Behavioral Cloning

Artem Sokolov

Institute for Computational Linguistics, Heidelberg University

11 October 2018

The purpose of imitation learning is to efficiently learn a desired behavior by imitating an expert's behavior.

We want to

- in general: learn how to navigate an environment like the expert
- in particular (structured prediction): make inference tractable

- finite horizon MDP $(\mathcal{S}, \mathcal{A}, P, C, \rho_0, T)$
  - ➡ $\mathcal{S}$ – set of S states
  - ➡ $\mathcal{A}$ – set of A actions
  - ➡ $P_t : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ – transition distribution
  - ➡ $C_t : \mathcal{S} \times \mathcal{A} \to [0,1]$ – cost distribution
  - ➡ $\rho_0 : \mathcal{S} \to [0,1]$ – initial state distribution
  - ➡ T – maximal horizon
- π^* – expert policy we wish to mimic
- $\pi : \mathcal{S} \times \mathcal{A} \to [0,1]$ – some stochastic policy
- d_π^t – state distribution at time step t (vector in $\mathbb{R}^S$)
- $d_\pi = \frac{1}{T} \sum_{t=1}^{T} d_\pi^t$ – state visitation frequency at time step t
- $J(\pi) = \sum_{t=1}^{T} \mathbb{E}_{s_t \sim d_\pi^t} \mathbb{E}_{a_t \sim \pi(s_t)} [C(s_t, a_t)]$ – cost we wish to minimize

- finite horizon MDP $(\mathcal{S}, \mathcal{A}, P, C, \rho_0, T)$
  - ➡ $\mathcal{S}$ – previous word and tag, and current word, $\langle x_{i-1}, y_{i-1}, x_i \rangle$
  - ➡ $\mathcal{A}$ – set of possible POS tags
  - ➡ $P_t : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ – deterministic:
    $P(\langle x_i, y_i, x_{i+1} \rangle \mid \langle x_{i-1}, y_{i-1}, x_i \rangle, y_i) = 1$
  - ➡ $C_t : \mathcal{S} \times \mathcal{A} \to [0, 1]$ – hamming cost
    $C(\langle x_{i-1}, y_{i-1}, x_i \rangle, y_i) = \mathbb{I}[y_i \neq y_i^*]$
  - ➡ $\rho_0 : \mathcal{S} \to [0, 1]$ – say, uniform
  - ➡ T – maximum number of input tokens
- π^* – deterministically outputs the correct label,
  $\pi^*(\langle x_{i-1}, y_{i-1}, x_i \rangle) = y_i^*$
- $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$ – e.g. deterministic $\pi(s) = \arg\max_a score(s, a)$

Expected cost:

$$J(\pi) = \sum_{t=1}^{T} \mathbb{E}_{s_t \sim d_\pi^t} \mathbb{E}_{a_t \sim \pi(s_t)} [C(s_t, a_t)]$$

May seem unintuitive at first if you used to think in terms of trajectories. Define:

- $\tau = (s_1, a_1, \ldots, a_{T-1}, s_T)$ – trajectory
- trajectory distribution

$$\rho_\pi(\tau) = \rho_0(s_1) \prod_{t=2}^{T} \pi(a_{t-1}|s_{t-1}) P_{t-1}(s_t|s_{t-1}, a_{t-1})$$

- state distribution

$$d_t^\pi(s_t) = \sum_{\{s_i, a_i\}_{i \leq t-1}} \rho_0(s_1) \prod_{i=2}^{t-1} \pi(a_{i-1}|s_{i-1}) P_{i-1}(s_i|s_{i-1}, a_{i-1})$$

$$J(\pi) = \mathbb{E}_{\tau\sim\rho}[\sum_{t=1}^{T} C(s_t, a_t)]$$ this is expected cost by definition

$$= \sum_{\tau} \rho_\pi(\tau) \sum_{t=1}^{T} C(s_t, a_t)$$

$$= \sum_{t=1}^{T} \sum_{\tau} \rho_\pi(\tau) C(s_t, a_t)$$

$$= \sum_{t=1}^{T} \sum_{\{s_i,a_i\}_{i\leq t-1}} \sum_{\{s_i,a_i\}_{i\geq t}} \rho_\pi(\tau) C(s_t, a_t)$$

$$= \sum_{t=1}^{T} \sum_{\{s_i,a_i\}_{i\leq t-1}} \sum_{\{s_i,a_i\}_{i\geq t}} \rho_0(s_1) \prod_{i=2}^{T} \pi(a_{i-1}|s_{i-1}) P_{i-1}(s_i|s_{i-1}, a_{i-1}) C(s_t, a_t)$$

