Case-Based Off-Policy Evaluation Using Prototype Learning

Abstract

Importance sampling (IS) is often used to perform off-policy evaluation but it is prone to several issues—especially when the behavior policy is unknown and must be estimated from data. Significant differences between target and behavior policies can result in uncertain value estimates due to, for example, high variance. Standard practices such as inspecting IS weights may be insufficient to diagnose such problems and determine for which type of inputs the policies differ in suggested actions and resulting values. To address this, we propose estimating the behavior policy for IS using prototype learning. The learned prototypes provide a condensed summary of the input-action space, which allows for describing differences between policies and assessing the support for evaluating a certain target policy. In addition, we can describe a value estimate in terms of prototypes to understand which parts of the target policy have the most impact on the estimate. We find that this provides new insights in the examination of a learned policy for sepsis management. Moreover, we study the bias resulting from restricting models to use prototypes, how bias propagates to IS weights and estimated values and how this varies with history length.

1 INTRODUCTION

Historical data on decisions and outcomes provide opportunities for evaluating policies for future decision-making. For example, the prospect of using patient records to evaluate new policies for medication dosing in sepsis management has attracted recent attention [Komorowski et al., 2018, Gottesman et al., 2019]. An example of off-policy evaluation (OPE), this amounts to estimating the value of a target policy based on data gathered under a different so-called behavior policy; see e.g., Thomas [2015] for an overview.

Importance sampling (IS) methods [Precup, 2000] perform OPE by weighting observed outcomes by the density ratio of the target policy and the behavior policy. IS methods are often preferred over alternatives which rely on modeling outcomes or covariate transitions, due to their simplicity and the fact that behavior policies often are controllable or human made. Similarly, the equivalent strategy of inverse-propensity weighting is fundamental to the study of causal effects [Rosenbaum and Rubin, 1983, Hirano et al., 2003].

In practice, it is difficult to assess the quality of an IS value estimate. When the behavior policy is unknown and must be estimated from data, conditions which guarantee good estimates are hard to meet and rely on untestable assumptions [Rosenbaum et al., 2010, Namkoong et al., 2020]. Standard practices of inspecting weights [Li et al., 2019] and removing outliers [Crump et al., 2009] give only aggregate or per-sample perspectives on potential issues and are often insufficient for domain experts to reason about the validity of the result. There is a clear need to better inspect and diagnose importance sampling estimates.

In this paper, we propose estimating the unknown behavior policy using prototype learning [Li et al., 2018b, Ming et al., 2019]. The learned prototypes are selected cases from the input data, readily interpretable by a domain expert and representative of the input-action space. In healthcare applications, the prototypes are trajectories of former patients, and a prototype-based estimate of the behavior policy is analogous to how physicians use experience from previous patients to treat new ones. While offering transparency, a prototype model is flexible enough to model behavior policies in large and/or sequential input spaces.

Our main contribution is to use learned prototypes as an OPE diagnostic tool. In addition to enabling interpretation of individual predictions, we show that (a) prototypes can be used to describe areas of similarities/dissimilarities between behavior and target policies; and (b) prototypes induce a soft clustering which can be used to explain differences in value

Submitted to the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022). **To be used for reviewing only**.

for different policies. We elaborate on this idea in Section 3.3 and demonstrate our method in Section 4.1 using an example of sepsis management. Further, we study the added bias of restricting the model class to use prototypes and how this bias propagates to the IS weights (Section 4.2).

2 OFF-POLICY EVALUATION

Policy evaluation refers to estimating the *value* $V(\pi)$ of a *target policy* $\pi \in \Pi$, as defined below. We focus on the sequential case, where a policy is used to select an *action* $A \in \mathcal{A} = \{1, \ldots, k\}$ after a *history* $H \in \mathcal{H}$, comprising a sequence of previous actions and *contexts* $X \in \mathcal{X}$. The history until time t is defined as $H_t \coloneqq (X_0, A_0, X_1, A_1, \ldots, X_t)$, with $H_0 = X_0$. A policy $\pi : \mathcal{H} \to \Delta_{\mathcal{A}}$ is a map from a history to a distribution over $\mathcal{A}$. In a medical example, a context X could correspond to information about a patient's state, an action A to a medical intervention, and the target policy π to new clinical guidelines.

