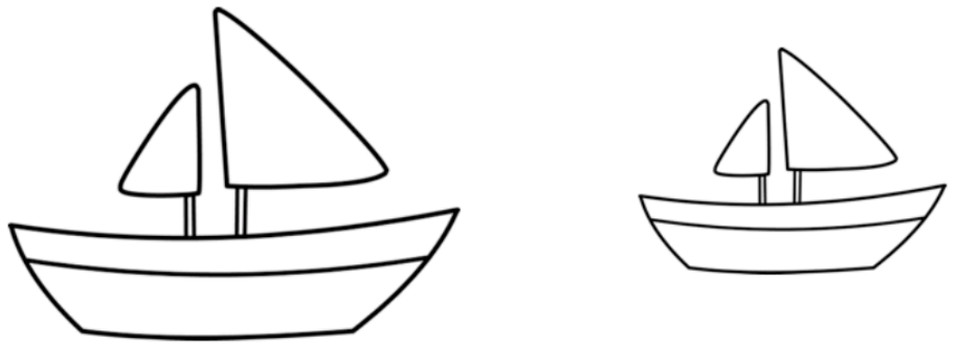
- This problem was solved by adding different variations of the same image during training.

What is the problem with CNNs?

Exaple: classify ships and horses.

- The 1st layer understands the small curves and edges.
- The 2nd layer might understand the straight lines or the smaller shapes, like the mast of a ship or the curvature of the entire tail. Higher up layers start understanding more complex shapes like the entire tail or the ship hull.
- Final layers try to see a more holistic picture like the entire ship or the entire horse.

What is the problem with CNNs?



- We use pooling after each layer to make it compute in reasonable time frames. But in essence it also loses out the positional data.

- Pooling helps in creating the positional invariance.

- This invariance also leads to triggering false positive for images which have the components of a ship but not in the correct order.

- So the system can trigger the right to match with the left in the above image. You as an observer clearly see the difference. The pooling layer also adds this sort of invariance.

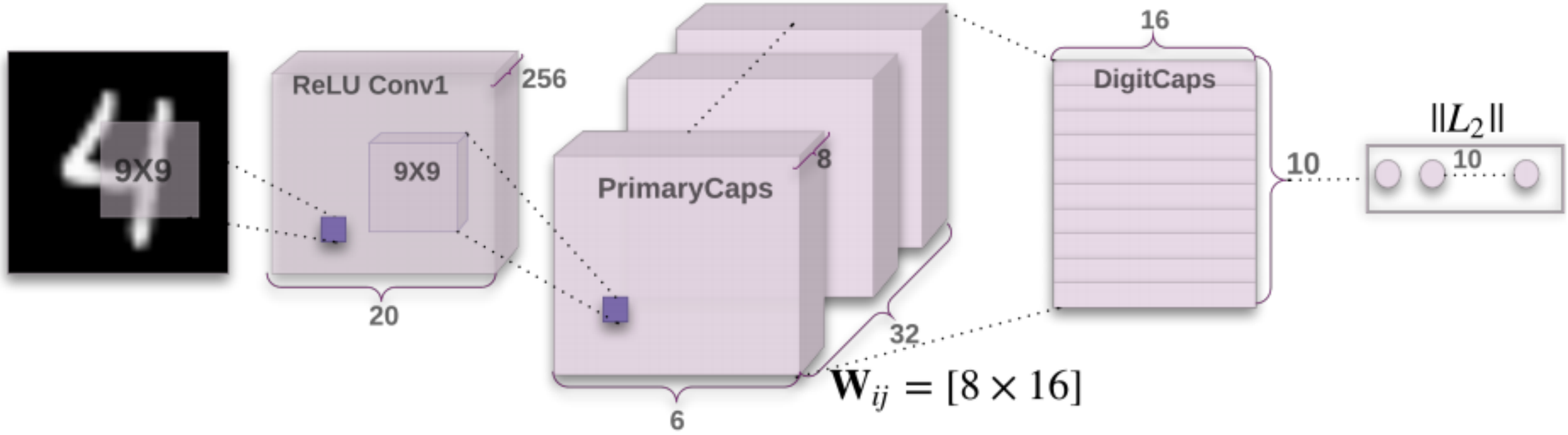
What is the problem with CNNs?

- This was never the intention of pooling layer. What the pooling was supposed to do is to introduce positional, orientational, proportional invariances.

- But the method we use to get this uses is very crude. In reality it adds all sorts of positional invariance. Thus leading to the dilemma of detecting right ship in image 2.0 as a correct ship.

- What we needed was not invariance but equivariance. **Invariance** makes a CNN tolerant to small changes in the viewpoint. **Equivariance** makes a CNN understand the rotation or proportion change and adapt itself accordingly so that the spatial positioning inside an image is not lost. A ship will still be a smaller ship but the CNN will reduce its size to detect that.

- This leads us to the recent advancement of Capsule Networks.



