問題分割と対称性検知を用いた、高速なプランニングアルゴリズムの開発

浅井政太郎
東京大学大学院 総合文化研究科

1. はじめに

本報告は、平成 29 年度スーパーコンピューター若手・女性利用者推進制度の前記課題「問題分割と対称性検知を用いた、高速なプランニングアルゴリズムの開発」の成果を報告する。内容は、国際会議 AAAI-2018 に採択された論文[Asai and Fukunaga, 2018] の日本語訳・要旨である。

2. Introduction

近年、ドメイン非依存プランニングソルバの性能は大幅に改善した。しかしプランニングソルバは、入力として、人間が環境を PDDL [McDermott, 2000] のようなモデリング言語を用いて離散表現に分割した **記号表現** を必要とし、そのような記号表現は、PDDL などのモデリング言語で人が直接入力するか、あるいは人力で作った他の記号表現を PDDL に変換する必要がある。その結果、プランニングの枠組みには、「知識獲得のボトルネック」と呼ばれる、現実世界を人間が理解しシンボルに翻訳するステップが存在する。

記号表現の要求は実応用における大きな障害である。自動プランニングは与えられたモデルで記述できる状況にしか対応できない。従って例えば、想定外の状況に事欠かず、かつ (モデル変更のための) 通信が容易でない環境（例:火星探査ロボット）には適用できない。この問題を解決するためには、生のセンサー入力からいかにして記号モデルを生成するかという問題、すなわちシンボル接地問題[Steels, 2008]の解決が必要である。

近年、物体認識[Ren *et al.*, 2015]、音声認識[Deng *et al.*, 2013]、問題解決システム[Mnih *et al.*, 2015; Graves *et al.*, 2016]など、様々なニューラルネットワーク (NN) ベースの手法が成功を収めつつある。しかし、**純粋な NN ベースの問題解決システムは、記号的システムの持つ理論的性質 (アルゴリズムの完全性や解の最適性など) を保証しない。**

この両者の欠点を補完するため筆者らは、NN ベースの知覚システムを用いて、ドメイン非依存プランニングの記号的入力モデルを画像から自動的に生成し、解く手法を開発した。このことで本研究は、自動プランニング技術の適用性を大幅に拡大し、両方のパラダイムに貢献する。今回主題となるのは、**記号的システムと NN ベースシステムをいかにして繋げるかである**。なぜなら、NN ベースシステムの出力は通常、記号的システムには扱えない連続値だからである。

Fig. 1 左の写真は、右にある写真を 3x3 スライディングパズル (8 パズル) としてランダムに動かした例である。我々は、このようなパズルをドメイン非依存に自動で最適に解くことの出来る**潜在空間プランナ (LatPlan)** を制作した。システムの入力は、許される動作の前後を写している多数の画像ペアであり、それ以上のラベルは与えられていない。通常の記号的ソルバにとってこのような 8 パズルは (記号表現がある限り) 自明な問題である。しかし今回扱うような、画像のみを入力として「動く物体」「格子状」「タイル」などの事前知識を一切持たずにドメイン非依存に問題を解決するシステムは、全く自明ではない。

LatPlan は以下の 3 つの要素からなる: (1) NN ベースの **State AutoEncoder** (SAE) はセンサー入力と記号表現の双方向の変換を担う。SAE の実装は Gumbel-Softmax[Jang *et al.*, 2017] 活性化関数を用いた Variational AutoEncoder [Kingma and Welling, 2013]で、その隠れ層の活性値は離散値 (カテゴリカル 1-hot 表現) に収束する。ここでカテゴリ数を 2 に制限することで、1-hot 表現を命題の真偽値に対応させることが出来る。SAE を画像データを用いて入力と出力が一致するよう訓練すると、SAE は 画像入力→命

図 1　画像によって与えられた 8 パズル。

題列→画像出力 (入力と一致) を達成するネットワークとして訓練される。結果、ネットワークの前段 (エンコーダ) および後段 (デコーダ) を用いて、ノイズのある実数値センサー入力を命題列に変換、および逆変換する関数が得られる。(2) **Action Model Acquisition (AMA)** は SAE が画像から生成した命題列から PDDL モデルを生成する。(3) 記号的ソルバが PDDL モデルを解き、記号的プランを探索結果として得る。このプランは未知の記号列で表現されているが、この命題列を SAE の逆変換により人間に理解可能な画像に戻すことができる。実験では、 LatPlan を 8-puzzle, LightsOut, およびハノイの塔の画像版を用いて評価した。

3. 背景

3.1. 古典プランニング

古典プランニングソルバの性能は、知識あり前方探索探索の下界関数の進歩によって近年劇的に進歩している。ソルバ (**プランナ**) の入力は $\Pi = \langle P, O, I, G, A \rangle$、 ここで P は一階述語論理の述語、 O は述語の引数となるオブジェクト、 I は初期状態、 G はゴール条件、そして A は探索空間で許される遷移を表している **アクションスキーマ集合** である。状態とは、命題変数に対する真偽値の割当であり、 条件はそのうち一部を指定する部分的割当となる。それぞれの命題変数は、述語にオブジェクトを適用したものである。アクションスキーマ $a \in A$ はさらに 5-tuple $\langle params, pre, e^+, e^-, c \rangle$ で表される。これは順にパラメータ、前提条件 (precondition)、追加効果、削除効果、コストを表す。アクションスキーマのパラメータを O 中のオブジェクトで代入すると、 **アクション** が得られる。c が指定されない場合は、通常 $c = 1$ が想定される。これらの入力は PDDL [Bacchus, 2000] のようなモデリング言語で記述される。

Fig. 2 は、3x3 パズル (8 パズル) を一階述語論理、および対応する PDDL で表現した一例である。

$$
\begin{aligned}
&Empty(x_0, y_0) \\
\wedge &At(x_1, y_0, panel_6) \\
\wedge &Up(y_0, y_1) \\
\wedge &Down(y_1, y_0) \\
\wedge &Right(x_0, x_1) \\
\wedge &Left(x_1, x_0) \ldots
\end{aligned}
$$

```
(empty x0 y0)
(at    x1 y0 panel6)
(up    y0 y1)
(down  y1 y0)
(right x0 x1)
(left  x1 x0)...
```

	6	8
7	3	2
5	1	4

図 2　人間の作った、3x3 スライディングタイルパズル (8 パズル) の一階述語論理表現と PDDL 表現の例。これには、**述語記号** empty, up, down, left, right, at および**オブジェクト記号**, 例えば x_i, y_i, panel_j ($i \in \{0..3\}$, $j \in \{1..8\}$) が含まれている。