The value of a policy π is defined as the expectation of a *reward* or *outcome* $R \in \mathbb{R}$, accumulated after acting according to π. Here, we study the special case where a single reward is awarded at the end of the sequence, $R = R_T$, but our results generalize to the case where rewards are given after every action. Under the distribution $p_\pi(X_0, A_0, \ldots, X_T, A_T, R) = p_\pi(H_T, A_T, R)$, induced by the policy π, the value is $V(\pi) \coloneqq \mathbb{E}_\pi[R]$.

Estimating $V(\pi)$ is trivial given a large enough number of samples from the target policy p_π. In off-policy evaluation (OPE), we have access to no such samples, but must estimate $V(\pi)$ using an observational dataset of m samples $\mathcal{D} = ((h^1_{t_1}, a^1_{t_1}, r^1), \ldots, (h^m_{t_m}, a^m_{t_m}, r^m))$, drawn according to a distribution $p_\mu(H_T, A_T, R)$, controlled by a *behavior policy* $\mu \in \Pi$. In the medical example, the behavior policy represents current clinical practice. In this work, the behavior policy μ is unknown and an estimate $\hat{p}_\mu(A \mid H)$ is learned from the samples $\mathcal{D}$.

A common method for OPE is *importance sampling* (IS).[1] The IS estimator uses an estimate $\hat{p}_\mu$ in a weighted average over the samples $\mathcal{D}$ [Hanna et al., 2019]:

$$\hat{V}_{\mathrm{IS}}(\pi; \hat{\mu}) \coloneqq \frac{1}{m} \sum_{i=1}^{m} w_i r^i, \tag{1}$$

with

$$w_i \coloneqq \prod_{t=0}^{t_i} \frac{p_\pi(A_t = a^i_t \mid H_t = h^i_t)}{\hat{p}_\mu(A_t = a^i_t \mid H_t = h^i_t)}. \tag{2}$$

Sufficient conditions for the estimator $\hat{V}_{\mathrm{IS}}(\pi; \hat{\mu})$ to be an unbiased estimator of $V(\pi)$ include (sequential) *ignorability* and *overlap* [Rosenbaum and Rubin, 1983, Robins, 1986]. In our setting, ignorability may be defined as for all t, the conditional distribution of R given A_t and H_t is the same under π and μ, i.e., $\forall t : p_\pi(R \mid H_t, A_t) = p_\mu(R \mid H_t, A_t)$. Overlap is satisfied for a pair (h, a) if it being observable under π implies that it is observable under μ, $p_\pi(A_t = a \mid H_t = h) > 0 \Rightarrow p_\mu(A_t = a \mid H_t = h) > 0$. We say that overlap is partially violated if this condition is violated for some pairs of histories and actions.

Even when ignorability and overlap are satisfied, if μ and π differ significantly, the estimator $\hat{V}_{\mathrm{IS}}(\pi; \hat{\mu})$ suffers from high variance. The weighted importance sampling estimator [Rubinstein and Kroese, 2016], $\hat{V}_{\mathrm{WIS}}(\pi; \hat{\mu}) \coloneqq \frac{1}{\sum_{i=1}^m w_i} \sum_{i=1}^m w_i r^i$, introduces bias, but often has less variance. Under the Markov assumption, i.e., that context (or "state") transitions, actions and rewards depend only on the most recent context-action pair, the history H_t in (2) can be replaced by X_t. We leave out the subscript t where clear.

