Procedural Content Generation via Machine Learning (PCGML) + PCG Review

Matthew Guzdial

guzdial@ualberta.ca

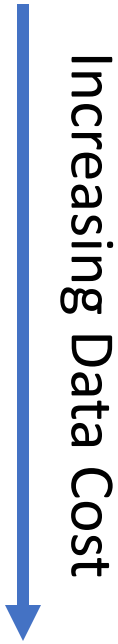
UNIVERSITY OF ALBERTA

Announcements

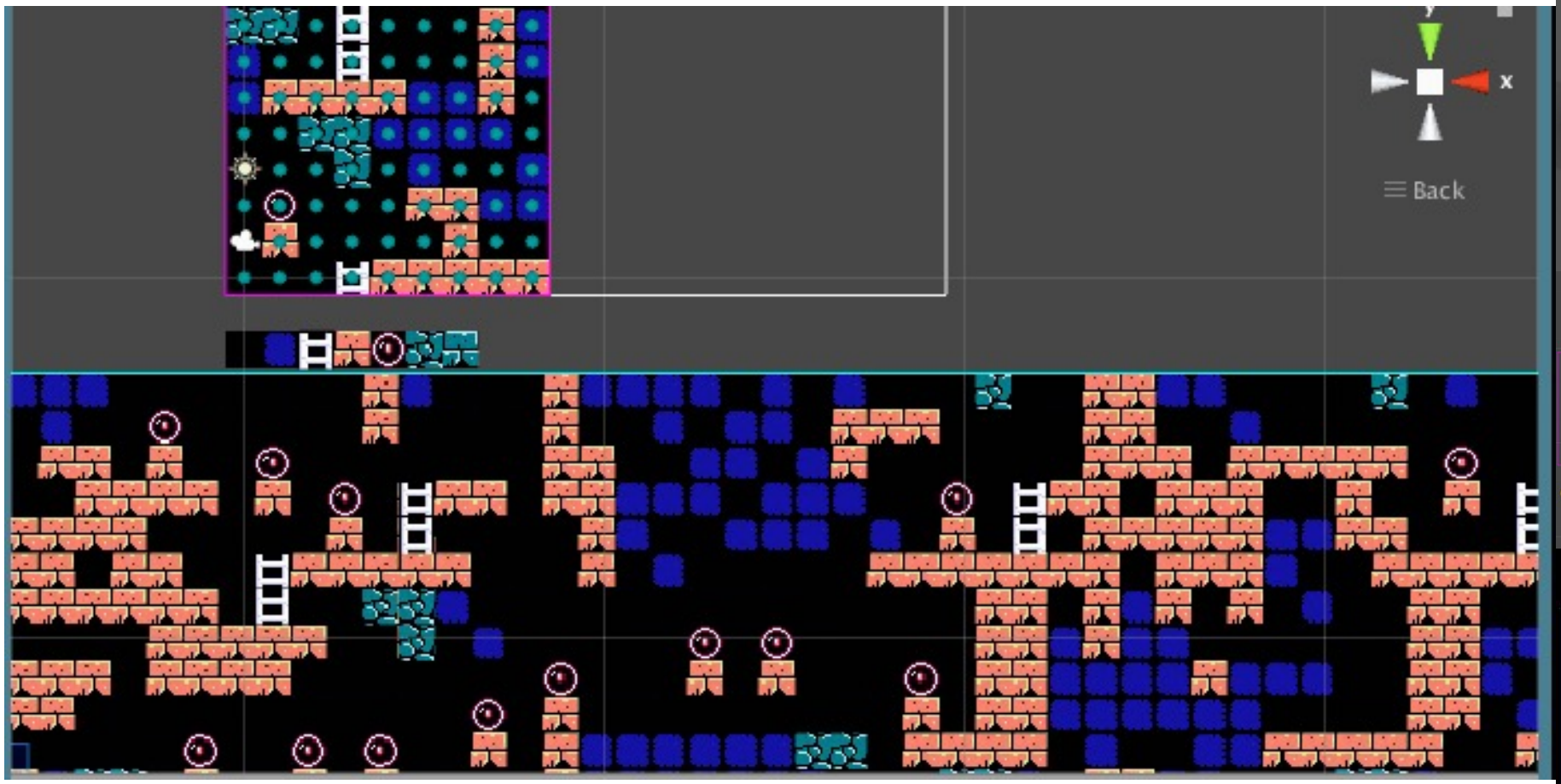
- Vote was 40% review, 40% cover new things, so first a lightning overview of popular PCGML approaches, then review
- HW4 Released (~8/10), HW5 Help Session next week
- Quiz 4 released (~7/9)
- Final Topic Voting Results: (1) Automated Game Playing, (2) AI-based Game Design, (3) Generating Dialogue + Story, (4) RL in Games
- Upcoming Lectures
  - Wednesday, Nov 24: RL in Games
  - Monday, Nov 29: Automated Game Playing
  - Wednesday, Dec 1: Generating Dialogue + Story
  - Friday, Dec 3: AI-based Game Design
  - (not a lecture) Monday, Dec 6: “Final” Quiz 6

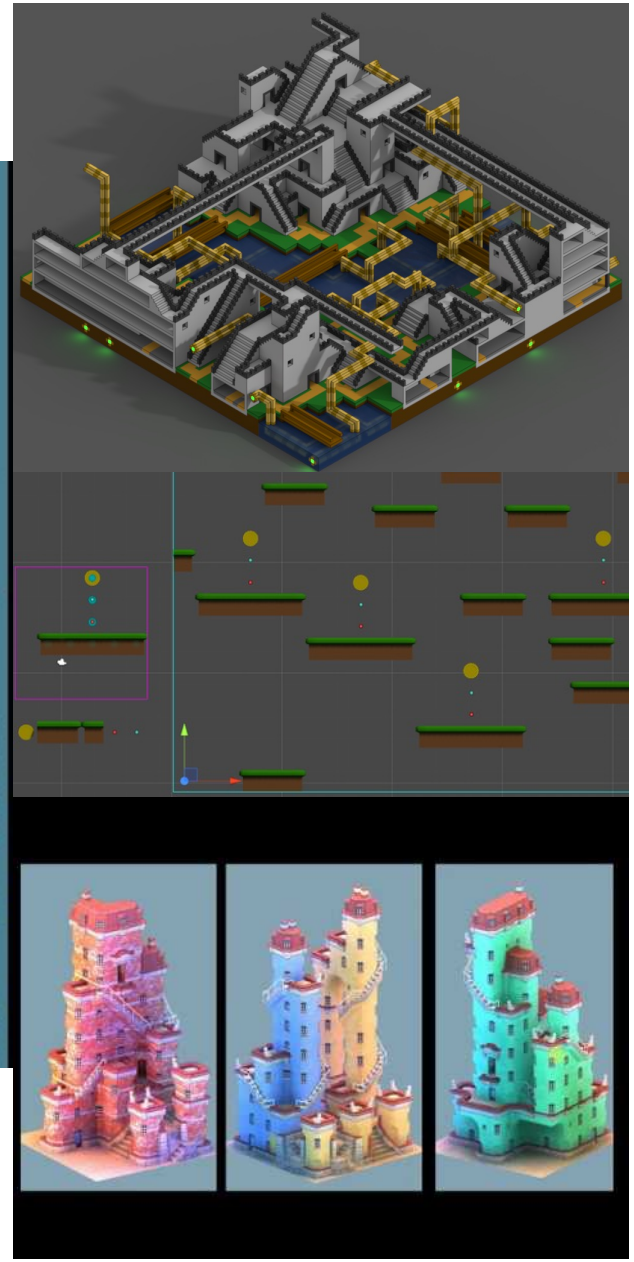
PCGML Level Generation Overview

1. Wave Function Collapse
2. Markov Methods
3. Bayesian Networks
4. Long Short-Term Memory Recurrent Neural Networks (LSTMs) and Autoencoders
5. Generative Adversarial Neural Networks (GANs)



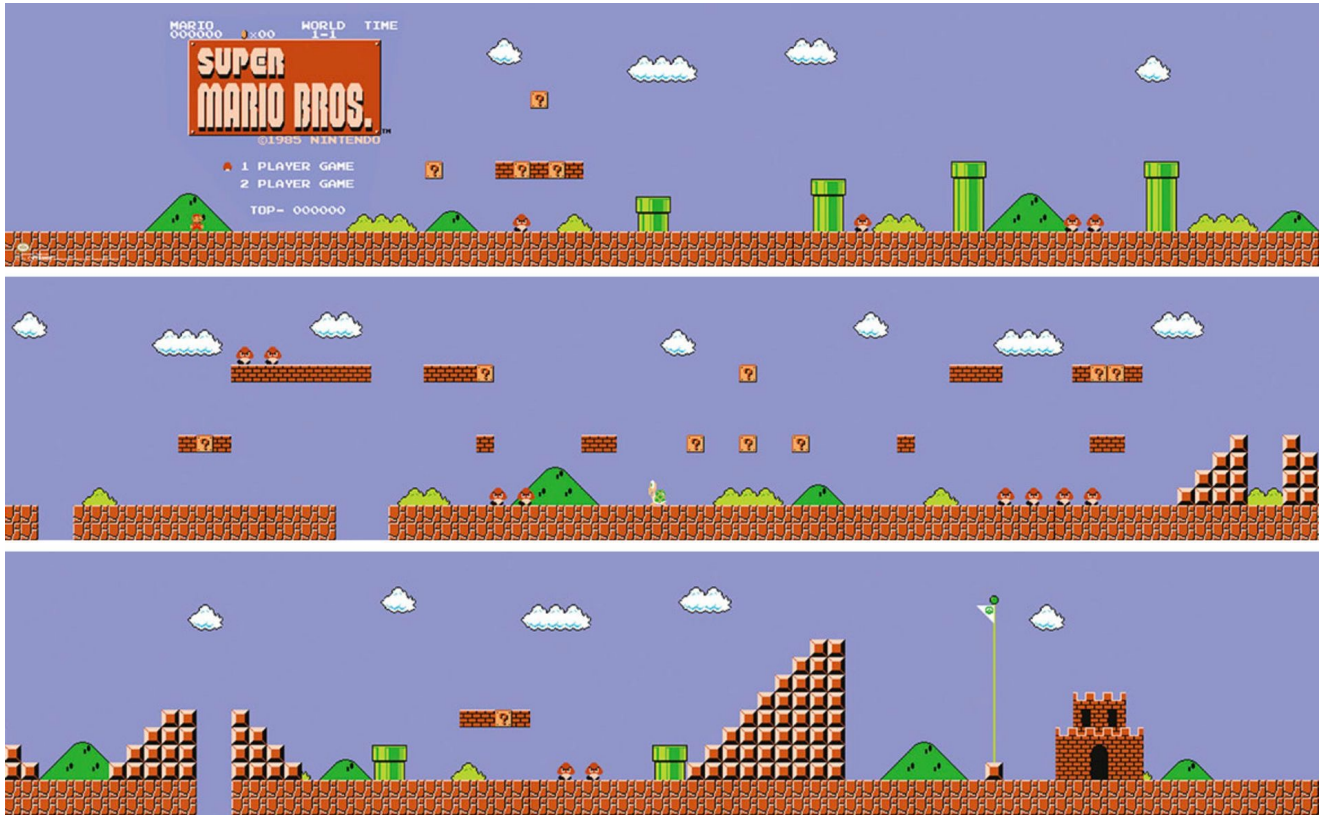
1. Wave Function Collapse (WFC)



