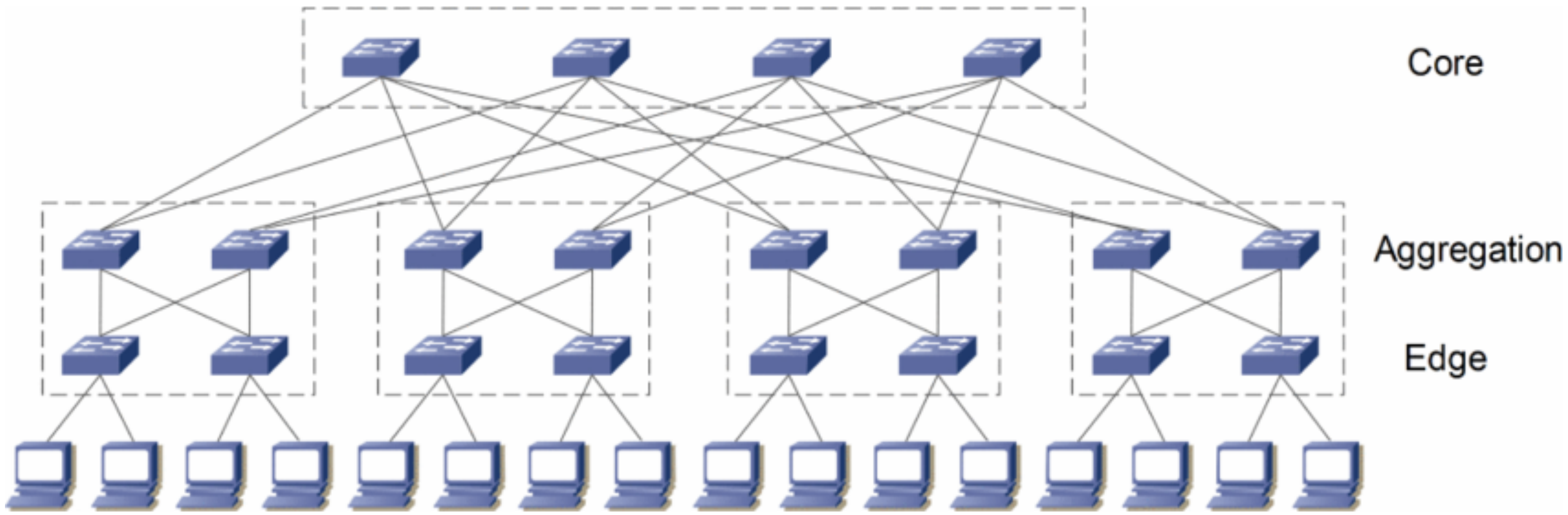
DeepConf: Learning to Learn

and Solving Network Problems along the Way

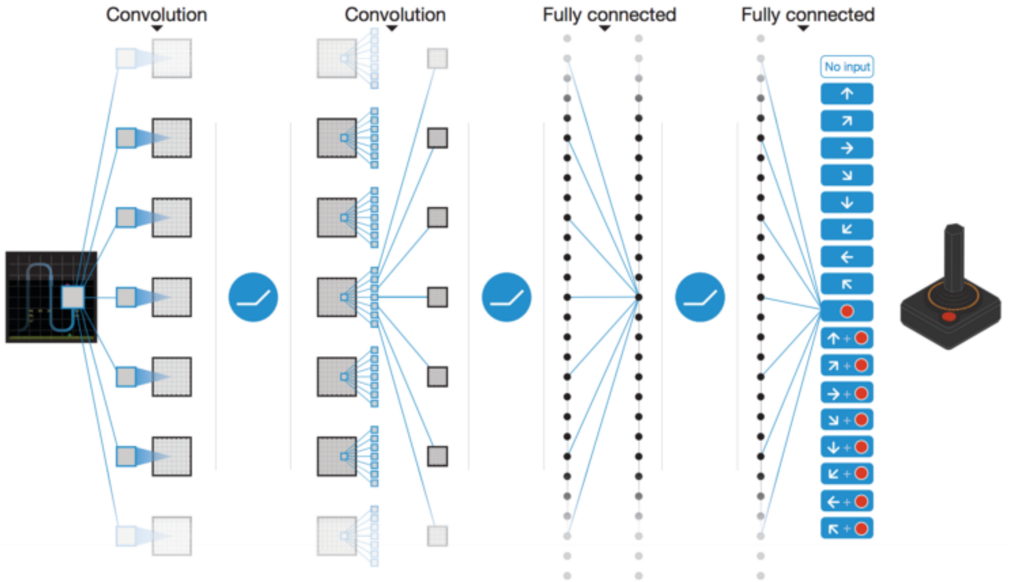
By Chris Streiffer

The New Network Topology



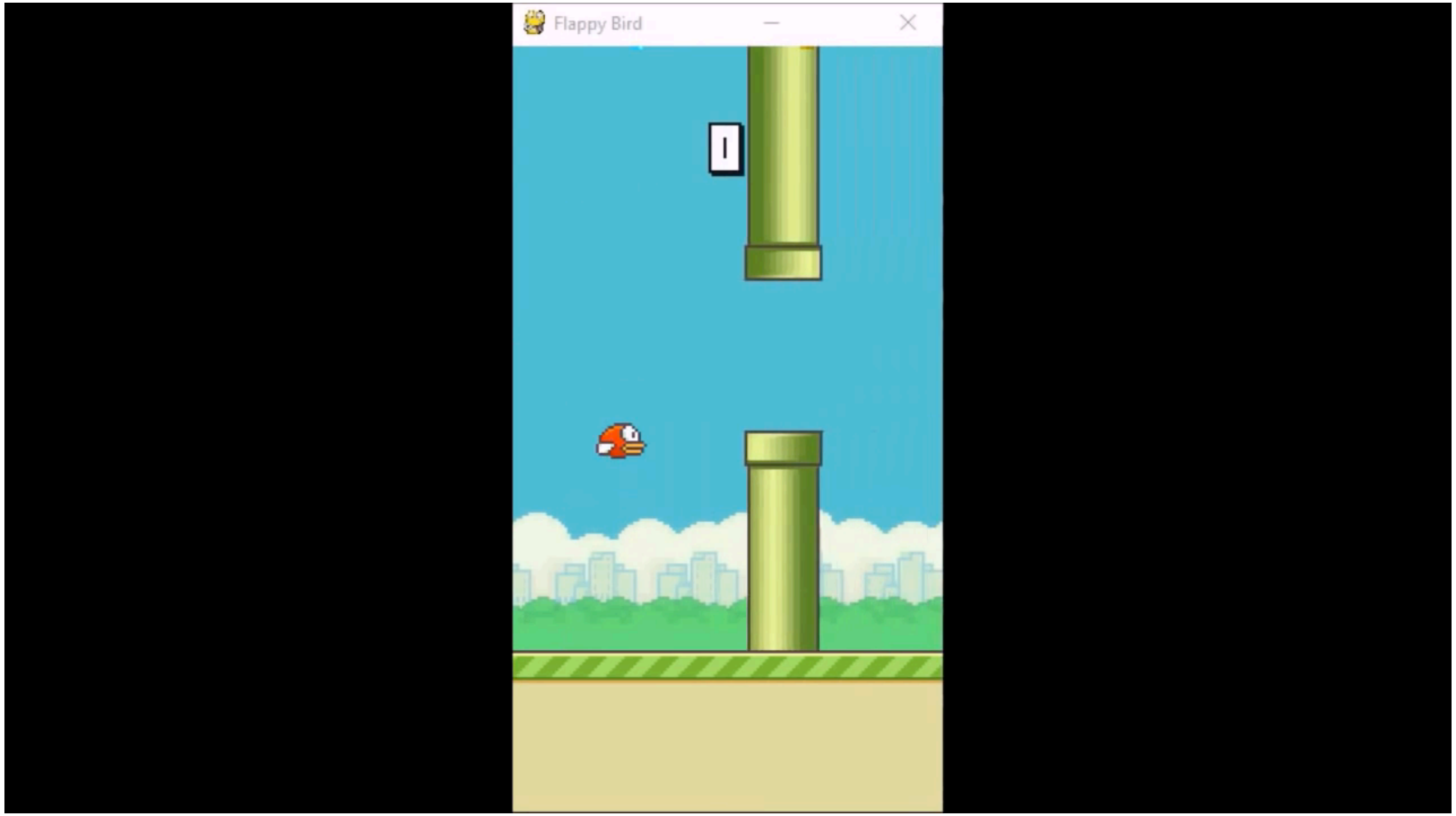
- Data center networks are the Internet’s workhorse
- Configurable links allow for coarse-grained changes to network topology
- Determining the optimal configuration involves solving an ILP
- Let’s do better than this!