$$= \sum_{t=1}^{T} \sum_{\{s_i,a_i\}_{i<t-1}} \sum_{a_t, s_t} \rho_0(s_1) \prod_{i=2}^{t-1} \pi(a_{i-1}|s_{i-1}) P_{i-1}(s_i|s_{i-1}, a_{i-1}) \pi(a_t|s_t) C(s_t, a_t$$

$$=\sum_{t=1}^{T} \sum_{\{s_i,a_i\}_{i\leq t-1}} \sum_{a_t,s_t} \rho_0(s_1) \prod_{i=2}^{t-1} \pi(a_{i-1}|s_{i-1}) P_{i-1}(s_i|s_{i-1},a_{i-1}) \pi(a_t|s_t) C(s_t,a_t)$$

$$= \sum_{t=1}^{T} \sum_{s_t \sim d_\pi^t} \sum_{a_t \sim \pi(s_t)} d_\pi^t(s_t) \pi(a_t|s_t) C(s_t,a_t)$$

$$= \sum_{t=1}^{T} \mathbb{E}_{s_t \sim d_\pi^t} \mathbb{E}_{a_t \sim \pi(s_t)} [C(s_t,a_t)]$$

IL with Behavioral Cloning

1. converting structured prediction into a search problem with specified search space and actions;
2. defining structured features over each state to capture the inter-dependency between output variables;
3. constructing a reference policy based on training data;
4. learning a policy that imitates the reference policy.

The question is – what is 'imitates'?

1. converting structured prediction into a search problem with specified search space and actions; $\leftarrow$reduction to classification
2. defining structured features over each state to capture the inter-dependency between output variables;
3. constructing a reference policy based on training data;
4. learning a policy that imitates the reference policy.

The question is – what is ‘imitates’?

Very natural thing to do:

- let's learn to predict the same thing as the expert!
- after all, isn't it the very meaning of imitation?

Very natural thing to do:

- let's learn to predict the same thing as the expert!
- after all, isn't it the very meaning of imitation?
- define an error of not predicting the expert's actions:

$$\text{simplest case } e(s,a) = \mathbb{I}[a \neq \pi^*(s)]$$
$$e_\pi(s) = \mathbb{E}_{a\sim\pi(s)}[e(s,a)]$$

Very natural thing to do:

- let's learn to predict the same thing as the expert!
- after all, isn't it the very meaning of imitation?
- define an error of not predicting the expert's actions:

$$\text{simplest case } e(s,a) = \mathbb{I}[a \neq \pi^*(s)]$$
$$e_\pi(s) = \mathbb{E}_{a\sim\pi(s)}[e(s,a)]$$

- learn by driving this error to minimum:

$$\hat{\pi} = \arg\min_\pi \mathbb{E}_{s\sim d_{\pi^*}}[e_\pi(s)]$$

Very natural thing to do:

- let's learn to predict the same thing as the expert!
- after all, isn't it the very meaning of imitation?
- define an error of not predicting the expert's actions:

$$\text{simplest case } e(s,a) = \mathbb{I}[a \neq \pi^*(s)]$$
$$e_\pi(s) = \mathbb{E}_{a\sim\pi(s)}[e(s,a)]$$

- learn by driving this error to minimum:

$$\hat{\pi} = \arg\min_\pi \mathbb{E}_{s\sim d_{\pi^*}}[e_\pi(s)]$$

- note: the state distribution comes from the expert (or labeled data)

Very natural thing to do:

- let's learn to predict the same thing as the expert!
- after all, isn't it the very meaning of imitation?
- define an error of not predicting the expert's actions:

$$\text{simplest case } e(s,a) = \mathbb{I}[a \neq \pi^*(s)]$$
$$e_\pi(s) = \mathbb{E}_{a\sim\pi(s)}[e(s,a)]$$

- learn by driving this error to minimum:

$$\hat{\pi} = \arg\min_\pi \mathbb{E}_{s\sim d_{\pi^*}}[e_\pi(s)]$$

- note: the state distribution comes from the expert (or labeled data)

Question: which known approach in NMT does this correspond to?