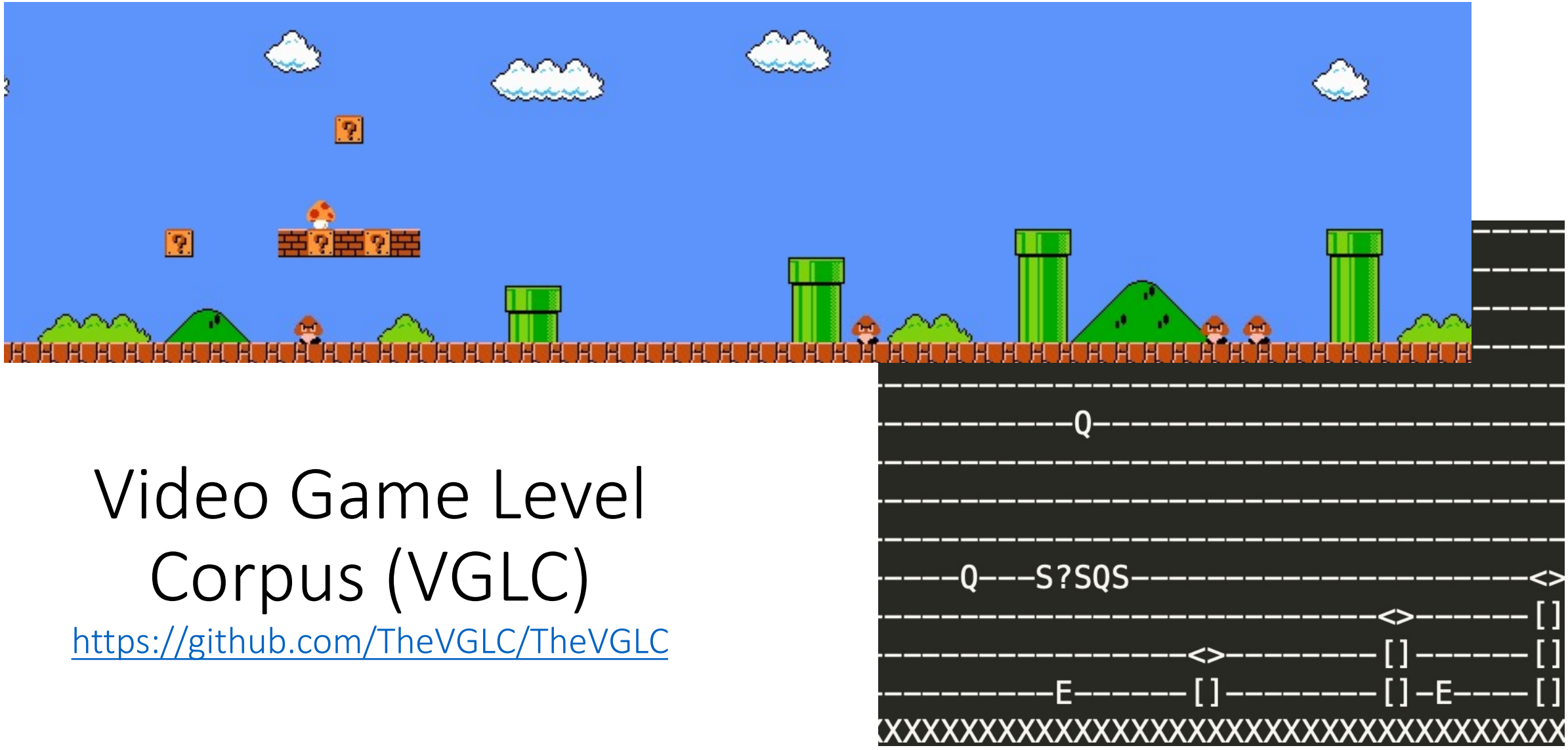


Example: https://www.youtube.com/watch?v=7ffT_8wViBA&feature=emb_title
WFC Tutorial: http://www.procjam.com/tutorials/wfc/

Remaining 2-5 Super Mario Bros. (PCGML's MNIST)



Tile-based Representation



Video Game Level Corpus (VGLC)

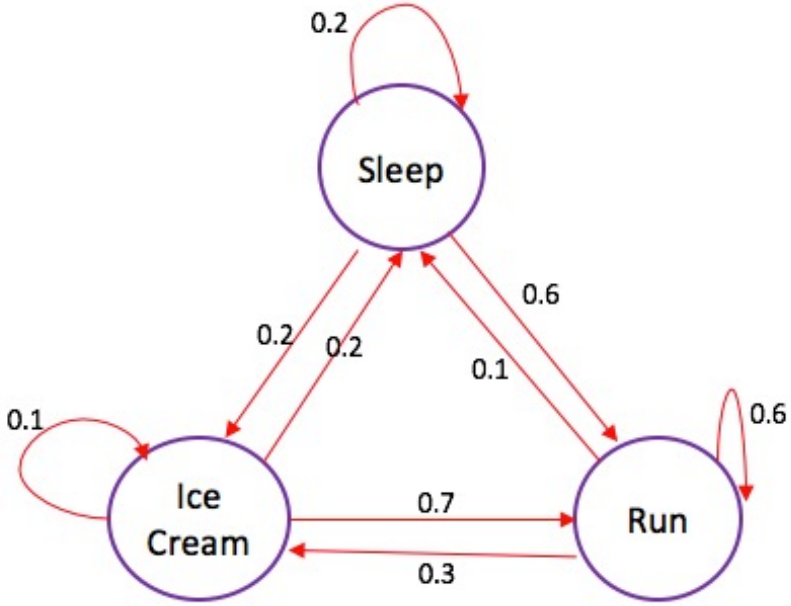
https://github.com/TheVGLC/TheVGLC

2. Markov Methods

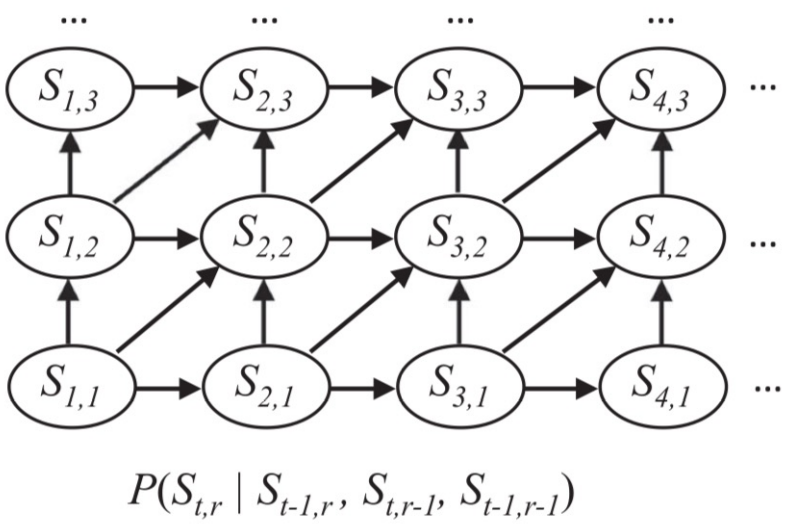
Probabilistic Graphical Model.

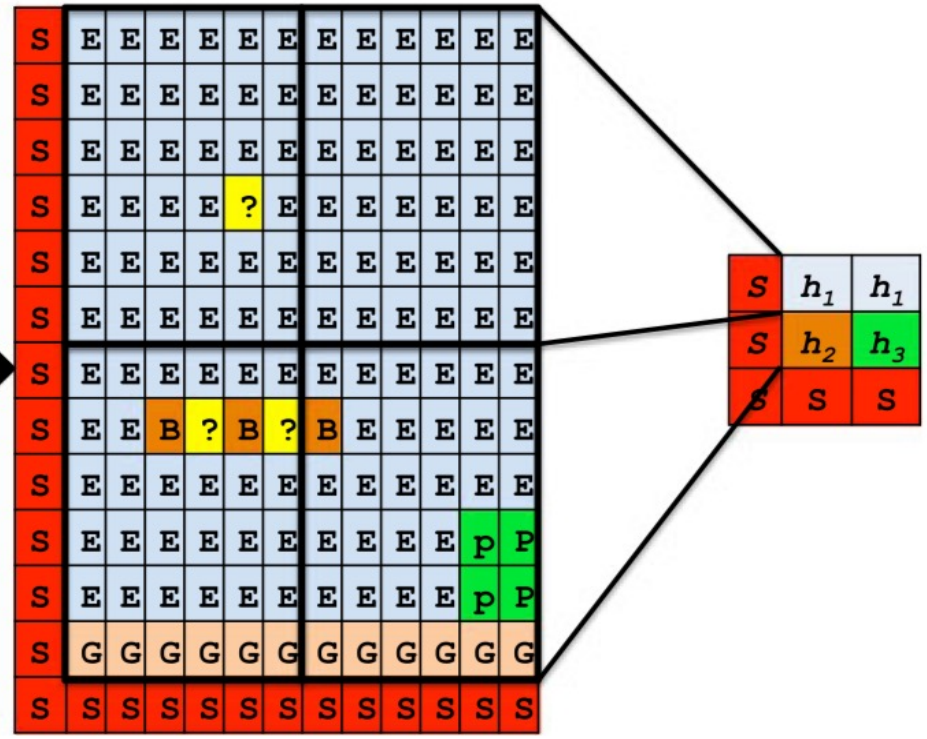
Simplest version is the **Markov Chain**, which represents some set of states *S* and the probabilities of transitioning from one state to another $P(s_1,s_2)$.

We can approximate these probabilities from training data of sequences of states.

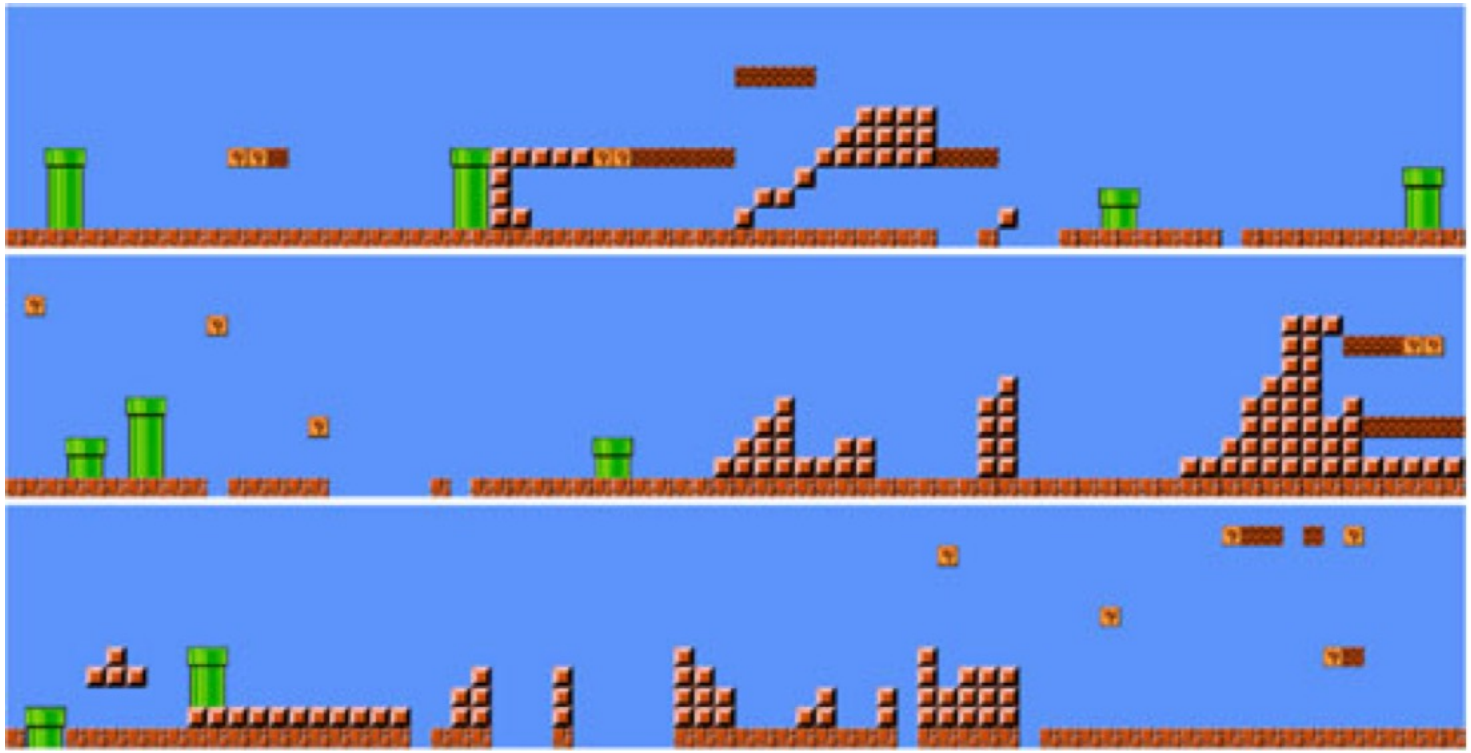


Markov Chains for Super Mario Bros.



