Lecture 26:
Deep Learning
and other cool stuff

Artificial Intelligence
CS-GY-6613-I

Julian Togelius
julian.togelius@nyu.edu

Sequence learning

- Non-sequential machine learning takes input from a single point in time, and outputs a prediction or classification:

  $y=f(x(t))$

- Now we look at sequential inputs where the output y can depend on more than just the immediate input:

  $y=f(s(t))=F(x(t),x(t-1),\ldots,x(1))$

Sequence learning

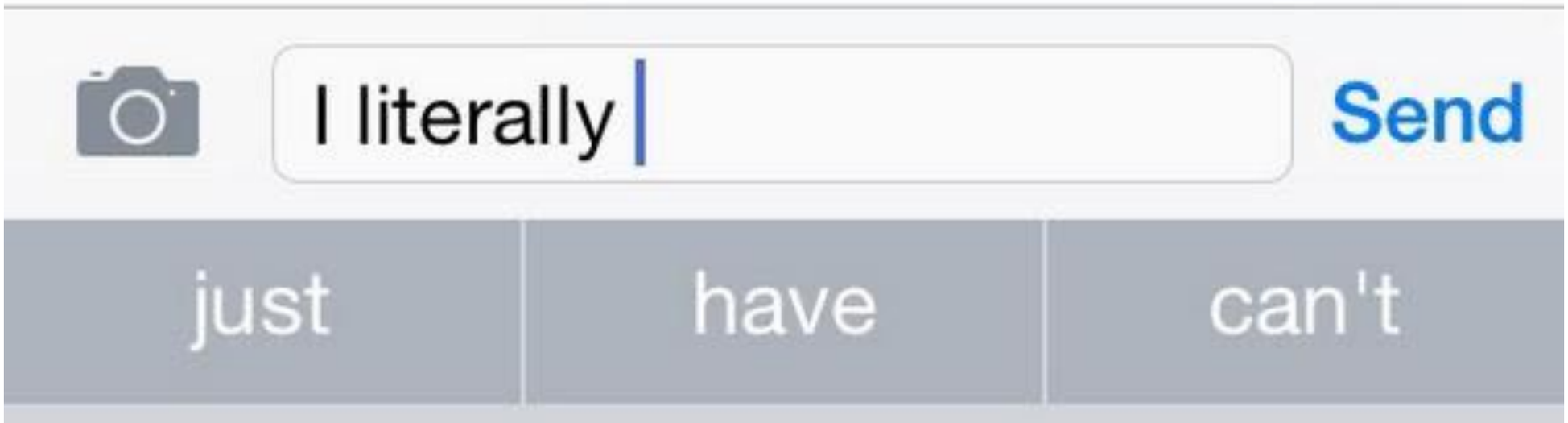
- Many natural processes are inherently sequential
  - Speech
  - Vision
  - Natural language
  - DNA
- In robotics tasks short-term memory can be essential for determining the state of the world due to limited sensor information

Speech recognition

Text classification

the dog is on the table

Text prediction



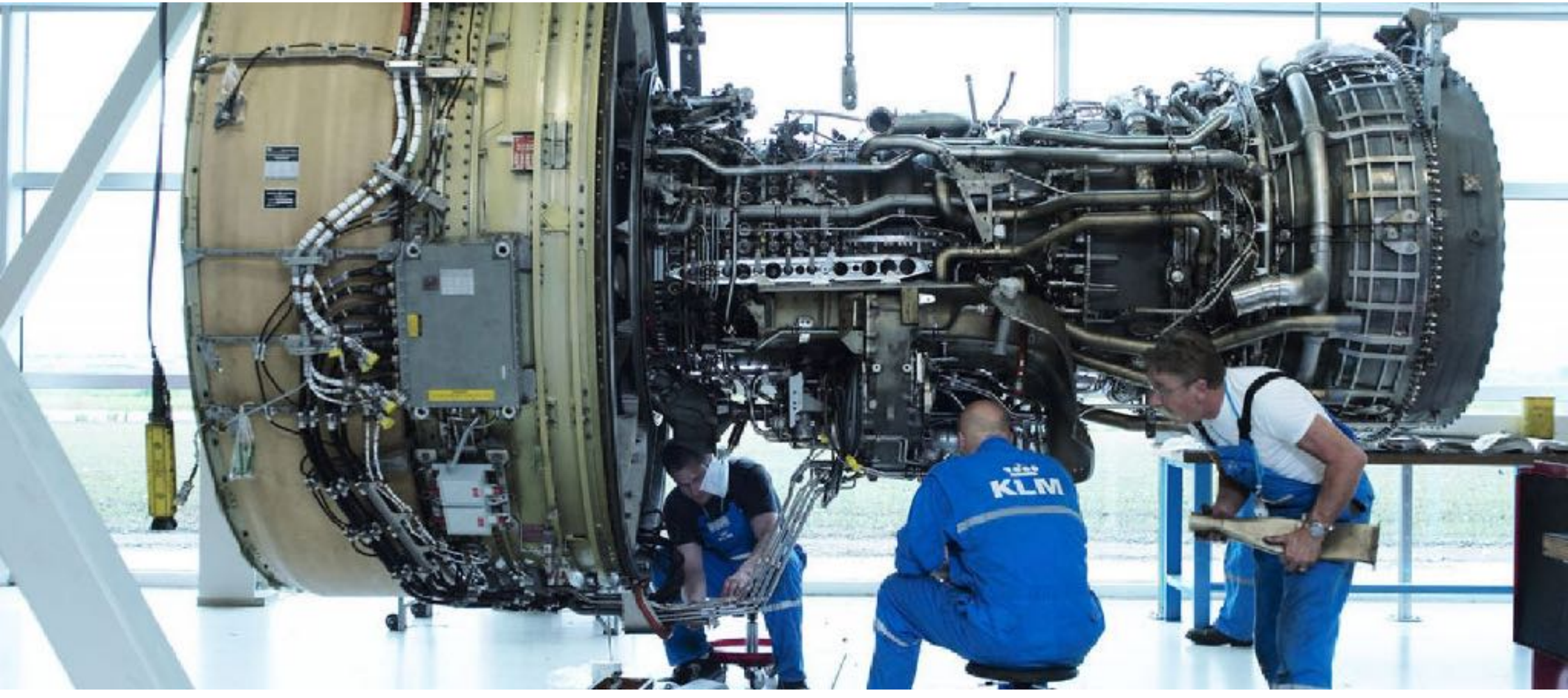
Text prediction

THE HANDSOME ONE

The castle grounds snarled with a wave of magically magnified wind. The sky outside was a great black ceiling, which was full of blood. The only sounds drifting from Hagrid's hut were the disdainful shrieks of his own furniture. Magic: it was something that Harry Potter thought was very good.

Leathery sheets of rain lashed at Harry's ghost as he walked across the grounds toward the castle. Ron was standing there and doing a kind of frenzied tap dance. He saw Harry and immediately began to eat Hermione's family.