Very natural thing to do:

- let's learn to predict the same thing as the expert!
- after all, isn't it the very meaning of imitation?
- define an error of not predicting the expert's actions:

$$\text{simplest case } e(s,a) = \mathbb{I}[a \neq \pi^*(s)]$$
$$e_\pi(s) = \mathbb{E}_{a\sim\pi(s)}[e(s,a)]$$

- learn by driving this error to minimum:

$$\hat{\pi} = \arg\min_\pi \mathbb{E}_{s\sim d_{\pi^*}}[e_\pi(s)]$$

- note: the state distribution comes from the expert (or labeled data)

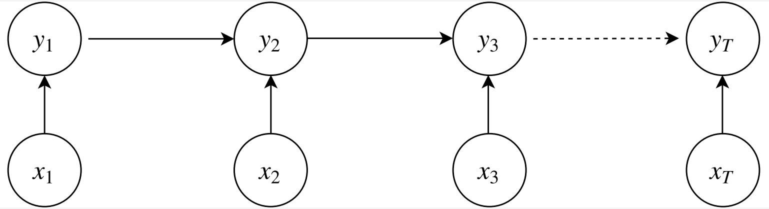
Teacher forcing

Behavioral cloning may not work

- we will show two examples where the error is unacceptably high
- intuitively this happens because the learner cannot recover from unseen situations
- this phenomenon is sometimes called ‘exposure bias’
- it is hypothesized that this could be the reason for NMT hallucinations

- Given $\mathbf{x} = (x_1, x_2, \ldots, x_T)$, predict $\mathbf{y} = (y_1, y_2, \ldots, y_T) \in \{0, 1\}^T$
- Assumption $(\mathbf{x}, \mathbf{y}) \sim \mathcal{D}$ iid
- Prediction $\hat{y}_i = f_i(\mathbf{x})$
- Loss of classifier: Hamming loss $\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}}\left[\sum_{t=1}^{T} \mathbb{I}[y_i \neq \hat{y}_i]\right]$

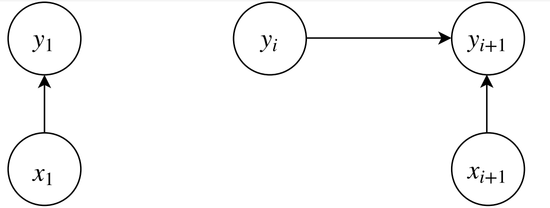
Structure of $\mathcal{D}$:



[Kääriäinen'05]

To achieve a reduction to binary classification, let's choose a class of binary classifiers for each position i:

$$
\begin{aligned}
f_1(\mathbf{x}) :& \mathcal{X}_1 \rightarrow \{0,1\} \\
f_i(\mathbf{x}) :& \{0,1\} \times \mathcal{X}_i \rightarrow \{0,1\}
\end{aligned}
$$



1. Obtain a set of training examples $\{\mathbf{x}, \mathbf{y}\}$ sampled from $\mathcal{D}$
2. Learn:

$$f_1(\mathbf{x}) : \mathcal{X}_1 \to \{0, 1\}$$
$$f_i(\mathbf{x}) : \{0, 1\} \times \mathcal{X}_i \to \{0, 1\}$$

3. Given a test example $\mathbf{x}$, predict

$$\hat{y}_1 = f_1(\mathbf{x})$$
$$\hat{y}_{i+1} = f_{i+1}(\hat{y}_i, \mathbf{x})$$

1. Obtain a set of training examples $\{\mathbf{x}, \mathbf{y}\}$ sampled from $\mathcal{D}$
2. Learn:

$$f_1(\mathbf{x}) : \mathcal{X}_1 \to \{0,1\}$$
$$f_i(\mathbf{x}) : \{0,1\} \times \mathcal{X}_i \to \{0,1\}$$

3. Given a test example $\mathbf{x}$, predict

$$\hat{y}_1 = f_1(\mathbf{x})$$
$$\hat{y}_{i+1} = f_{i+1}(\hat{y}_i, \mathbf{x})$$

- assume we learned the classifiers f_i
- and all f_i happen to have some small error, $\mathbb{E}[f_i(y_i, x_i) \neq y_i^*] = \epsilon$
- **Question**: what will be the Hamming loss on the full sequence $\mathbf{x}$, if the f_i is applied in every position? Basically, how successful is our reduction?