古典プランニング問題を解くことは、許されたアクションを用いて初期状態からゴール条件を満たす状

When
$Empty(x, y_{old})$ ∧
$at(x, y_{new}, p)$ ∧
$up(y_{new}, y_{old})$;
then
$\neg Empty(x, y_{old})$ ∧
$Empty(x, y_{new})$ ∧
$\neg at(x, y_{new}, p)$ ∧
$at(x, y_{old}, p)$

```
;; Translates to a PDDL model below:
(:action slide-up ...
 :precondition
 (and (empty ?x ?y-old)
      (at ?x ?y-new ?p) ...)
 :effects
 (and (not (empty ?x ?y-old))
      (empty ?x ?y-new)
      (not (at ?x ?y-new ?p))
      (at ?x ?y-old ?p)))
```

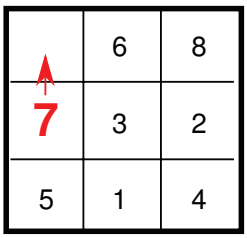

図 3　同様に、タイル 7 を上にスライドさせるアクションの一階述語論理表現と PDDL 表現。ここでは **アクション記号** slide-up が用いられている。

態へ至る経路を発見することである。この際、状態 s に アクション a を適用して状態 t に遷移することを $t = a(s)$ と書き、a は状態遷移関数と呼ばれる。この状態遷移は $s \supseteq pre$ が満たされたときのみ許され、結果として $t = (s \setminus e^-) \cup e^+$ が得られる [Bacchus, 2000].

State-of-the-Art プランナはこの問題をグラフ上の経路探索問題として解く。定義からわかるようにグラフの形状は (例えば隣接行列などで) 明示的には与えられてはおらず、初期状態から指数関数的に大きな、メモリには収まらないサイズのグラフが生成される。したがって、これらの問題を解くためには知識あり探索を用いて高度な枝刈りを行いながら探索を行う必要がある。代表的なアルゴリズムとしては、最短経路と求める場合には A^* 、そうでない場合には Greedy Best-First Search がある. 様々な下界関数のおかげで[Helmert and Domshlak, 2009; Sievers *et al.*, 2012; Helmert *et al.*, 2007; Bonet, 2013; Hoffmann and Nebel, 2001; Helmert, 2004; Richter *et al.*, 2008]、現在の state-of-the-art プランナは、最小でも数千ステップのプランを必要とする巨大な問題を解くことができる [Asai and Fukunaga, 2015]。

4. 知識獲得のボトルネック

理想的には、前節で見たような記号的表現を自動的に学習できる仕組みがあれば好ましいが、そのタスクを人間の助けなくまったく自動で行えるシステムはいままで実現できておらず、実用的なシステムは人間の手によるモデリングに依存している。そのため、記号的 AI システムは **知識獲得のボトルネック** [Cullen and Bryman, 1988]、すなわち、**人間が現実世界を記号表現にモデリングするのにかかるコスト** を避けられない。

古典プランニングのためのモデルを完全自動で生成するためには、**記号接地** と **Action Model Acquisition** (AMA) が必要である。**記号接地** とは、「ノイジーで、連続値の、構造を持たない巨大な入力から、コンパクトで、ノイズがなく、離散値で、分離された個別の要素すなわち記号への変換関数を、教師無しで学習する」操作である。PDDL は 6 種の記号を持つ: オブジェクト、述語、命題、アクション、問題、問題ドメインである (Table 1)。異なる種類の記号はそれぞれ別の方法で接地される。たとえば、画像認識のコミュニティでは画像中の物体 (例:顔)、その属性 (例:男女)、あるいは動画中の動き (例:料理) を検出する手法がそれぞれ提案されている。これらはそれぞれオブジェクト、述語、アクションの記号を接地していると考えられる。

一方、 **アクションモデル** とは、環境の状態遷移を表すなんらかのデータ構造である (記号的であるとは限らない)。PDDL でのアクションモデルは、それぞれのアクション記号に付随する前提条件と追加・削除効果に相当する (Fig. 2)。

この研究では、命題記号とアクション記号の接地、およびアクションモデルの生成に焦点を絞る。

記号の種類	
オブジェクト記号	**panel7, x_0, y_0** ...
述語記号	(**empty** ?x ?y) (**up** ?y_0 ?y_1)
命題記号	**empty$_5$** = (empty x_2 y_1) (6th application)
アクション記号	(**slide-up** $panel_7$ x_0 y_1)
問題記号	**eight-puzzle-instance1504**, etc.
ドメイン記号	**eight-puzzle**, **hanoi**

表 1　PDDL 中の 6 種の記号。

4.1. 既存の AMA 手法

既存の AMA 手法は、すでに接地された記号からアクションモデルを生成することを仮定している。したがって、その入力は記号的、あるいは「ほぼ」記号的な入力である。ARMS [Yang *et al.*, 2007], LOCM [Cresswell *et al.*, 2013], Mourão *et al.* (2012) はアクション記号、オブジェクト記号、述語記号を仮定している。Framer [Lindsay *et al.*, 2017] システムは自然言語を入力として PDDL を出力するが、用いられる自然言語入力には一定の文法構造や一貫した語の使用などが仮定されており、ほぼ記号的な入力を扱っている。

Konidaris *et al.* は PDDL を semi-MDP (2014) と呼ばれる連続値モデルから生成する。しかし、semi-MDP は move、interact など 人間によって与えられたアクション記号を持つ **別の形式のモデル**に過ぎず、実際に行われたことはモデルからモデルへの変換であり、生の現実世界から自動で記号を抽出したとはいえない。また、状態を表す入力も、画像などの非構造データではなく、物体への x/y 軸距離、ライトの光量、スイッチの状態など、それぞれ区別可能な構造にすでに分解された記号的な入力である。

4.2. オートエンコーダと隠れ表現

オートエンコーダ (AE) は、出力が入力と一致するように学習を行う前方伝播ニューラルネットである [Hinton and Salakhutdinov, 2006]。その中間レイヤは入出力よりも小さく、 入力の**圧縮された隠れ表現** を保持していると捉えられている。AE は入力からの誤差を最小化するように逆誤差伝播法 (Backpropagation) により訓練される。ニューラルネットのニューロンは一般に連続値の活性化値を持つため、これらのシステムを記号的なシステムと接続する方法は自明ではない。

5. LatPlan: システム構成

この章は、LatPlan の全体構造を俯瞰する (Fig. 4)。LatPlan は 2 つの入力を持つ。第 1 の入力は、生データの対のセットである**遷移入力** Tr である。各対 $tr_i = (pre_i, post_i) \in Tr$ は何らかのアクションが実行された前後の環境の変化を表している。第二の入力は **プランニング入力** (i, g) (生データの対) であり、これは環境の初期状態とゴールに対応する。LatPlan の出力は、 i から g へのプラン実行の様子を示すデータ列である。本研究ではこの「データ」の種類として画像を用いる実装を制作したが、アーキテクチャ自身にはそのような仮定は置いていない。したがって、このアーキテクチャは音声・テキストなど他の種類に拡張されることが予想される。

