Knowledge-free domain independent planning for games

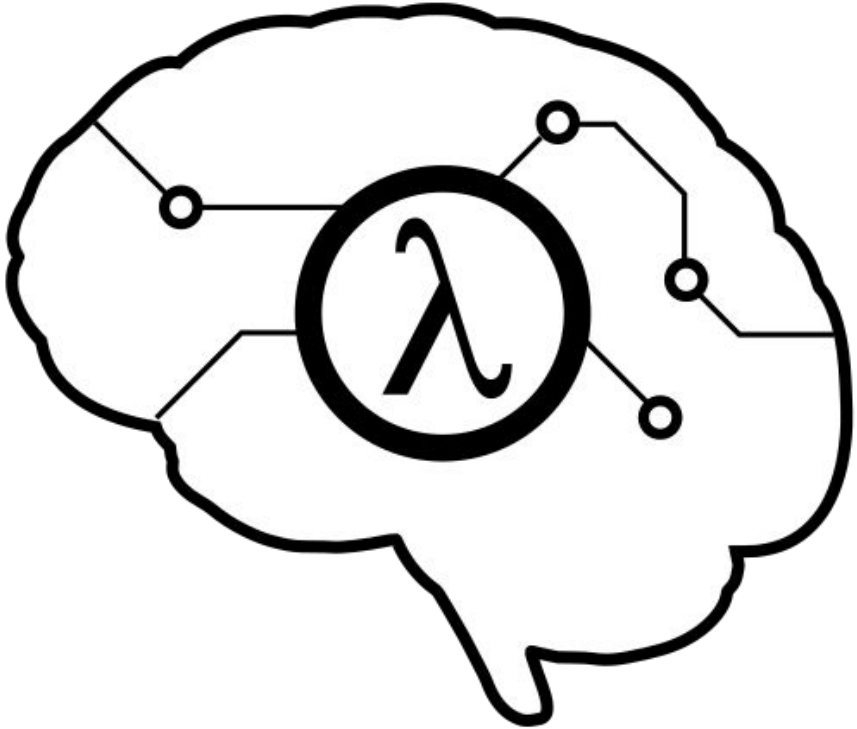
Results on the Atari video game

Raphael Baldi

Introduction

- Knowledge acquisition bottleneck
  - It is hard to formalize domains using symbolic representation
  - Expressing temporal planning domains requires extensions that are not

- Planning in games
  - Exponential growth of transitions
  - Domains are hard to formalize