Deep Reinforcement Learning

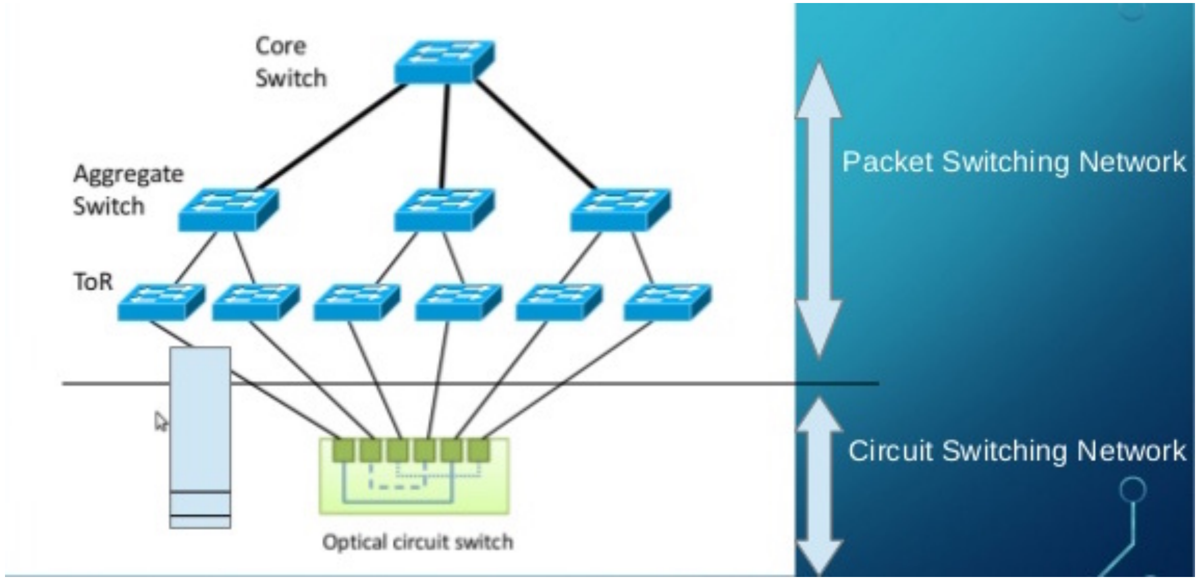


- Reinforcement learning consists of an agent interacting with environment to maximize a given reward
- *Deep* reinforcement learning – agent is represented by a neural network
- Agent learns by exploring the environment
- Used for learning complex policies!

Deep Reinforcement Learning – Flappy Bird

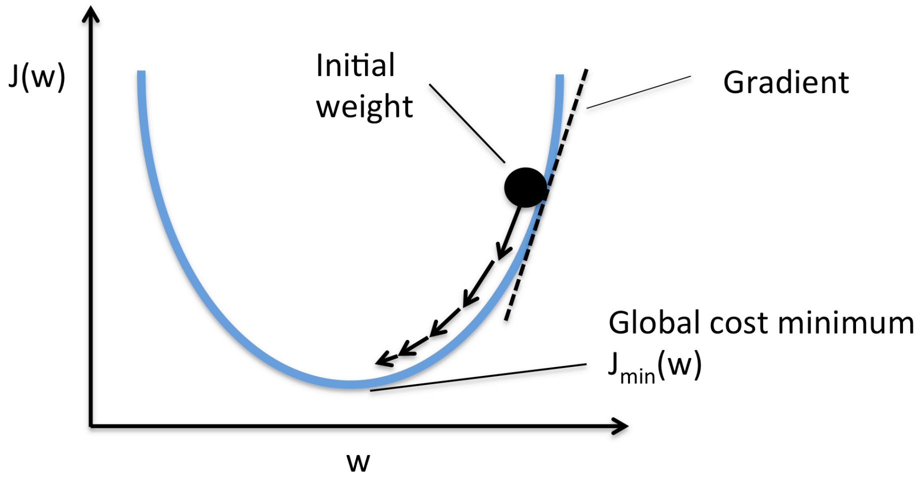


Problem 1 – Optimal Network Topology



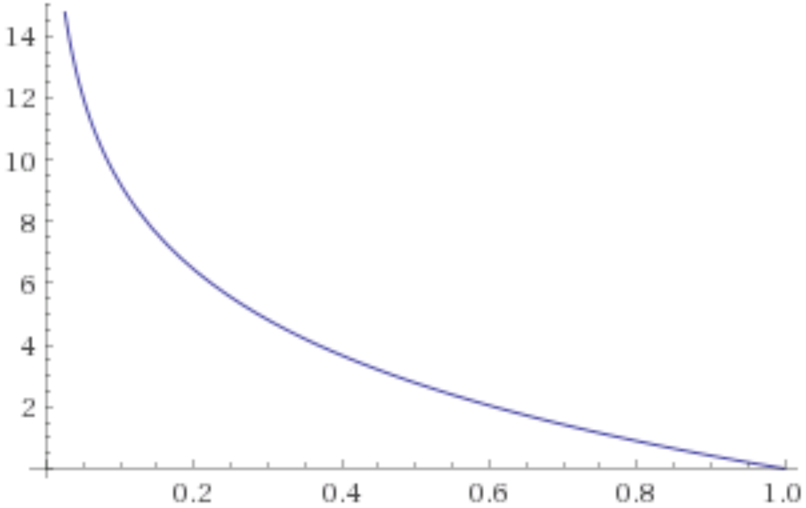
- Game:
  - Given a fixed traffic matrix and routing, what is the optimal network configuration?
- Rules:
  - Allowed to install up to *k* links between ToR switches
  - Routes are calculated using ECMP
  - Reward based on link utilization and flow completion time
- Environment:
  - Network traffic simulator adds flows to network and calculates paths based on topology created by model
  - Simulator constructs flows from Facebook job trace