- will obtain a lower bound for reducing structural sequence learning to binary classification
- such reductions permit reusing known results from other tasks
- lower bounds highlighting difficulties

Simplifications:

- assume for simplicity that

$$P(f_{i+1}(y_i, x_{i+1}) \neq y^*_{i+1}) \mid y_i) = \epsilon$$

  for any y_i (0 or 1)
- test time predictions: $\hat{y}_{i+1} = f(\hat{y}_i, x_i)$
- <u>in combination with the 1st asumption, this means $\hat{y}_{i+1}$ is biased to stick to whatever previous prediciton was</u>
- assume $f_1(x_1) = y_1$ and $\mathbf{y} = (y_1, y_1, \ldots, y_1)$ (all the same)
- define a random variable $z_i = \mathbb{I}[y_i \neq y^*_i]$, that flips with prob. ϵ and stays on the previous value with prob. $1 - \epsilon$

Simplifications:

- assume for simplicity that

$$P(f_{i+1}(y_i, x_{i+1}) \neq y^*_{i+1}) \mid y_i) = \epsilon$$

  for any y_i (0 or 1)
- test time predictions: $\hat{y}_{i+1} = f(\hat{y}_i, x_i)$
- in combination with the 1st asumption, this means $\hat{y}_{i+1}$ is biased to stick to whatever previous prediciton was
- assume $f_1(x_1) = y_1$ and $\mathbf{y} = (y_1, y_1, \ldots, y_1)$ (all the same)
- define a random variable $z_i = \mathbb{I}[y_i \neq y^*_i]$, that flips with prob. ϵ and stays on the previous value with prob. $1 - \epsilon$

They form a Markov Reward (here, loss) Process with the transition matrix:

Simplifications:

- assume for simplicity that

$$P(f_{i+1}(y_i, x_{i+1}) \neq y^*_{i+1}) \mid y_i) = \epsilon$$

  for any y_i (0 or 1)
- test time predictions: $\hat{y}_{i+1} = f(\hat{y}_i, x_i)$
- in combination with the 1st asumption, this means $\hat{y}_{i+1}$ is biased to stick to whatever previous prediciton was
- assume $f_1(x_1) = y_1$ and $\mathbf{y} = (y_1, y_1, \ldots, y_1)$ (all the same)
- define a random variable $z_i = \mathbb{I}[y_i \neq y^*_i]$, that flips with prob. ϵ and stays on the previous value with prob. $1 - \epsilon$

They form a Markov Reward (here, loss) Process with the transition matrix:

$$P = \begin{pmatrix} 1-\epsilon & \epsilon \\ \epsilon & 1-\epsilon \end{pmatrix}$$

Simplifications:

- assume for simplicity that

$$P(f_{i+1}(y_i, x_{i+1}) \neq y^*_{i+1}) \mid y_i) = \epsilon$$

  for any y_i (0 or 1)
- test time predictions: $\hat{y}_{i+1} = f(\hat{y}_i, x_i)$
- in combination with the 1st asumption, this means $\hat{y}_{i+1}$ is biased to stick to whatever previous prediciton was
- assume $f_1(x_1) = y_1$ and $\mathbf{y} = (y_1, y_1, \ldots, y_1)$ (all the same)
- define a random variable $z_i = \mathbb{I}[y_i \neq y^*_i]$, that flips with prob. ϵ and stays on the previous value with prob. $1 - \epsilon$

They form a Markov Reward (here, loss) Process with the transition matrix:

$$P = \begin{pmatrix} 1-\epsilon & \epsilon \\ \epsilon & 1-\epsilon \end{pmatrix}$$

After t transitions:

Simplifications:

- assume for simplicity that

$$P(f_{i+1}(y_i, x_{i+1}) \neq y^*_{i+1}) \mid y_i) = \epsilon$$

  for any y_i (0 or 1)
- test time predictions: $\hat{y}_{i+1} = f(\hat{y}_i, x_i)$
- in combination with the 1st asumption, this means $\hat{y}_{i+1}$ is biased to stick to whatever previous prediciton was
- assume $f_1(x_1) = y_1$ and $\mathbf{y} = (y_1, y_1, \ldots, y_1)$ (all the same)
- define a random variable $z_i = \mathbb{I}[y_i \neq y_i^*]$, that flips with prob. ϵ and stays on the previous value with prob. $1 - \epsilon$

They form a Markov Reward (here, loss) Process with the transition matrix:

$$P = \begin{pmatrix} 1-\epsilon & \epsilon \\ \epsilon & 1-\epsilon \end{pmatrix}$$

After t transitions:

$$P^t = \frac{1}{2} \begin{pmatrix} 1+(1-2\epsilon)^t & 1-(1-2\epsilon)^t \\ 1-(1-2\epsilon)^t & 1+(1-2\epsilon)^t \end{pmatrix}$$