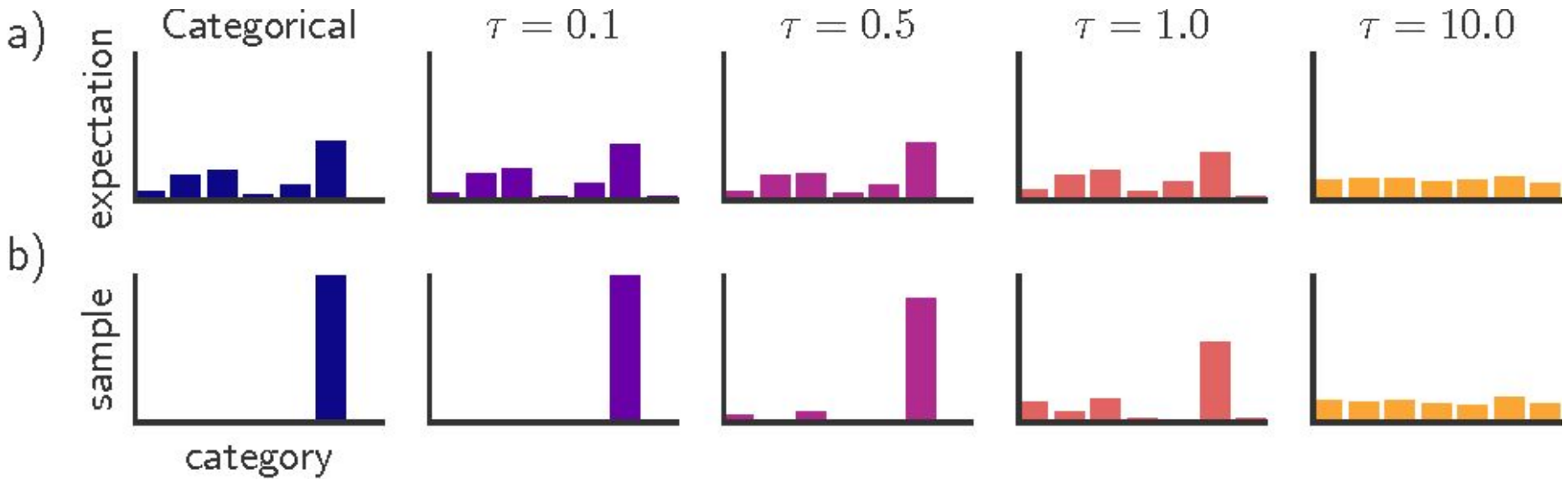


λatPlan

Symbolic classical planning
+ neural perception

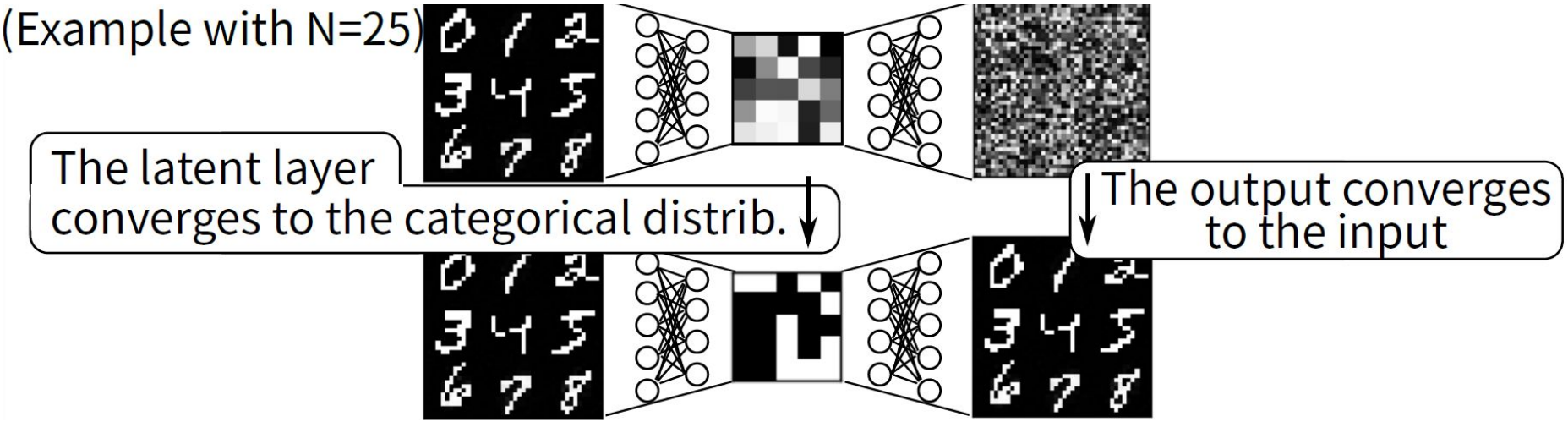
Latplan

- Encode states using a categorical distribution
- Use Gumbel-Softmax as bottleneck (reparameterization trick)
  - Temperature annealing