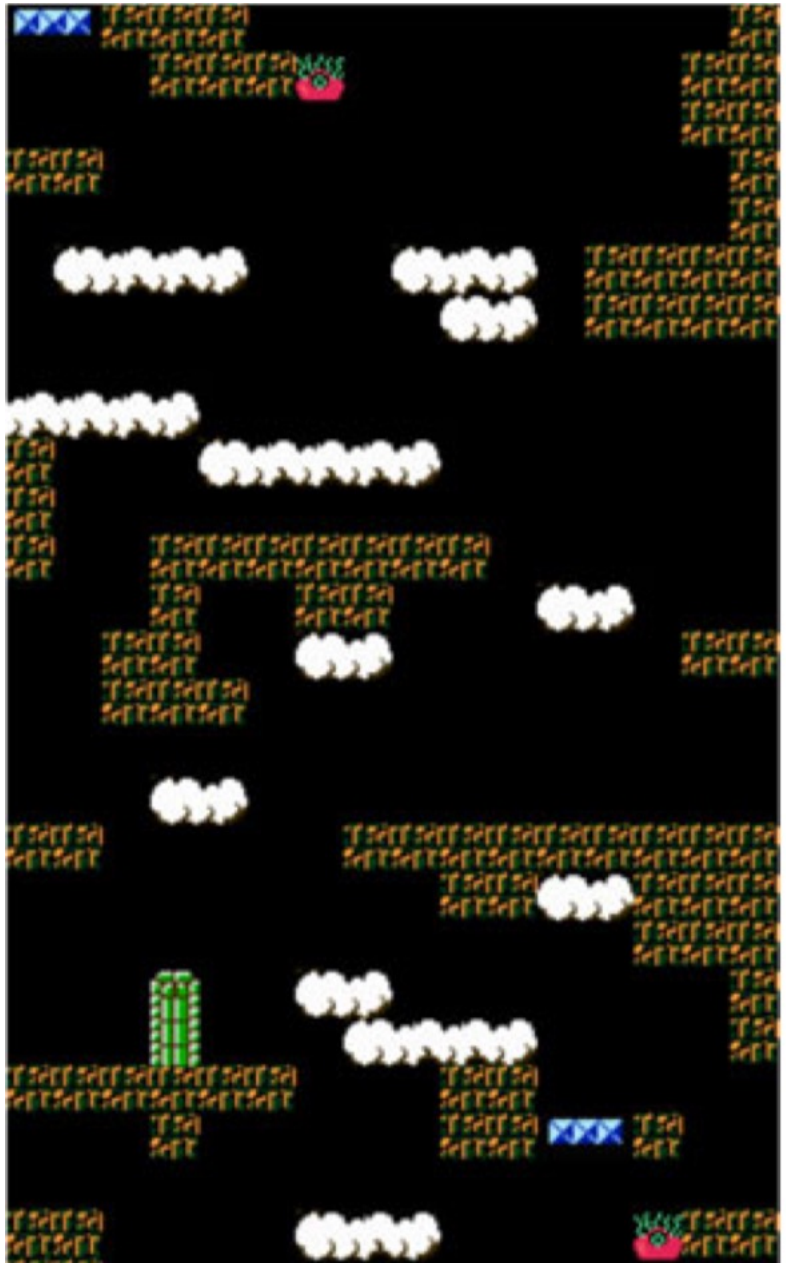


Snodgrass and Ontañón. 2017:
https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7728021



^Super Mario Bros.

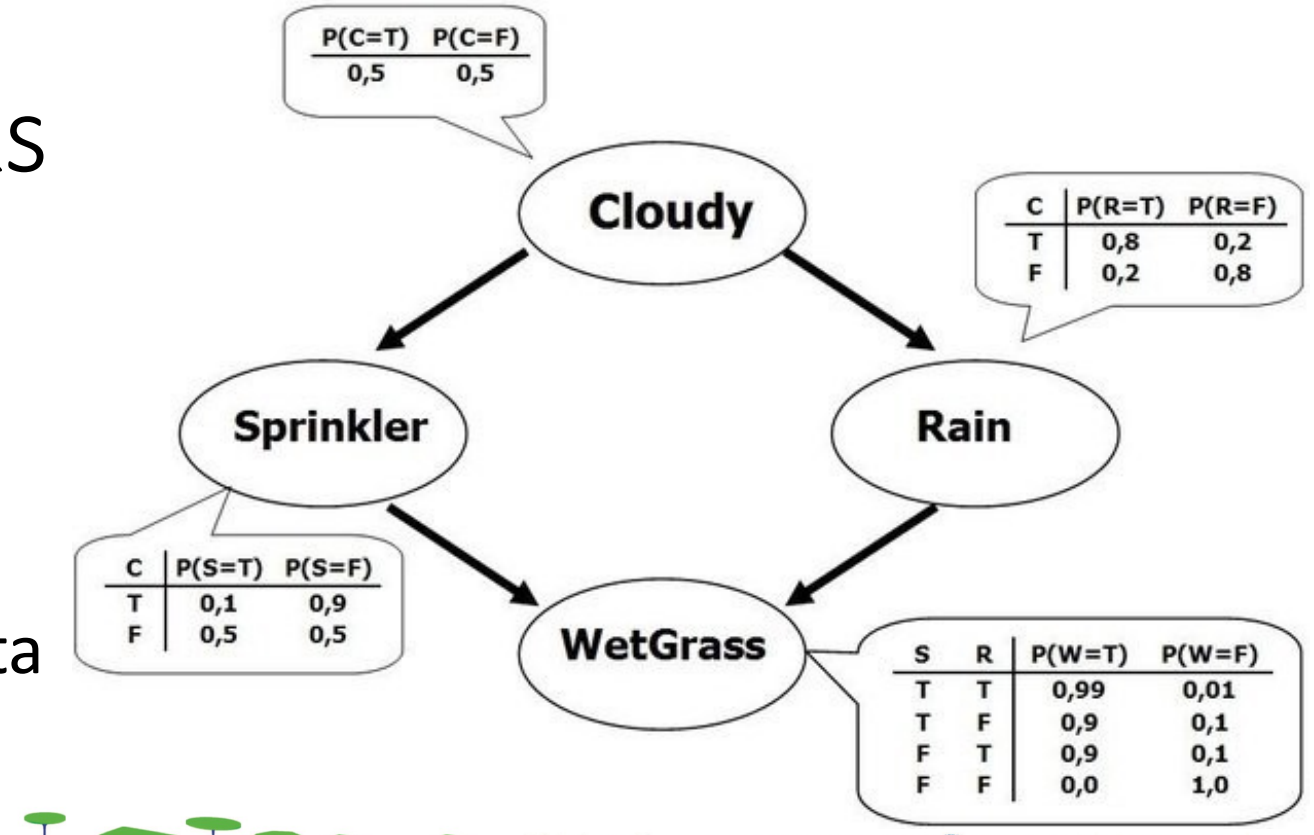
Kid Icarus ->

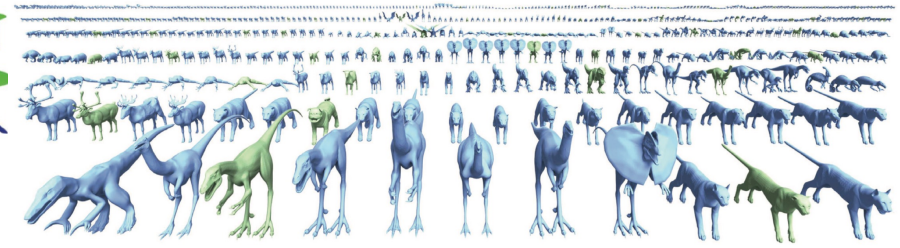


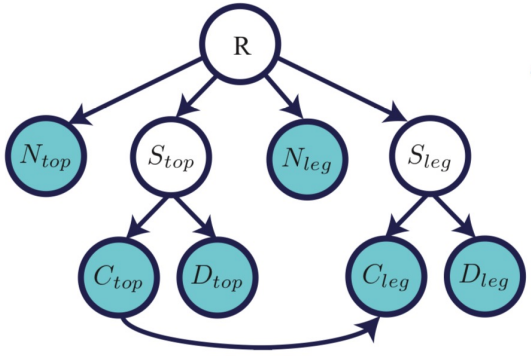
3. Bayesian Networks

Probabilistic Graphical Model.

Major differences from Markov chain are that input training data is a graph and that there are hidden nodes.

