Classical Planning in Deep Latent Space: From Unlabeled Images to PDDL (and back)

Masataro Asai, Alex Fukunaga
The University of Tokyo

1. Background

- **Symbols: Labels for distinguishing entities**
- **Symbols in PDDL:**

Propositions	(handempty)
Objects	A, B, C, D
Predicates	(clear ?x)
Actions	(pick-up ?x)

- **Existing Action Model Learning (AML) systems ALL require symbolic (or near-symbolic) inputs**
- **Cannot directly handle complex unstructured high-dimensional data e.g. images, audio**

Action-Relation Modelling System *[Yang et al 07]*
input: symbolic relations (predicates, actions)

Semi-MDP → Classical Planning *[Konidaris et al 14,15]*
Not entirely subsymbolic: converts probabilistic → propositional model
Symbolic action labels: **given** *(SMDP option e.g. move,interact)*
Input sensors: **given,structured,low-dim**
(set of 33 real/int variables with distinct meanings:x/y distance, light level, whether monkey cries)

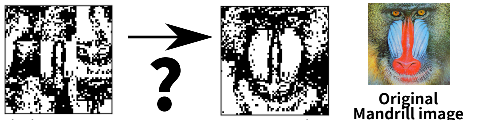
Learning from Observation: *[Argall et al 09, Mourao et al 12]*
Noisy, incomplete, but symbolic states/actions
Learns the preconditions from state/action sequence
Object/predicate/action labels: **given**

Learning from Video for board game:
Images with strong assumptions (almost symbol)
e.g. 3x3 Ellipse Detector *[Barbu et al 10; Kaiser12; Kirk&Laird16]*
almost immediately provides propositions

2. Our Objectives

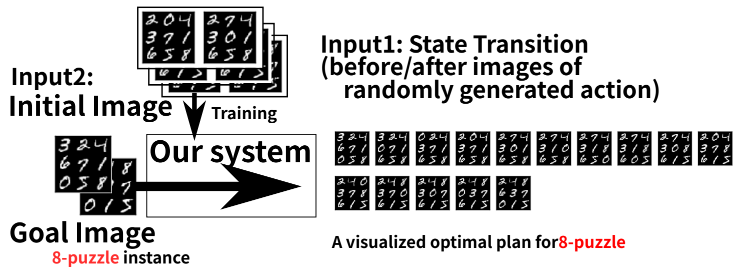
Visually presented classical planning domains

The system should have no idea that this is an 8-puzzle



Initial state image **Goal state image (black/white)**

- with **NO** prior assumptions/symbolic descriptions
("grids","tiles","disks","move","toggle") i.e. domain-independent
- with **NO** expert plan traces



Existing AML systems cannot handle raw images

- High-dimensional input (42x42 pixels = 1764)
- Each pixel does not have a significant meaning
- "Meaning" emerges from nonlin. entanglements
- No simple detector (e.g. stone exisitence)
- Need Robustness & Generalization for noisy inputs

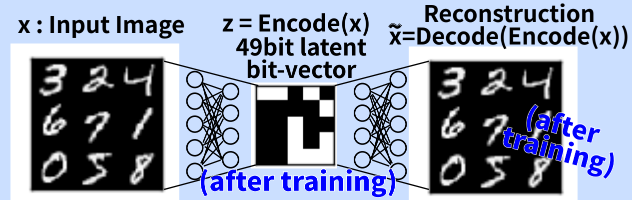
3. OPEN PROBLEM: Convert images/ arbitrary unstructured data to/from symbols

Generate the symbolic inputs for AML Systems !

Raw data → Symbolic data
→ **Exisiting AML methods**
→ Symbolic Planning → **Symbolic plan → Raw output**

4. Solution: State AutoEncoder

- A neural network that learns a **bidirectional mapping** **z=Encode(x)** and **x=Decode(z)** between a subsymbolic input **x** and a discrete boolean vector **z**
- The only specification is the number of bits:
An upper bound of state encoding length $\log_2(|S|)$



- For training, optimize the loss via Stoc. Grad. Descent
Minimize reconstruction loss $||x-\tilde{x}||$ (Binary crossentropy etc.)
+ variational loss KL(Z, Categorical(49))
(KL divergence: distance between distributions)