Exercise: Proof this by induction with base case $t = 1$.

$$P^t = \frac{1}{2}\begin{pmatrix} 1+(1-2\epsilon)^t & 1-(1-2\epsilon)^t \\ 1-(1-2\epsilon)^t & 1+(1-2\epsilon)^t \end{pmatrix}$$

- $t \to \infty$

$$P^t \to \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

- $\epsilon = 0$

$$P^t = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

- $\epsilon = 1$
  - $t = 2k$

$$P^t = \frac{1}{2}\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$$

  - $t = 2k + 1$

$$P^t = \frac{1}{2}\begin{pmatrix} 0 & 2 \\ 2 & 0 \end{pmatrix}$$

$$P^t = \frac{1}{2}\begin{pmatrix} 1+(1-2\epsilon)^t & 1-(1-2\epsilon)^t \\ 1-(1-2\epsilon)^t & 1+(1-2\epsilon)^t \end{pmatrix}$$

- to track errors we're interested in elements (2,1) and (1,2)
- thanks to the assumptions, they look the same
- let's sum them up to figure out the error over the whole sequence:

$$\begin{aligned}\frac{1}{2}\sum_{t=1}^{T} 1-(1-2\epsilon)^t =& \frac{T}{2} - \frac{1}{2}\sum_{t=1}^{T}(1-2\epsilon)^t \\ =& \frac{T}{2} - \frac{1}{2}\left(\frac{1-(1-2\epsilon)^{T+1}}{1-(1-2\epsilon)} - 1\right) \\ =& \frac{T}{2} - \frac{1-(1-2\epsilon)^{T+1}}{4\epsilon} + \frac{1}{2}\end{aligned}$$

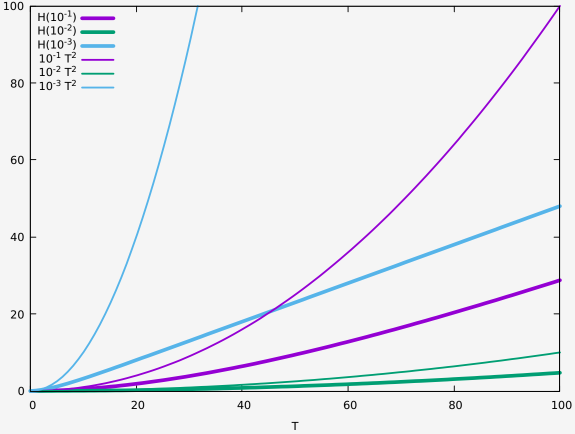
$$H = \mathbb{E}[\text{error}] = \frac{T}{2} - \frac{1-(1-2\epsilon)^{T+1}}{4\epsilon} + \frac{1}{2}$$

Exercise: Do a Taylor expansion of $(1-2\epsilon)^{T+1}$ up to $O(\epsilon^2)$

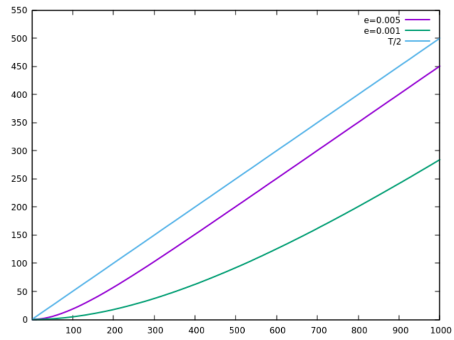
$$H = \mathbb{E}[\text{error}] = \frac{T}{2} - \frac{1-(1-2\epsilon)^{T+1}}{4\epsilon} + \frac{1}{2}$$

Exercise: Do a Taylor expansion of $(1-2\epsilon)^{T+1}$ up to $O(\epsilon^2)$

$$H \simeq \frac{1}{2}T(T+1)\epsilon + \cdots = \Theta(\epsilon T^2)$$

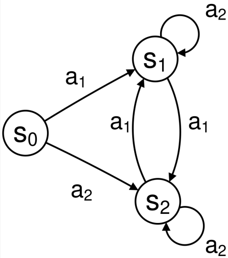


- you'd expect that if per-step error probability is ϵ then you'll make $\simeq \epsilon T$ of them over a sequence of length T
- instead **errors grow like ϵT^2 instead of ϵT**
- of course this is for small ϵ, and you can make $> T$ in total