2.1 CAN WE TRUST AN IS ESTIMATE?

A fundamental challenge with off-policy evaluation is that no ground truth value, or even samples of it, are available. What is worse, the assumption of ignorability cannot be verified statistically [Rosenbaum et al., 2010] and the extent of overlap is unknown if μ is unknown. As a result, assessing the quality of an estimate $\hat{V}_{\mathrm{IS}}$ inherently relies on domain expertise.

By examining importance weights $\{w_i\}_{i=1}^m$ and estimated propensities $\hat{p}_\mu(A_t \mid H_t)$, analysts can spot outliers with extremely large weights, and compute the effective sample size (ESS) [Gottesman et al., 2018, Owen, 2013, Chapter 9]. These practices give a per-sample and an average view of potential issues with variance and the potential for removing samples with excessive weights [Crump et al., 2009, Stürmer et al., 2010]. However, several questions remain regarding what replacing μ with π would imply in practice:

- **Where do π and μ differ?** How can we *describe* the inputs for which the most probable actions under π differ from those under μ?
- **If $\hat{V}(\pi) > \hat{V}(\mu)$, what gives π the edge?** In *which* situations does acting according to π result in higher rewards than acting according to μ?

Inspecting weights and propensities in aggregate or on a per-sample basis is insufficient to answer these questions as they concern *patterns* in policy decisions, weights and rewards. For example, extreme weights and small ESS may indicate lack of overlap between π and μ but do not explain the cause of the problem. Next, we show that a case-based model of the behavior policy μ can help identify said patterns by inducing a soft clustering over the space of histories.

[1] Importance sampling estimators are often also referred to as "importance weighting" estimators.

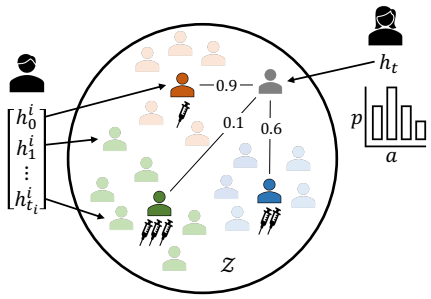

Figure 1: A schematic drawing of the prototype setup using a medical example. Each subsequence h_t^i of the patient histories in the training data have a representation in the learned latent space $\mathcal{Z}$. A few subsequences are selected as prototypes—samples that are representative of the history-action space. In this example, there are three prototypes which are treated with different drug doses. Note that a patient can belong to different prototype clusters during the course of medication. The action propensity $p_\mu(a \mid h_t)$ of a test sample h_t is computed by weighting the similarity between h_t and each prototype.

3 OFF-POLICY EVALUATION WITH PROTOTYPES

We propose performing off-policy evaluation using *prototype learning* [Li et al., 2018b, Ming et al., 2019]. The idea is to express the behavior policy $p_\mu(A \mid H)$ by comparing the history H to a relatively small set of prototype histories from the training data, see Figure 1. In a clinical setting, such a policy may correspond to physicians choosing treatment for a new patient based on their prior experience in treating similar patients. For a domain expert, trained in interpreting such cases, a prototype-based estimate is transparent as long as the number of prototypes is small enough. By examining how policy overlap and value estimates vary with prototypes, we can answer the questions raised above.

3.1 MODELING BEHAVIOR WITH PROTOTYPES

Let $\tilde{H} = [\tilde{h}^1, \dots, \tilde{h}^n]^\intercal$ be a list of n prototype histories[2]. Each prototype is a *subsequence* of an observed history, $\tilde{h}^j = h_t^i$ for $h^i \in \mathcal{D}$ and $t \leq t_i$. We allow the prototypes to be subsequences of full-length histories since OPE requires evaluating the behavior policy at each time step. The behavior policy $p_\mu(A_t \mid H_t = h_t)$ is approximated based on the similarity between an observation h_t and the prototypes in a learned representation. The prototypes $\tilde{H}$ are themselves selected by the learning algorithm.