LatPlan は 3 段階にわけて動作する。ステップ 1 では、 **State Autoencoder** (SAE) が、環境を写した生の状態データ (画像) とその命題表現の双方向変換関数を教師無しで学習する。SAE の $Encode$ 関数は画像を命題表現に変換し、 $Decode$ 関数は命題表現を画像に戻す。遷移入力 $Tr = \{pre_i, post_i \ldots\}$ を用いて SAE を訓練した後、LatPlan は Tr をすべて記号表現 $(Encode(pre_i), Encode(post_i)) = (s_i, t_i) = \overline{tr}_i \in \overline{Tr}$ に変換する。

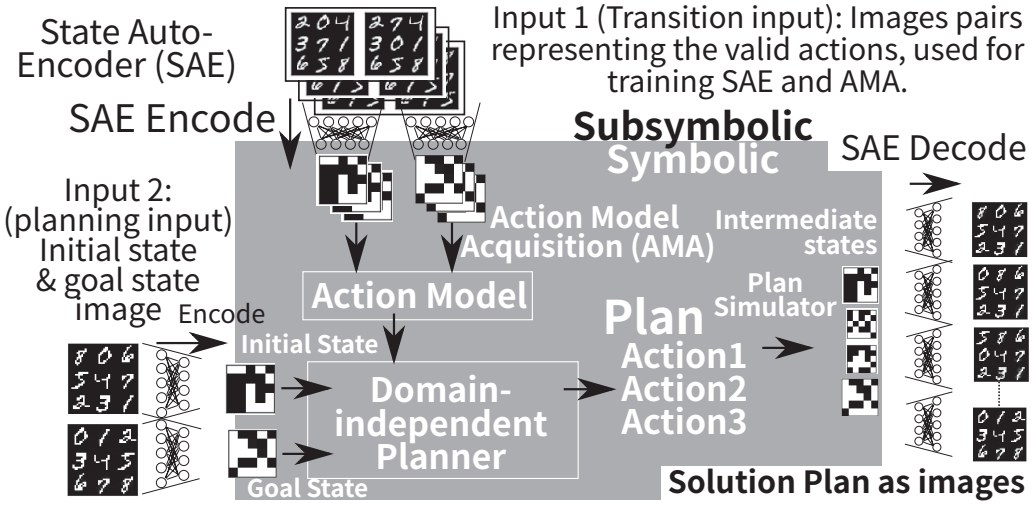

図 4　隠れ空間での古典プランニング：学習済みの State Autoencoder (Sec. 6) を用い、画像ペア $(pre, post)$ を記号的遷移に変換する。次に、記号的状態遷移からアクションモデルを学習する。次に、初期状態・ゴール状態を記号的初期状態・ゴール状態に変換する。次に、古典プランナが、記号的空間でゴールへ遷移するプランを求める。最後に、プランの中間状態が State Autoencoder より画像へとデコードされ可視化される。

ステップ 2 では, AMA 手法が $\overline{Tr}$ から アクション記号とアクションモデルを同時に教師なし学習する。本研究では 2 つのアプローチを提案する: AMA_1 は PDDL を直接生成し、一方 AMA_2 は PDDL は生成しないがグラフ探索に用いる **後者関数**を生成する。各々に利点と欠点がある。AMA_1 は **自明な** AMA であり、その目的は SAE が生成した命題表現がプランナによる論理推論と互換性があることを証明することである。AMA_1 は少数例から汎化する能力を持たず、したがって許可された全遷移の集合を入力として必要とする。したがって AMA_1 は AMA として実用的な手法とは言えないが、直接 PDDL を生成できることから、SAE を用いることで生データを既存の古典プランナに与える手法の実現可能性を示すことができる。AMA_2 は、本論文で示す新しいニューラルネットであり、少ない状態遷移データを汎化しつつアクション記号とアクションモデルを同時に教師無しで学習する。既存の AMA 手法と異なり、 AMA_2 はアクション記号を必要としない。しかし、PDDL を出力しないので、一般的な PDDL ソルバは使えず、それ専用の探索アルゴリズム実装 (A*など) を必要とする。

ステップ 3 では、記号的プランニング問題インスタンスがプランニング入力 (i, g)　から生成される。(i, g) は SAE によって記号表現に変換され、記号的ソルバがこの探索問題を解く。

最後に、生成されたプランが画像にデコードされ可視化される。SAE の出力する命題表現は人間の作った表現ではないので、それぞれの命題の「意味」は人間の観測者にとって自明ではない。しかし、SAE が *Decode* 関数を持つ双方向関数であることから、命題表現で表された探索結果を画像に戻し、可視化することができる (例:Fig. 6)。

6. SAE as a Gumbel-Softmax VAE

画像から命題への 1 対 1 のマッピング自体は、自明に求めることができる。単に離散化された画素値の配列や、または画像ハッシュ関数を用いればよい。しかし、そのような SAE には、一般化 — すなわち、始めて観測した世界の状態を同じシンボルで表す能力、ロバストネス — 同じ状態を表す 2 つの類似した観測を同じ命題表現にマップする能力、双方向性 — 命題表現をもとの表現に戻す能力、が欠けている。Latplan を実

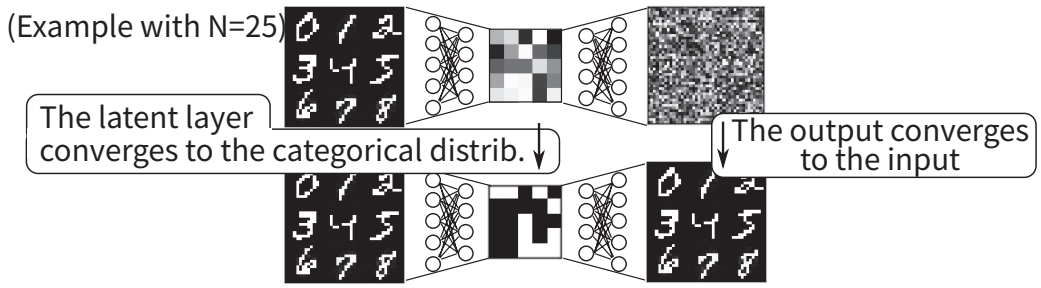

図 5　Step 1: 入力との誤差を最小化すると同時に離散値との KL ダイバージェンスを最小化することで、State Autoencoder を訓練する。訓練が進むに連れ、出力は入力に収束し、かつ中間層は離散値に収束する。

現するためには双方向関数が必要であり、Latplan の SAE は生の画素ベクトルのような命題表現ではなく、画像の「エッセンス」をキャプチャした命題表現を得なくてはならない。

本論文の第一の貢献は、SAE を Gumbel-Softmax (GS) [Jang *et al.*, 2017] 活性化関数を用いた Variational Autoencoder [Kingma *et al.*, 2014] として実装する手法である (Fig. 5)。

Variational Autoencoder (VAE) [Kingma and Welling, 2013] は、隠れ層の活性化の分布をある一定の分布 (例:正規分布) に収束させられる特殊なオートエンコーダである。正規分布のようなランダム変数は微分不可能なので、VAE は **reparametrization trick**、すなわち対象となる分布をランダム変数と学習可能な変数に分割する操作を行う。例えば、正規分布 $N(\sigma, \mu)$ は $\mu + \sigma N(1, 0)$ に分解できる。