- but for small ϵ the errors can quickly accumulate
- intuitively, once an error is committed the learner cannot recover

To see this phenomenon better, let's consider not a chain, but a real FST:



- expert policy π^* always picks a_1 in s_0, a_2 in s_1, and a_1 in s_2
- $d^*_\pi = (\frac{1}{T}, \frac{T-1}{T}, 0)$
- policy π with prob. $(1 - \epsilon T)$ executes a_1 in s_0, and a_2 in other states
- $\epsilon \leq 1/T$
- error of π: $\mathbb{E}_{s \sim d_{\pi^*}}[\mathbb{I}[\pi^*(s) \neq \pi(s)]] = \epsilon T \frac{1}{T} + \frac{T-1}{T} \cdot 0 + 0 \cdot 1 = \epsilon$
- so it could have been found by behavioral cloning
- T-step expected cost: 0 with prob. $(1 - \epsilon T)$, and T with prob. ϵT, so ϵT^2

[Ross & Bagnell'10]

Let $\hat{\pi}$ be such that $\mathbb{E}_{s\sim d_{\pi^*}}[e_{\hat{\pi}}(s)] \leq \epsilon$. Then $J(\hat{\pi}) \leq J(\pi^*) + \epsilon T^2$.

[Ross & Bagnell'10]

Let $\hat{\pi}$ be such that $\mathbb{E}_{s\sim d_{\pi^*}}[e_{\hat{\pi}}(s)] \leq \epsilon$. Then $J(\hat{\pi}) \leq J(\pi^*) + \epsilon T^2$.

- this is an upper bound, so maybe it's not that bad?

[Ross & Bagnell'10]

Let $\hat{\pi}$ be such that $\mathbb{E}_{s \sim d_{\pi^*}}[e_{\hat{\pi}}(s)] \leq \epsilon$. Then $J(\hat{\pi}) \leq J(\pi^*) + \epsilon T^2$.

- this is an upper bound, so maybe it's not that bad?
- actually, the examples above have just showed that this is tight
- so there are MDPs where the error actually scales as $O(\epsilon T^2)$

- assume $\epsilon_i = \mathbb{E}_{s \sim d_{\pi^*}}[e_{\hat{\pi}}(s)] = \epsilon, \forall i$
- consider two cases at time t:
  1. $\hat{\pi}$ did not make any mistake during $1 \ldots t-1$
  2. it did at least once
- p_t prob. of case 1, d_t state distribution of π in case 1, e_t prob of $\hat{\pi}$'s mistake at t in case 1
- d'_t state distribution of π^* in case 2, and e'_t prob. of mistake at t in case 2
- $d^t_{\pi^*} = p_t d_t + (1 - p_t) d'_t$
- $\epsilon_t = p_t e_t + (1 - p_t) e'_t$

$$
\begin{aligned}
J(\pi) &\leq \sum_{t=1}^{T}[p_t \mathbb{E}_{s\sim d_t}[C_{\hat{\pi}}(s)] + (1-p_t)\mathbb{E}_{s\sim d_t}[C_{\hat{\pi}}(s)]] \\
&\leq \sum_{t=1}^{T}[p_t \mathbb{E}_{s\sim d_t}[C_{\hat{\pi}}(s)] + (1-p_t)]] \qquad (C(.) \leq 1) \\
&\leq \sum_{t=1}^{T}[p_t \mathbb{E}_{s\sim d_t}[C_{\pi^*}(s) + e_t] + (1-p_t)]] \qquad \text{(making an error at } t\text{)} \\
&= \sum_{t=1}^{T}[p_t \mathbb{E}_{s\sim d_t}[C_{\pi^*}(s)] + p_t e_t + (1-p_t)]] \\
&\text{(note } J(\pi^*) = p_t \mathbb{E}_{d_t}[C_{\pi^*}] + (1-p_t)\mathbb{E}_{d'_t}[C_{\pi^*}]) \\
&\leq J(\pi^*) + \sum_{t=1}^{T}[p_t e_t + (1-p_t)] \leq J(\pi^*) + \sum_{t=1}^{T}[\epsilon_t + (1-p_t)] \\
&\leq J(\pi^*) + \sum_{t=1}^{T}[\epsilon_t + \sum_{i=1}^{t-1}\epsilon_i] = J(\pi^*) + \sum_{t=1}^{T}\sum_{i=1}^{t}\epsilon_i \leq J(\pi^*) + T^2\epsilon
\end{aligned}
$$