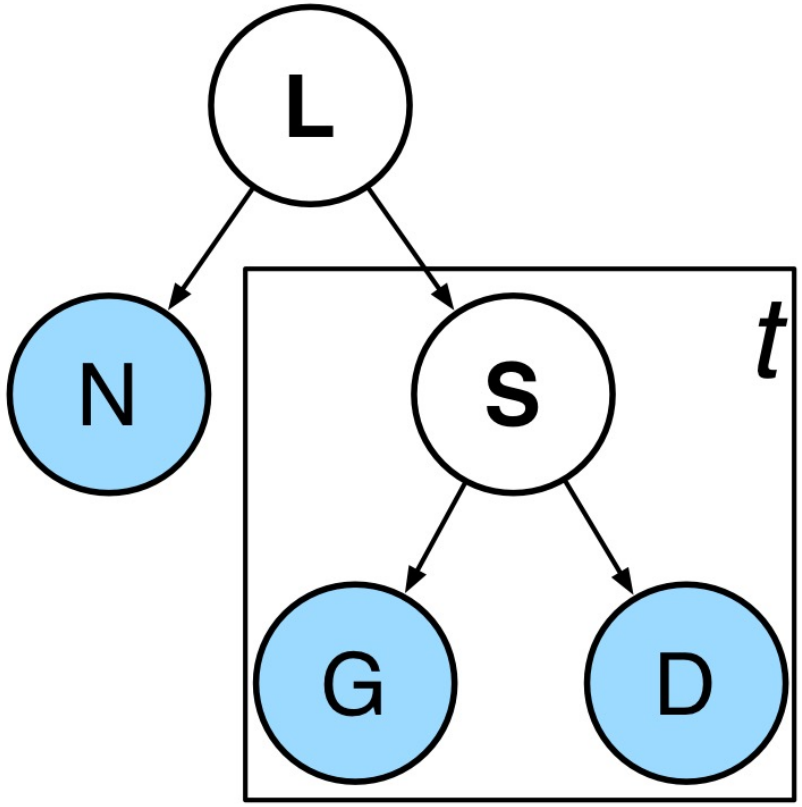


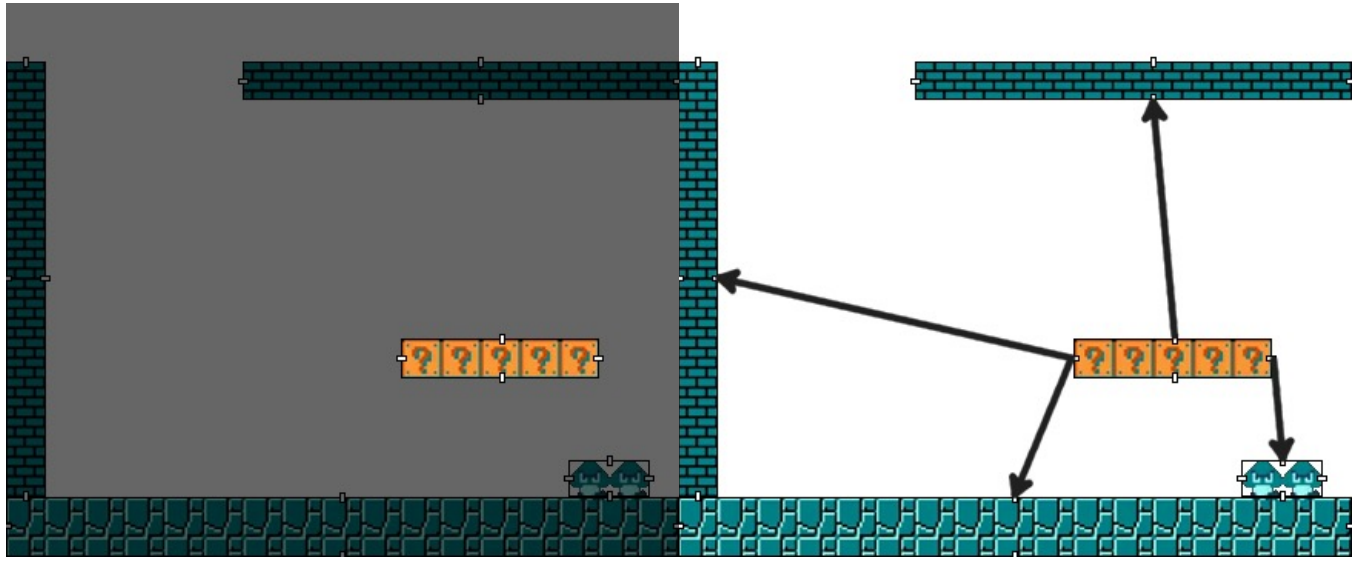




Kalogerakis et al. A Probabilistic Model for Component-Based Shape Synthesis 2017. https://people.cs.umass.edu/~kalo/papers/ShapeSynthesis/ShapeSynthesis.pdf

Bayesian Networks for Super Mario Bros.



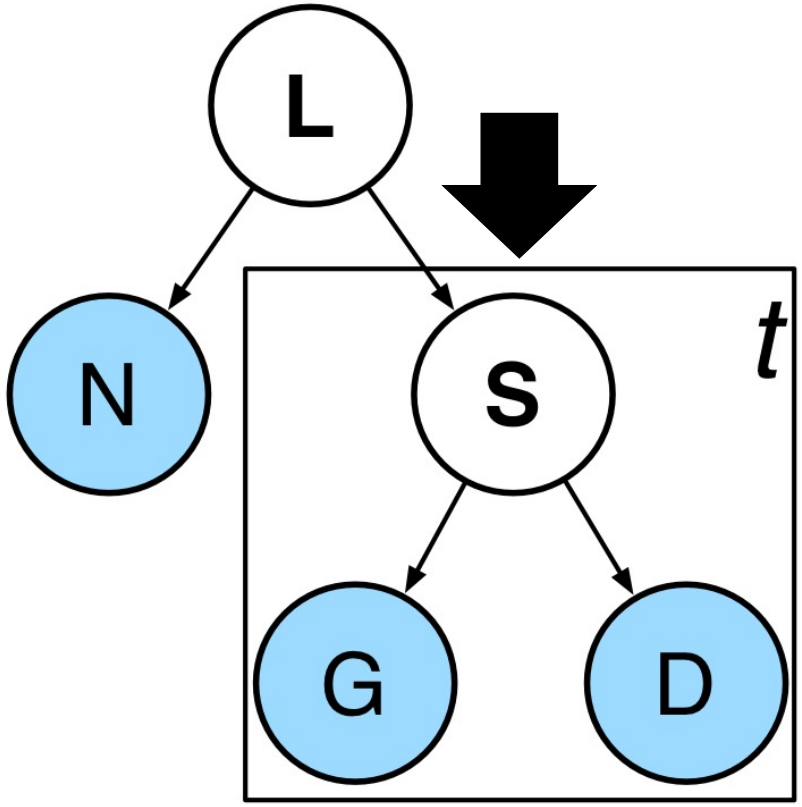


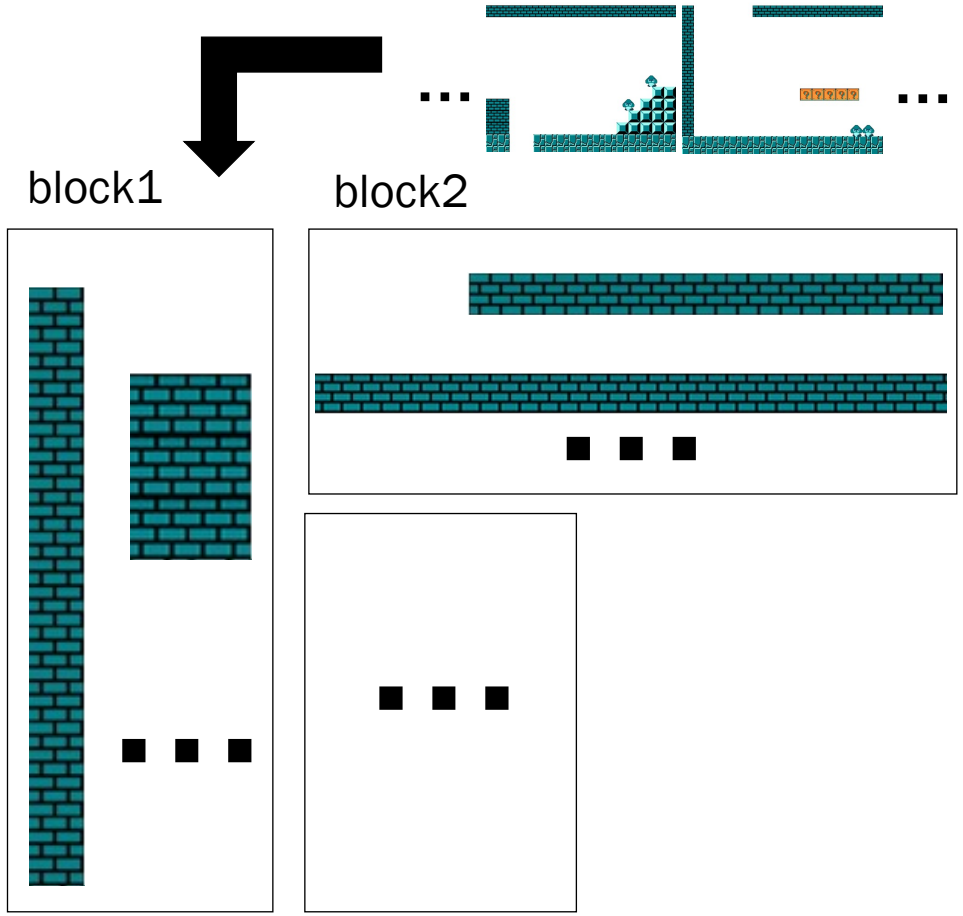
G node value

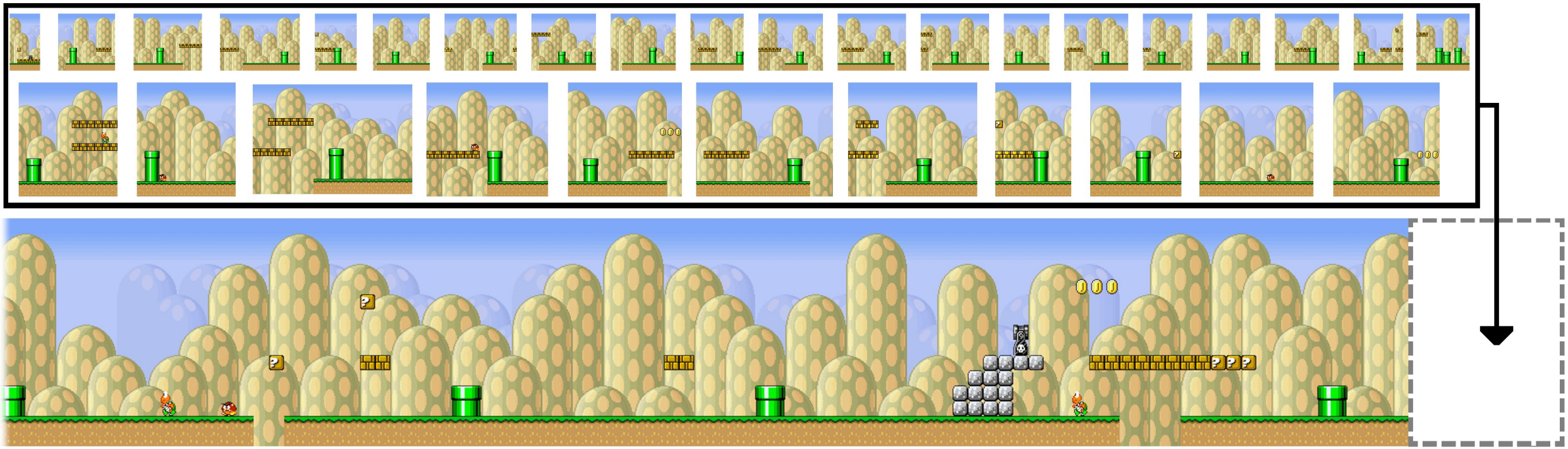
D node value

Guzdial and Riedl. Game Level Generation from Gameplay Videos. 2016.
https://www.aaai.org/ocs/index.php/AIIDE/AIIDE16/paper/download/14008/13593