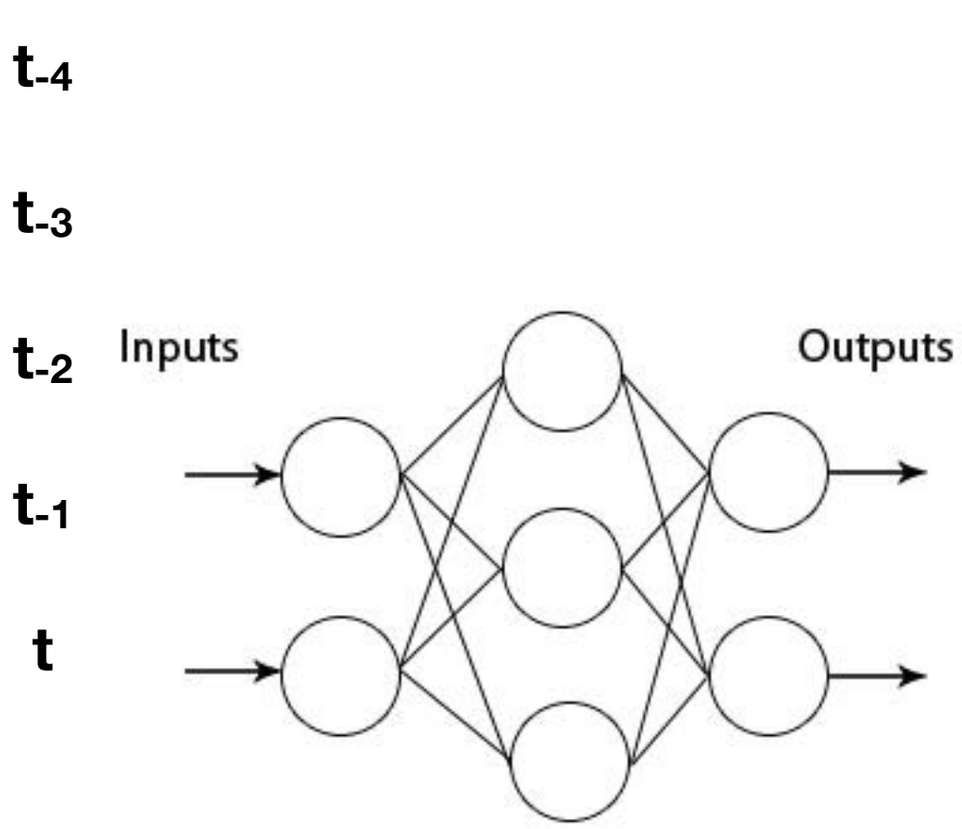
Fault prediction



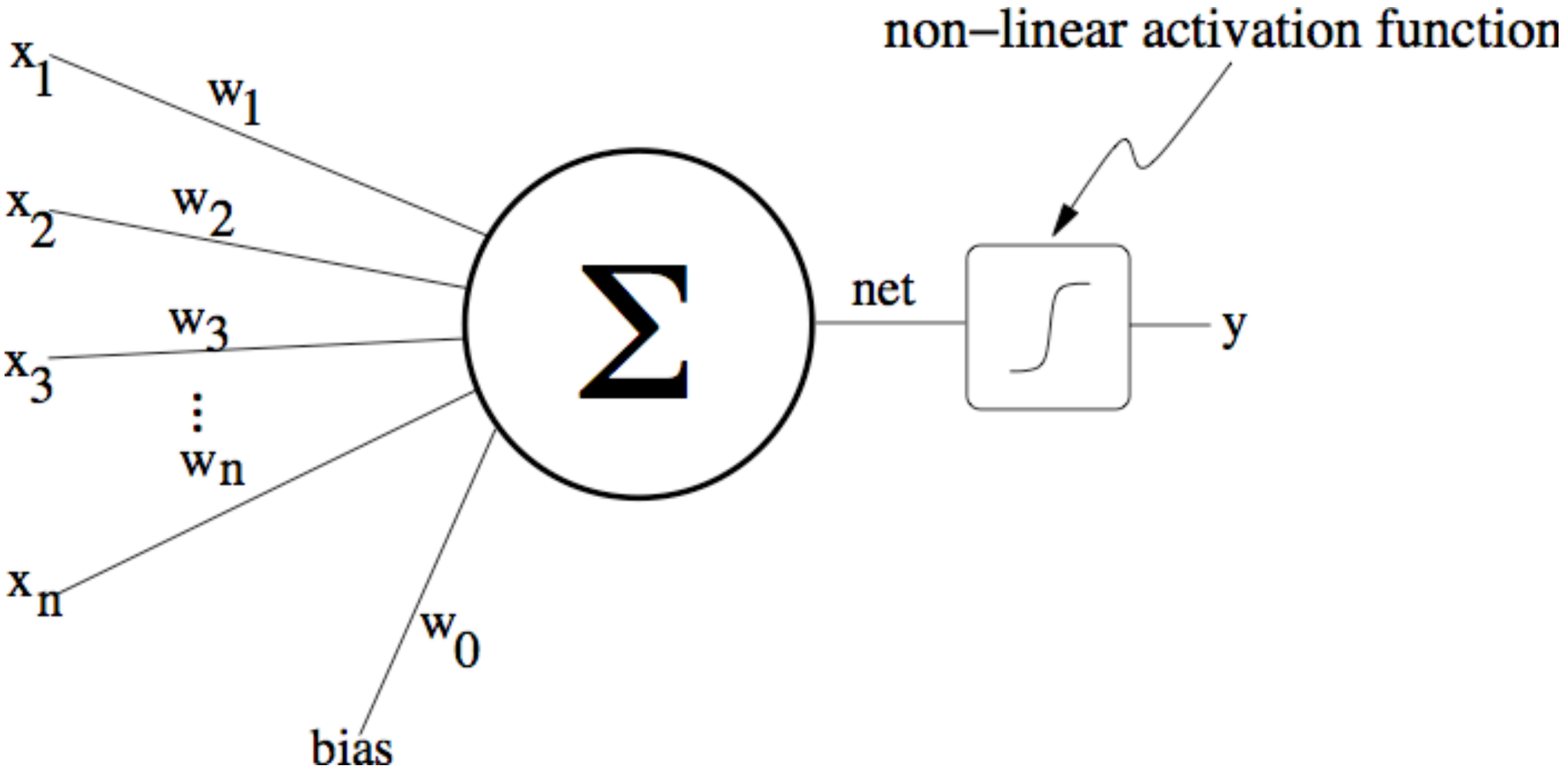
Time-window approach

- The “naïve” way of doing time-series prediction/ classification with a neural network

- Feed the the last *n* time-steps into the *n* network inputs



Feedforward neuron

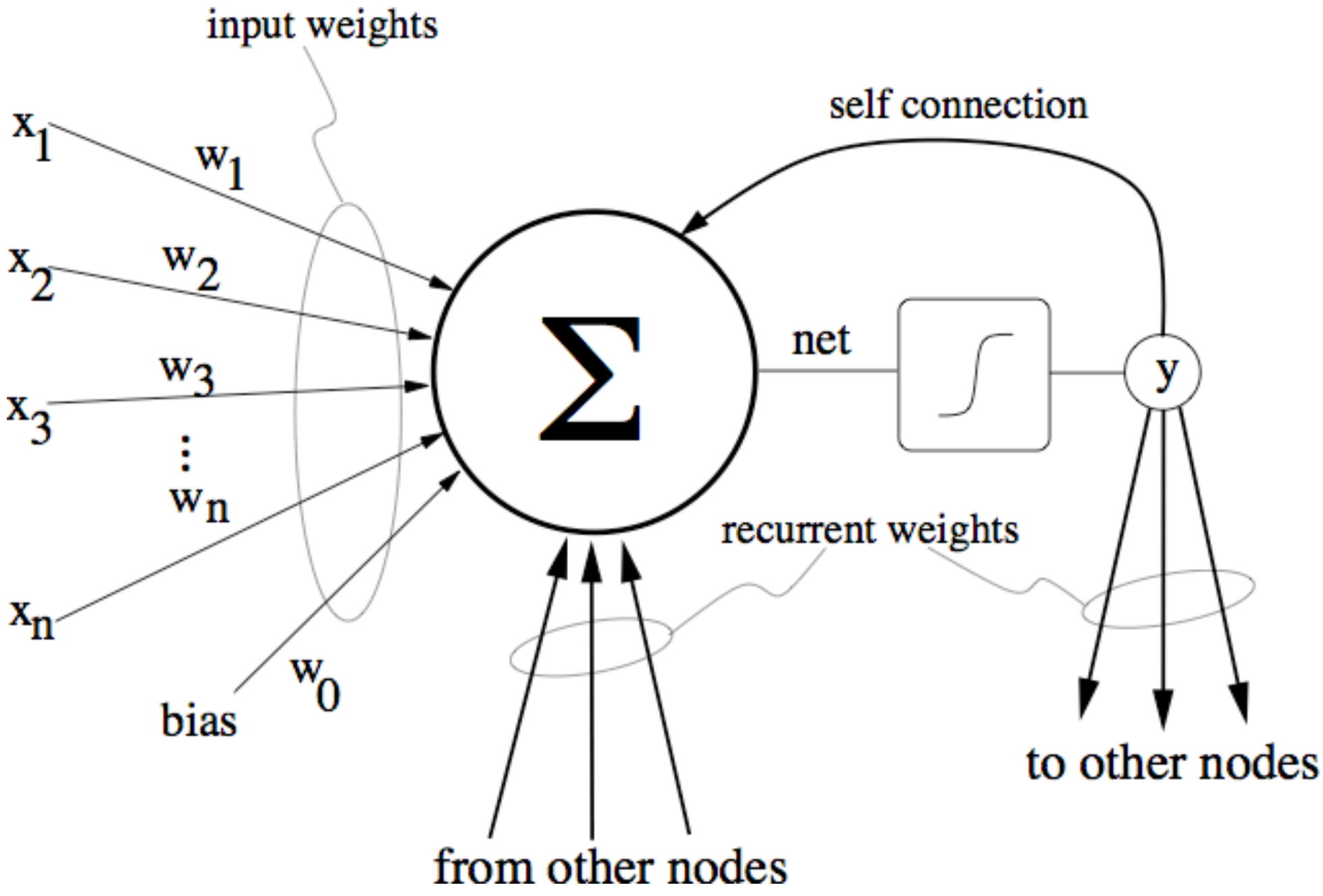


Recurrent networks

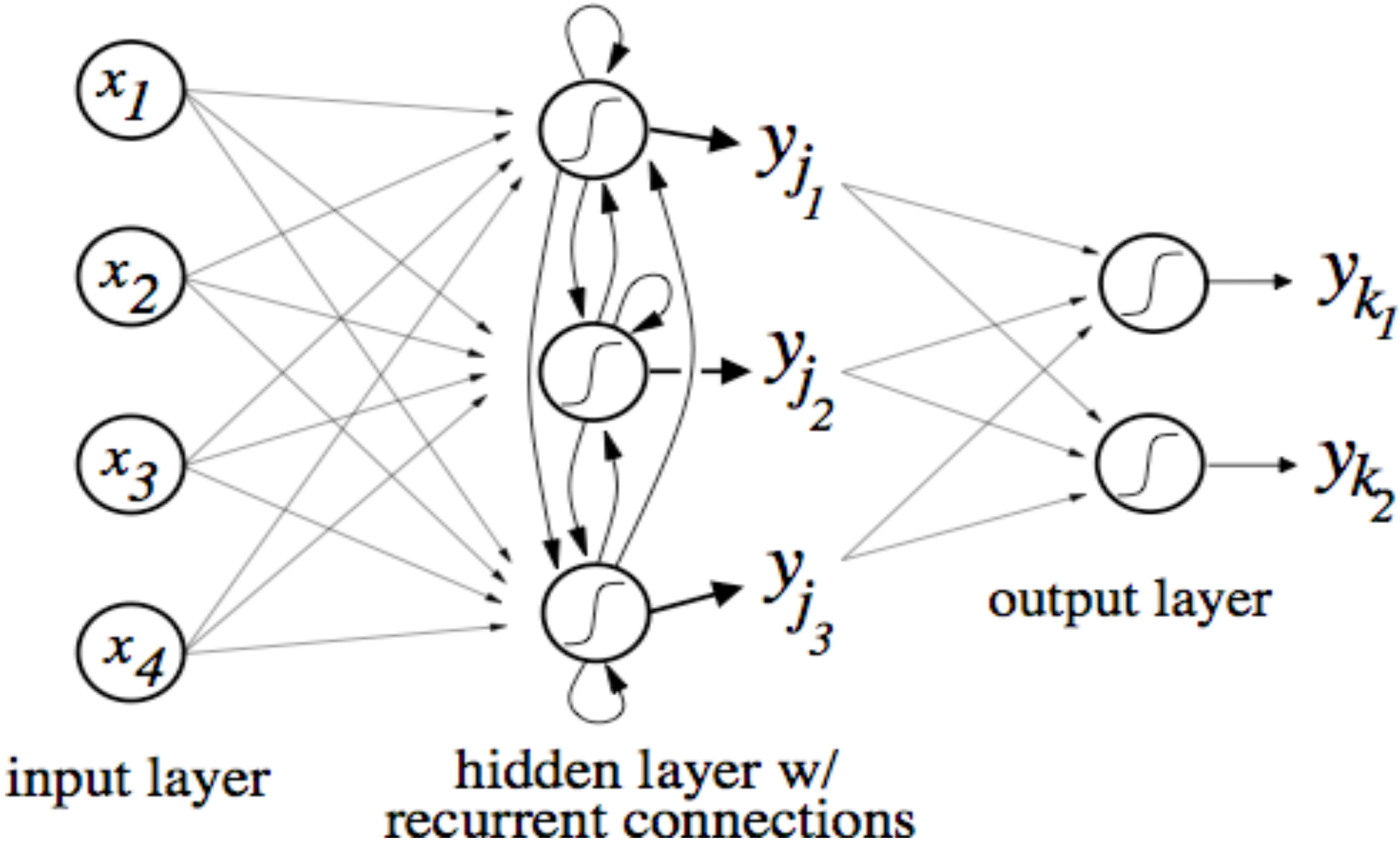
- The more general, and ultimately more powerful, way of doing sequence prediction/classification

