A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning

Stéphane Ross, Geoffrey J. Gordon, J. Andrew Bagnell

Presented by Markus Loide and Martin Liivak

Overview

- Glossary
- Preceding works
- Dataset Aggregation Algorithm
- Experiments

Some Definitions

- Policy - a state-action mapping, state is the agent’s idea of the world
- Stationary - unchanging
- Stochastic - having a random probability distribution or pattern
- Expert - some manner of active guidance

Imitation Learning

Learner needs to be able to predict future sequences shown by the expert

If learner’s prediction affects future, it may be put into unencountered states

This may cause accumulating mistakes

Traditional Supervised Approach to Imitation

Change in distribution is ignored

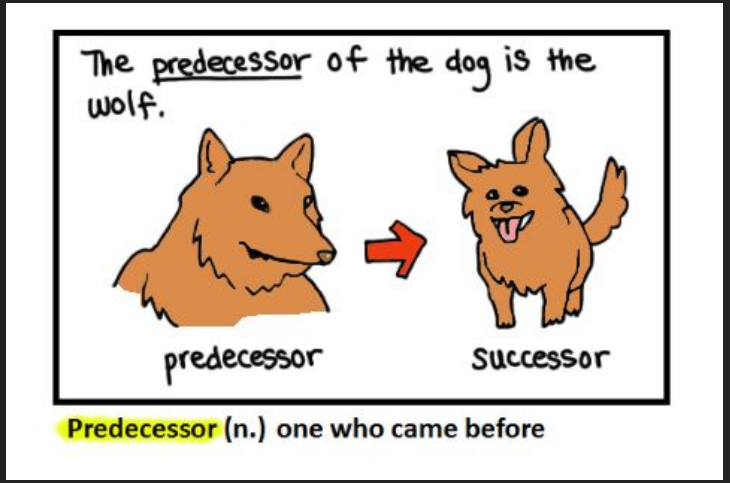
Single stationary policy is trained

Policy performs well under distribution of states encountered by the expert

Preceding Algorithms

The following are algorithms upon which this algorithm is iteratively built:

- Search based learning algorithm (SEARN)
- Forward training
- Stochastic Mixing Iterative Learning (SMILe)



Forward Training algorithm

Trains a non-stationary policy iteratively

Each policy is trained on states induced by previous policies

This algorithm is impractical when there’s a large number of iterations meaning it cannot be applied to real-world applications

Stochastic Mixing Iterative Learning (SMILe)

In principle similar to forward training algorithm

A stochastic policy is trained over the iterations

Policy is updated from previous policies

Dataset Aggregation Algorithm (DAgger)

Initially:

- Expert’s policy is used to gather a dataset of trajectories

Iteratively:

- New policy is trained that best mimics existing trajectories
- New policy is used to collect more trajectories, which are added to the dataset

DAgger

Initialize $\mathcal{D} \leftarrow \emptyset$.
Initialize $\hat{\pi}_1$ to any policy in Π.
for $i = 1$ **to** N **do**
  Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$.
  Sample T-step trajectories using π_i.
  Get dataset $\mathcal{D}_i = \{(s, \pi^*(s))\}$ of visited states by π_i and actions given by expert.
  Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{D}_i$.
  Train classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$.
end for
Return best $\hat{\pi}_i$ on validation.

Algorithm 3.1: DAGGER Algorithm.

DAgger

DAgger collects a dataset at each iteration under the current policy

The next policy is trained under the aggregate of all collected datasets.

Follow-The-Leader algorithm, best policy is picked in hindsight

DAgger Theoretical Analysis

Imitation learning is reduced to no-regret online learning

Regret - knowledge of a better action choice that could have been taken

No-regret algorithms guarantee policy which has good performance guarantees under its own distribution of states

Online Learning

Online learning algorithm must iteratively provide i-th policy and its loss

This produces a sequence of policies and loss functions

No Regret Algorithm

A no-regret algorithm produces a sequence of policies such that the average regret with respect to the best policy in hindsight approaches 0 asymptotically

Can be used to find a policy which has good performance guarantees under its own distribution of states in the imitation learning setting

Experiments

The efficacy and scalability of DAgger is demonstrated by applying it to two imitation learning problems:

- Super Tux Kart
- Super Mario Bros.