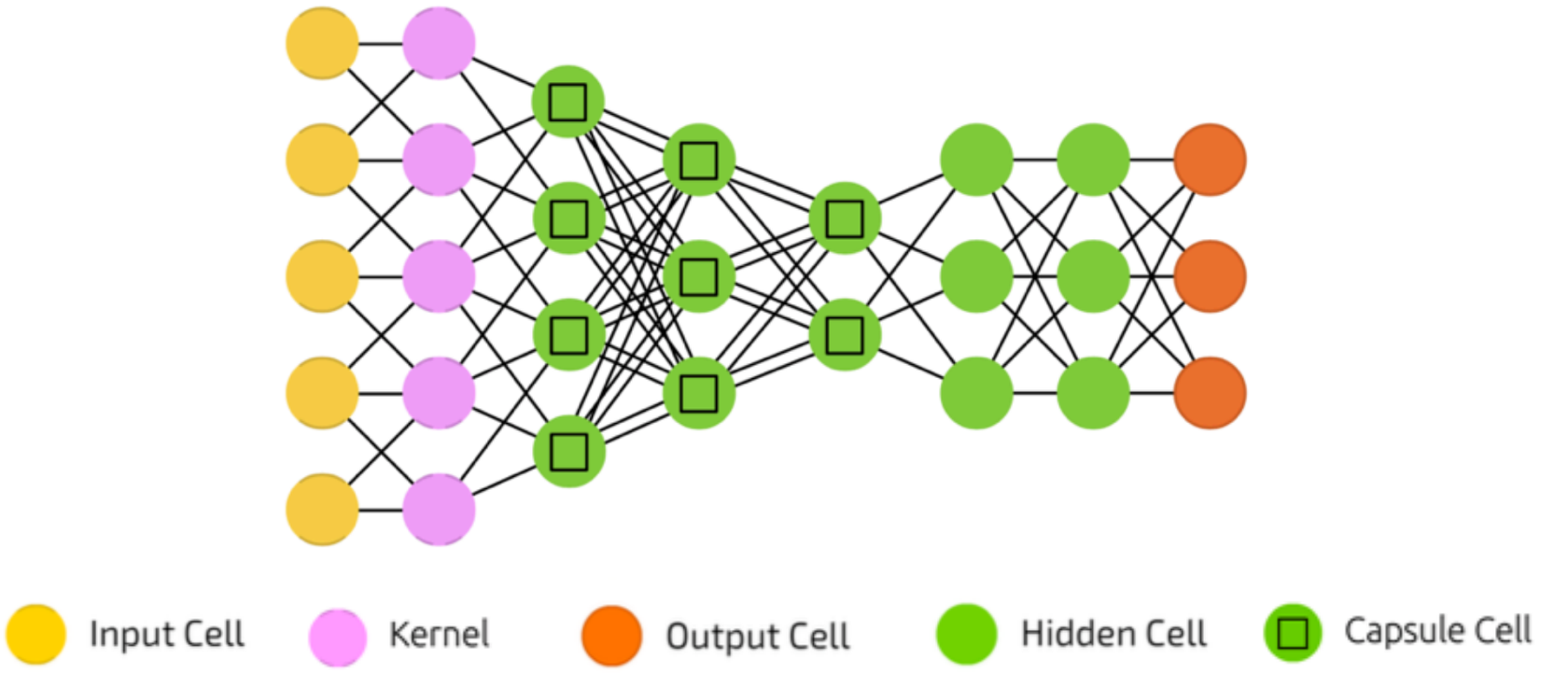
- In CapsNet you would add more layers inside a single layer
    - The state of the neurons inside a capsule capture the above properties of one entity inside an image.

- The authors think that human brain have modules called “**capsules**”.

- These capsules are particularly good at handling different types of visual stimulus and encoding things like pose (position, size, orientation), deformation, velocity, albedo, hue, texture etc.

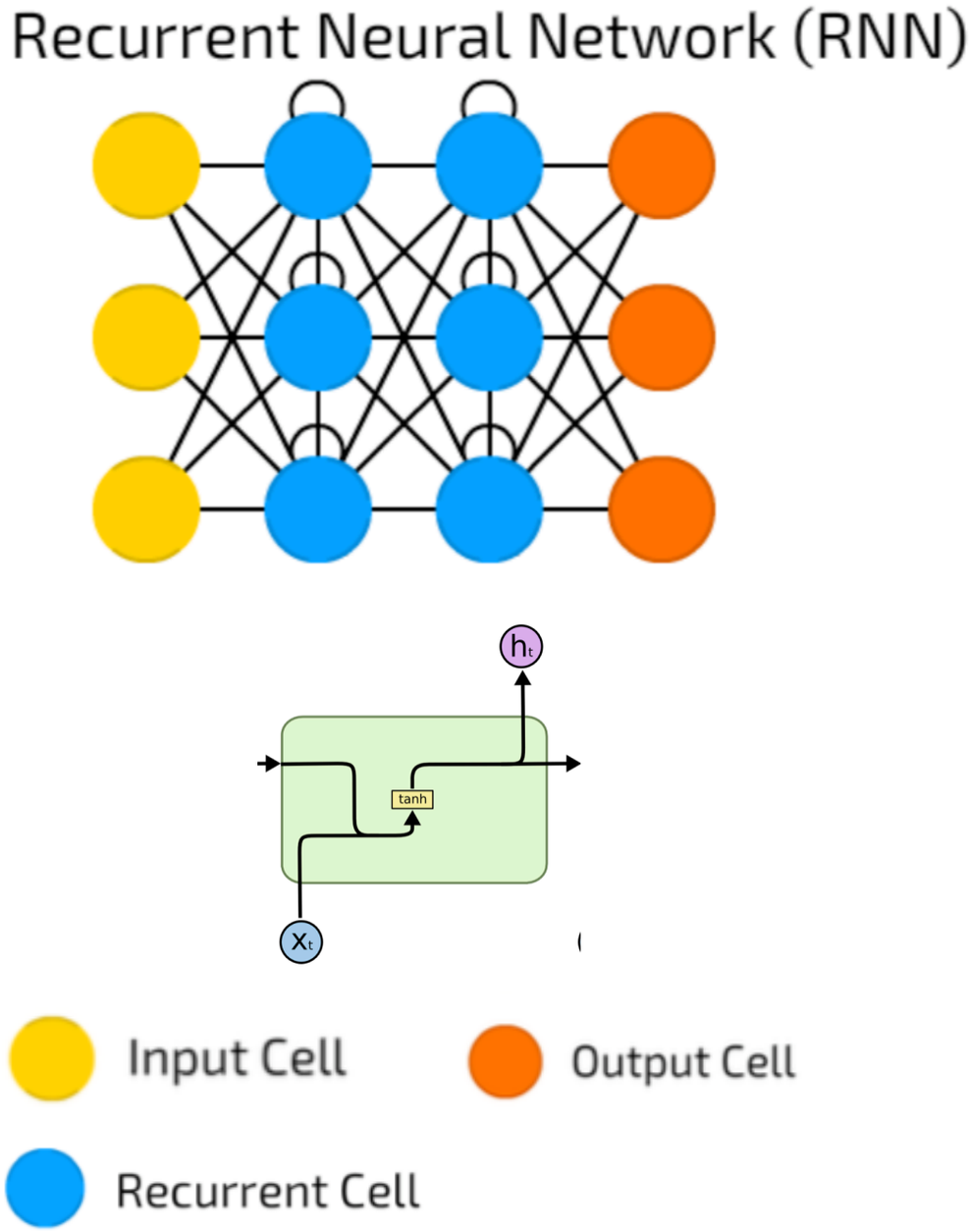
- The brain must have a mechanism for “routing” low level visual information to what it believes is the best capsule for handling it.