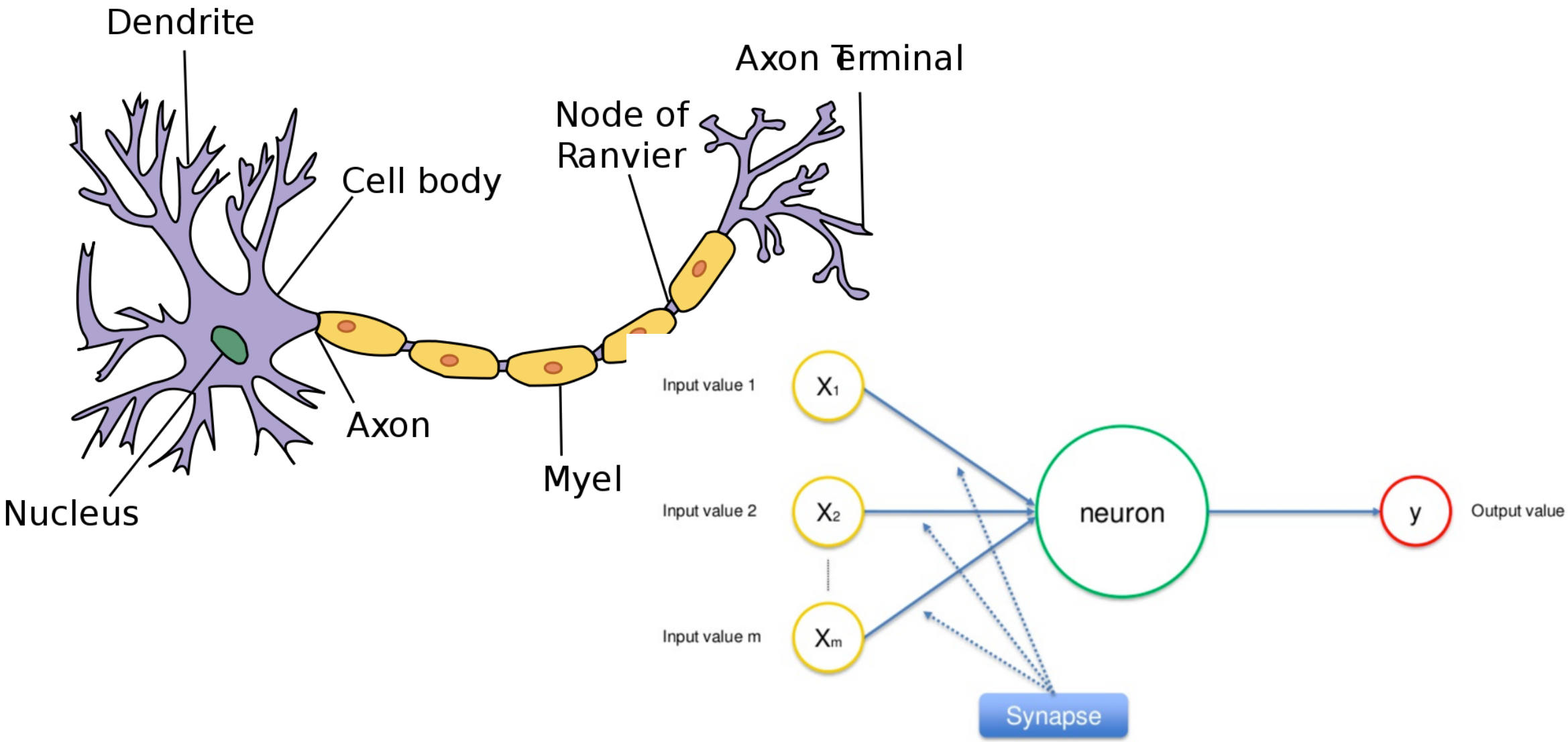
Bayesian Networks for Super Mario Bros.



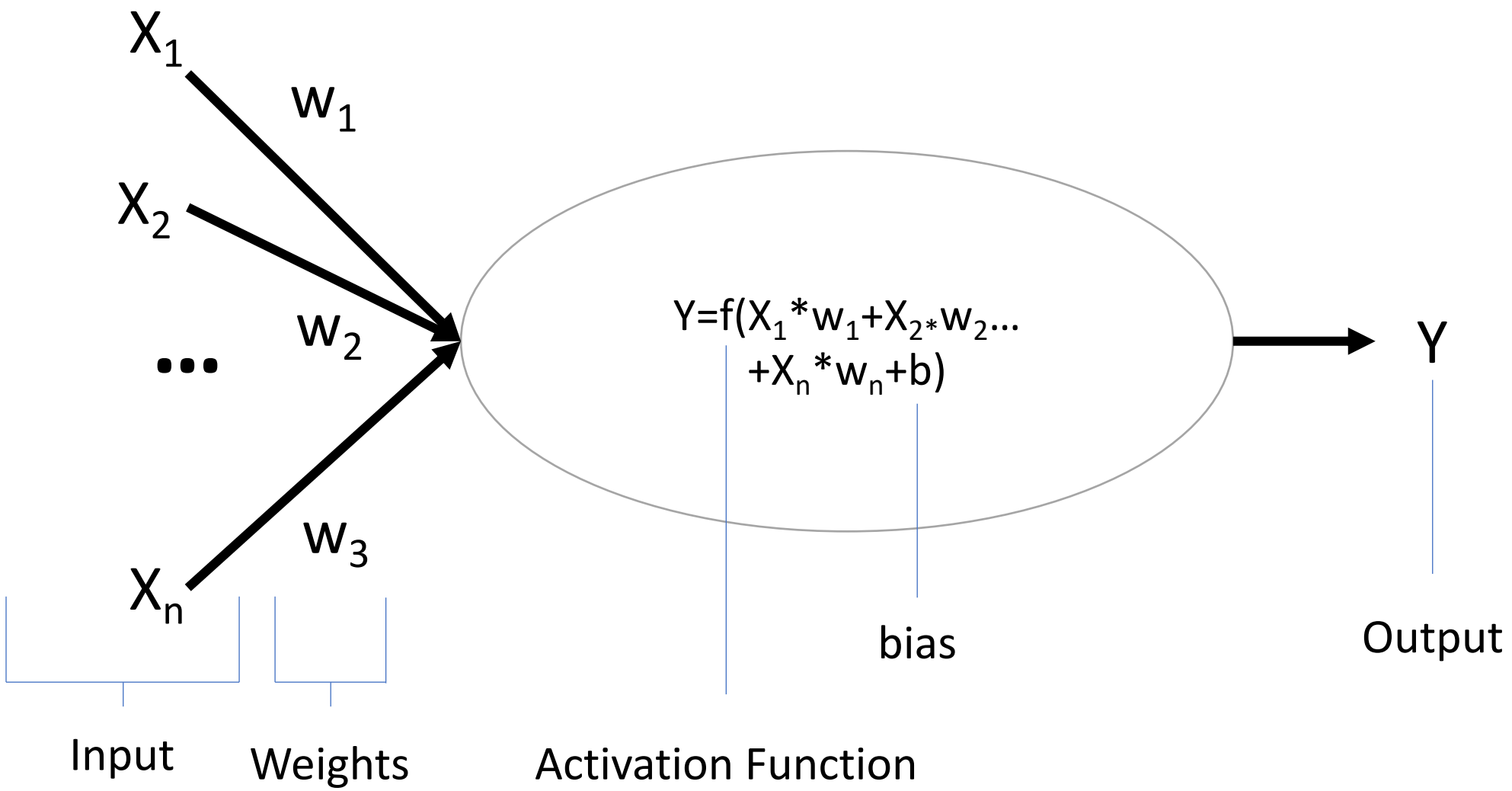




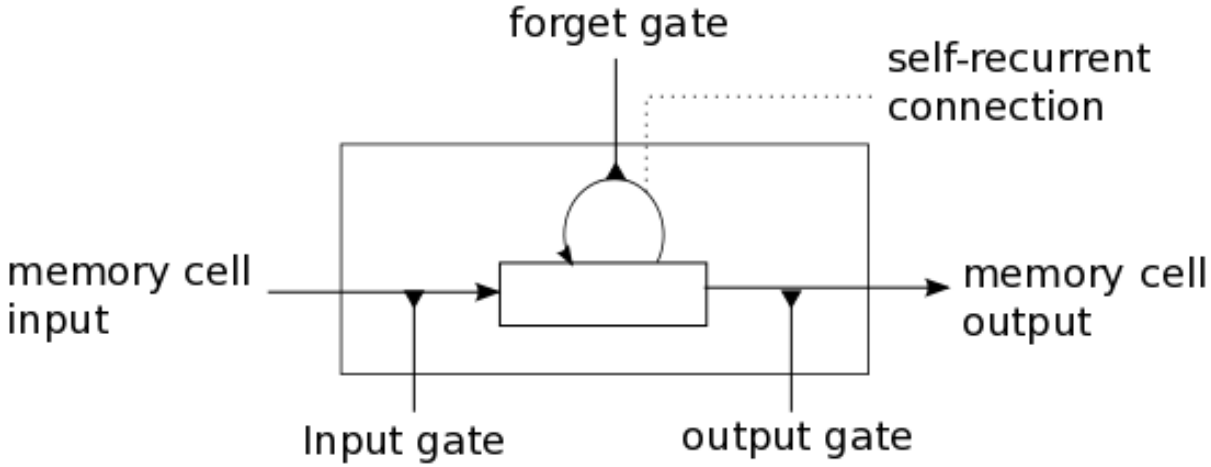
4-5 Deep Neural Networks: Lots of “Neurons”



Neuron Components

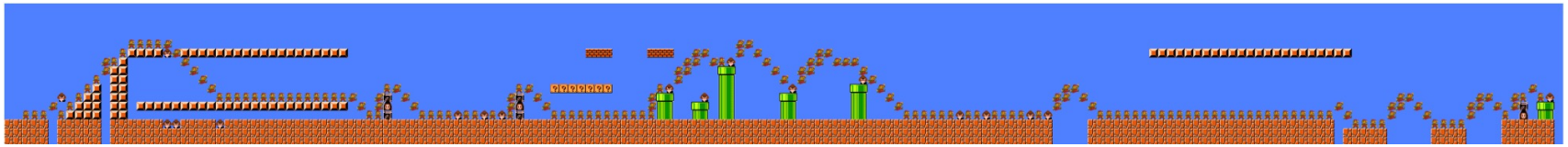


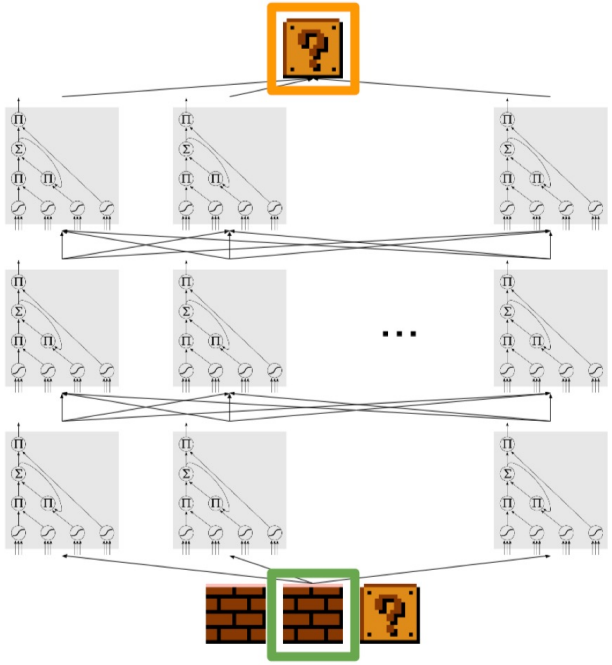
4. Long Short-Term Memory Recurrent Neural Networks (LSTMs)

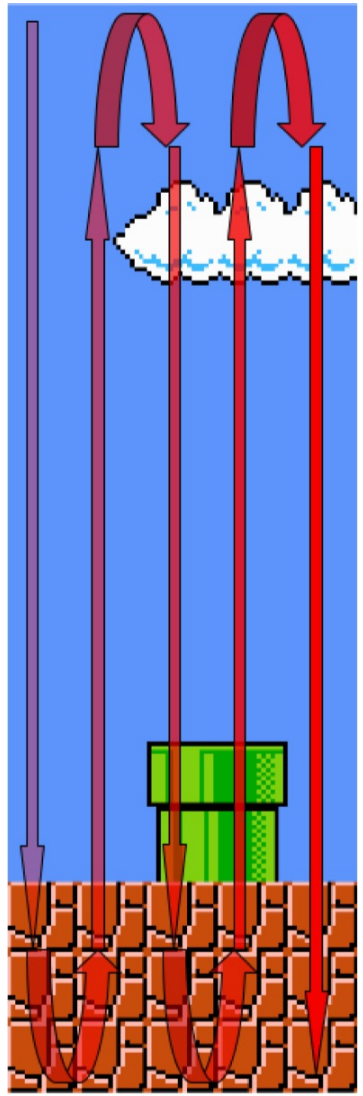


LSTMs (prior to Transformers) were the SOTA for sequence modeling, based on holding a particular index of an input sequence in memory for a learned number of indexes.