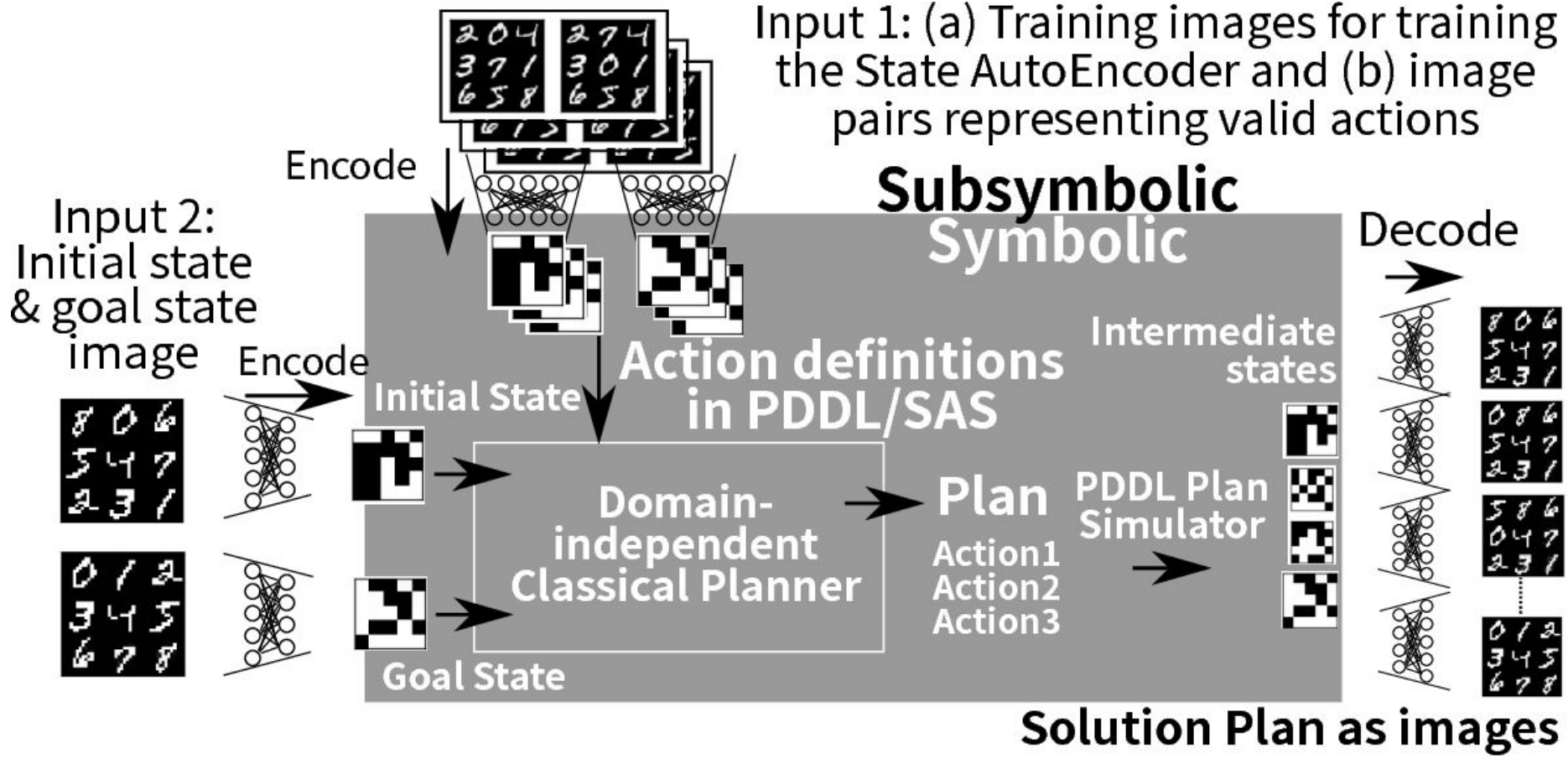
Latplan

- Categorical convergence
- Optimization objective: reconstruct input + reduce softmax loss



Latplan - AMA1

- State autoencoder
  - Encode frames into a binary representation
- Trivial conversion from binary encoding into PDDL
  - N predicates (where N is the size of the latent layer)
  - Actions: encode all known transitions
  - Pre: encode s_i
  - Suc: encode s_{i+1}
- Requires us to know ALL transitions
- Run using an off-the-shelf planner (FD, for example)

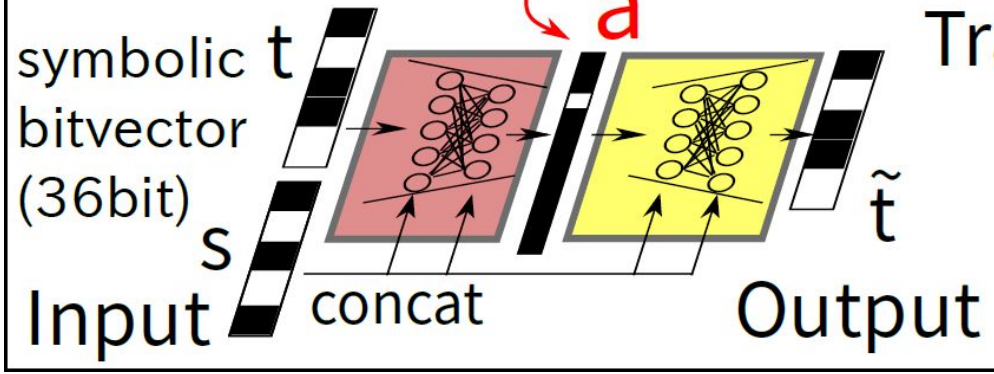

Input 1: (a) Training images for training the State AutoEncoder and (b) image pairs representing valid actions
Encode
Subsymbolic
Symbolic
Input 2: Initial state & goal state image
Encode
Decode
Intermediate states
Initial State
Action definitions in PDDL/SAS
Domain-independent Classical Planner
Plan
Action1
Action2
Action3
PDDL Plan Simulator
Goal State
Solution Plan as images

Latplan - AMA2

- State discriminator
  - Learn to identify valid states
- Action autoencoder
  - Learn a transition function, and encode transitions as a one-hot categorical vector
- Action discriminator
  - Learn to identify valid transitions
- Trivial planner
  - Based on A*
  - Sample actions using the AAE

Latplan - AMA2 - Action Autoencoder

One-hot action label ((N,M)=(1,128) Gumbel-Softmax)



Train NN to Minimize |t - t̃|

Obtain two functions:

- **action(t, s) = a**
- **apply(a, s) = t̃**

Arcade Learning Environment

- Clear API
  - Extract current frame and memory
  - Full control over the simulation
  - Can run in non-stochastic mode
- Supports several Atari games
- We are using Alien
  - Pac-man clone
  - With guns
  - And Aliens

Project management

1. Organize the infrastructure to accommodate Atari frames.
2. Train all the networks.
3. Integrate the networks into the planner.
4. Collect results and compare our approach to other planners on the same domain.
5. Write the final report.

Project management

Table 1: Tentative time schedule for the project

Activity	W1	W2	W3	W4	W5
1	X	X			
2		X	X		
3		X	X		
4		X	X	X	
5				X	X

Conclusion

- We are proposing to evaluate Latplan in a complex domain

- The expected result is for us to be able to plan without prior knowledge of the environment

References

[1] Masataro Asai and Alex Fukunaga. Classical planning in deep latent space: Bridging the subsymbolic-symbolic boundary. CoRR

[2] Malik Ghallab, Dana Nau, and Paolo Traverso. Automated Planning and Acting. Cambridge University Press, New York, NY, USA, 1st edition, 2016.

[3] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In Fifth International Conference on Learning Representations (ICLR), 2017