Capsule Networks (CapsNets)



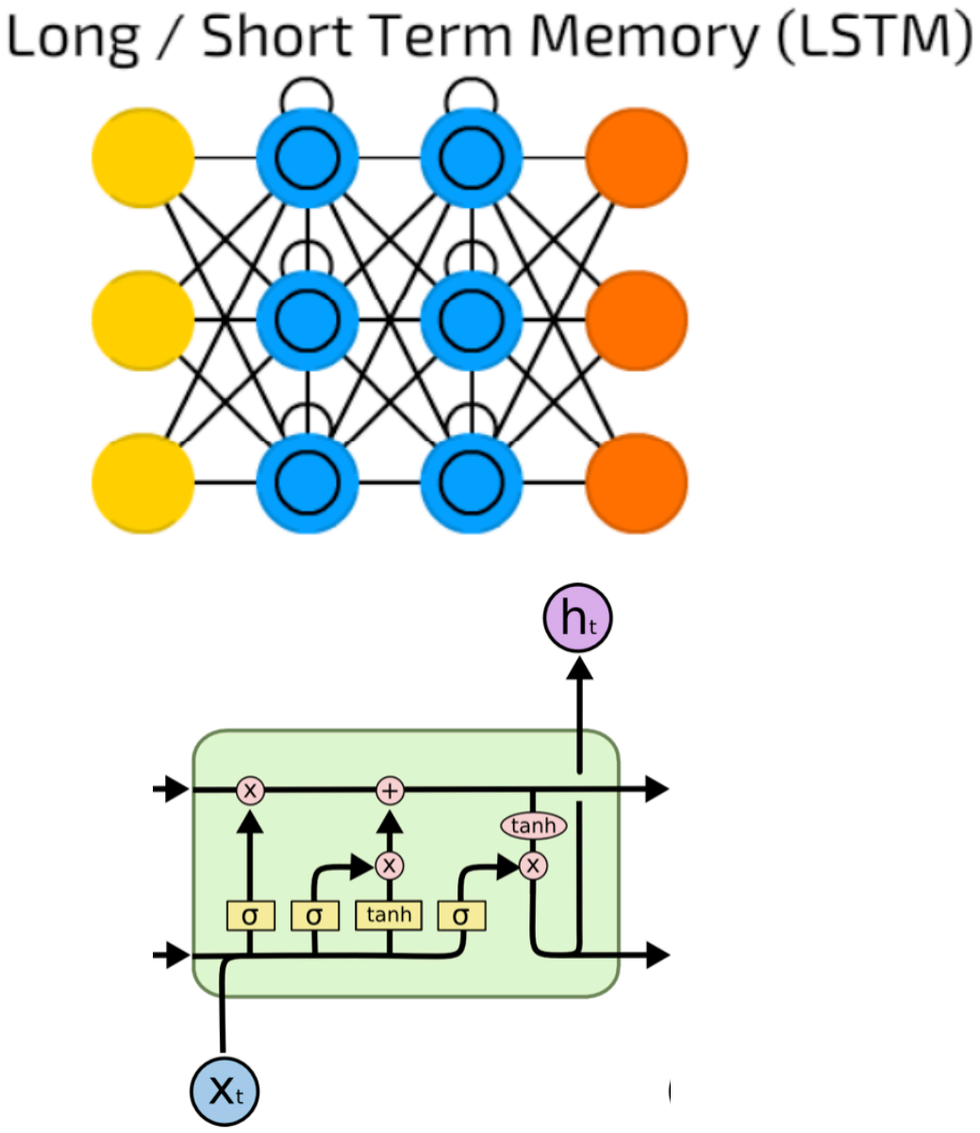
- DRNs are very deep FFNNs with extra connections passing input from one layer to a later layer (often 2 to 5 layers) as well as the next layer.
- It adds an identity to the solution, carrying the older input over and serving it freshly to a later layer
- It has been shown that these networks are very effective at learning patterns up to 150 layers deep, much more than the regular 2 to 5 layers one could expect to train.