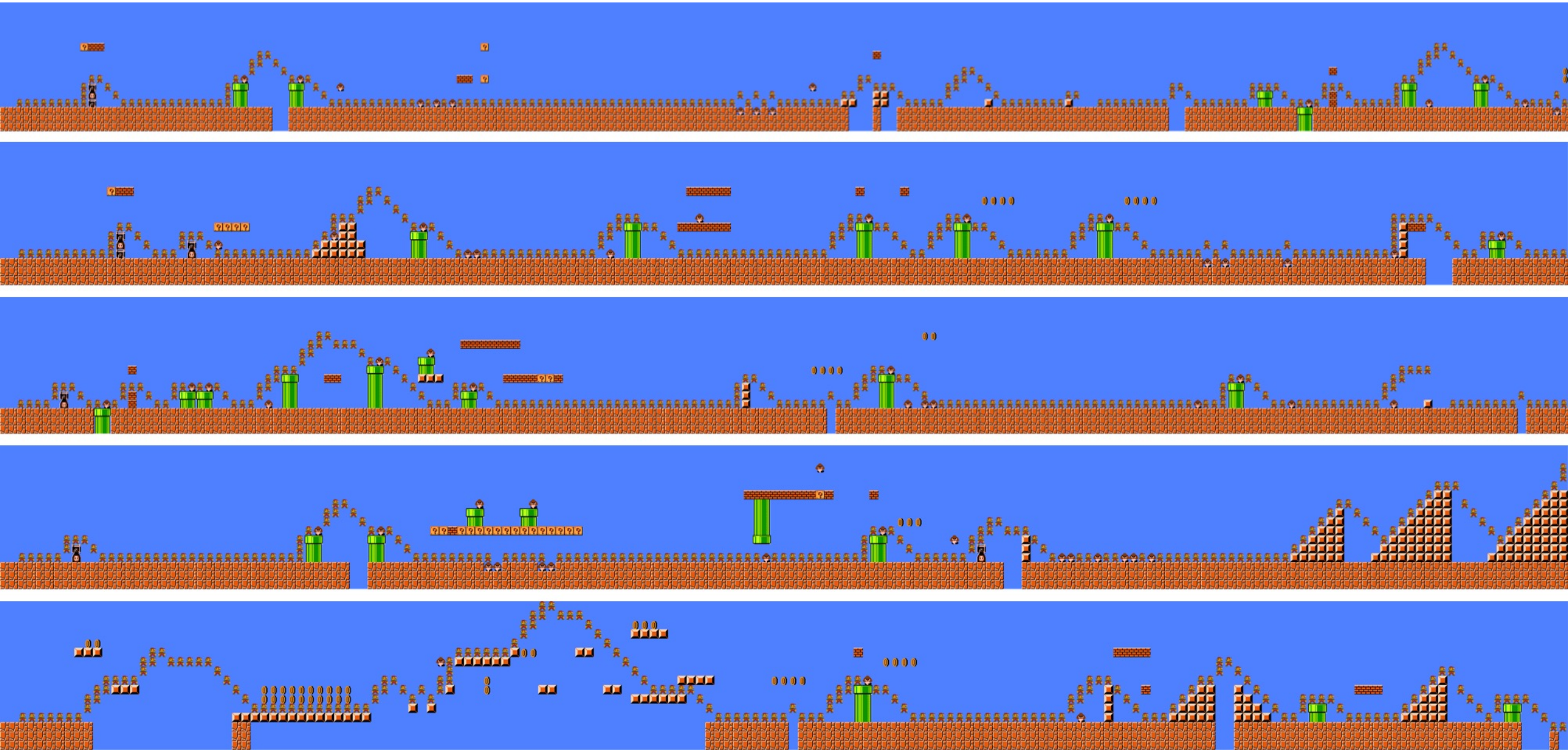
LSTM for Super Mario Bros.







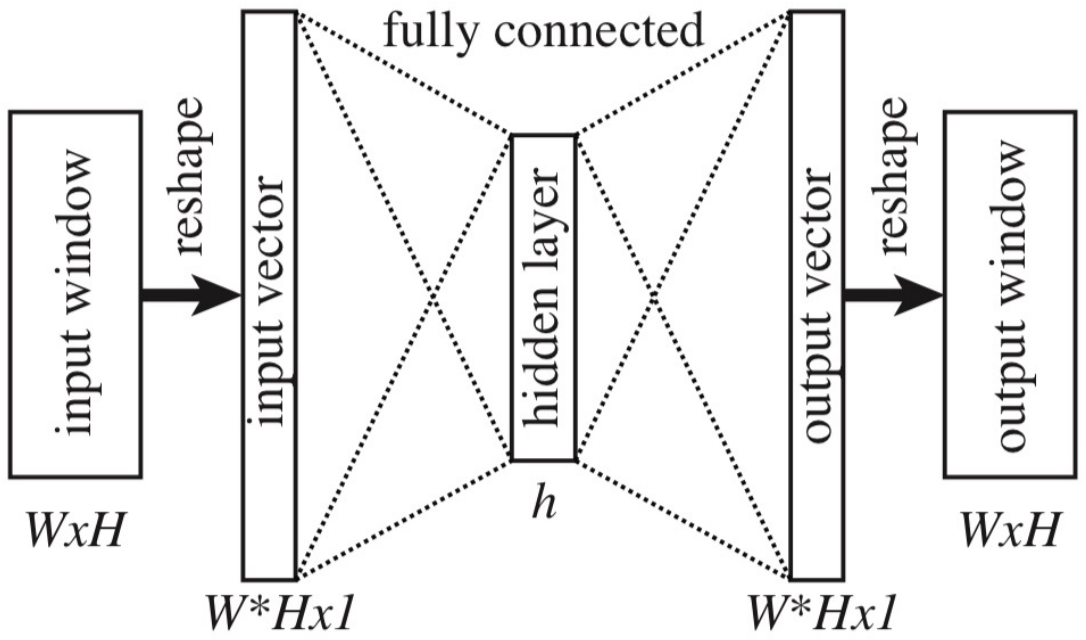
Summerville and Mateas. Super Mario as a String: Platformer Level Generation via LSTMS. 2016. https://arxiv.org/pdf/1603.00930.pdf



4.) Autoencoders

Pass in the same thing as input as output, but force it to squish down into an **embedding/hidden layer**.

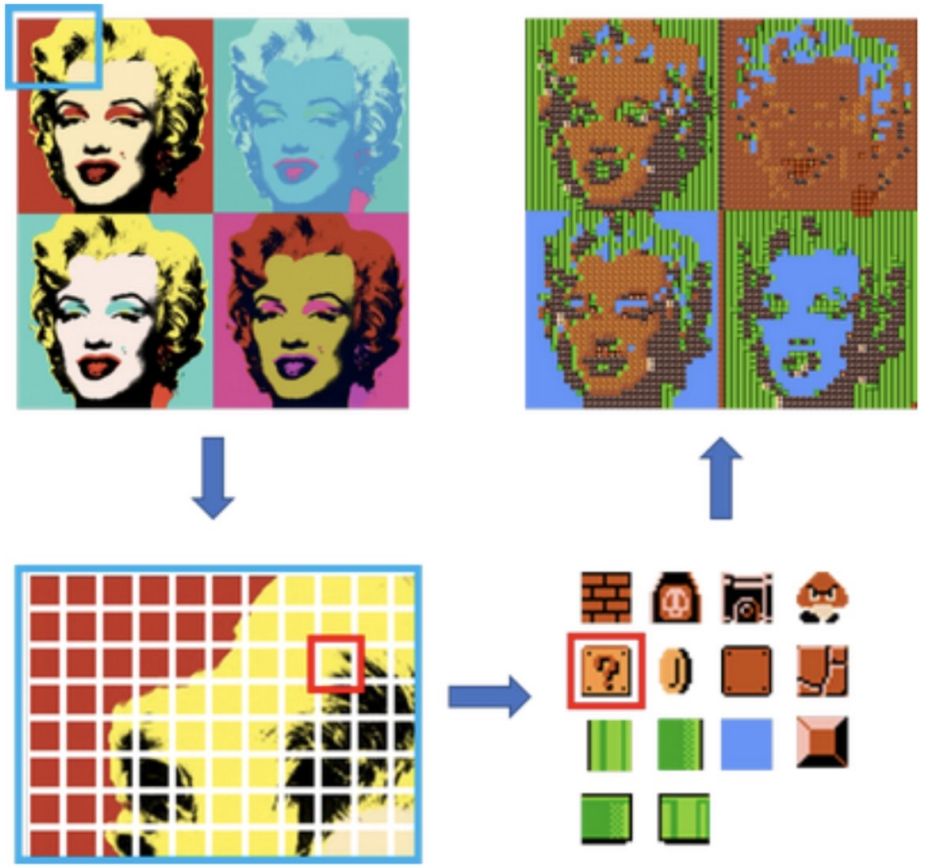
Embedding layer then defines a space that is initially random but comes to represent a space of possible outputs.



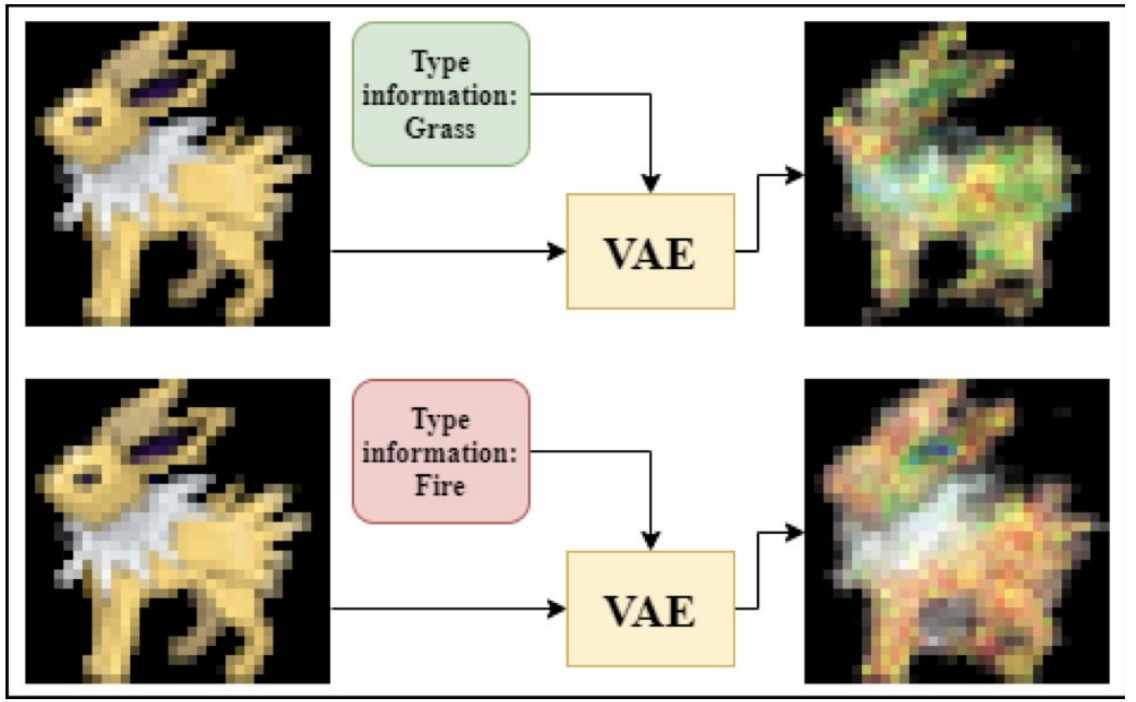
Autoencoders for Super Mario Bros.