Gumbel-Softmax (GS) は近年提案された reparametrization trick [Jang *et al.*, 2017] で、活性化値の分布を離散カテゴリカル分布に収束させることが出来る。GS は Gumbel-Max [Maddison *et al.*, 2014] というサンプル手法の微分可能な近似手法と捉えられる。出力 z が one-hot 表現、例えば値域が $D = \{a, b, c\}$ であるとき $\langle 0, 1, 0\rangle$ は “b” を表す。Gumbel-Max の入力はクラスに属する確率を表すベクトル π、例えば $\langle .1, .1, .8\rangle$ である。このとき Gumbel-Max は D から π に従って以下のように離散値をサンプリングする: $z_i \equiv [i = \arg\max_j (g_j + \log \pi_j)\ ?\ 1 : 0]$ ただし、 g_j は Gumbel$(0, 1)$ [Gumbel and Lieblein, 1954] から相互に独立に得られたサンプルである。Gumbel-Softmax は argmax を softmax で近似することにより全体を微分可能にする: $z_i = \mathrm{Softmax}((g_i + \log \pi_i)/\tau)$. ここで、「温度」 τ は近似の度合いを決定し、0 へと冷却される。温度が 0 に近づく時、GS の出力は Gumbel-Max の離散サンプルに収束する。

ここで鍵となるのは、これらの離散変数が命題として直接使用できるということである。すなわち、これは Latplan アーキテクチャの命題記号接地に対する解を与える。SAE では、潜在層に GS を使用する。その入力はエンコーダネットワークに接続される。GS の出力は (N, M) 行列である。ここで N はカテゴリ変数の数、M はカテゴリの数を表す。 ここで $M = 2$ と指定すれば、結果として得られる表現は N 個の命題変数として捉えることが可能になる。

訓練後の SAE は、生の入力と命題表現の双方向マッピングを提供する:

- $b = \mathit{Encode}(r)$ は画像 r を Bool 配列 b に変換する。
- $\tilde{r} = \mathit{Decode}(b)$ は Bool 配列 b を画像 $\tilde{r}$ に変換する。

$\mathit{Encode}(r)$ は、生の入力 r をエンコーダネットに入力し、GS 層の出力を取り出し、$N \times 2$ 配列の一列目を取り出すことで長さ N の bool 配列を得る。$\mathit{Decode}(b)$ は bool 配列 b をその補数 $\bar{b}$ と結合して $N \times 2$ 配列を得、これをデコーダネットに入力することで画像を取り出す。

通常の活性化関数を丸めることで離散値を得る手法では、デコーダネットが 0/1 の値を用いて訓練されていないので、命題表現を正しくデコードすることが出来ない。SAE の実装はディープラーニングコミュ

ニティの様々な技術革新から直接恩恵を受けることが出来る。実際、筆者らが用いた実装は Denoising AE [Vincent *et al.*, 2008] によって AE にノイズ耐性をつけている。

7. AMA_1: Oracular PDDL Generator

最初の AMA 方式である AMA_1 は、PDDL を直接出力する。AMA_1 は、環境で許された全ての遷移を入力として動作する **神託機械的** 戦略であり、理論的に最適な AMA 方式をシミュレートしている。従って、その手法自体はスケーラビリティの観点からは実用的ではない。ただし、AMA_1 が全ての遷移 (すなわち全ての状態を含む) を入力とする一方、SAE は少数の画像データから訓練されたことを明記しておきたい。繰り返すが、AMA は理想的には少数の訓練サンプルから学習を行うべきだが、AMA_1 の目的は SAE を用いた命題表現を用いてプランナが推論を行えるかを示すために意図的に制作されたものである。

AMA_1 は $\overline{Tr}$ を以下のようにして PDDL に変換する。それぞれの遷移 $(s_i, t_i) \in \overline{Tr}$ は各々個別のアクション a_i に対応する。Bool 配列 s_i および t_i 中のそれぞれのビット $b_j (1 \leq j \leq N)$ は、1 のときには命題 `(b`$_j$`-true)`、0 のときには `(b`$_j$`-false)` に変換される。s_i はアクション a_i の前提条件として直接用いられる。追加・削除効果は s_i と t_i のビット毎の差分から作られる。例えば、 b_j が 1 から 0 になるとき、対応する効果は `(and (b`$_j$`-false) (not (b`$_j$`-true)))` である。初期状態とゴールも同様に変換される。

AMA_1 の生成した PDDL インスタンスは既存のプランナによって解かれる。我々は Fast Downward [Helmert, 2006] ソルバを用いた。従って、LatPlan は用いるソルバの探索アルゴリズムの理論的性質を全て引き継ぐ。例えば、プランナが完全かつ最適であれば、 LatPlan は 解が存在すれば必ずそれを返却し、また返却される解は最短の解である。

7.1. AMA_1 の評価

我々は LatPlan + AMA_1 を複数のパズルドメインで評価した。結果として生成されたプランを Fig. 6-7 に示す。

MNIST 8-puzzle は 8 パズルの画像ベースのバージョンで、タイルには MNIST データベース[LeCun *et al.*, 1998]の数字が書かれている。このドメインの有効なアクションは、「0」タイルを隣接するタイルと交換する操作である。すなわち、「0」は通常の「空」タイルとして機能する。マンドリル、スパイダー 8 パズルは、一般の店で売っているような写真を持ちいた 8 パズルである。これらは、MNIST の 8 パズルとは違って各「タイル」が黒色の領域できれいに分離されていない。これは、LatPlan には「格子状の領域」といったような概念を必要としていないことを示している。**ハノイの塔（ToH）**では、円盤 4 つのインスタンスを解き、15 ステップの最適なプランを発見している。LightsOut は小型の電子ゲームをベースとしたドメインで、ライトのグリッドがオン/オフ設定（＋：オン）になっていて、ライトを押すとそのライトおよびその隣接ライトの状態が反転する。ゴールはすべて消灯された状態である。以前とは異なり 1 つのアクションが 5/16 の位置を一度に反転し、またいくつかの「オブジェクト」（ライト）を取り除くため、このドメインは、LatPlan が「局所的な変化」や「一定のオブジェクトが安定して存在すること」などを仮定していないことを表す。最後に、 **Twisted LightsOut** は元の LightsOut の画像をスワール効果で歪ませて入力としたもので、これは LatPlan が長方形の「オブジェクト」/領域に限定されないことを示している。

7.2. ノイズ耐性

Fig. 8 は学習済みの SAE にノイズを含んだ入力を与えたときのデコード結果である。SAE が Denoising AE であることから、ノイズを含まない出力を返す。この SAE を用いることで、初期状態やゴール状態がノイズを含んでいても、同様にプランを生成できる。