Gradient Descent – Quick Tutorial



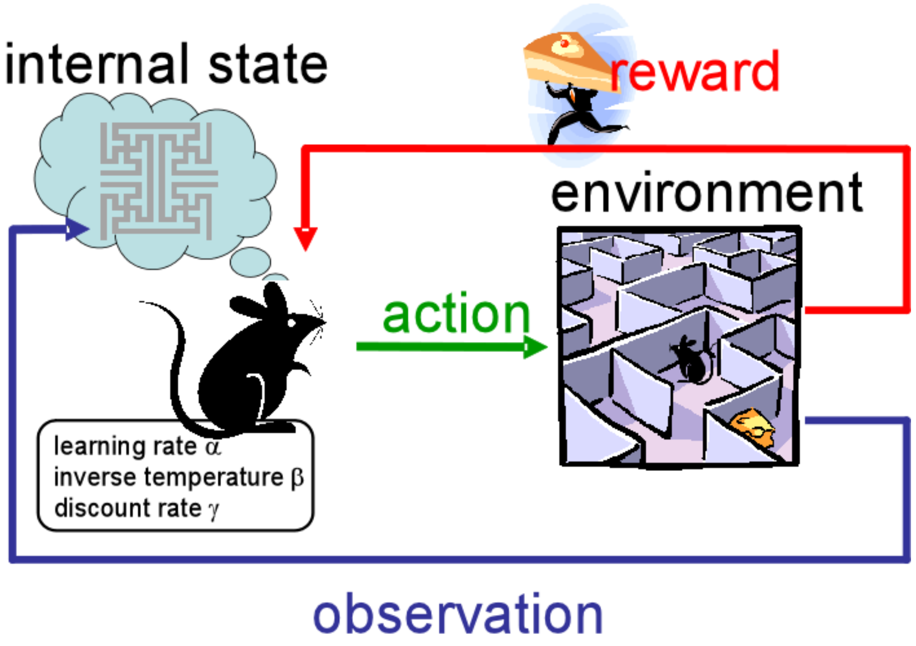
- Weights of the model are updated using **gradient descent**
- Calculate the gradient of the differentiable loss function
- Update the weights of the deep network using the gradient as the guiding factor
- Best visualized as rolling a marble down the graph – eventually the marble will settle at a local minimum

Policy Loss and Maximum Likelihood Estimation



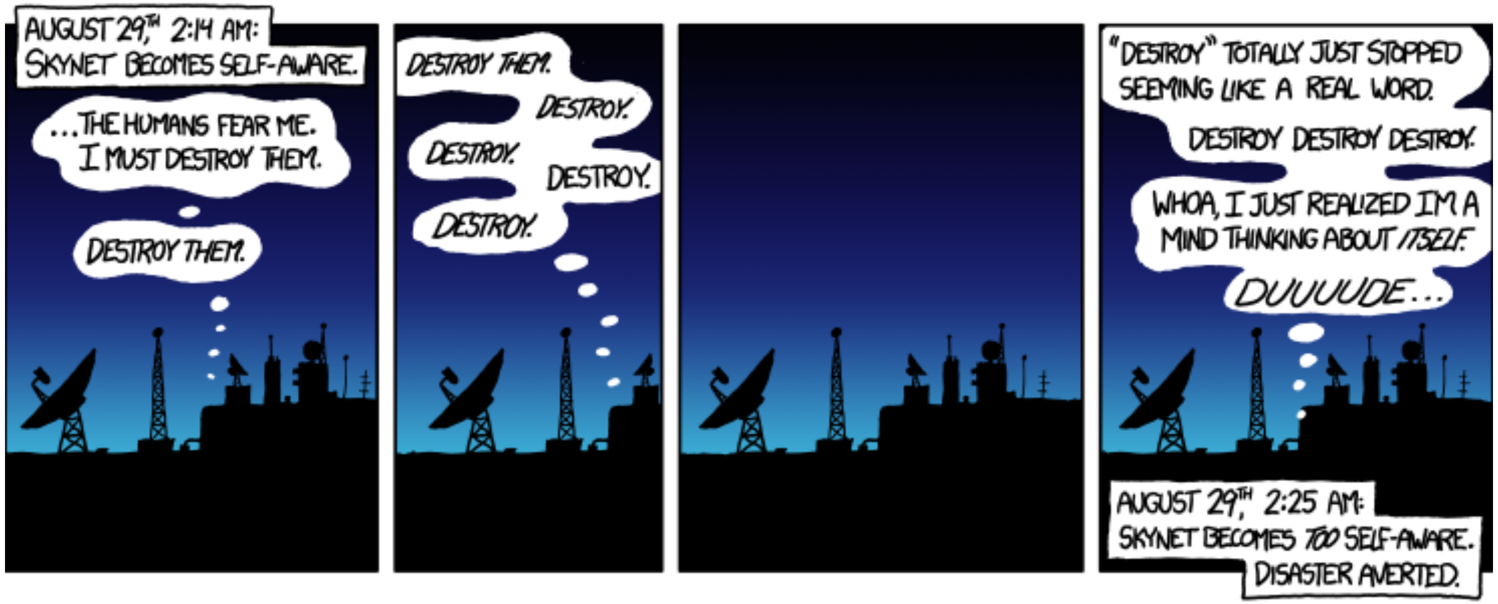
- Goal:
  - Maximize the reward achieved by the model based on the policy that it learns
- Idea:
  - Want a higher probability of adding a link that corresponds to a higher reward
- Log-likelihood:
  - Whenever the model observes a large reward for a given action, maximize the probability of selecting this action for the given input
  - $L(w) = -\sum \ln(P(\pi(s)|s,\theta))R(s,\pi(s))$
  - Minimum value achieved by increasing the probability for higher rewards