Additionally its efficacy was demonstrated on a sequence labeling task:

- Handwriting recognition

Super Tux Kart

Super Tux Kart is a 3D racing game similar to the popular Mario Kart.

Performances are compared on a specific race track.

The kart can fall off the track.

Super Tux Kart

Human expert provides correct steering demonstration in the form of analog joystick value in [-1, 1]

Linear controllers are used as the base learner that update the steering

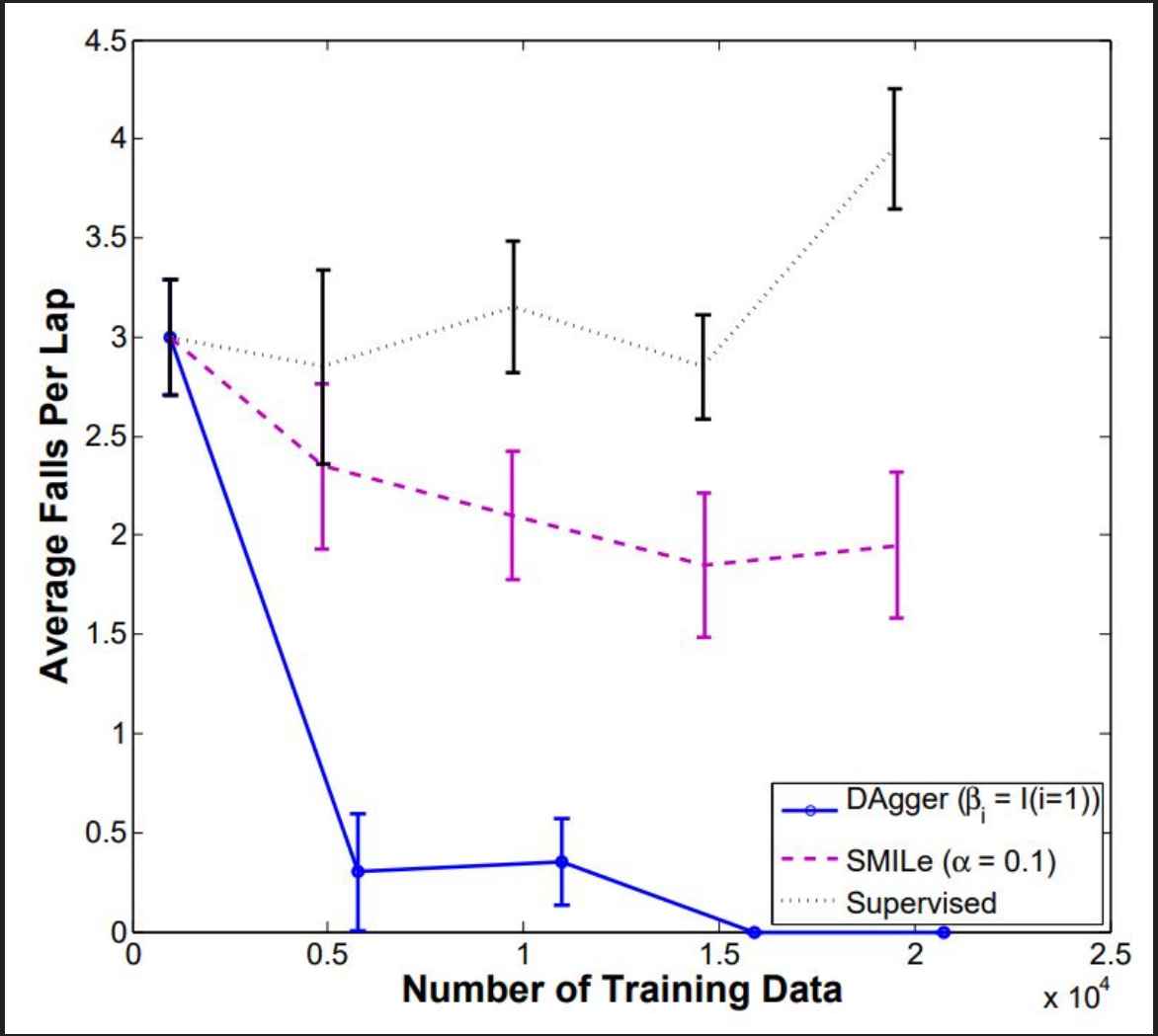
Input is the original 800x600 image resized to 25x19, output is found by minimizing a ridge regression objective

Super Tux Kart

The algorithm performances are measured in terms of the average number of falls per lap

DAgger algorithm is compared to the SMILe algorithm and a baseline supervised learning.