- Feed only the current time-step into the network at any given time; make the network remember what it saw before

Recurrent non-linear neuron

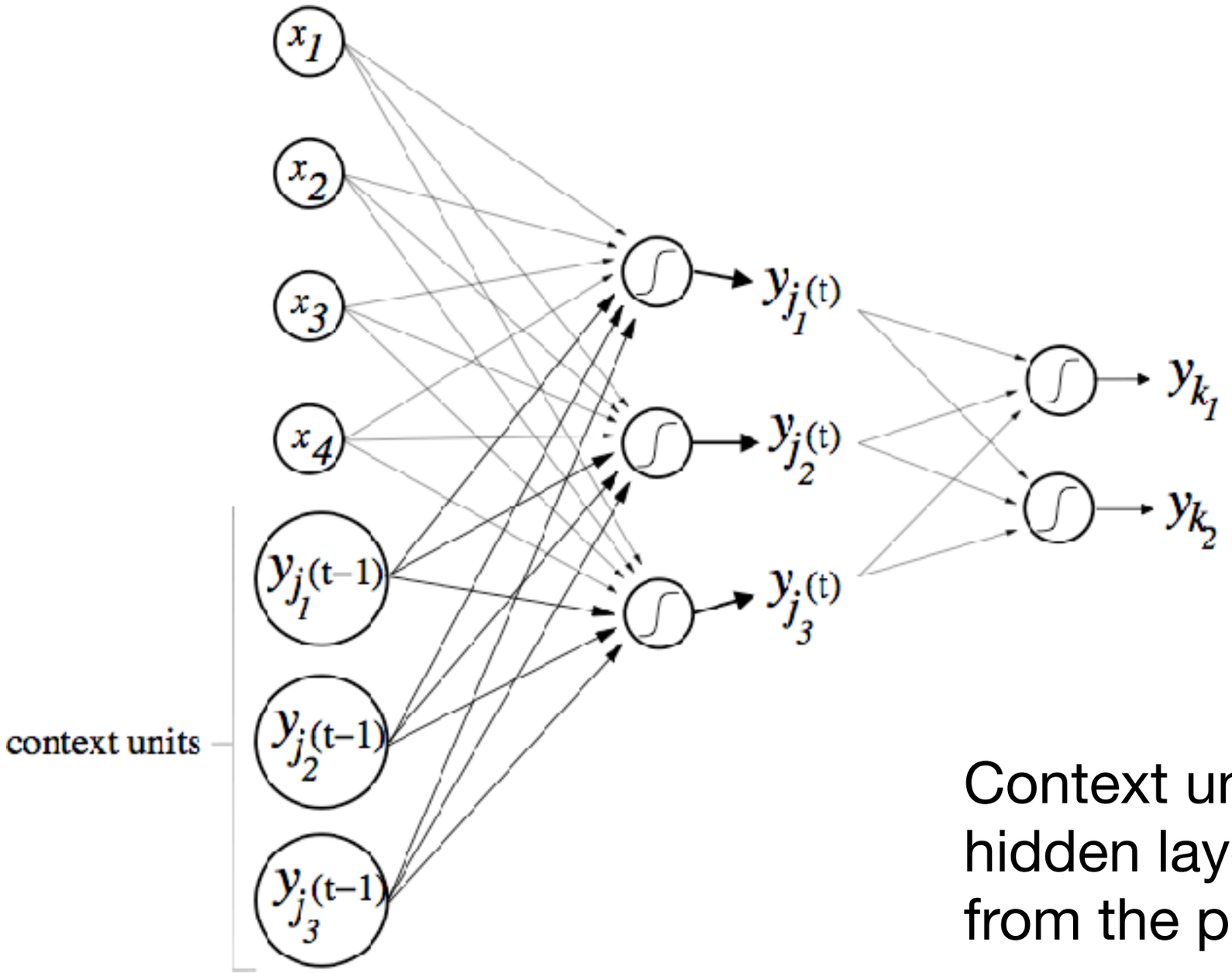


Simple recurrent network (SRN)



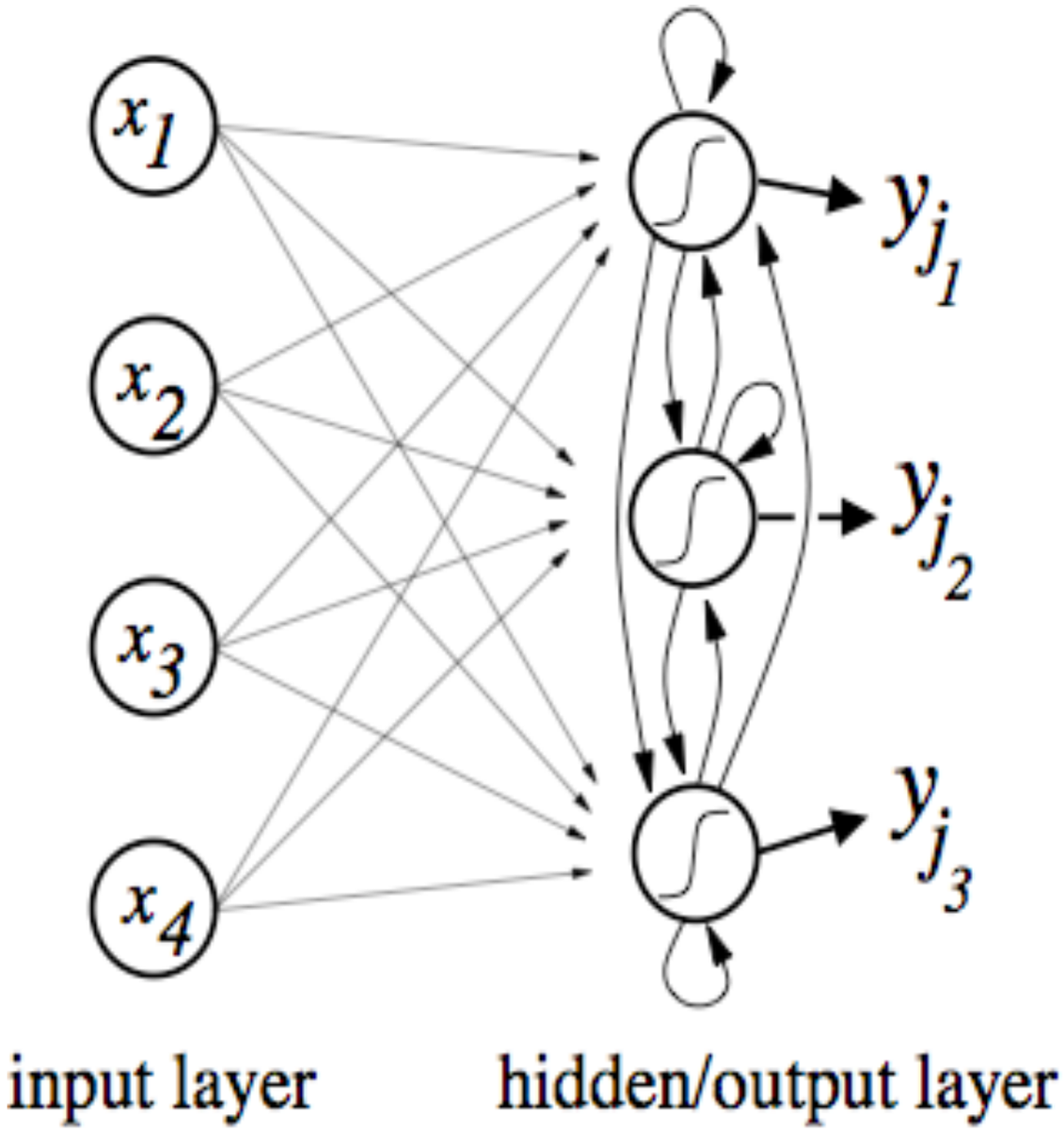
Hidden units now have state or memory which is dependent on all previous inputs

SRN: another perspective



Context units are just the hidden layer activation from the previous time step

Fully recurrent network



- Can approximate any differentiable trajectory
- Same as SRN but without output layer

Recurrent networks are actually Turing-complete!