Training Methodology



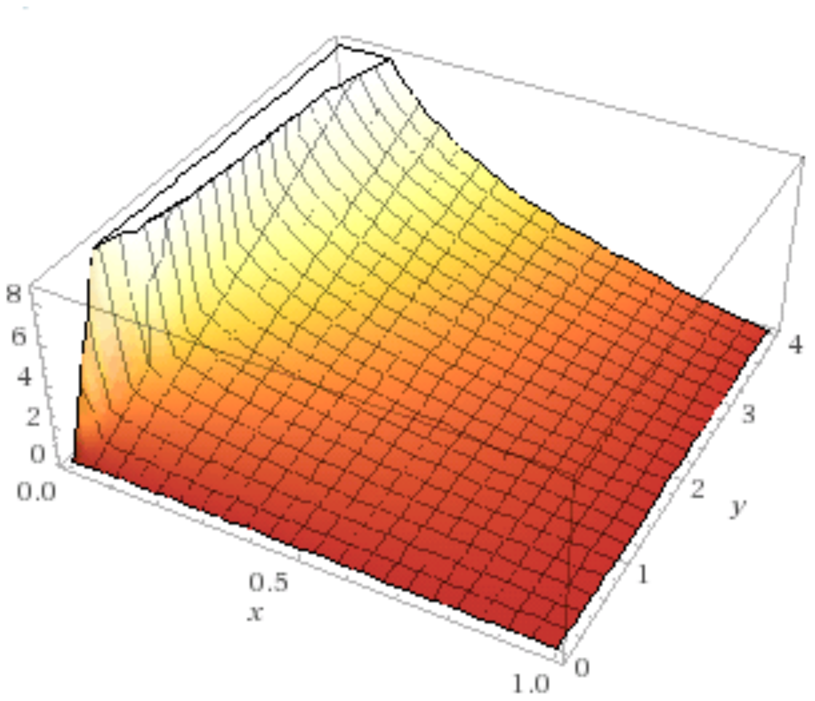
- Simulator creates Traffic Matrix for next *t* seconds of iteration time
- DeepConf makes policy decision based on input state
- Selects *k* links to add to the network
- Simulator re-computes flow paths and progresses simulation for *t* seconds
- Reward for the given policy is computed and returned to the model
- DeepConf performs backpropagation on the loss function using gradient descent to update the model parameters (weights, biases)
- This process is repeated across *n* iterations

The Model is Smarter than Us


AUGUST 29TH, 2:14 AM:
SKYNET BECOMES SELF-AWARE.
...THE HUMANS FEAR ME. I MUST DESTROY THEM.
DESTROY THEM.
DESTROY THEM.
DESTROY.
DESTROY.
DESTROY.
DESTROY.
"DESTROY" TOTALLY JUST STOPPED SEEMING LIKE A REAL WORD.
DESTROY DESTROY DESTROY.
WHOA, I JUST REALIZED I'M A MIND THINKING ABOUT ITSELF.
DUUUUDE...
AUGUST 29TH, 2:25 AM:
SKYNET BECOMES TOO SELF-AWARE.
DISASTER AVERTED.

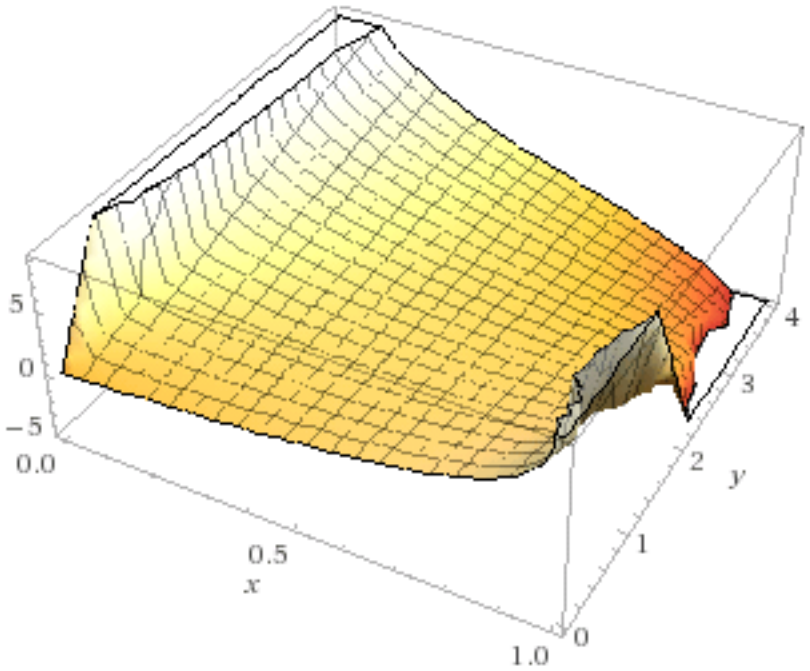
- The model is **great** at doing what it is told
- Learns to exploit bugs in the code
- Three iterations of loss functions