Jain, Rishabh, et al. "Autoencoders for level generation, repair, and recognition." 2016.
http://julian.togelius.com/Jain2016Autoencoders.pdf

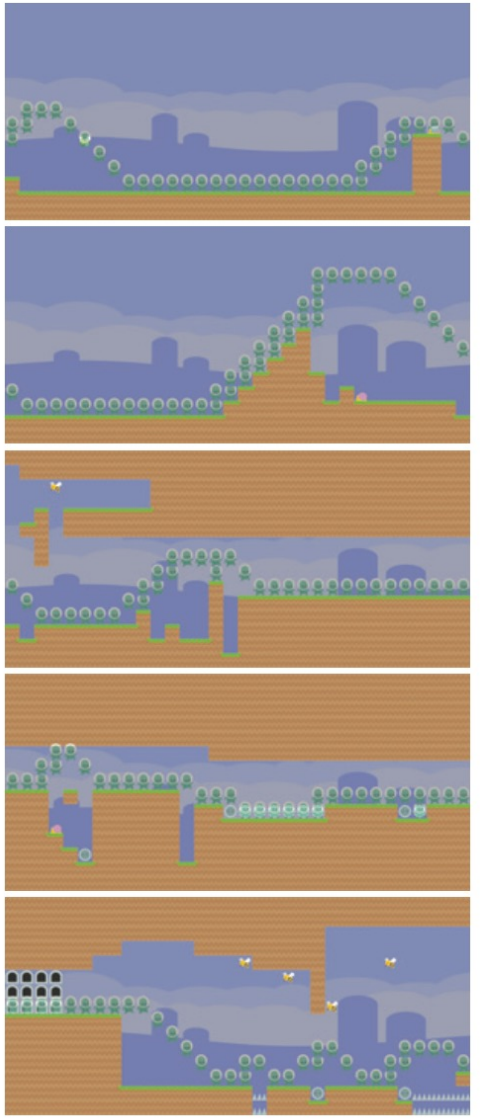
Lots of recent autoencoder work



Chen, Eugene, et al. "Image-to-Level: Generation and Repair."
http://imagetolevel.com



Gonzalez et al. “Generating Gameplay-Relevant Art Assets with Transfer Learning”
https://arxiv.org/pdf/2010.01681.pdf



(b) GRU SMB ↓ Met

Sarkar et al. “Exploring level blending across platformers via paths and affordances”
https://www.aaai.org/ojs/index.php/AIIDE/article/download/7442/7282

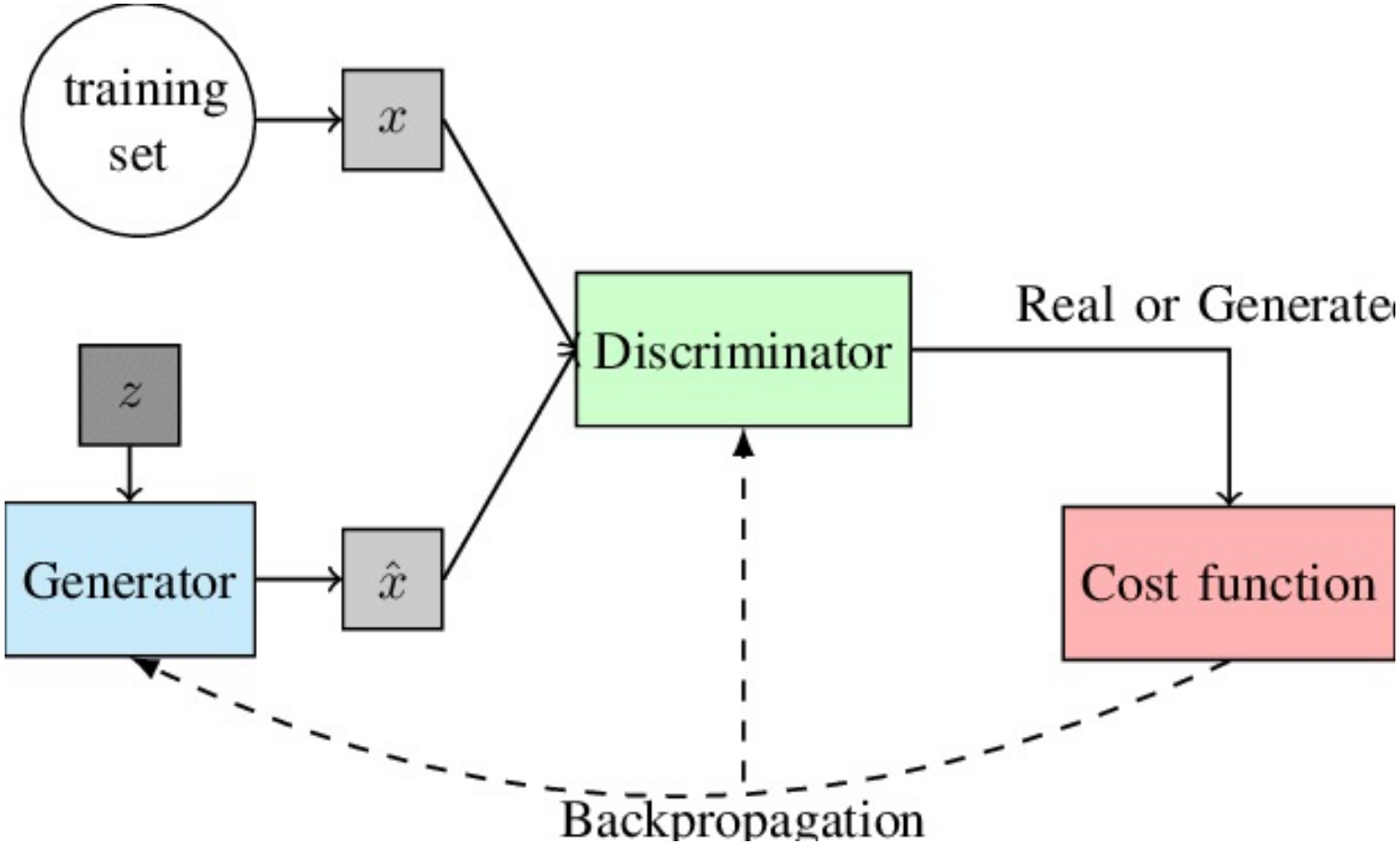
5. Generative Adversarial Networks (GANs)

GANs have two parts, a Discriminator and a Generator.

The generator produces fake output.

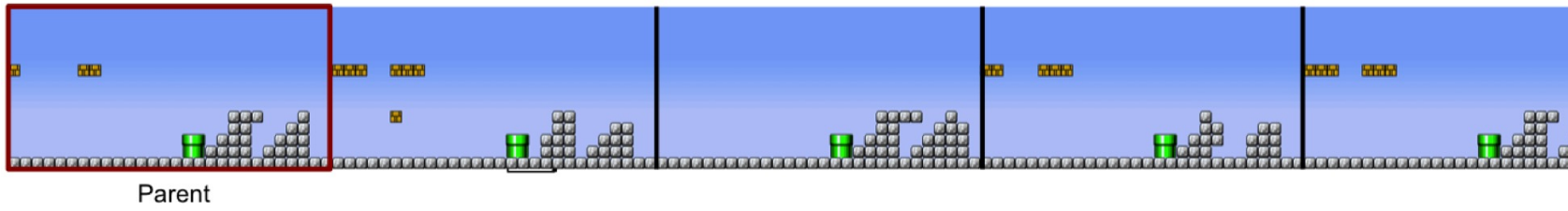
The discriminator tries to tell the difference between real and fake.

The generator gets rewarded when it fools the generator.

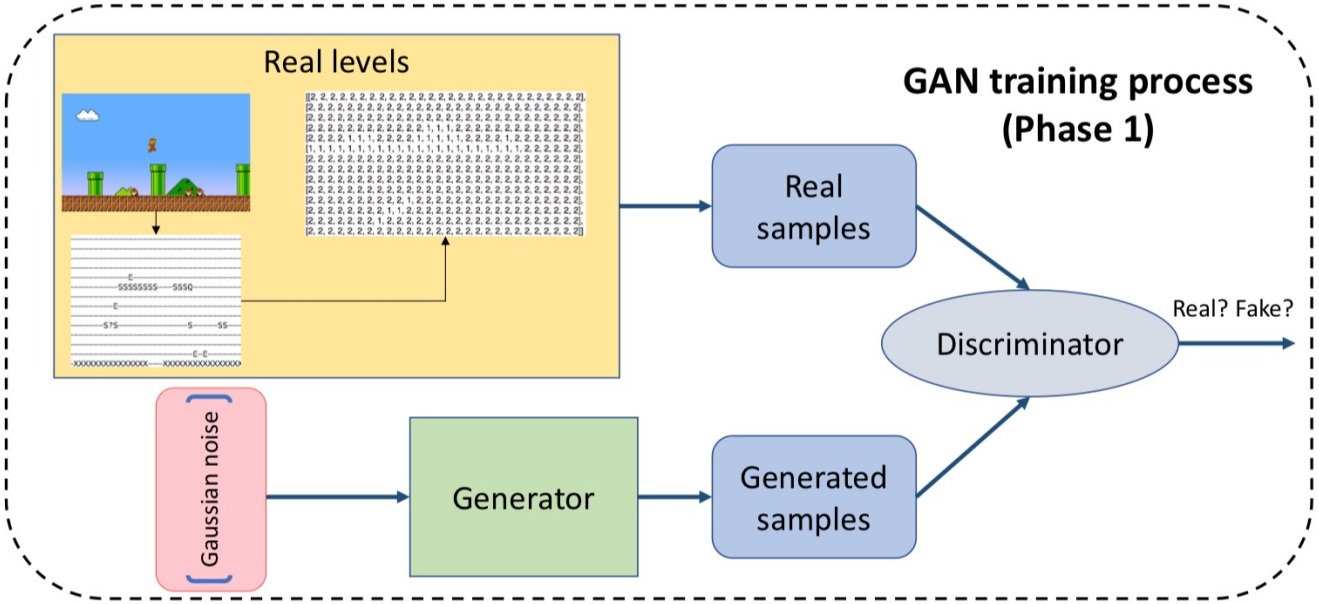


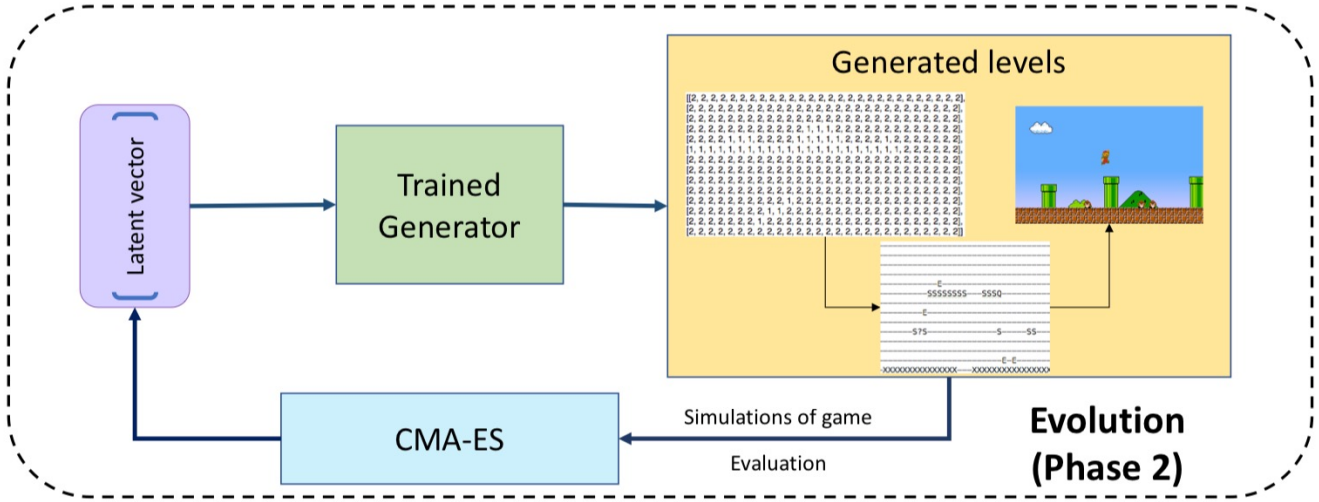
GANs typically require *massive* datasets, how do we get around this?