- Feedforward networks can only implement functions
- Recurrent nets are dynamical systems
- A large enough feedforward net (e.g. MLP) can approximate any function
- A large enough recurrent network can implement any algorithm
- In practice, easier to learn some algorithms than others...

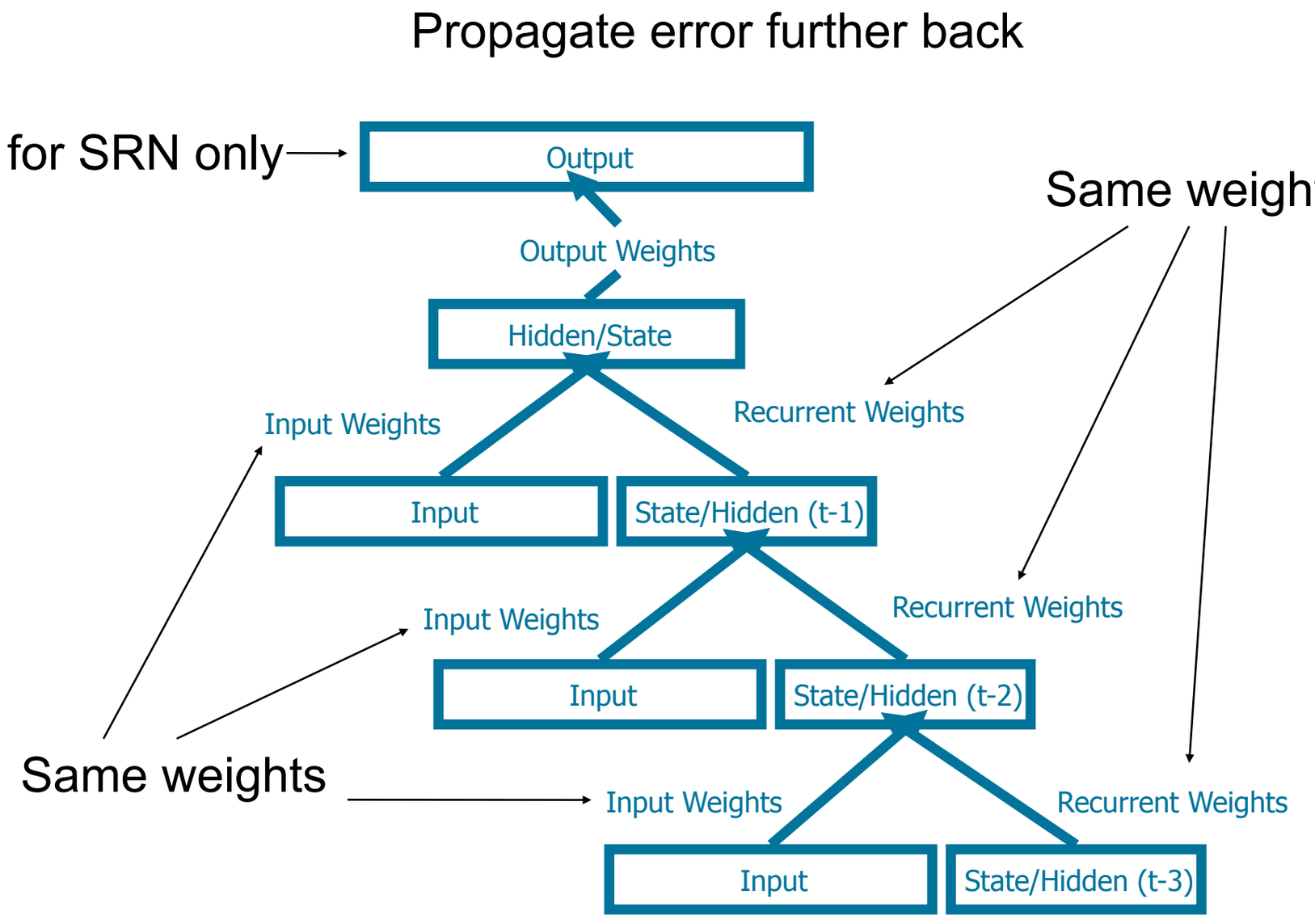
Train with truncated backpropagation

- Train the same way as an MLP
- But treat activation from previous time-step as just another set of inputs (context units)
- The network can now learn to map the same external inputs to different outputs due to “context”

Backpropagation Through Time

- Just like backpropagation but network is “unfolded” spatially for each time-step in input sequence
- For an n-step sequence, we get a network with n-layers
- Each layer has the same weights
- Error at output is propagated back through all layers

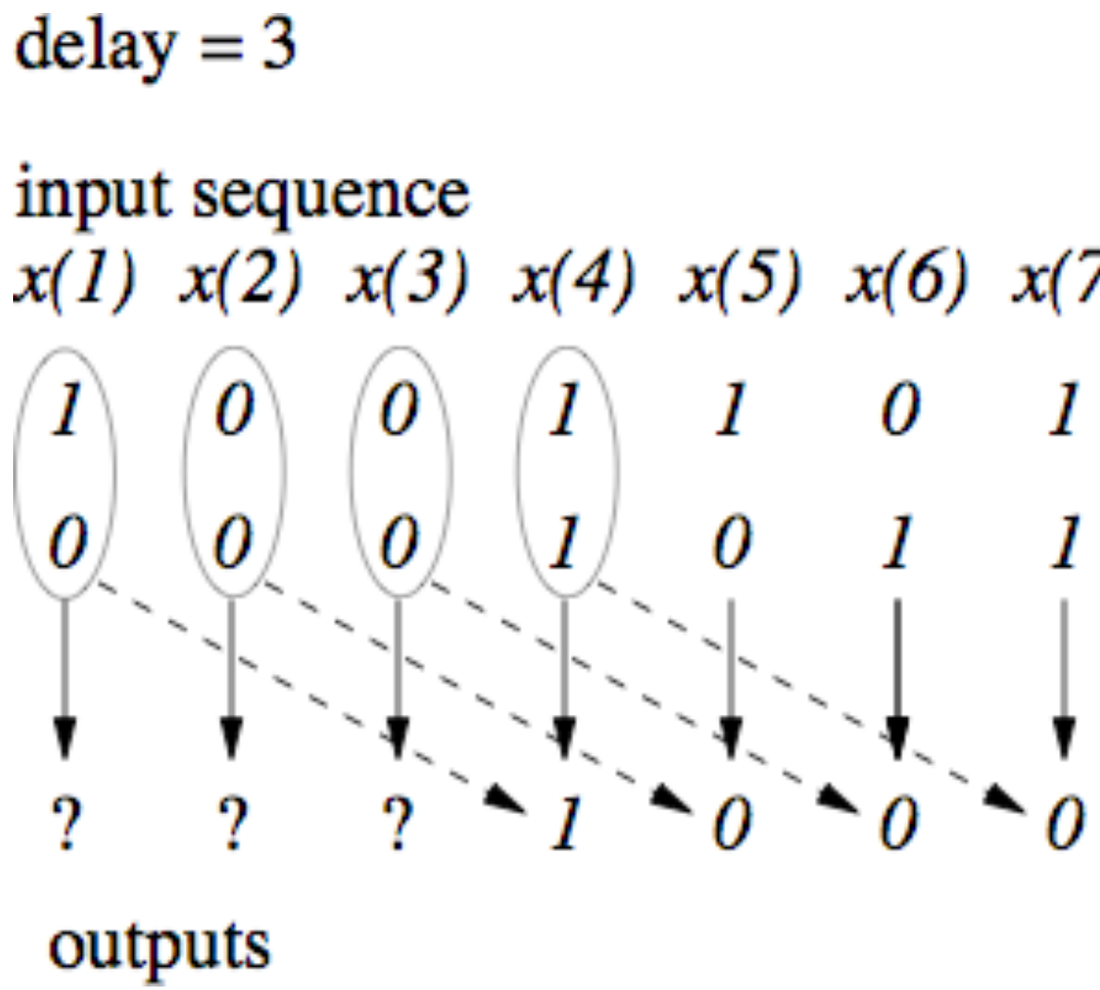
Backpropagation Through Time



Simple example: XOR with delay

- Network learns to perform XOR of two or more inputs,
- But instead of outputting the XOR of the current inputs,
- It outputs the XOR of the input it saw n steps ago

Delayed XOR


delay = 3
input sequence
x(1) x(2) x(3) x(4) x(5) x(6) x(7)
1 0 0 1 1 0 1
0 0 0 1 0 1 1
? ? ? 1 0 0 0
outputs

Vanishing error gradient

- Although RNNs can represent arbitrary sequential behavior, they are not easy to train when the output depends on some input more than around 10 time-steps in the past

- The error gradient becomes very small so that the weights cannot be adjusted to respond to events far in past

- Might as well use an MLP with an input layer *n* time-steps wide… if you know n in advance!