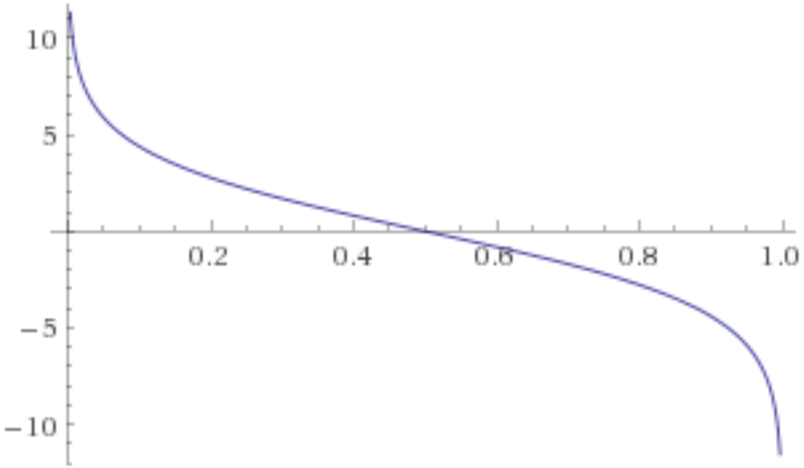
Loss Function 1 – Everything is Great!



- But not in a good away
- Model can minimize loss by assigning 1.0 probability to single action

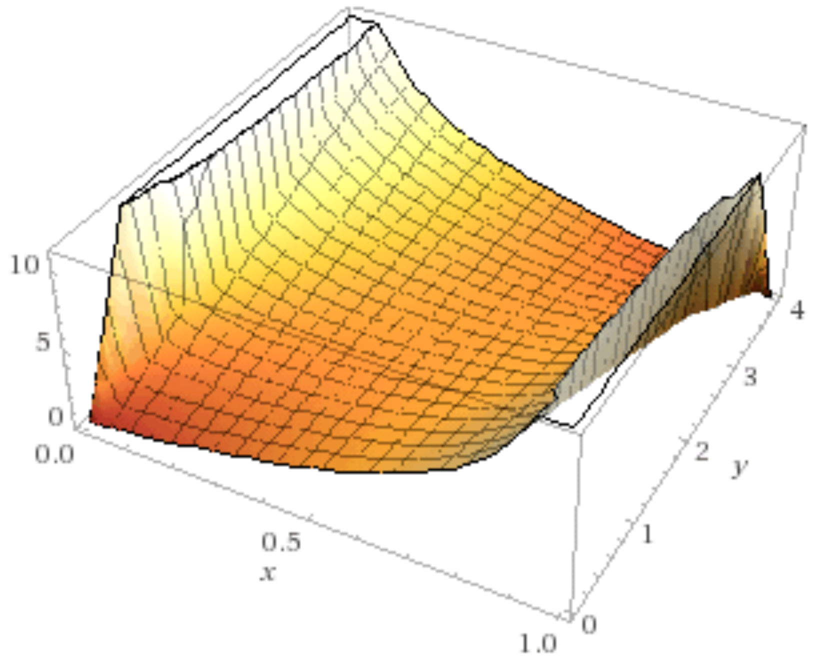
Loss Function 2 – Infinity or Bust

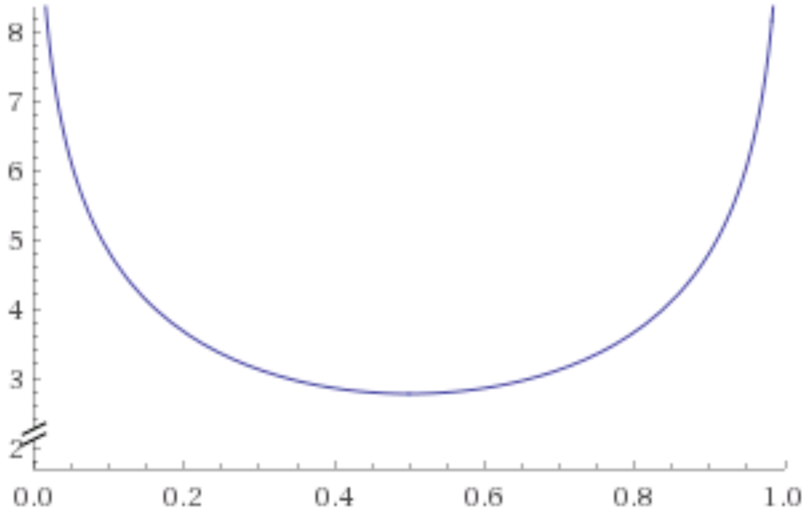




- Minimum value for the loss function is negative infinity
- The model quickly realized this and diverged