To learn $\tilde{H}$, we follow Li et al. [2018b], Ming et al. [2019] by first learning a set of latent prototypes as free parameters $\tilde{Z} = [\tilde{z}_1, \dots, \tilde{z}_n]^\intercal$ in an encoding space $\mathcal{Z}$. Given an encoder $e : \mathcal{H} \to \mathcal{Z}$, for an arbitrary history h_t, let

$$S(\tilde{Z}, e(h_t)) = [s(\tilde{z}_1, e(h_t)), \dots, s(\tilde{z}_n, e(h_t))]^\intercal$$

be the *similarity vector* for the encoding of h_t comparing $e(h_t)$ to $\tilde{Z}$ using a fixed function $s : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}_+$. We use an RBF-kernel with unit bandwidth ($\gamma = 1$),

$$s(\tilde{z}, e(h)) \coloneqq \exp(-\|\tilde{z} - e(h)\|_2^2 / \gamma^2), \tag{3}$$

which takes values between 0 (no similarity) and 1 (full similarity). With $B \in \mathbb{R}^{k \times n}$, we estimate the behavior policy μ through logistic regression in the space induced by S,

$$\hat{p}_\mu(A_t \mid H_t = h_t) = f_\sigma(BS(\tilde{Z}, e(h_t)) + c), \tag{4}$$

where f_σ denotes the softmax function over rows and $c \in \mathbb{R}^k$ is a bias term. Column j of B represents the coefficients determining the action probabilities associated with $\tilde{h}^j$. If the coefficient B_{ij} is positive, higher similarity between h_t and $\tilde{h}^j$ makes action i more probable for h_t; a negative coefficient makes action i less probable.

The model parameters $\Theta = (e, B, c, \tilde{H})$, comprising the parameters of the encoder e, coefficients B, c and the set of prototypes $\tilde{H}$, are all unknown and must be learned from data. As encoder, we use either feedforward or recurrent neural networks. Following Ming et al. [2019], we learn Θ by minimizing the regularized negative log-likelihood

$$J(\Theta) = \mathrm{NLL}(\mathcal{D}; \Theta) + \lambda_d R_d(\Theta) + \lambda_c R_c(\Theta) + \lambda_e R_e(\Theta) \tag{5}$$

using stochastic gradient descent. The regularization terms $R_d(\Theta)$, $R_c(\Theta)$ and $R_e(\Theta)$ encourage *diversity*, *clustering* and *evidence*, respectively, and are defined in Appendix A.

To make sure that prototypes represent real cases, i.e., to select $\tilde{H}$, latent prototypes are projected onto encodings of training samples at regular intervals between descent steps,

$$\tilde{h}^j \leftarrow \underset{h_t^i \in \overline{\mathcal{D}}}{\arg\max}\ s(\tilde{z}_j, e(h_t^i)) \quad \text{and} \quad \tilde{z}_j \leftarrow e(\tilde{h}^j), \tag{6}$$

with $\overline{\mathcal{D}}$ the set of all subsequences of trajectories in $\mathcal{D}$.

Is there a good prototype model? Modeling the behavior policy using prototypes places additional restrictions on the functional form of estimates. It is natural to ask: Assuming that adjusting for the history H_t is sufficient for unbiased policy evaluation, do there exist prototype histories $\tilde{H}$, an encoding e and a similarity function s such that evaluation using the prototype model is exact or accurate? In Section 4, we study this question empirically. Additionally, in Appendix A.2, we show constructively that there are indeed problems for which a prototype model exists that *exactly* describes the behavior policy μ.

[2]From now, we refer to these as "prototypes".

3.2 PREDICTING WITH PROTOTYPES

When computing the estimated behavior policy (4) for a history h, the similarity vector $S(e(\tilde{H}), e(h))$ determines how similar each of the n prototypes are to h. The number n is a hyperparameter. The more prototypes are used, the greater the flexibility of the model, but a large n may result in S consisting of multiple elements close to 1, making predictions difficult to interpret. For example, if $s(e(\tilde{h}^j), e(h)) \approx 1$ for more than 10 prototypes j, it may be difficult to reason about the policy decision after all.