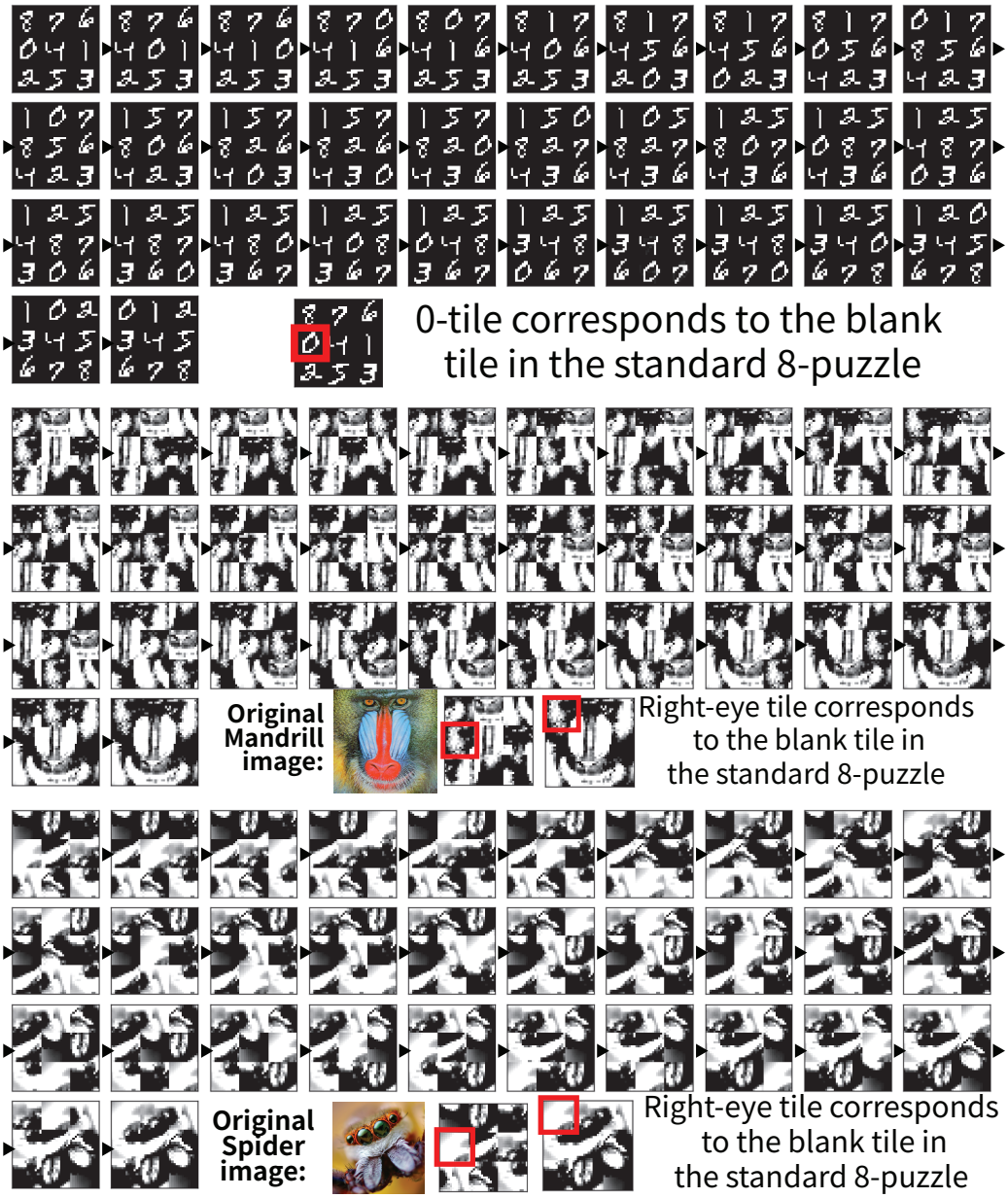

図 6　LatPlan ＋ AMA_1 が MNIST/Mandrill/Spider 8 パズルを解いた例。このインスタンスは、8 パズル中最も長い解を持つ (31 ステップ) 問題例であり、LatPlan は最適解を見つけている (Reinefeld 1993)。これは、神託的 AMA_1 を与えられれば、 LatPlan が SAE による命題変数を用いて最適解を正しく探索できることを示している。これにより、 LatPlan は 「スライド」「タイル」などの事前知識は全く与えられること無しに記号的推論を行えることが示された。

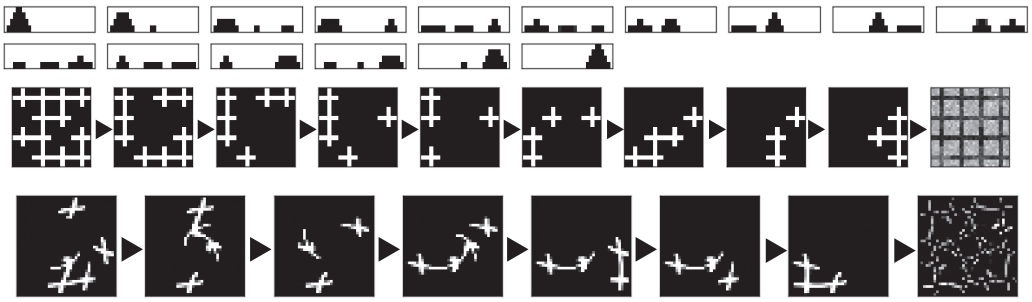

図 7　(1) ハノイの塔の例。(2-3) LightsOut と Twisted LightsOut の例。ゴール状態に見える残渣は、小さなノイズが可視化ライブラリによって正規化・強調されたもの。

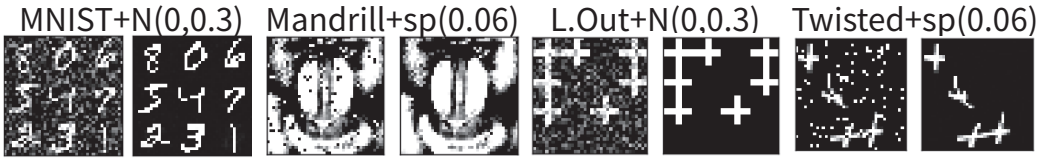

図 8　SAE のノイズ耐性: ノイズを追加された初期状態 r とその復号結果 $Decode(Encode(r))$。ノイズにはガウシアンノイズと ごま塩ノイズを用いた。LatPlan はこれらの問題に対してもプランを返却した。

8. AMA_2: Action Symbol Grounding

LatPlan + AMA_1 は、(1) SAE が画像 ↔ 命題ベクトルを学習できること、および (2) 全ての有効な画像から画像への遷移（つまり状態空間全体）を与えられれば、LatPlan が最適な計画を正しく生成できることを示した。しかし、AMA_1 は状態空間全体を入力として必要とするために実用的ではなく、実際には少数の状態遷移の例からアクションモデルを学習/一般化する能力が必要である。そこで、AMA_2 を提案する。これは、少ない状態遷移データを汎化しつつアクション記号とアクションモデルを同時に教師無しで学習する新たなニューラルアーキテクチャである。