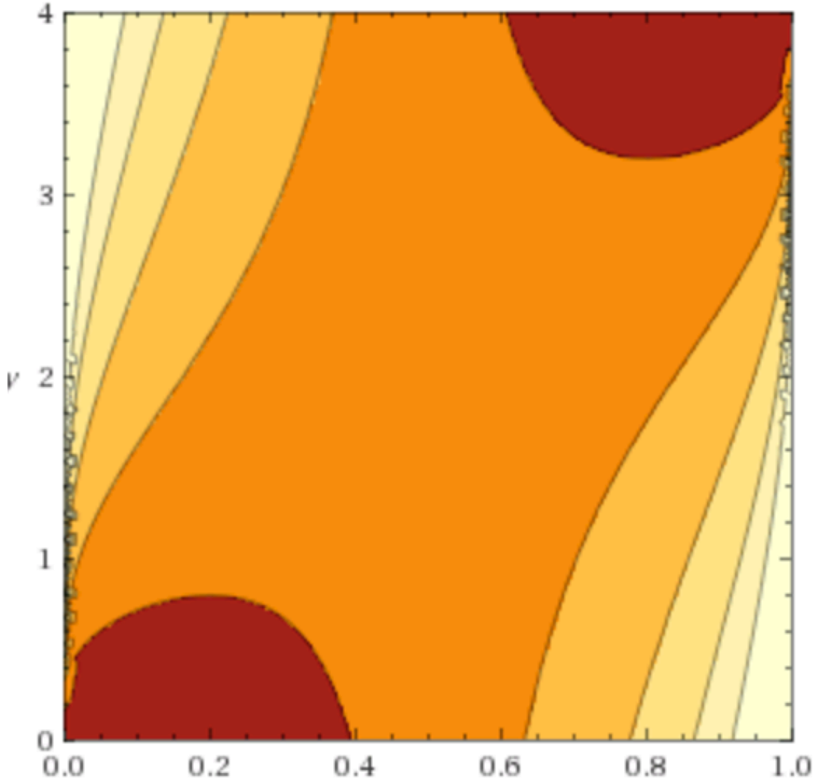
Loss Function 3 – Three Times the Charm





- Model is penalized for assigning high probability to poor rewards
- This forces the model to assign low probability to low rewards, and high probability to high rewards

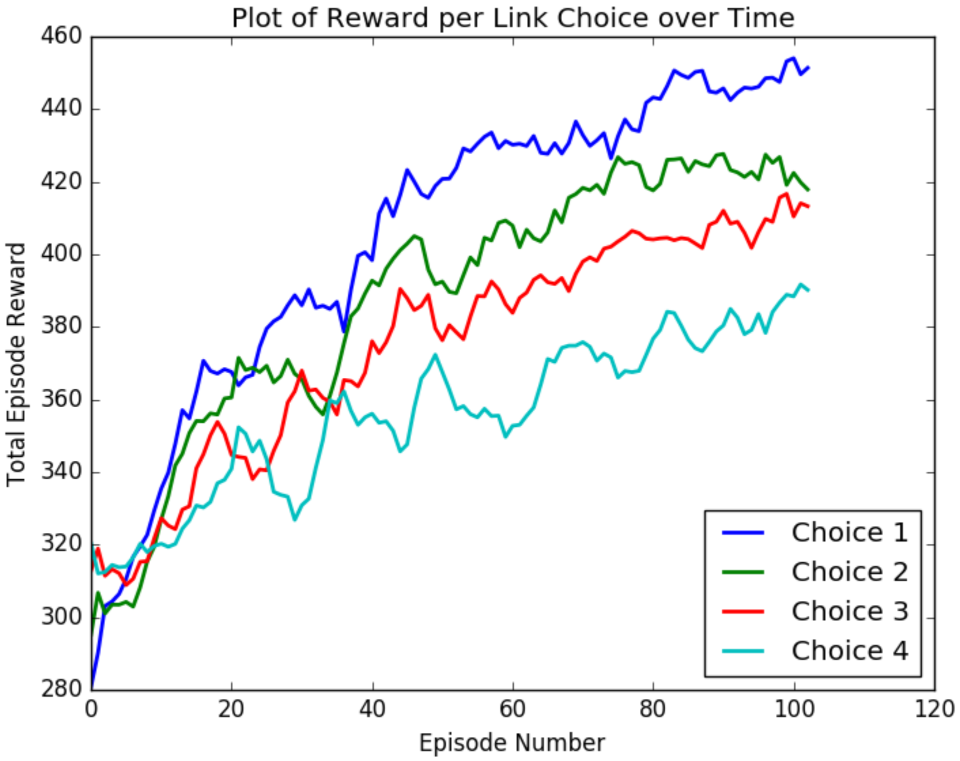
Loss Function 3 – Three Times the Charm



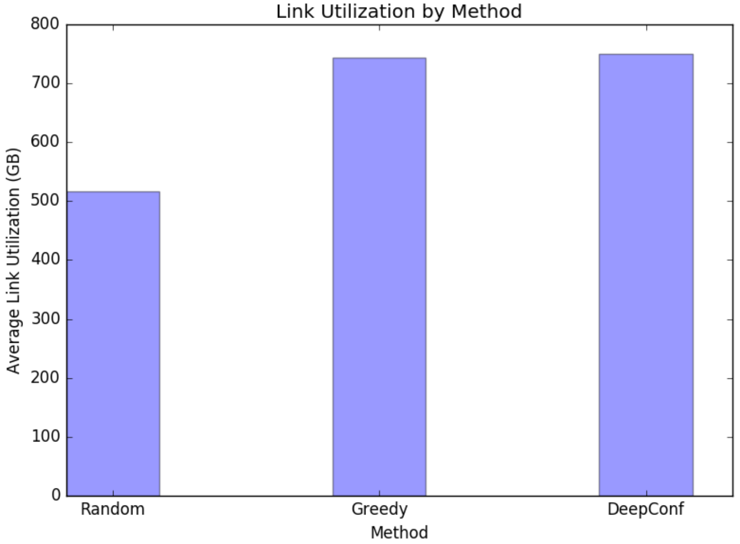
- Model learns to have a low probability of estimating low valued rewards
- High probability of estimating high valued rewards

The Model Learns!