First two methods are run for 20 iterations with one lap of training used per iteration


Average Falls Per Lap
Number of Training Data
x 10^4
DAgger (β_i = I(i=1))
SMILe (α = 0.1)
Supervised
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5

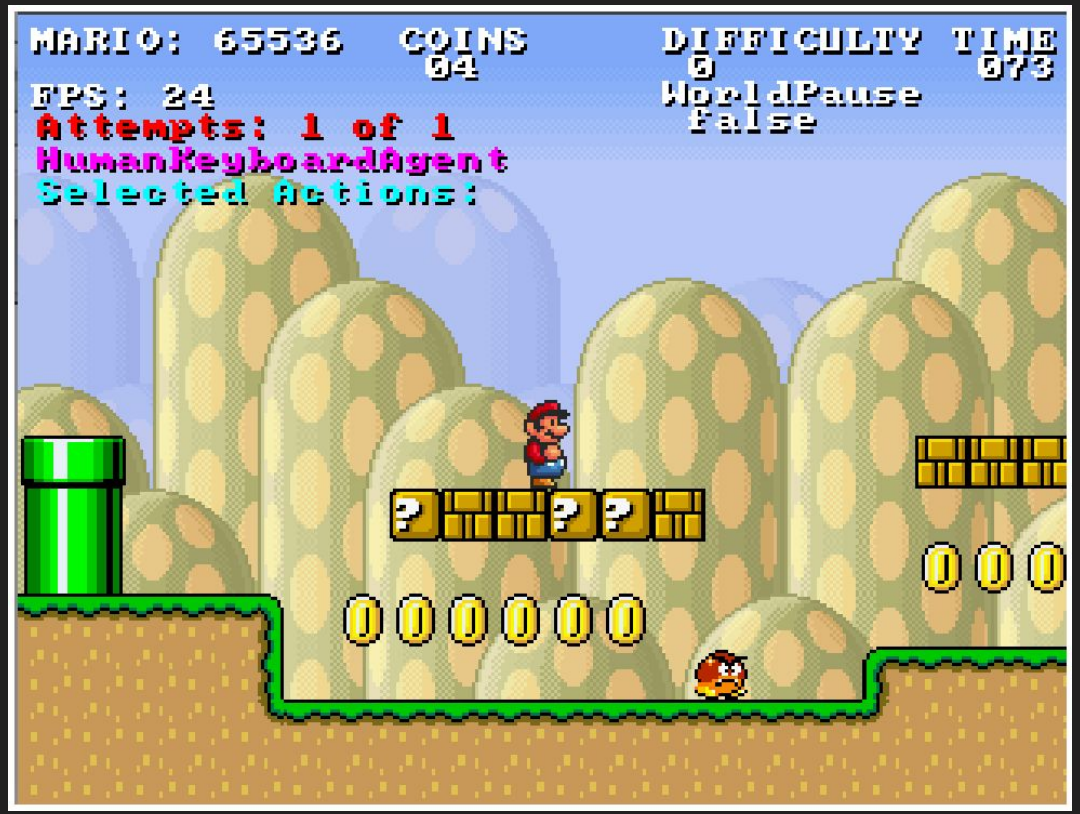

Features
0:40:4
Lap
1/1
1st

Super Mario Bros

Popular 2D platformer

Played using a simulator

The expert is a near-optimal planning algorithm



Super Mario Bros

Action consists of “what buttons are pressed”