SAE: A Variational AutoEncoder w/ Gumbel-Softmax

- Allows NNs to learn a categorical/discrete distribution

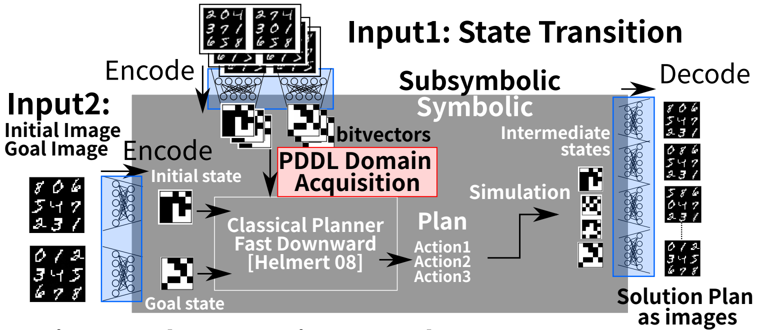
$$z_i = \text{Softmax}((g_i + \log \pi_i)/\tau)$$

π_i: Probability for Category i τ : Annealing Temparature
g_i : A random sample from *Gumbel(0,1)* [Gumbel et.al, 1954]

Number of categories = 2 : Propositional variables

- the "meaning" of the propositions may or may not correspond to human intuitions about the domain

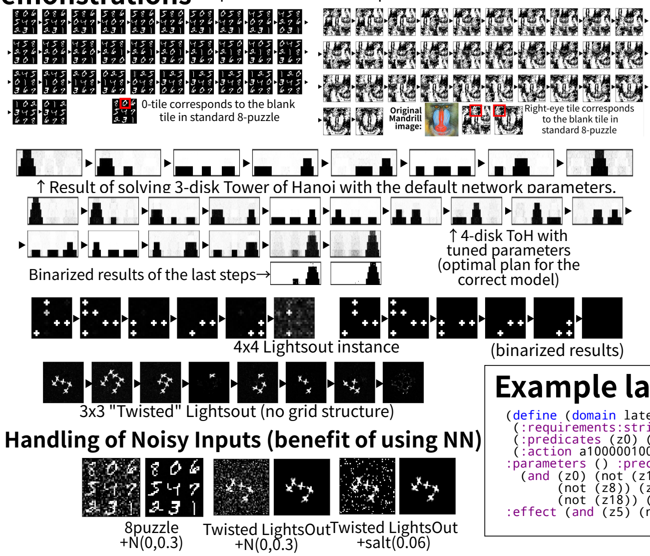
5. Image-based planning architecture: LatPlan



A First Implementation: LatPlan α

- SAE: Trained w/ smaller subset of states
(cross validated, SAE learns a generalized/robust state encoding)
- Trivial AML method: action = individual state transition
we provide all state transitions (i.e. no action generalization)
  - Future work: adapt existing AML method

Demonstrations Optimal solutions to 8-puzzle instances



6. Domain-Independent SearchHeuristics are Effective in Latent-Space Planning

Somewhat surprising result: Heuristics reduce search in domains based on features generated by neural networks

- Exploit common characteristics (e.g. abstraction)
- PDBs can underperform blind search in some standard IPC domains (Edelkamp12)

Heuristics	Search Time (sec)	Expansion
MNIST 8-puzzle (6 different instances)		
blind	1.904	193924
pdb	**1.780**	**109096**
blind	1.751	201156
pdb	**1.508**	**111642**
blind	1.657	186767
pdb	**1.215**	**84561**
blind	1.514	183336
pdb	**1.474**	**82518**
blind	1.460	169907
pdb	**0.685**	**52084**
blind	1.489	130863
pdb	**0.382**	**26967**
Hanoi (4 peg)		
blind	0.0008	55
pdb	**0.0006**	**17**
LightsOut (4x4)		
blind	0.0159	952
pdb	**0.0013**	**27**
Spiral LightsOut (3x3)		
blind	0.0040	522
pdb	**0.0026**	**214**
Mandrill 8-puzzle		
blind	2.759	335378
pdb	**1.113**	**88851**

Example latent-space domain

Example with |z| = 25bits

```
(define (domain latent)
 (:requirements:strips :negative-preconditions)
 (:predicates (z0) (z1) (z2) (z3) (z4) ...  (z24))
 (:action a1000001001011011110001111100001000101111111110011111
:parameters () :precondition
  (and (z0) (not (z1)) (not (z2)) (not (z3)) (not (z4)) (not (z5)) (z6) (not (z7))
       (not (z8)) (z9) (not (z10)) (z11) (z12) (not (z13)) (z14) (z15) (z16) (z17)
       (not (z18)) (not (z19)) (not (z20)) (z21) (z22) (z23) (z24))
:effect (and (z5) (not (z6)) (z13) (z20))) ...
```

pre: before-state

eff: state diff