AMA_2 は、Action Autoencoder（AAE）と Action Discriminator（AD）という 2 つのネットワークで構成されている。AAE は同時にアクション記号と効果を学習し、現在の状態から遷移可能な状態の候補を列挙する能力を提供する。AD は、候補のうちどの遷移が実際に有効であるか、すなわち前提条件を学習する。後者状態 (Successor states) の候補を列挙・フィルタリングすることで、AMA_2 は正しい後者関数を提供する。

8.1. Action Autoencoder

仮に、枝分かれのない線形な探索空間を考えよう。この場合、アクション記号は不要であり、AMA の目的は、現在の状態 s の唯一の後者状態 t を **予測** することと等価になる。従って NN a' は、損失関数 $|t - a'(s)|$ を最小化することで後者関数 $a(s) = t$ を近似するように訓練することができる。このような単純化は ビデオからのシーン予測[Srivastava *et al.*, 2015] など多くの既存研究で行われている。

しかし、現在の状態がプランニング問題ののように複数の後者状態を持つ場合、そのような単純化は適用できない。1 つのアクションごとに別々の NN をトレーニングすることも考えられるが、(1) 何個のアクションがあるかもわからない、(2) 状態によって許されるアクションの数が変わる、(3) そもそもどの状態遷移がそのタイプのアクションに属するかも解らない、ということから、訓練のしようがない。単一の NN がマルチモーダル分布を学習することは可能かもしれないが、それは探索アルゴリズムの重要な必要条件である**後**

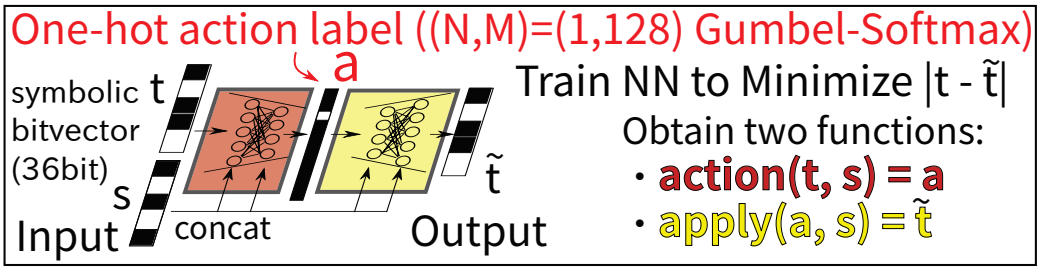

図 9　Action Autoencoder.

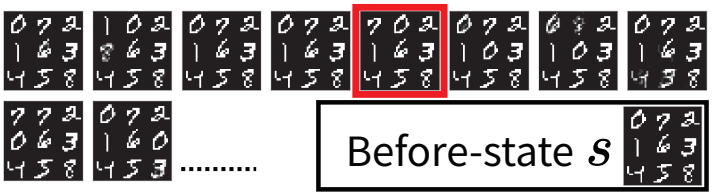

図 10　右下の状態の後者状態候補を AAE が列挙したもの。このうち正しいものは赤枠のものだけである。

者状態の列挙が不可能である。

これを解決するために、Action Autoencoder(AAE, Fig. 9) を提案する。 AAE の重要なアイデアは、遷移を $apply(a, s) = t$ と再定式化することである。この定式化から、アクション記号は訓練されるべき変数であり、また s は状態遷移の背景情報であることが分かる。そこで AAE は、s, t を入力とし、t を $\tilde{t}$ と自己符号化するよう誤差 $|t - \tilde{t}|$ を最小化する。典型的な AE との主な違いは次のとおりである。(1) 潜在層は Gumbel-Softmax one-hot 表現であり、これが **アクションラベル**を示すベクトル a である。(2) 全ての層がそれぞれ s と concatenate される。これは、ネットワークの表現全体が s で条件付けされることを意味する。これにより、128 個のアクション記号（7 ビット）が、「s を与えられた場合に t を再構成する」ために必要な条件付き情報（差分）のみを保持することになる。一方、通常の AE は入力の全情報をエンコードする。

結果、 AAE は t と a の双方向変換関数を学習するが、変換関数は共に s によって条件付けられている:

- $Action(t, s) = a$ は t からアクション記号 a を返す。
- $Apply(a, s) = \tilde{t}$ は アクション a を s **に適用し**、後者状態 $\tilde{t}$ を返す。

アクション記号の数 (128) は、実際に必要な記号の数の上界となっており、学習後には、どの入力も対応しないような記号が存在しうる。少数のアクション記号を得たことから、LatPlan は現在の状態の後者状態の候補を定数時間で返却することが出来る。仮に AAE が無ければ、2^N 個の全ての状態を列挙・枝刈りする必要があるが、これはあきらかに現実的ではない。

8.2. Action Discriminator

AAE はアクションの効果 (差分) を学習しているが、それぞれのアクションがいつ許されるかという前提条件は学習していない。前提条件による制限がなければ、実際に例えば、3 つのタイルを同時に移動したり、同じ番号のタイルを複製したりという推論を行ってしまう Fig. 10. そこで、*Action Discriminator* (AD, Fig. 11) が、それぞれの遷移に対して正しいか否かの確率を返すよう訓練される。これは通常の教師あり二値学習機であり、s, t を入力にとってそれが正しい確率を返す。

ここで問題になるのは、 二値分類機を訓練するためには 「間違い」とラベル付された遷移が必要だが、手

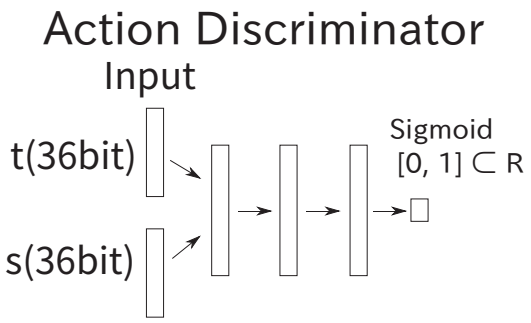

図 11　Action Discriminator.