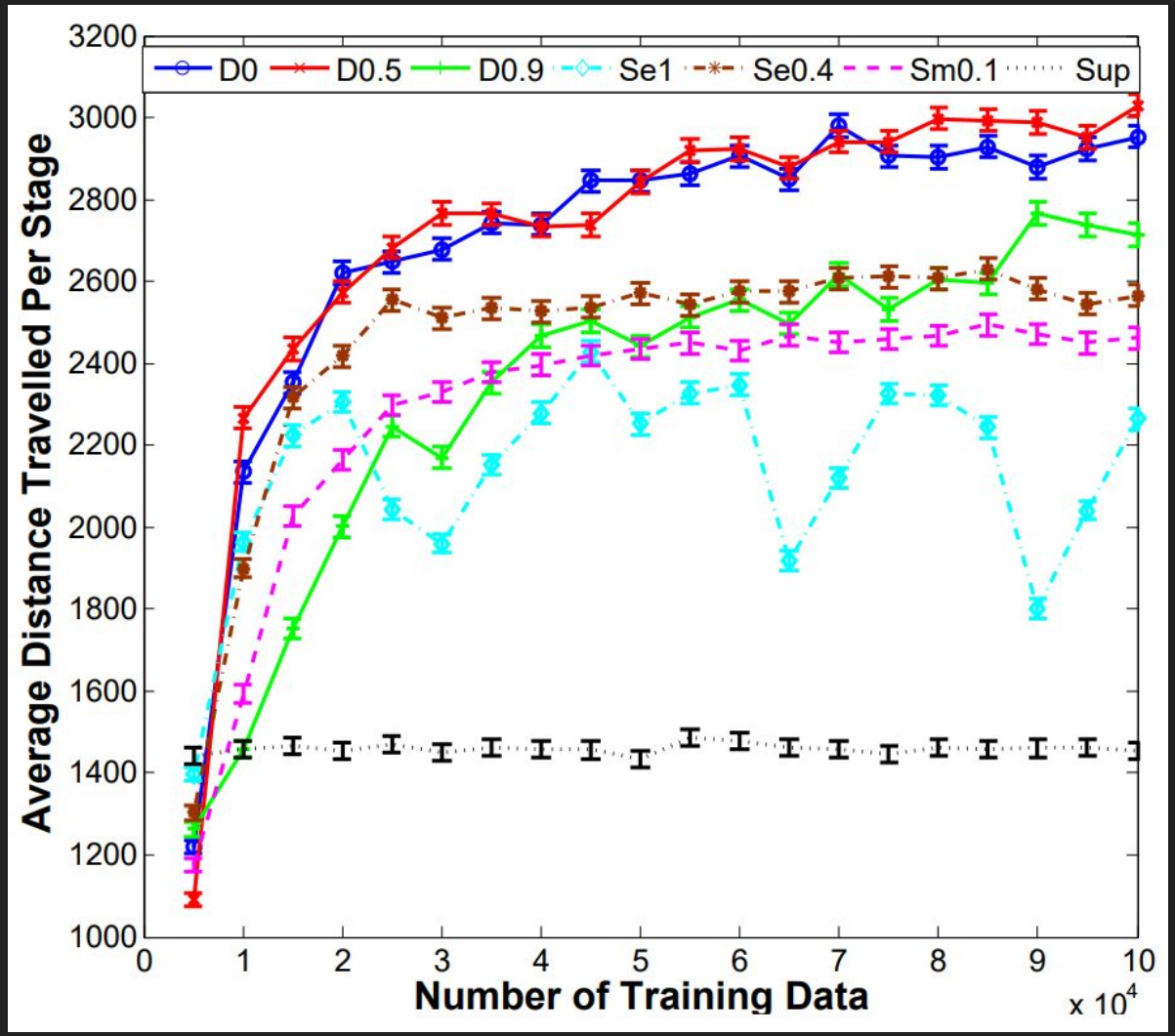
4 independent linear SVM as the base learners

Total of 27152 binary features

Super Mario Bros

Distance as the measure of success

More data is not always better


D0
D0.5
D0.9
Se1
Se0.4
Sm0.1
Sup
Average Distance Travelled Per Stage
Number of Training Data
x 10^4
3200
3000
2800
2600
2400
2200
2000
1800
1600
1400
1200
1000
0
1
2
3
4
5
6
7
8
9
10


MARIO: 1024
COINS 07
DIFFICULTY 1
TIME 037
FPS: 24
Attempt: 1 of 1
AgentLinear
Selected Actions:
WorldPause false
RIGHT
JUMP
SPEED

Handwriting Recognition

DAgger’s efficacy is demonstrated on a structured prediction problem

Task is to recognize handwritten words given a sequence of letters

Dataset of about 6600 words (52000 characters) is used for 10-fold training

Learner performance was measured in terms of character accuracy on test folds.