Recurrent Neural Networks



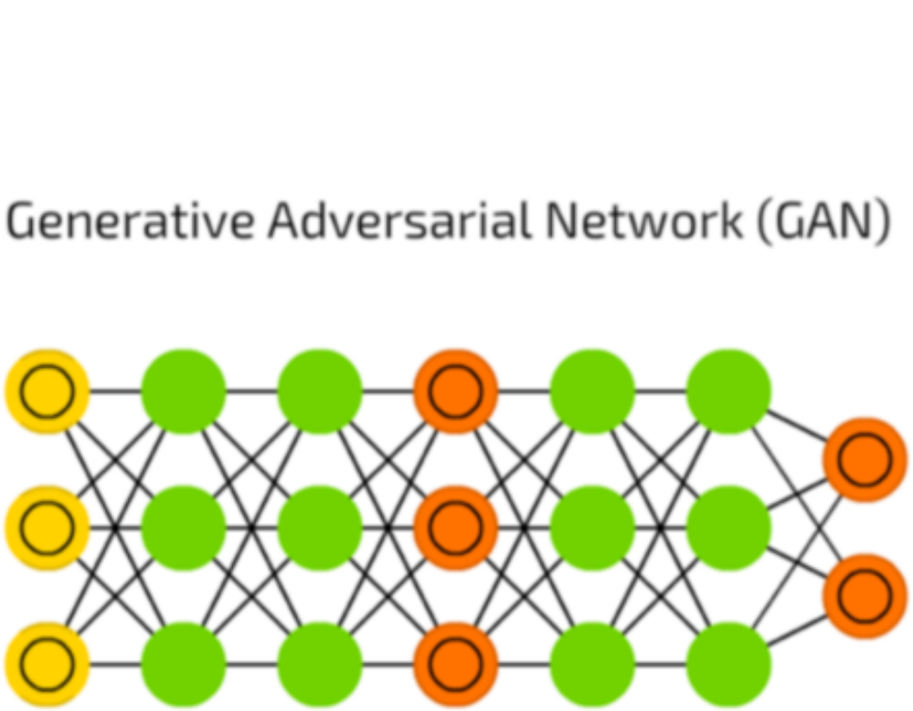
- RNNs have connections between passes, connections through time.
- One big problem with RNNs is the vanishing (or exploding) gradient problem where, depending on the activation functions used, information rapidly gets lost over time, just like very deep FFNNs lose information in depth.
- It can in principle be used in many fields as most forms of data that don't actually have a timeline (i.e. unlike sound or video) can be represented as a sequence.

Long / short term memory (LSTM) Networks



- LSTM networks try to combat the vanishing / exploding gradient problem by introducing gates and an explicitly defined memory cell.
- Each neuron has a memory cell and three gates: input, output and forget.
- Be able to learn complex sequences, such as writing like Shakespeare or composing primitive music.

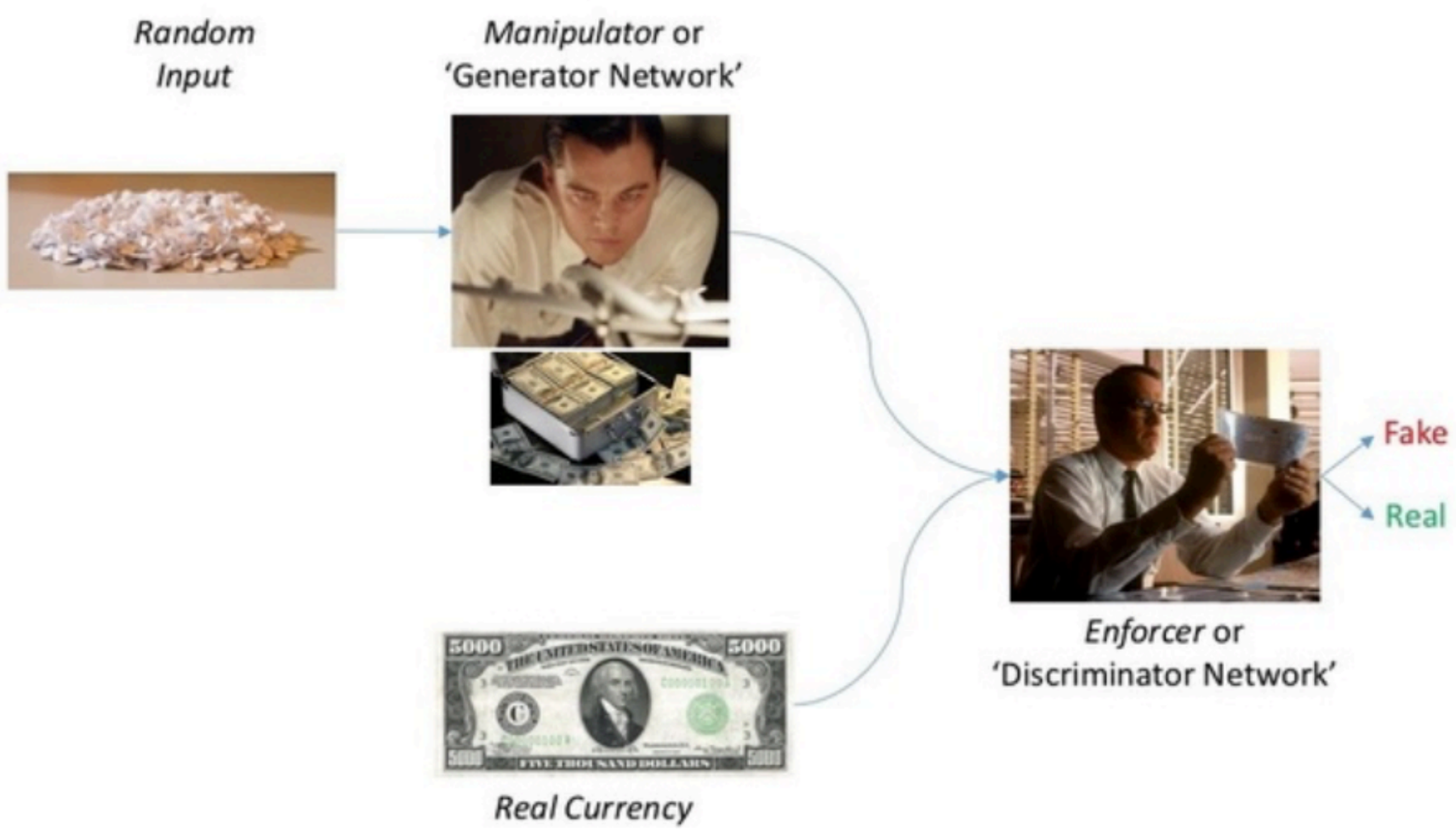
Generative adversarial networks (GANs)