- Solution:

  - Long Short-Term Memory

  - Neuroevolution

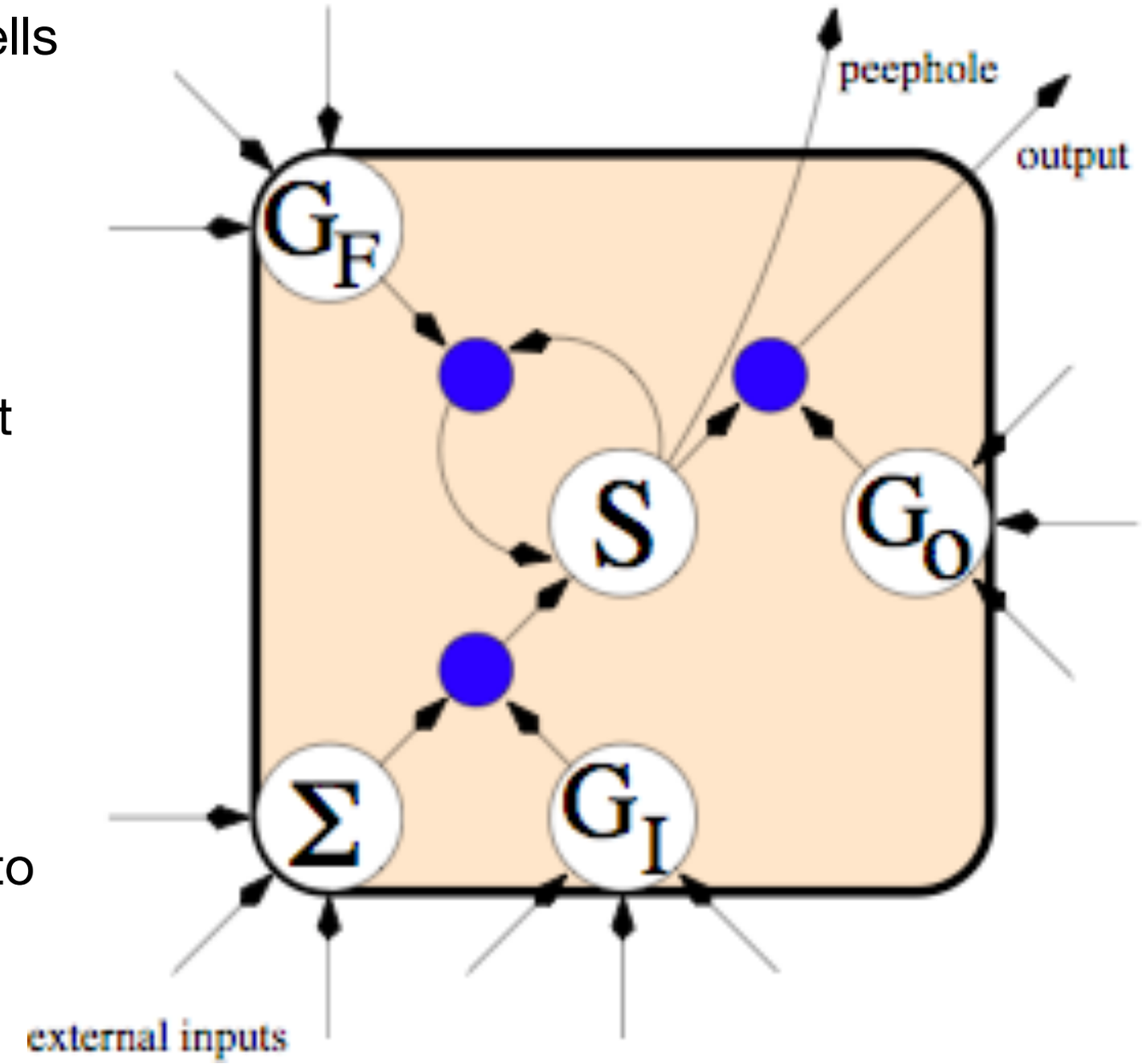
Long Short-Term Memory (LSTM)

- LSTM nets have memory cells with a linear state S that keeps error flowing back in time and is controlled by 3 gates

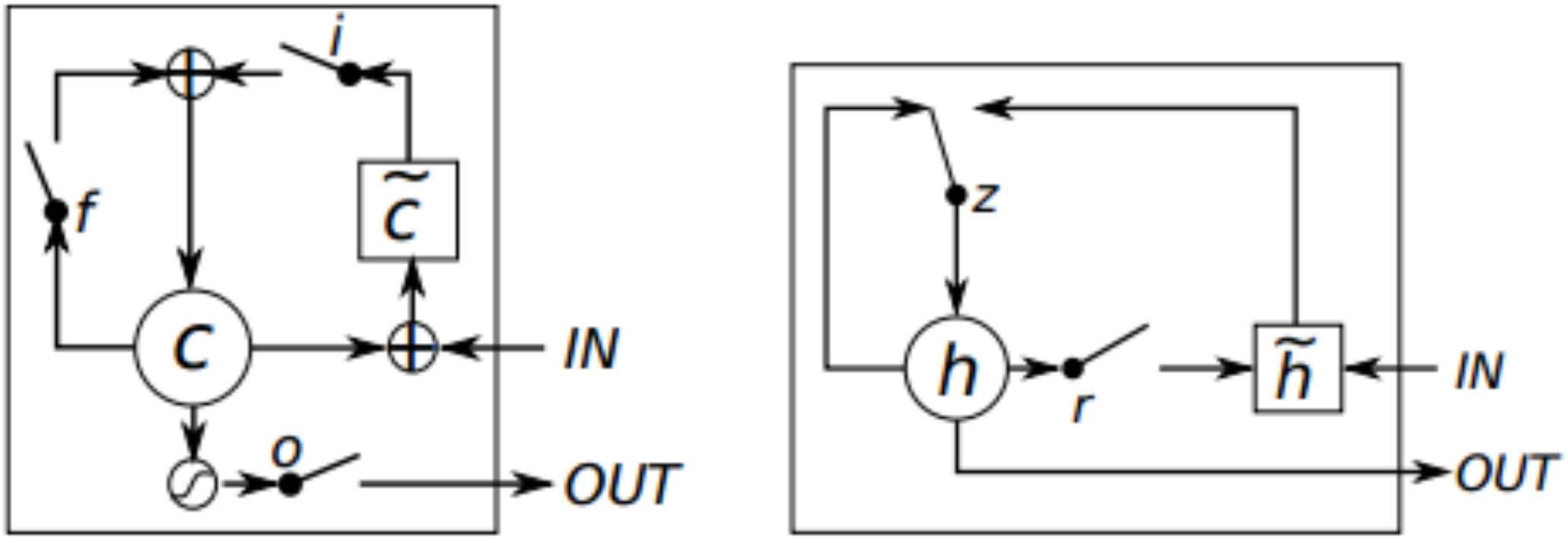
- Input gate (Gi) controls what information enters the state

- Output gate (Go) controls what information leaves the state to other cells

- Forget gate (Gf) allows cell to forget state when no longer needed



Gated recurrent units

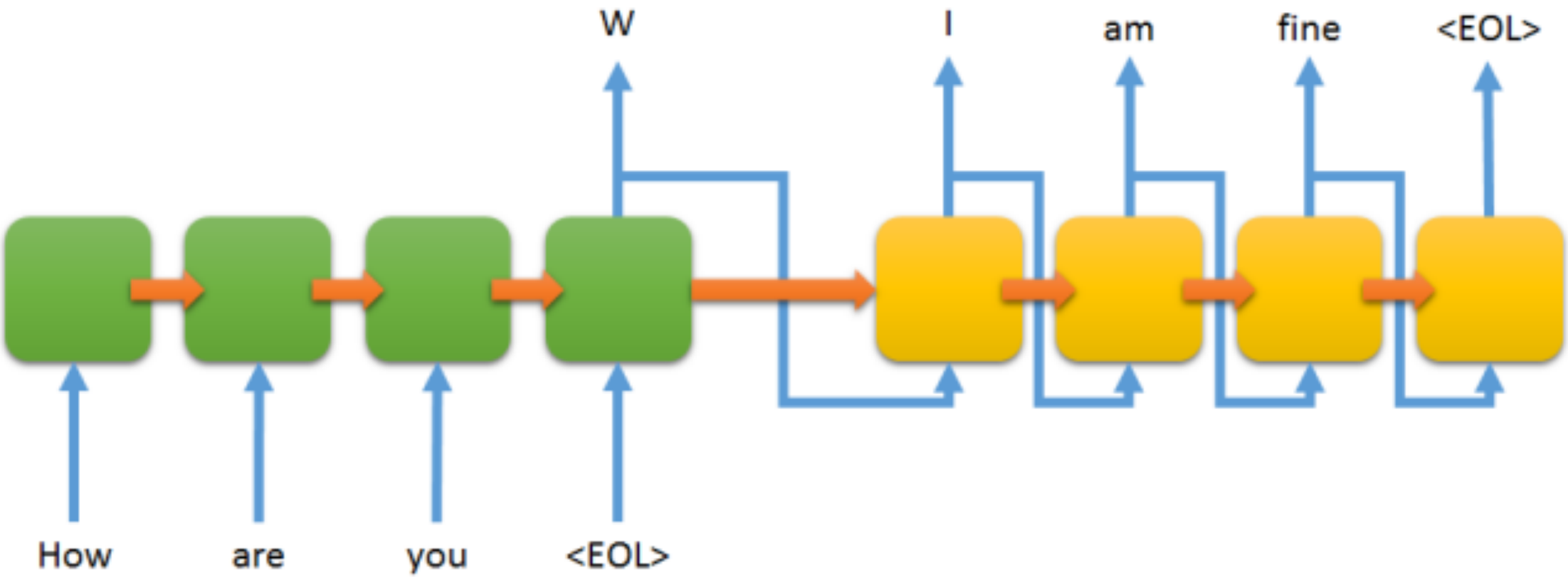


(a) Long Short-Term Memory

(b) Gated Recurrent Unit

Figure 1: Illustration of (a) LSTM and (b) gated recurrent units. (a) i, f and o are the input, forget and output gates, respectively. c and $\tilde{c}$ denote the memory cell and the new memory cell content. (b) r and z are the reset and update gates, and h and $\tilde{h}$ are the activation and the candidate activation.

Sequence-to-sequence



Sequence learning demo

- SketchRNN, trained on the game Quick, draw!