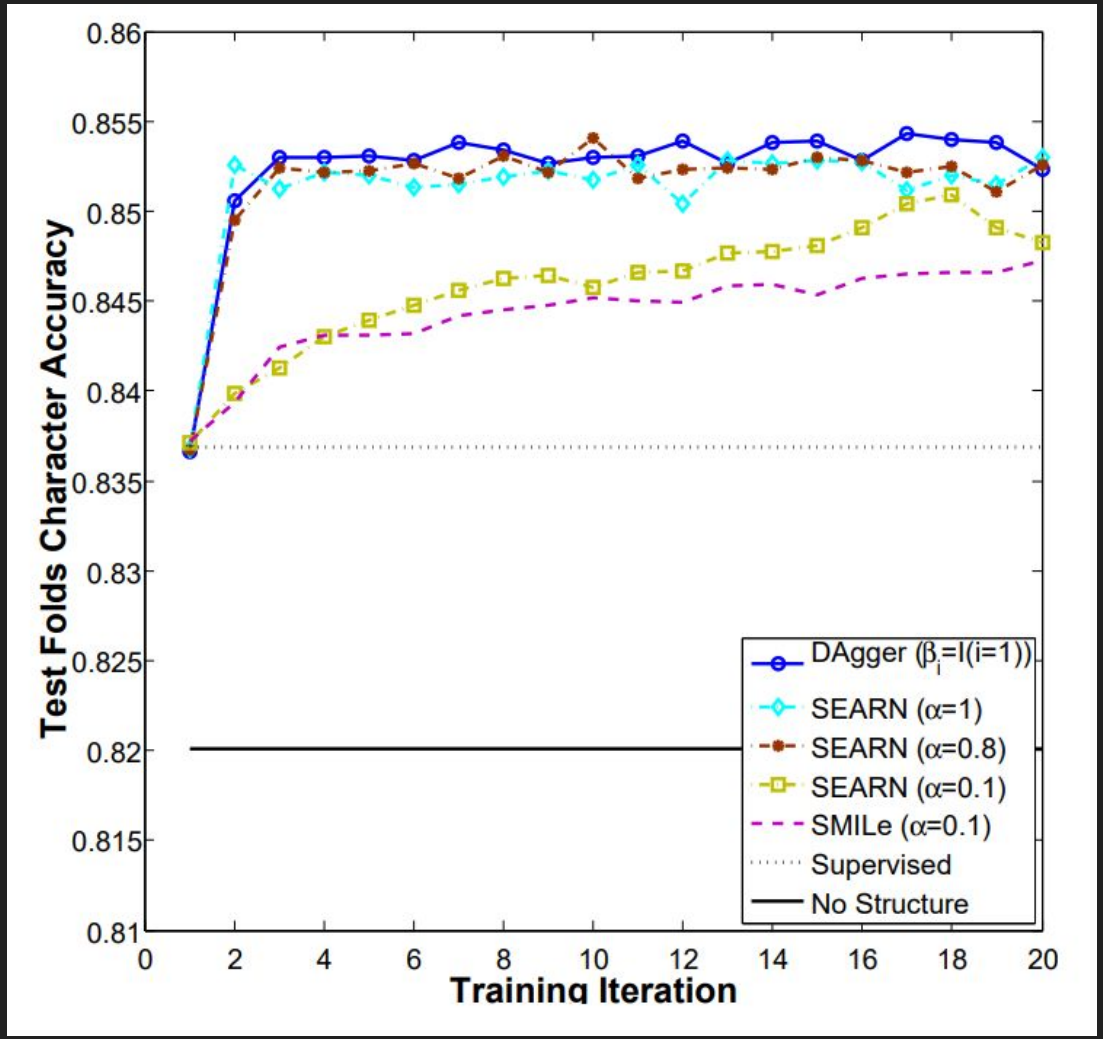

Unexpected
Volcanic
Embraces

Handwriting Recognition

Words are predicted by predicting each character in sequence from left to right

A linear multiclass SVM is used as the learner

DAgger is compared to SEARN and SMILe algorithms in addition to two baseline methods


Test Folds Character Accuracy
Training Iteration
0.86
0.855
0.85
0.845
0.84
0.835
0.83
0.825
0.82
0.815
0.81
0
2
4
6
8
10
12
14
16
18
20
DAgger (β_i=I(i=1))
SEARN (α=1)
SEARN (α=0.8)
SEARN (α=0.1)
SMILe (α=0.1)
Supervised
No Structure

Future work

Inverse Optimal Control techniques to learn a cost function for a planner to aid prediction

Thank you for listening!