- GNNs are from a different breed of networks, they are twins: two networks working together.
- It consist of any two networks (although often a combination of FFs and CNNs), with one tasked to generate content and the other has to judge content.
- The discriminating network receives either training data or generated content from the generative network.
- How well the discriminating network was able to correctly predict the data source is then used as part of the error for the generating network.
- This creates a form of competition where the discriminator is getting better at distinguishing real data from generated data and the generator is learning to become less predictable to the discriminator.

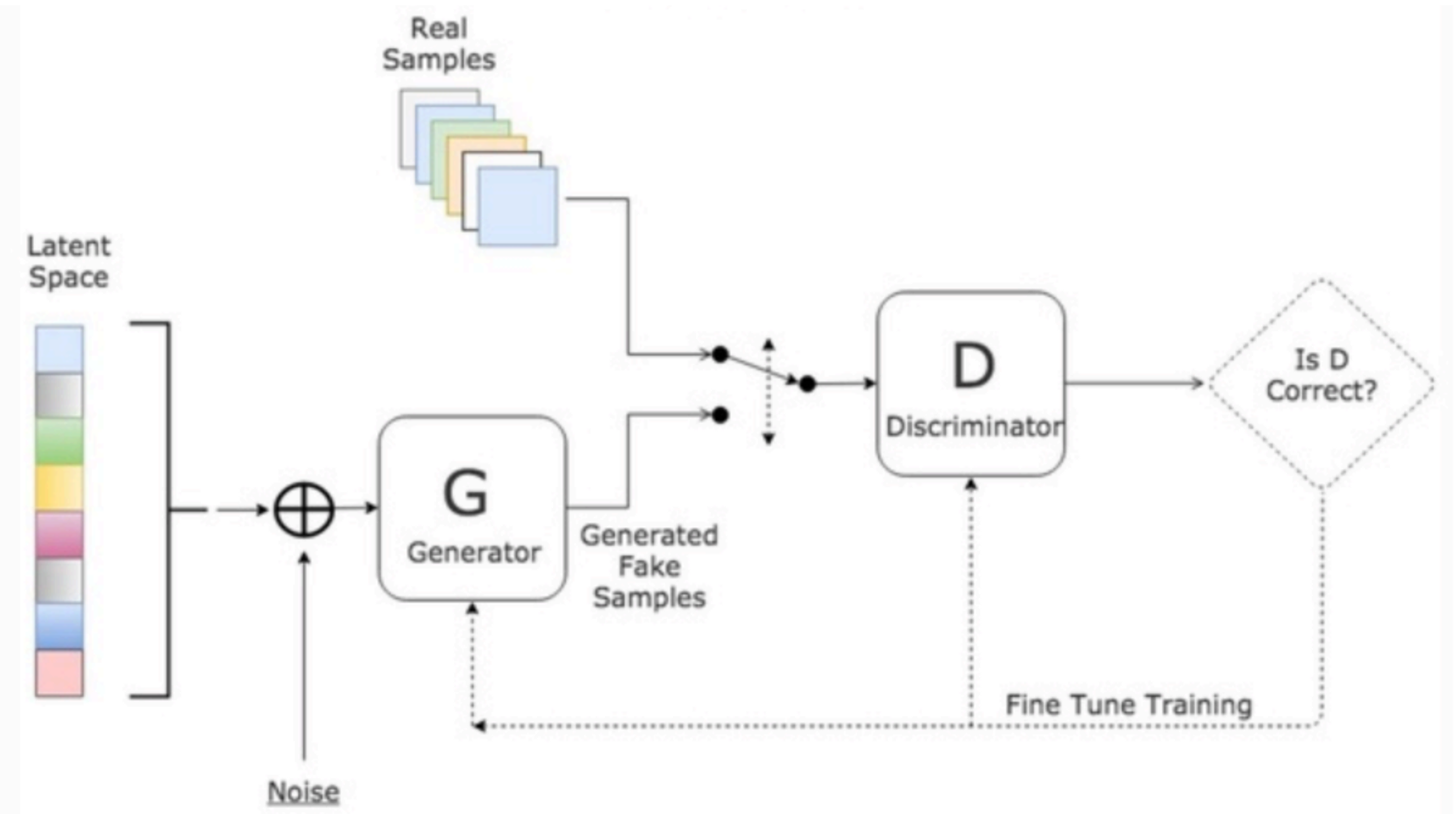
GANs

The generative model can be thought of as analogous to a team of counterfeiters, trying to produce fake currency and use it without detection, while the discriminative model is analogous to the police, trying to detect the counterfeit currency.