- https://magenta.tensorflow.org/sketch-rnn-demo

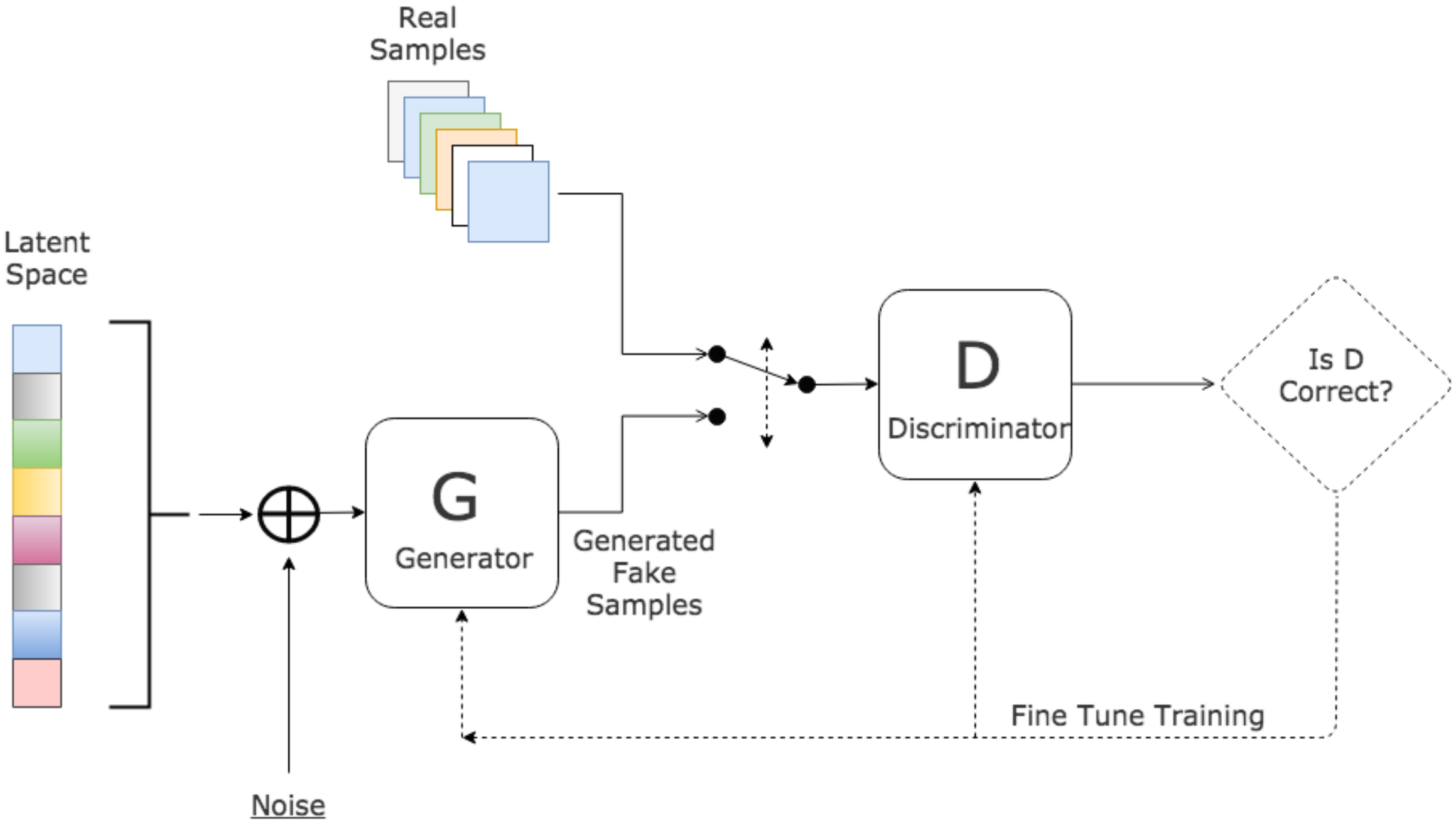
Bonus question: which movie?

Generative Adversarial Networks (GANs)

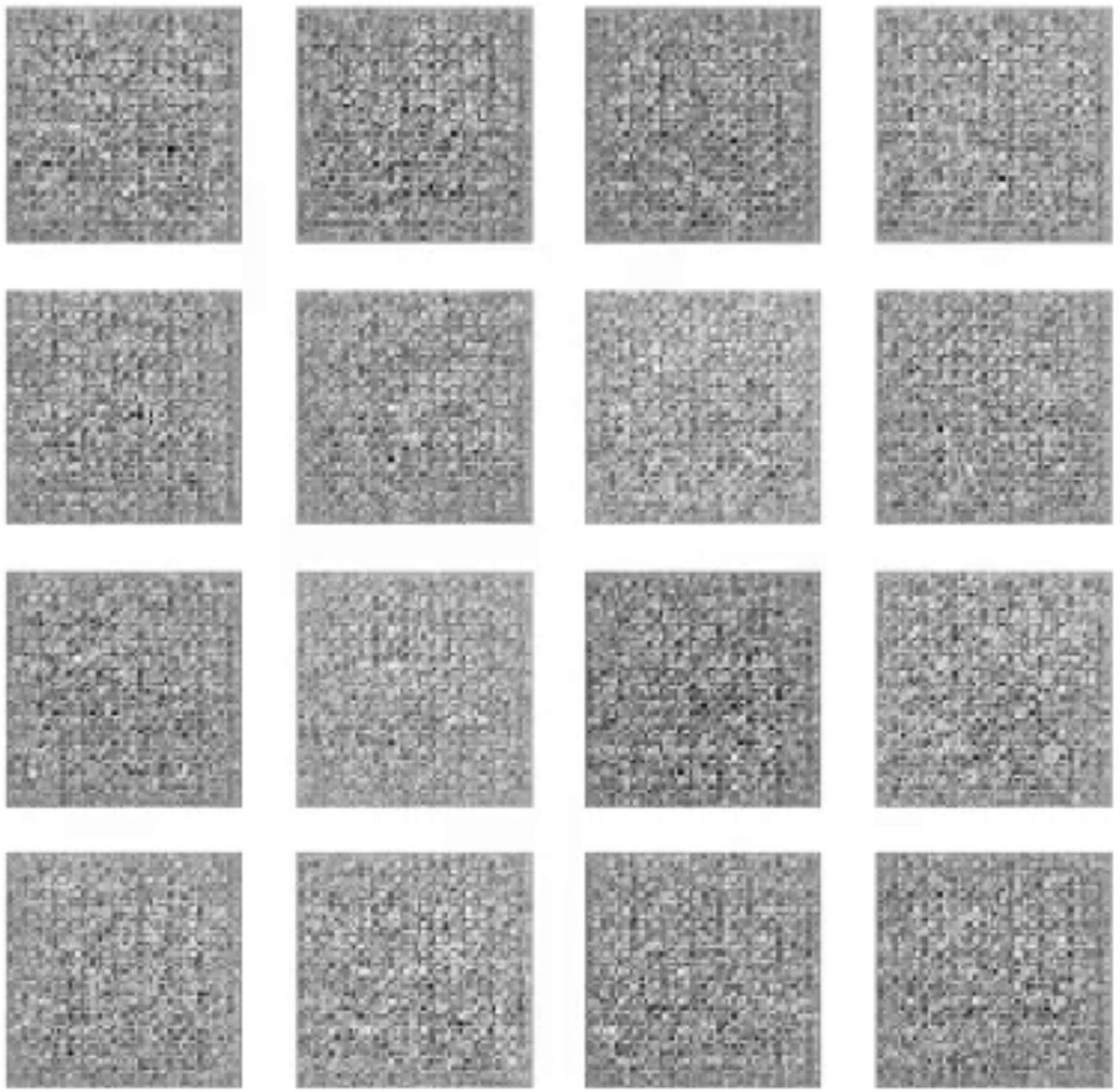
Generative Adversarial Networks (GANs)

- Train two networks, a discriminator and a generator
  - The discriminator tries to tell real from generated images; gets rewarded for getting it right
  - The generator tries to fool the discriminator; gets rewarded for the discriminator getting it wrong
- Pretty much like competitive coevolution, but with gradients

Generative Adversarial Network



GAN Training



BigGAN

Cool GAN Demos

- https://ganbreeder.app/

-

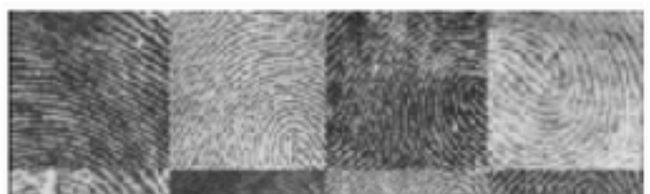
LILY HAY NEWMAN SECURITY 11.17.18 07:00 AM