元にそのような負データがないということである。ここで注意したいのは、**現実世界では、「間違った」遷移は観測することができない**ということである。例えば、テレポーテーションや念力などといった非現実的アクションは「起こり得ず」従って「データは存在しない」。物理環境で動作する現実的ロボットには、従って、明示的な負データを集めることが出来ない。負データを機械的に生成することも考えられるが、そもそも何が正データなのかすら解らないので、負データに対する生成のルールも存在しない。

この問題を解決するために、AMA_2 は PU-Learning [Elkan and Noto, 2008] を用いる。これは、正負の二値分類機を 正例と **未分類例** から学習するための手法である。ここで、$\overline{Tr}$ は観測データであるから正例である。未分類例 (正例と負例の混ざったデータ) は、AAE が生成した後者状態の候補である。

8.3. AMA_2 を用いたプランニング

AMA_2 は PDDL を出力しないので、PDDL ベースの既存プランナを用いることが出来ない。しかし、AAE と AD を用いれば後者関数を得られ、これによって A*[Hart *et al.*, 1968] を用いた記号的探索が可能になる。AAE は (間違った遷移を含む) 後者状態候補を返し、 AD はその候補を分類する:

$$Succ(s) = \{t = apply(a, s) \mid a \in \{0 \dots 127\}, \\ \wedge\, AD(s, t) \geq 0.5\}$$

我々は、状態を bit 配列とし、上記の $Succ$ 関数を後者関数とする A* を実装した。下界関数には goal-count heuristic (ゴールと異なる bit の数) を用いた。Goal-count heuristics は許容的ではないため結果は非最適だが、この実験の目的はそもそも AMA_2 が正しいプランを生成できるのかを示すことであり、最適性は重要ではない。

8.4. 評価

我々は、AAE によって学習されたアクション記号とアクションモデルの正しさを評価した。評価するドメインには 8 パズル（mnist、mandrill、spider)、LightsOut（+ Twisted）を用いた。それぞれのドメインごとに 200 インスタンスを生成し、正しいプランが返された数を計測した。それぞれのインスタンスにはガウシアン・ごま塩ノイズを加えたバージョンも加えた。200 インスタンスのうち 100 題はゴールから 7 歩のランダムウォークで生成し、残りは 14 歩で生成した。プランナは 3 分の時間制限で評価した。生成されたプランの正しさは、8 パズルなどドメインごとにバリデータを書いて用い、また適宜目視も行った。

Table 2 からは、 LatPlan が十分に高い成功率を保っていることが分かる。また、解かれなかった場合の多くは時間切れに依るものである (通常のプランニングと異なり、後者状態の生成のたびに重いニューラルネットの計算が行われるため)。

domain	A:step=7			B:step=14			AD error (in %)	
	std	G	s/p	std	G	s/p	type1	type2
MNIST	72	64	64	6	4	3	1.55	6.15
Mandrill	100	100	100	9	14	14	1.10	2.93
Spider	94	99	98	29	36	38	1.22	4.97
L. Out	100	99	100	59	60	51	0.03	1.64
Twisted	96	65	98	75	68	72	0.02	1.82

表 2　AMA_2: (**左**) 3 分間の間に正しく解けた問題の数。第二/第三列は G(aussian) / ごま塩 (s/p) ノイズを加えた結果。7 歩のランダムウォークで生成されたベンチマーク A では、LatPlan は過半数の問題を解けている。より難しい 14 歩のベンチマークでも、依然として多くの問題が解かれている。(**右**) AD の type-1/2 エラー (%): (type-1) 全ての正しい遷移を生成し、AD が否定したものの割合。(type-2) まず、1000 個の状態を生成し、AAE によって後者状態候補を作り、バリデータによって正しいものを取り除いて負例を生成する。このうち、AD が正例と判定したものの割合。一番エラー率の高い MNIST で、失敗したインスタンスが多くなっている。

次に、人工的に作成したデータで AD の正確さを検証した (Table 2)。具体的には、全て正例/負例のデータセットに対して AD の type-1/2 エラーを計測した。全体に渡ってあるていど低いエラー率が観測され、またエラー率がおおよそ失敗の数に対応する傾向があることが分かる。

9. Related Work

ニューラルネットを探索と組み合わせる既存手法には、まず NN を組合せ最適化問題の探索に直接使う手法、たとえば有名な NN ベース巡回セールスマンソルバ [Hopfield and Tank, 1985]がある。Neurosolver は探索ノードを NN のノードに対応させてハノイの塔などを解く [Bieszczad and Kuchar, 2015] いっぽう、これらのシステムの入力はもとから記号表現である。

その他の手法には、知識あり探索の下界関数を得るために NN を探索の**内側で**用いる手法がある。例えば、スライディングタイルパズルやルービックキューブ [Arfaee *et al.*, 2011]、古典プランニング [Satzger and Kramer, 2013]、あるいは囲碁 [Silver *et al.*, 2016]などに適用されている。

深層強化学習 (DRL) は、ビデオゲームなど複雑な問題を解くことができる[Mnih *et al.*, 2015, DQN]。ただし、これらのシステムは訓練と実行の際にシミュレータに自由にアクセスできることを仮定している。また、DRL システムは特定の初期状態に強く条件付けられているということが知られており、かつ長い時間がたった後に得られる報酬に弱いことが知られている。一方 LatPlan は、限られた数のラベルなしの状態遷移例を与えられるだけで自ら環境のモデル (≈ シミュレータ) を生成し、そのなかで探索を行ってプランを生成するため、外部シミュレータを必要としないばかりか、DRL のように報酬関数を必要とせず、かつ一回の訓練で複数の初期状態・ゴールの問題を解け、また AlphaGo のように専門家による例も必要としない。さらには、DQN や AlphaGo は「石の置き場所」「ボタン」「コントロールレバー (上下左右)」など、決められたセットのアクションの存在を仮定しているが、LatPlan はこれらも必要としない。

10. 結論

我々は LatPlan を提案した。LatPlan は、学習と計画の統合アーキテクチャであり、ラベルなし画像のセットから事前知識なしに古典的な計画問題を生成し、これを記号的ソルバで解き、その計画を人間が理解できる画像列として提示する。本論文では、画像ベースの探索問題（8 パズル、ハノイの塔、LightsOut）を用いてこのようなアプローチが実現可能であることを示した。本論文の主な技術的貢献は以下のとおりであ

る。(1) **Gumbel-Softmax を用いて生データと記号表現の双方向マッピングを学習する SAE 実装。**8 パズルでは、42x42 の訓練画像の「要点」が命題表現に圧縮され、プランナはその命題表現上で動作する。(2) アクション記号とアクションモデルを同時に学習する AMA_2 システム。これは少数の状態遷移例から、状態遷移の分類ラベルであるアクション記号と、それらの表す効果、および前提条件を学習する。

このシステムがもつ唯一の仮定は、対象となる環境が (1) 完全情報・決定的環境であり、かつ (2) ニューラルネットが与えられたデータから学習できることのみである。従って、我々は、同じシステムが、様々な種類の問題を、一行もコードに変更を加えること無く等しく解くことが出来ることを示した。言い換えるならば、**LatPlan は、ドメイン非依存の画像ベース古典プランニングソルバである。**様々な問題に対して **記号的システムと直接互換性のある** 論理的表現をラベルなしデータから直接生成できるこのようなシステムは、我々の知る限りでは LatPlan が初めてである。

我々はシミュレータ無し、エキスパートデータなし、報酬関数無し、アクション前後の状態を表す画像対のみが与えられたときに、深層学習を活用して A*のような古典的な探索アルゴリズムを使用する記号的プランニングを実現可能であることを示した。このアプローチのスケーラビリティを判断するためにはより多くの検証が必要だが、これは記号的・非記号的推論のギャップを橋渡しする重要な第一歩であり、将来の研究のための多くの道を開くと考えている。

参 考 文 献

[Arfaee *et al.*, 2011] Shahab Jabbari Arfaee, Sandra Zilles, and Robert C. Holte. Learning Heuristic Functions for Large State Spaces. *Artificial Intelligence*, 175(16-17):2075–2098, 2011.