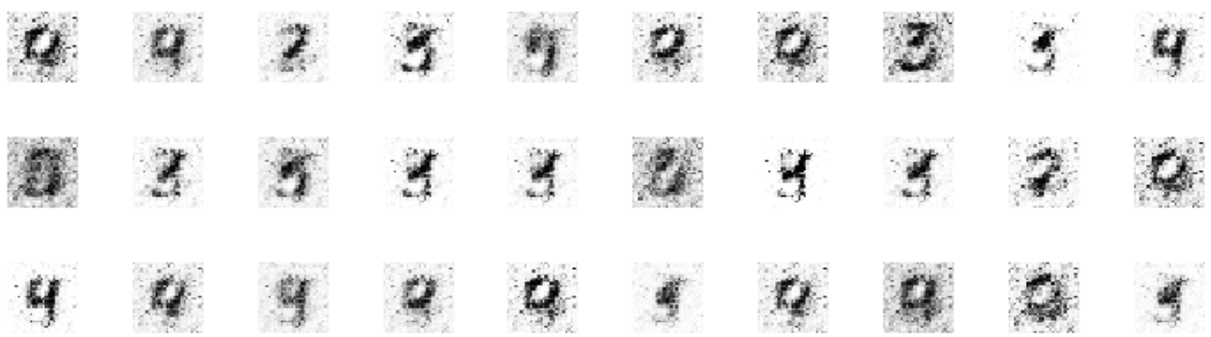


GANs

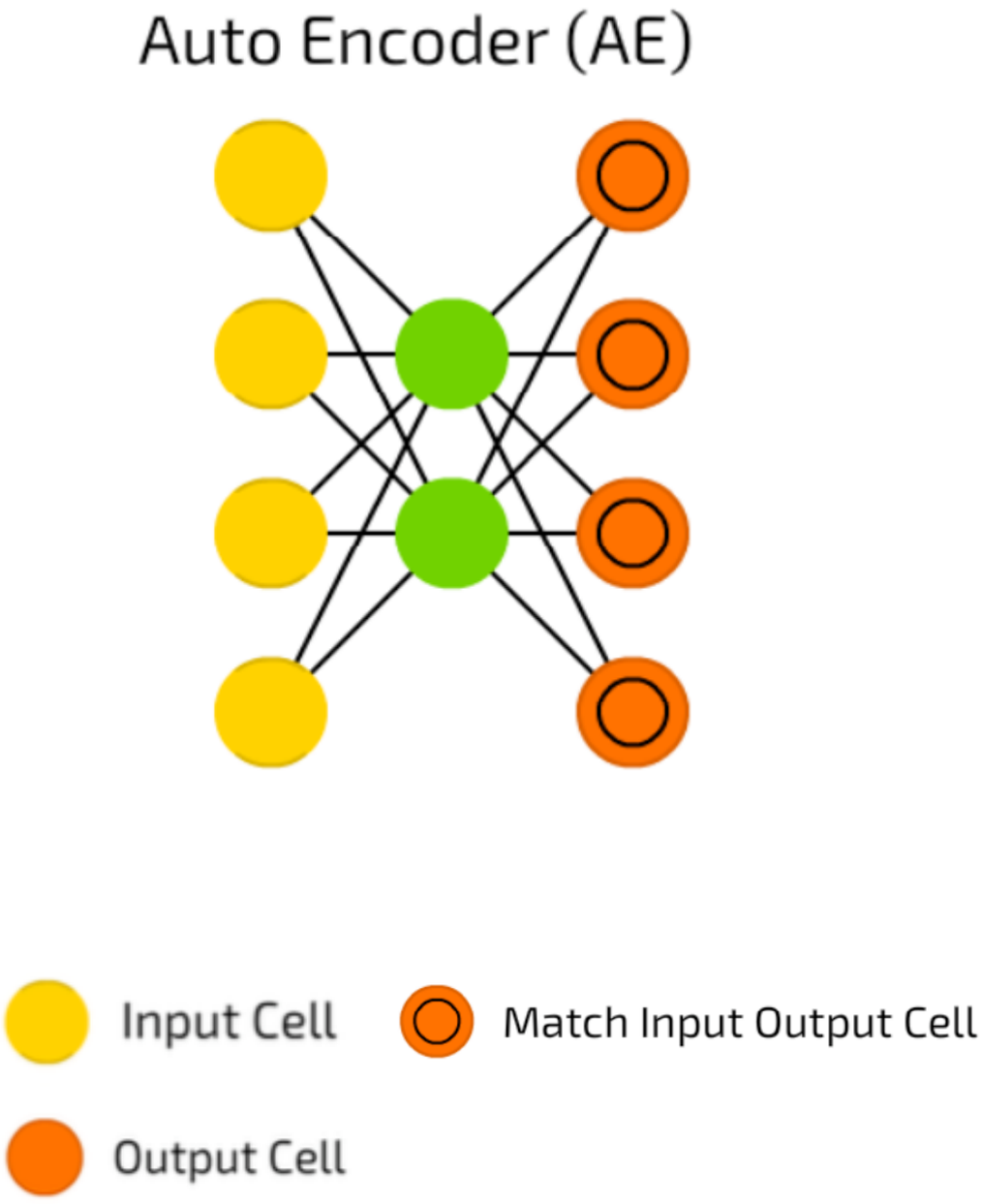
- **Generator** (counterfeiter)
- **Discriminator** (cashier)



A GAN that can generate numbers from arbitrary noise.