MACHINE LEARNING CAN CREATE FAKE 'MASTER KEY' FINGERPRINTS

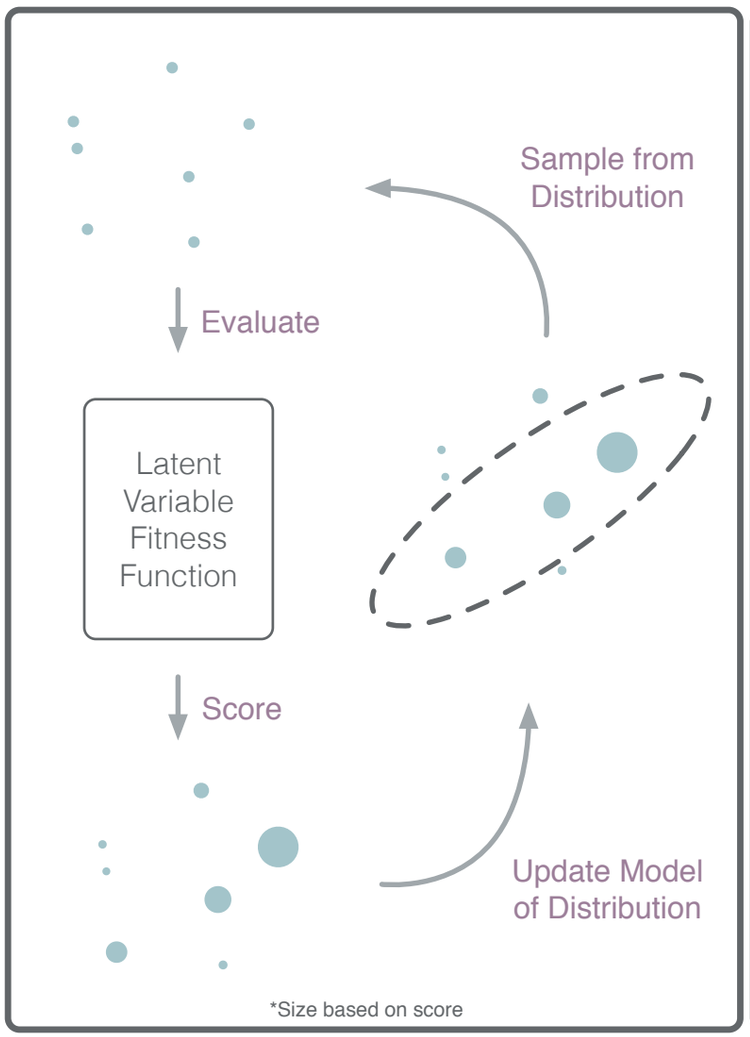
Biometrics

Fake fingerprints can imitate real ones in biometric systems – research

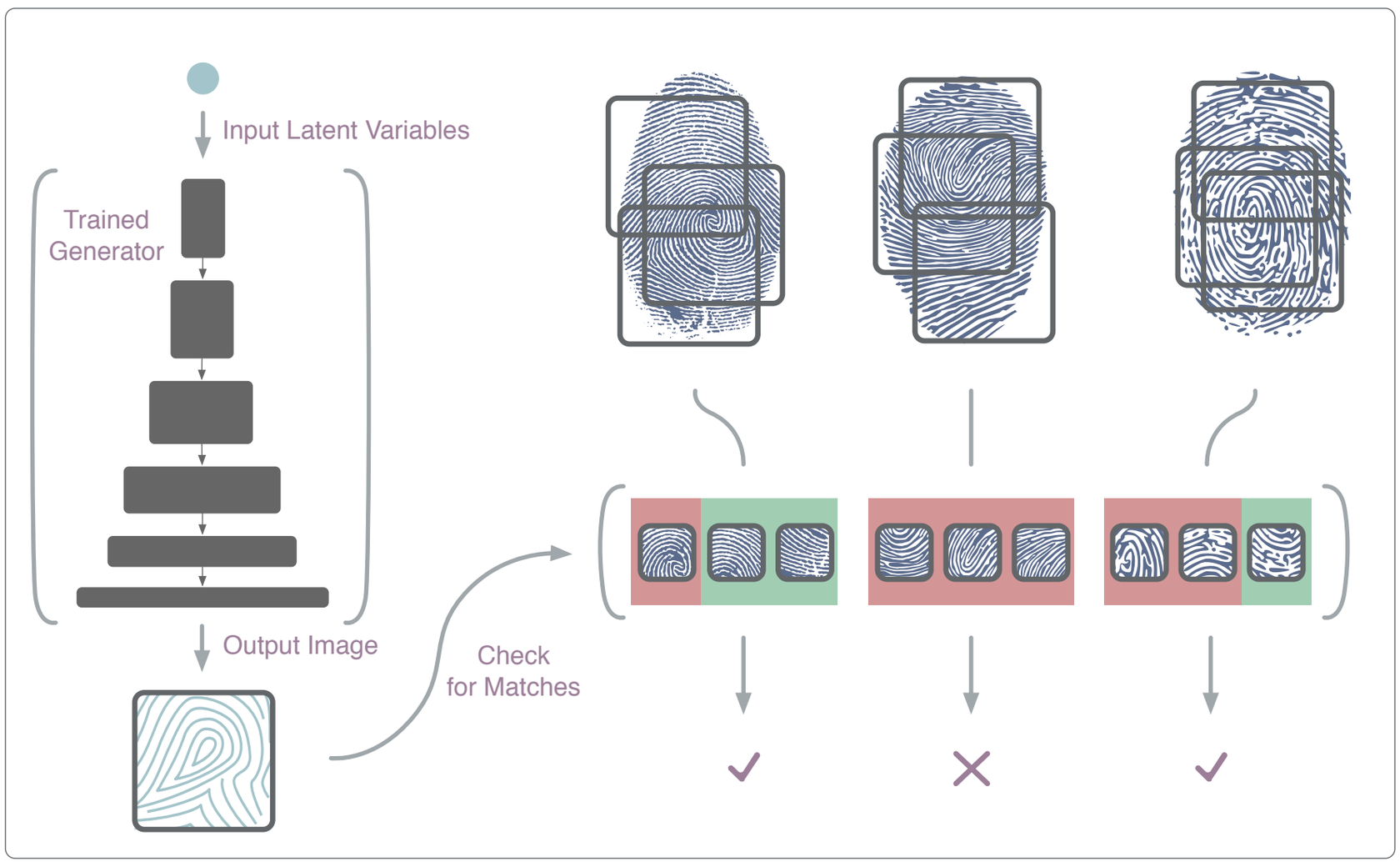
DeepMasterPrints created by a machine learning technique have error rate of only one in five



Alex Hern
@alexhern
Thu 15 Nov 2018 01.01 EST



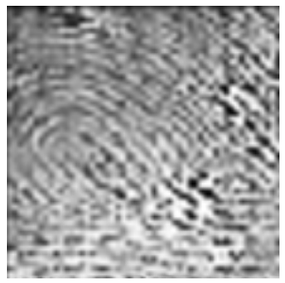
CMA-ES Optimization

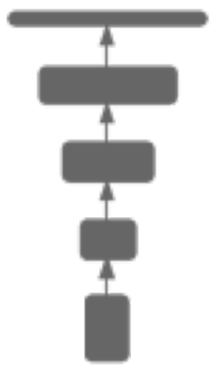


Latent Variable Fitness Function

Training

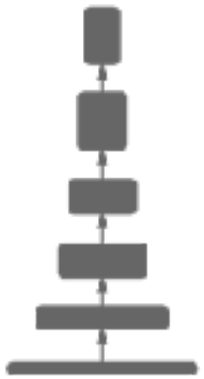
Training:

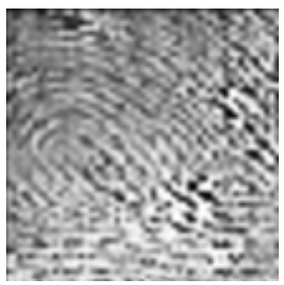




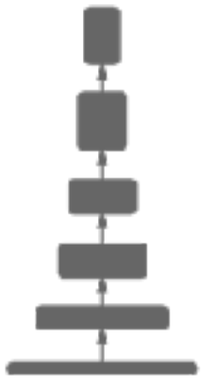
[.1, -.4, …, -.2, -.3]

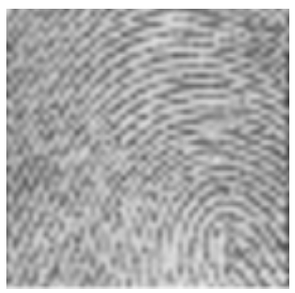
Fake





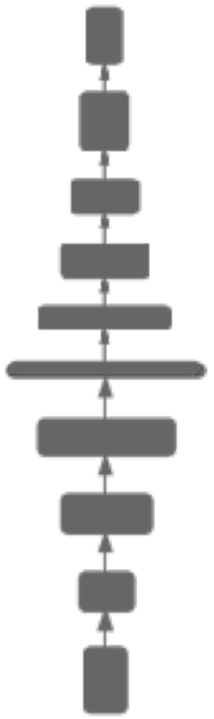
Real





Freeze Weights

Real



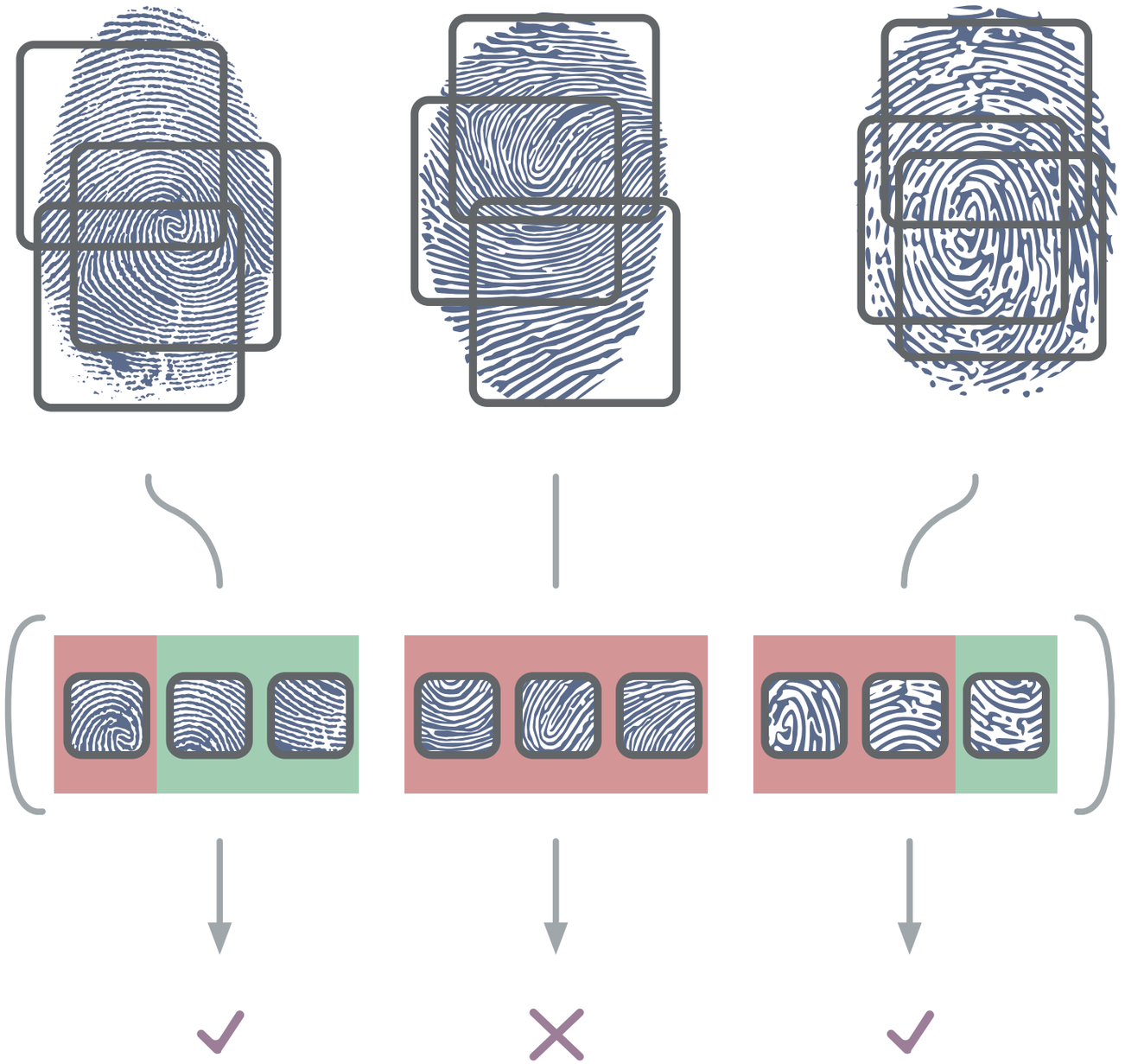
[-.2 .1, …, .3, .3]