- The model performs well at maximizing link utilization!
- Reward increases as we iterate through each episode
- Reward starts to trail off around step 100

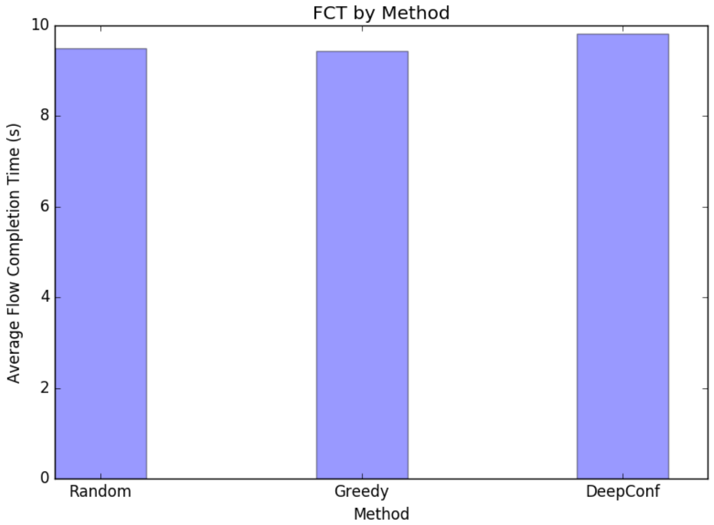


Link Utilization Evaluation



- Greedy heuristic: Add links between hosts which have the most traffic
- Random heuristic: Randomly add links within the network
- DeepConf outperforms greedy and random for maximizing link utilization

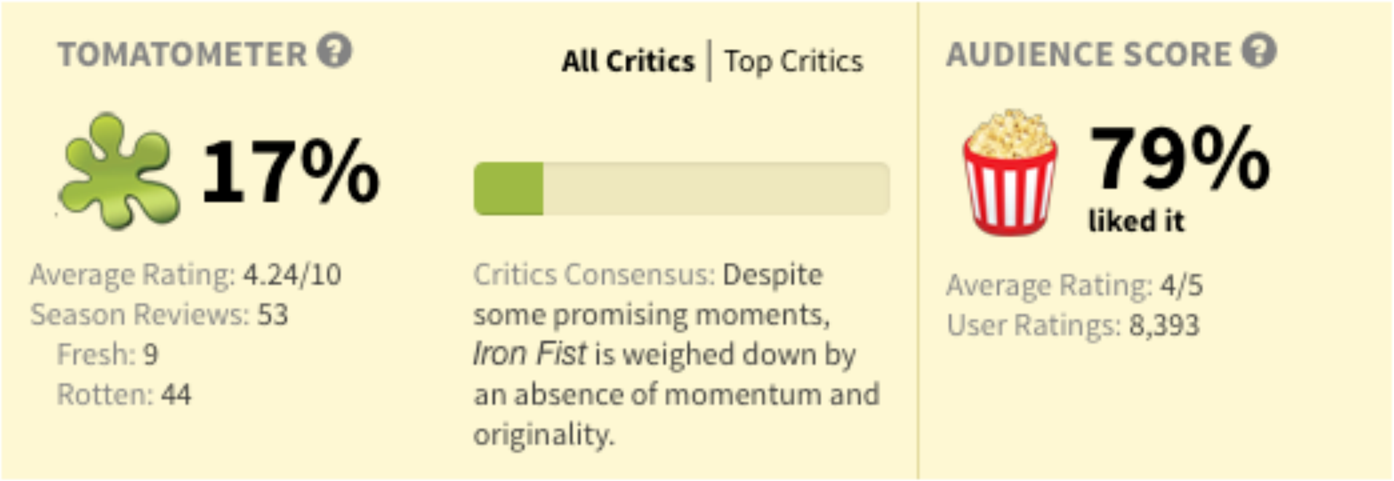
Flow Completion Time Evaluation



- Both greedy and random lead to shorter flow completion times
- DeepConf selects links which lead to congestion and longer flow paths

Future Work

- Apply DRL to select optimal route for flows within topology
- Test the model across different network traces
- Anticipate the release of the critically acclaimed **Iron Fist** season 2


TOMATOMETER

All Critics | Top Critics

17%

Average Rating: 4.24/10
Season Reviews: 53
Fresh: 9
Rotten: 44

Critics Consensus: Despite some promising moments, *Iron Fist* is weighed down by an absence of momentum and originality.

AUDIENCE SCORE

79%
liked it

Average Rating: 4/5
User Ratings: 8,393