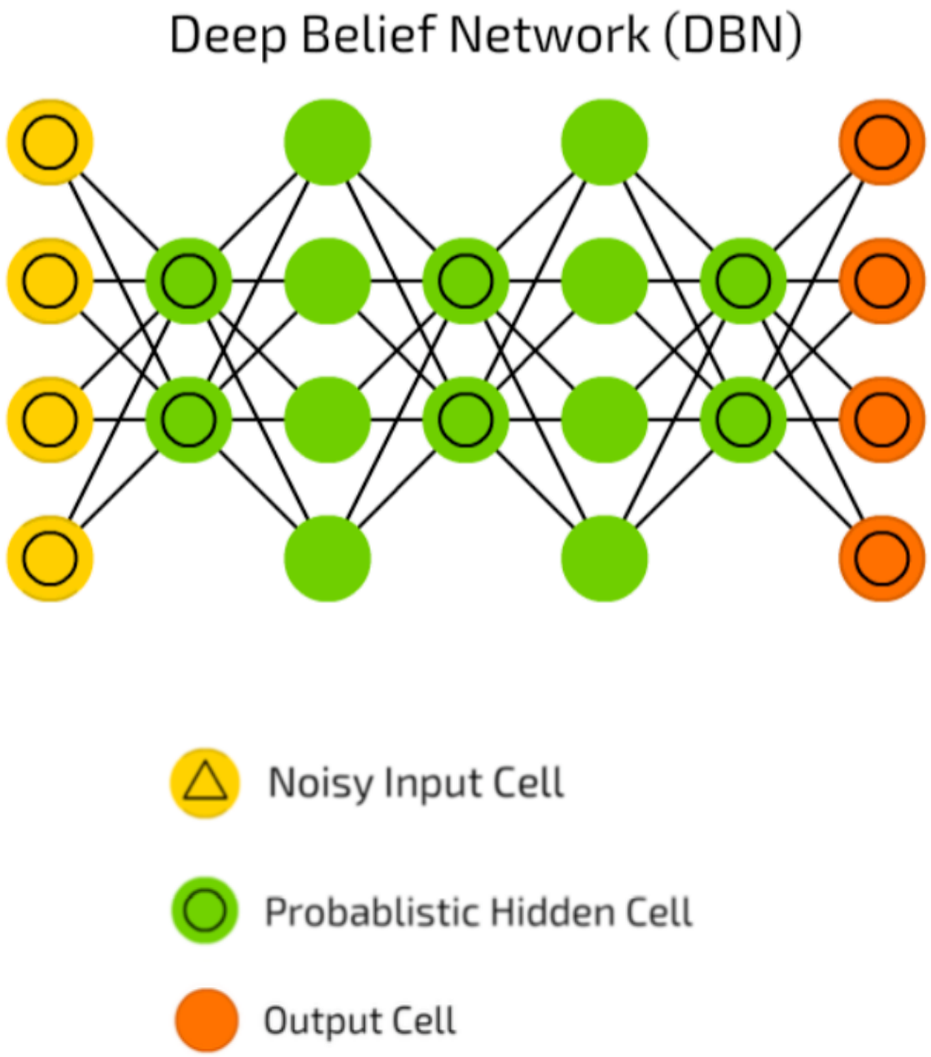


Autoencoders (AE)



- A different use of FFNNs than a fundamentally different architecture
- The basic idea behind autoencoders is to encode information (as in compress, not encrypt) automatically.
- The entire network always resembles an hourglass like shape.
- Everything up to the middle is called the encoding part, everything after the middle the decoding and the middle (surprise) the code.

Deep belief networks (DBNs)