GAN (+Genetic Algorithms) for Super Mario Bros.

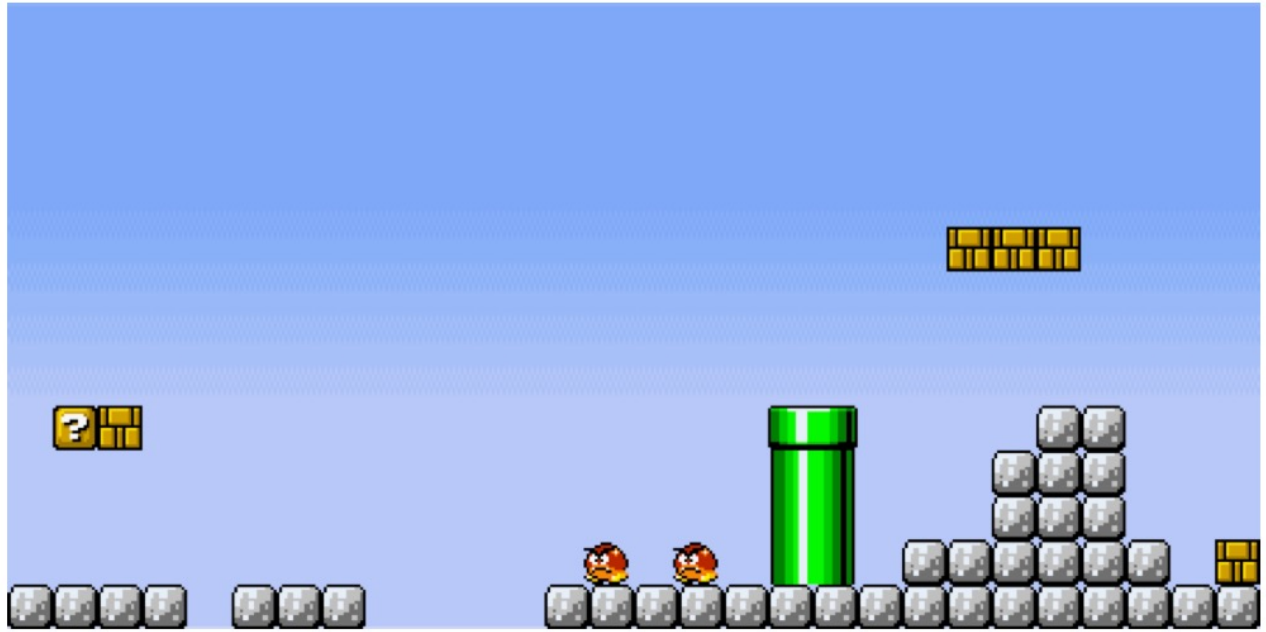


(b) Mutations

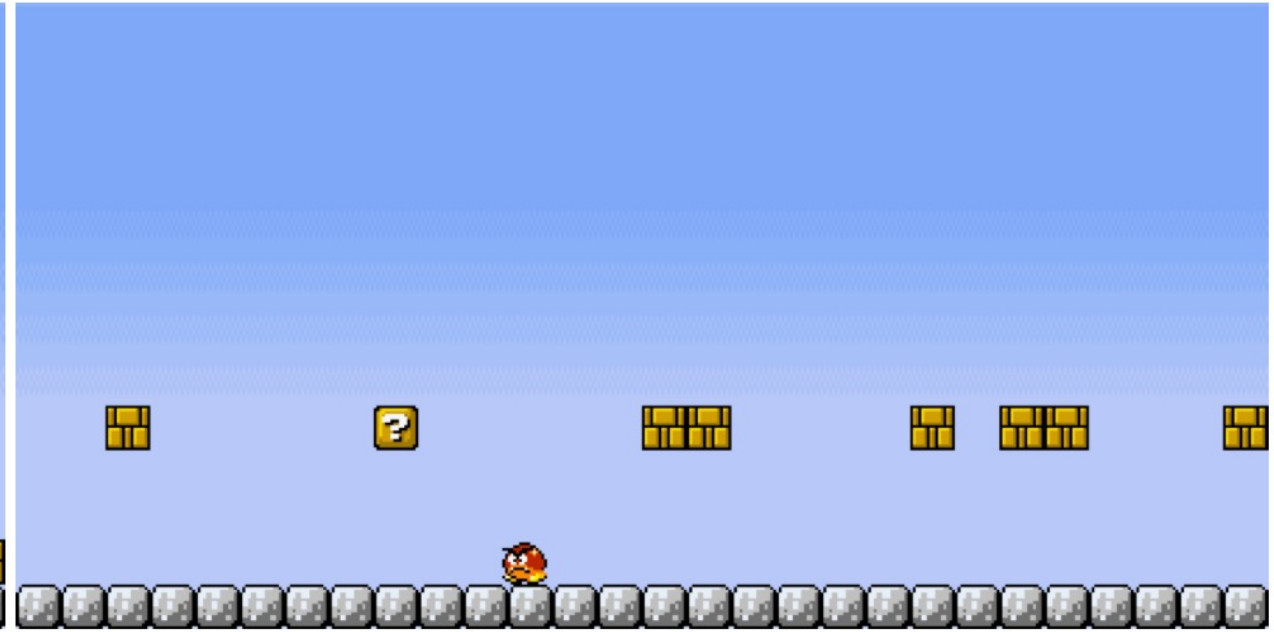




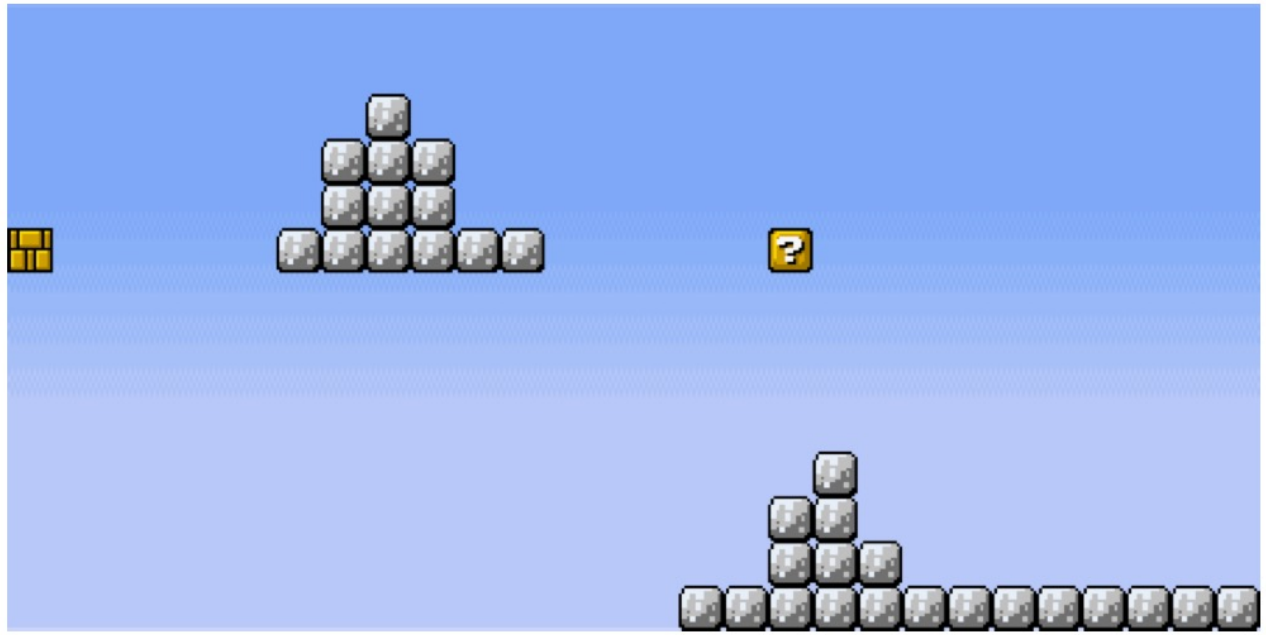
Volz et al. “Evolving Mario Levels in the Latent Space of a Deep Convolutional Generative Adversarial Network” 2018. https://arxiv.org/pdf/1805.00728.pdf



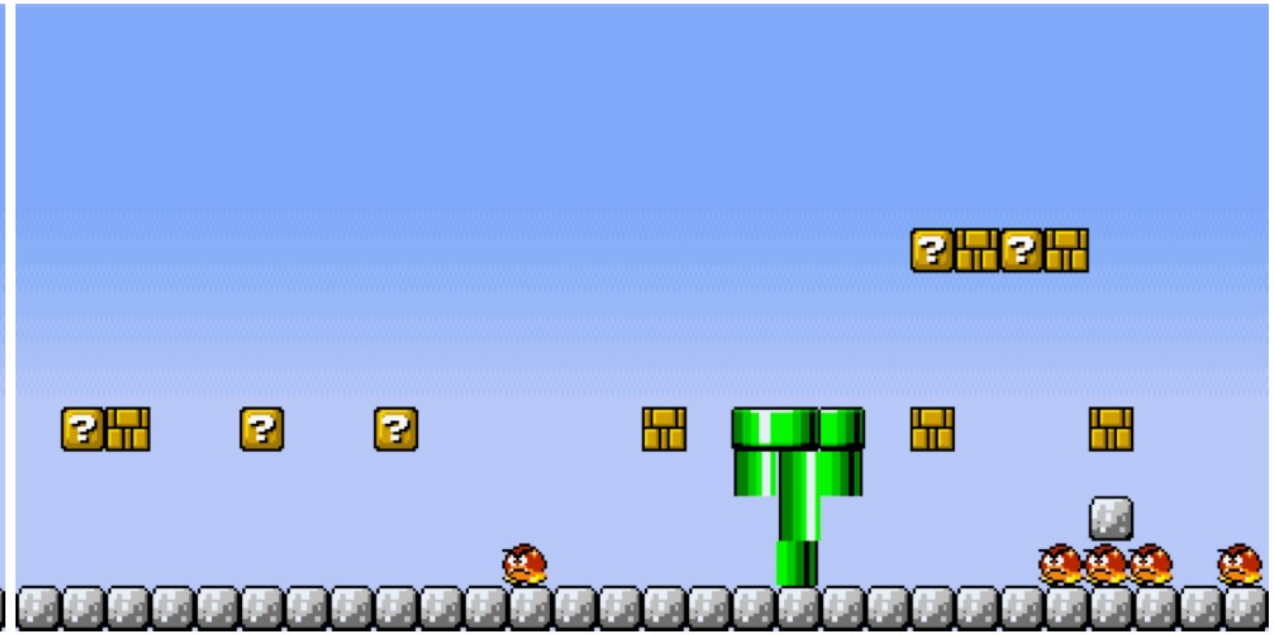
(a) Playable level maximizing jumps



(b) Playable level minimizing jumps



(c) Unplayable level



(d) Broken titles

PQ1 https://forms.gle/bghnjonecy9A7c7p9 https://tinyurl.com/guz-pq31a

Pick one of the approaches introduced today.

Can you think of a situation where you would use it over some other PCG approach we discussed previously?

Quiz 5 Review

- **Topics to be able to run**: PCG Intro (constraint-based PCG example), Search-based PCG (Greedy, Simulated Annealing, GA), Constructive PCG (Grammars and Noise), WFC (WFC)
- **Topics to remember**: Spatial Representations, Path planning, Decision Making, and Player Modelling