[Asai and Fukunaga, 2015] Masataro Asai and Alex Fukunaga. Solving Large-Scale Planning Problems by Decomposition and Macro Generation. In *ICAPS*, Jerusalem, Israel, June 2015.

[Asai and Fukunaga, 2018] Masataro Asai and Alex Fukunaga. Classical Planning in Deep Latent Space: Bridging the Subsymbolic-Symbolic Boundary. In *AAAI*, New Orleans, USA, February 2018.

[Bacchus, 2000] Fahiem Bacchus. Subset of PDDL for the AIPS2000 Planning Competition. In *IPC*, 2000.

[Bieszczad and Kuchar, 2015] Andrzej Bieszczad and Skyler Kuchar. Neurosolver Learning to Solve Towers of Hanoi Puzzles. In *IJCCI*, volume 3, pages 28–38. IEEE, 2015.

[Bonet, 2013] Blai Bonet. An Admissible Heuristic for SAS+ Planning Obtained from the State Equation. In *IJCAI*, 2013.

[Cresswell *et al.*, 2013] Stephen Cresswell, Thomas Leo McCluskey, and Margaret Mary West. Acquiring planning domain models using **LOCM**. *Knowledge Eng. Review*, 28(2):195–213, 2013.

[Cullen and Bryman, 1988] J Cullen and A Bryman. The knowledge acquisition bottleneck: Time for reassessment? *Expert Systems*, 5(3), August 1988.

[Deng *et al.*, 2013] Li Deng, Geoffrey Hinton, and Brian Kingsbury. New Types of Deep Neural Network Learning for Speech Recognition and Related Applications: An Overview. In *ICASSP*, pages 8599–8603. IEEE, 2013.

[Elkan and Noto, 2008] Charles Elkan and Keith Noto. Learning Classifiers from Only Positive and Unlabeled Data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 213–220. ACM, 2008.

[Graves *et al.*, 2016] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. Hybrid Computing using a Neural Network with Dynamic External Memory. *Nature*, 538(7626):471–476, 2016.

[Gumbel and Lieblein, 1954] Emil Julius Gumbel and Julius Lieblein. Statistical theory of extreme values and some practical applications: a series of lectures. 1954.

[Hart *et al.*, 1968] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *Systems Science and Cybernetics, IEEE Transactions on*, 4(2):100–107, 1968.

[Helmert and Domshlak, 2009] Malte Helmert and Carmel Domshlak. Landmarks, Critical Paths and Abstractions: What's the Difference Anyway? In *ICAPS*, 2009.

[Helmert *et al.*, 2007] Malte Helmert, Patrik Haslum, and Jörg Hoffmann. Flexible Abstraction Heuristics for Optimal Sequential Planning. In *ICAPS*, pages 176–183, 2007.

[Helmert, 2004] Malte Helmert. A Planning Heuristic Based on Causal Graph Analysis. In *ICAPS*, pages 161–170, 2004.

[Helmert, 2006] Malte Helmert. The Fast Downward Planning System. *J. Artif. Intell. Res.(JAIR)*, 26:191–246, 2006.

[Hinton and Salakhutdinov, 2006] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786):504–507, 2006.

[Hoffmann and Nebel, 2001] Jörg Hoffmann and Bernhard Nebel. The FF Planning System: Fast Plan Generation through Heuristic Search. *J. Artif. Intell. Res.(JAIR)*, 14:253–302, 2001.

[Hopfield and Tank, 1985] John J Hopfield and David W Tank. "Neural" Computation of Decisions in Optimization Problems. *Biological Cybernetics*, 52(3):141–152, 1985.

[Jang *et al.*, 2017] Eric Jang, Shixiang Gu, and Ben Poole. Categorical Reparameterization with Gumbel-Softmax. In *ICLR*, 2017.

[Kingma and Welling, 2013] Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. In *ICLR*, 2013.

[Kingma *et al.*, 2014] Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-Supervised Learning with Deep Generative Models. In *NIPS*, pages 3581–3589, 2014.

[Konidaris *et al.*, 2014] George Konidaris, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Constructing Symbolic Representations for High-Level Planning. In *AAAI*, pages 1932–1938, 2014.

[LeCun *et al.*, 1998] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-Based Learning Applied to Document Recognition. *Proc. of the IEEE*, 86(11):2278–2324, 1998.

[Lindsay *et al.*, 2017] Alan Lindsay, Jonathon Read, Joao F Ferreira, Thomas Hayton, Julie Porteous, and Peter J Gregory. Framer: Planning Models from Natural Language Action Descriptions. In *ICAPS*, 2017.

[Maddison *et al.*, 2014] Chris J Maddison, Daniel Tarlow, and Tom Minka. A* sampling. In *NIPS*, pages 3086–3094, 2014.

[McDermott, 2000] Drew V. McDermott. The 1998 AI Planning Systems Competition. *AI Magazine*, 21(2):35–55, 2000.

[Mnih *et al.*, 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-Level Control through Deep Reinforcement Learning. *Nature*, 518(7540):529–533, 2015.

[Mourão *et al.*, 2012] Kira Mourão, Luke S. Zettlemoyer, Ronald P. A. Petrick, and Mark Steedman. Learning STRIPS Operators from Noisy and Incomplete Observations. In *UAI*, pages 614–623, 2012.

[Ren *et al.*, 2015] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-time Object Detection with Region Proposal Networks. In *NIPS*, pages 91–99, 2015.

[Richter *et al.*, 2008] Silvia Richter, Malte Helmert, and Matthias Westphal. Landmarks Revisited. In *AAAI*, 2008.

[Satzger and Kramer, 2013] Benjamin Satzger and Oliver Kramer. Goal Distance Estimation for Automated Planning using Neural Networks and Support Vector Machines. *Natural Computing*, 12(1):87–100, 2013.

[Sievers *et al.*, 2012] Silvan Sievers, Manuela Ortlieb, and Malte Helmert. Efficient Implementation of Pattern Database Heuristics for Classical Planning. In *SOCS*, 2012.

[Silver *et al.*, 2016] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature*, 529(7587):484–489, 2016.

[Srivastava *et al.*, 2015] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised Learning of Video Representations using LSTMs. In *ICML*, pages 843–852, 2015.

[Steels, 2008] Luc Steels. The Symbol Grounding Problem has been Solved. So What's Next? In Manuel de Vega, Arthur Glenberg, and Arthur Graesser, editors, *Symbols and Embodiment*. Oxford University

Press, 2008.

[Vincent *et al.*, 2008] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and Composing Robust Features with Denoising Autoencoders. In *ICML*, pages 1096–1103. ACM, 2008.

[Yang *et al.*, 2007] Qiang Yang, Kangheng Wu, and Yunfei Jiang. Learning Action Models from Plan Examples using Weighted MAX-SAT. *Artificial Intelligence*, 171(2-3):107–143, 2007.