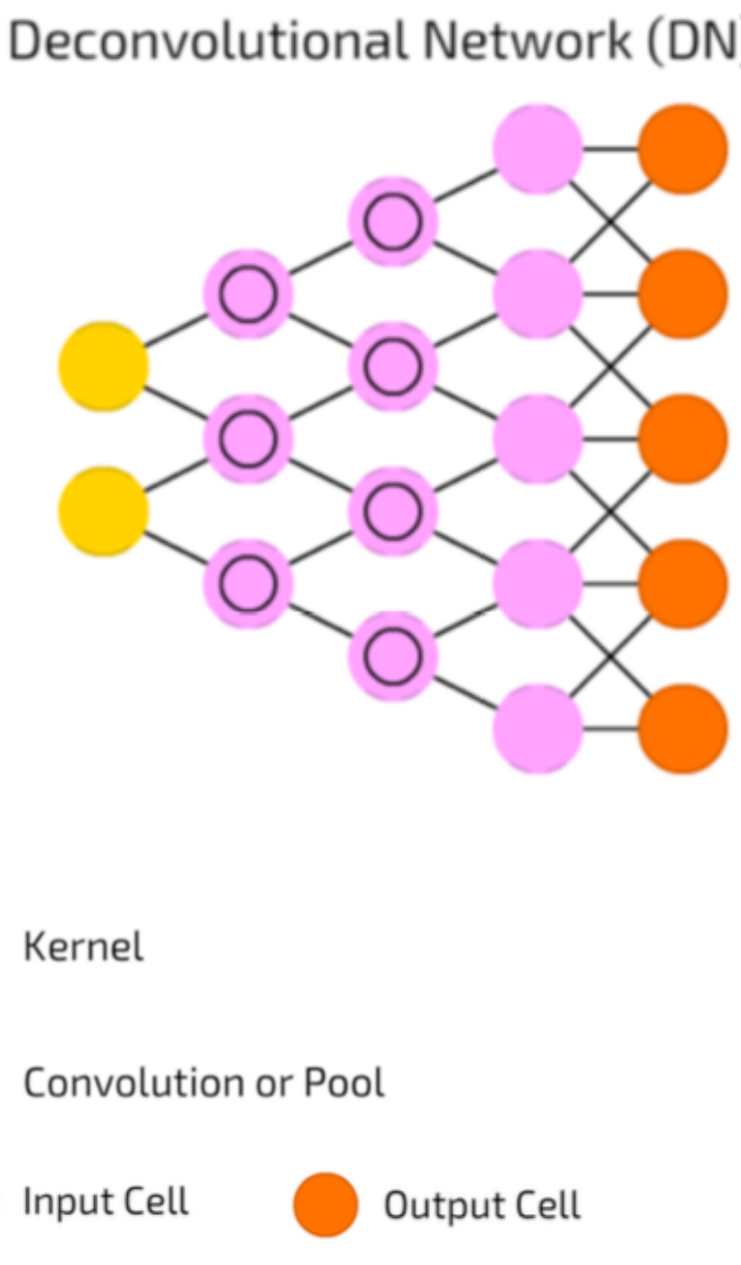


- These networks have been shown to be effectively trainable stack by stack which also known as greedy training

- Greedy means making locally optimal solutions to get to a decent but possibly not optimal answer

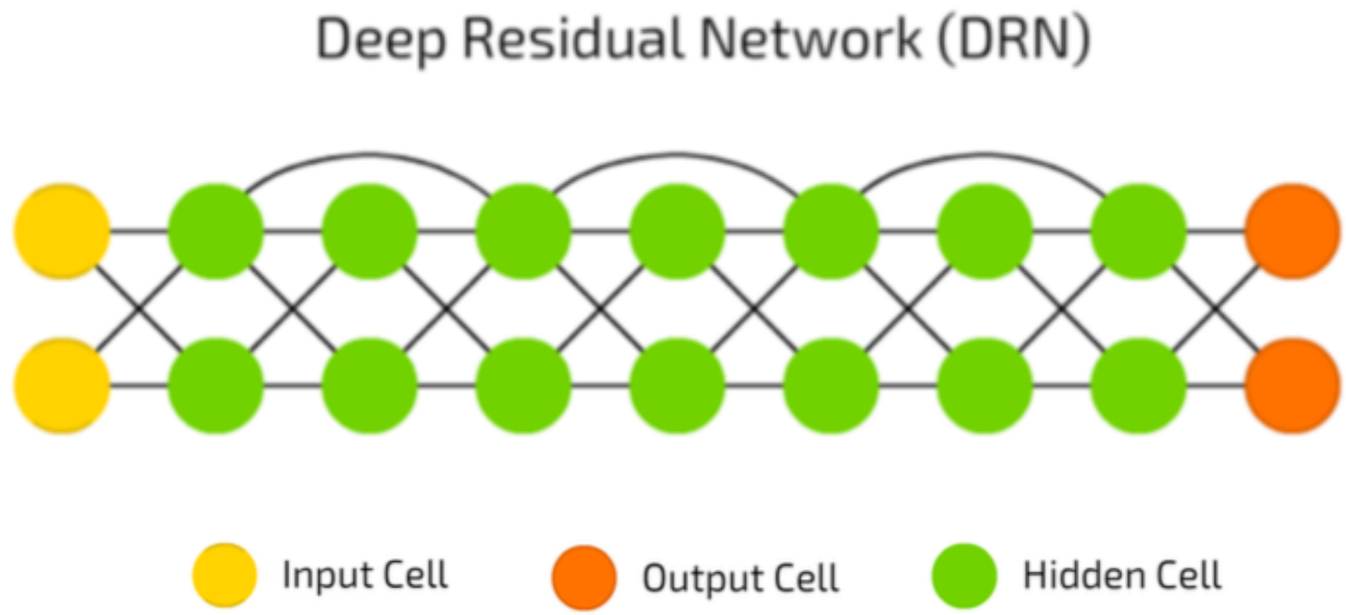
- DBNs can be trained through contrastive divergence or back-propagation and learn to represent the data as a probabilistic model, just like regular RBMs or VAEs.

Deconvolutional networks (DNs)



- DNs are reversed convolutional neural networks.
- Imagine feeding a network the word “cat” and training it to produce cat-like pictures, by comparing what it generates to real pictures of cats.
- A typical use case is where you feed the network images and the network classifies the data, e.g. it outputs “cat” if you give it a cat picture and “dog” when you give it a dog picture.
- Most applications feed a binary classification input vector. Think <0, 1> being cat, <1, 0> being dog and <1, 1> being cat and dog.
- The pooling layers commonly found in CNNs are often replaced with similar inverse operations

Deep residual networks (DRNs)



- DRNs are very deep FFNNs with extra connections passing input from one layer to a later layer (often 2 to 5 layers) as well as the next layer.
- It adds an identity to the solution, carrying the older input over and serving it freshly to a later layer
- It has been shown that these networks are very effective at learning patterns up to 150 layers deep, much more than the regular 2 to 5 layers one could expect to train.