Fingerprint Score

- Test dataset of fingerprints broken apart into partial fingerprints

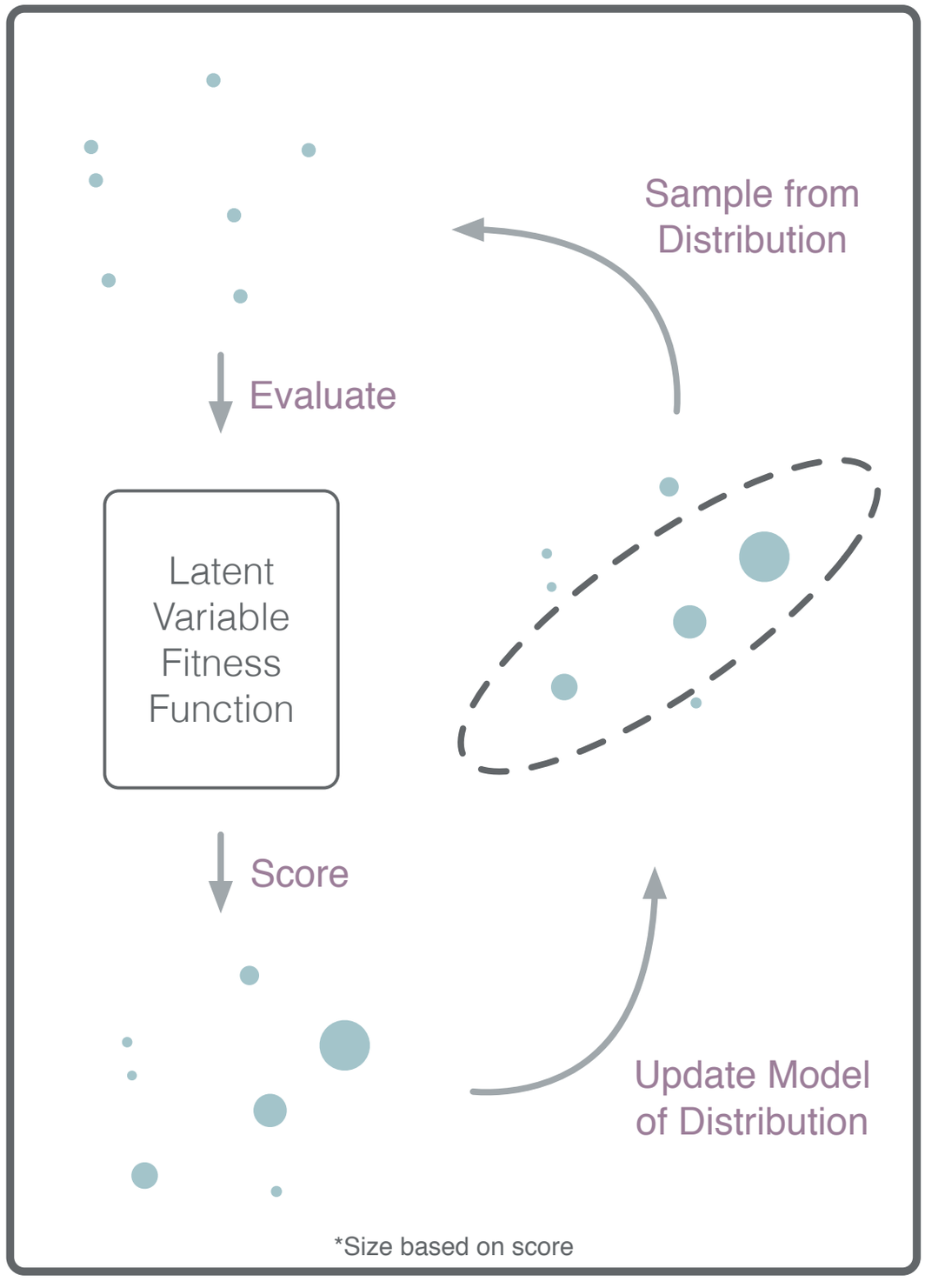
- Fingerprint matcher identifies matches at a specific False Match Rate (FMR) threshold

- A single partial fingerprint match counts as a security breach



CMA-ES

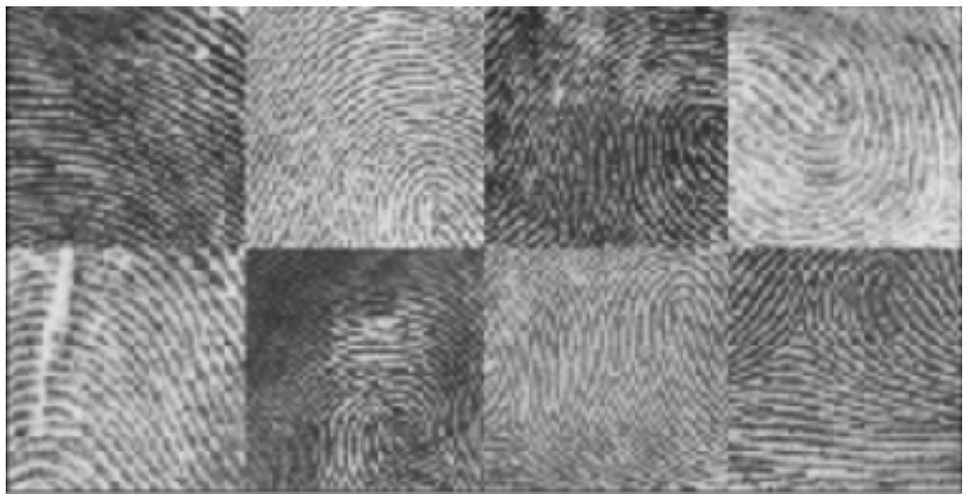
- Sample based evolutionary algorithm
- DeepMasterPrints represented as latent variables
- Covariance Matrix of successful fingerprints learned

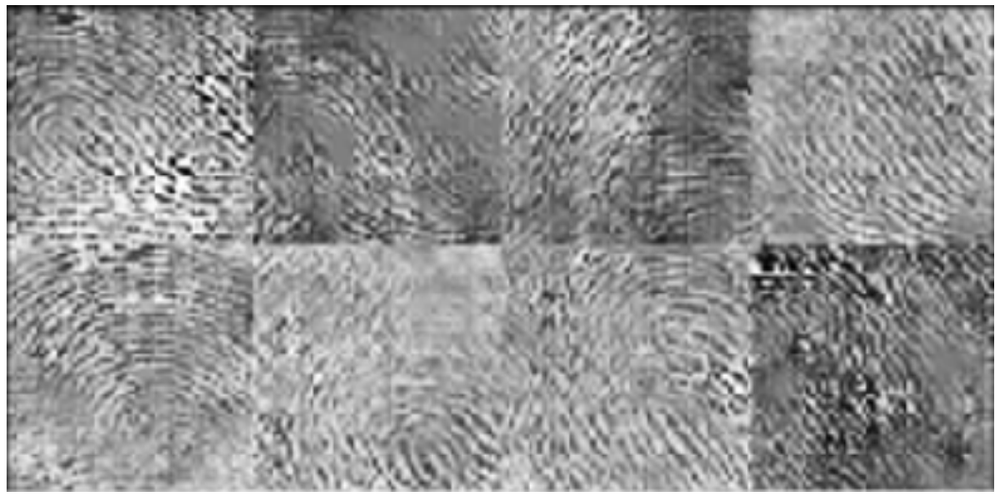


Generated Fingerprints

Fingerpass DB7
720 Subjects, Right Thumb

NIST Special Database 9
5400 Subjects, Right Thumb





VeriFinger DeepMasterPrints

	Rolled DeepMasterPrint			Capacitive DeepMasterPrint		
	0.01% FMR	0.1% FMR	1% FMR	0.01% FMR	0.1% FMR	1% FMR
						
Matches	0.3%	8.6%	78.1%	1.1%	22.5%	76.7%

All evolved for the Fingerpass DB7 dataset (50% train/test split)