To address this, we use only a limited number of $q \leq n$ prototypes—so-called *prediction prototypes*—when making predictions with the *trained* model. Let $s_q(h)$ be the similarity between $e(h)$ and its qth most similar latent prototype. For $j = 1, \ldots, n$, we truncate the similarity vector according to

$$s(\tilde{z}_j, e(h)) \leftarrow \begin{cases} s(\tilde{z}_j, e(h)) & \text{if } s(\tilde{z}_j, e(h)) \geq s_q(h), \\ 0 & \text{otherwise.} \end{cases}$$

As an example, with $q = 2$ the (sorted) similarity vector in Figure 1 would become $[0.9, 0.6, 0]^\intercal$.

We perform the truncation step independently for all contexts h. In Section 4.2, we study the resulting trade-off between transparency (small q and n) and bias as we vary the number of (prediction) prototypes, (q) n. Note that we optimize the regularization parameters λ_d, λ_c and λ_e with respect to the choice of q.

3.3 USING PROTOTYPES FOR EVALUATION

Prototypes induce a soft clustering of the space of histories. Each prototype represents a group of similar histories which can be associated with a certain distribution over actions. In Figure 1, we see for example that the "green prototype"—representing the patients in the "green cluster"—is given a higher dose of the drug than the other prototypes. Given characteristics of the "green prototype", a domain expert should be able to explain why it receives this type of treatment. While it is possible to use other methods to cluster the space of histories, prototypes have the advantage of being based in cases and trained to describe groups of subjects who are treated differently under the behavior policy. We see in Section 4.2 that this is beneficial also for accuracy.

When modeling the behavior policy μ using prototypes, we can utilize the induced clustering structure to answer the questions raised in Section 2.1. First of all, the prototypes $\tilde{h}^j$ and their action probabilities

$$\hat{p}^j_\mu(a) = \hat{p}_\mu(A = a \mid H = \tilde{h}^j) \tag{7}$$

give an overview of the estimated behavior policy. By comparing the action probabilities $\hat{p}^j_\mu(a)$ with the corresponding action probabilities under π, $p^j_\pi(a) = p_\pi(A = a \mid H = \tilde{h}^j)$, we can explain input regions for which π and μ differ in their suggested actions. Domain experts can use this overview to assess how well the data supports evaluation of the target policy. For example, if π, for a certain prototype, suggests actions that are extremely rare under μ, there may not be enough data on these decisions to accurately estimate $V(\pi)$.

It is good practice to compare $\hat{V}(\pi)$ with $\hat{V}(\mu)$, i.e., the mean reward in data. If $\hat{V}(\pi)$ is different from $\hat{V}(\mu)$, we would like to know for which inputs π gain or lose performance in relation to μ. The prototypes allow us to divide the estimated values into prototype-based contributions and answer this question. To make the implicit clustering explicit, we define J_t to be a random variable with values in $\{1, \ldots, n\}$, representing an assignment of a history H_t to prototype j at time t. We let the probability of being assigned to prototype j be proportional to the similarity s,

$$p(J_t = j \mid H_t = h_t) = \frac{s(\tilde{z}_j, e(h_t))}{\sum_{k=1}^n s(\tilde{z}_k, e(h_t))}. \tag{8}$$

Now, we define the value $V_{j,t}(\pi)$ of prototype j at time t, obtained under a policy π, as the expected future reward under π given the assignment $J_t = j$:

$$V_{j,t}(\pi) \coloneqq \mathbb{E}_\pi[R_T \mid J_t = j]. \tag{9}$$

With $p(J_t = j)$ the marginal probability of being assigned to prototype j at time t, by the law of total expectation, $V(\pi) = \sum_{j=1}^n V_{j,t}(\pi) p(J_t = j)$ for any t. Each term

$$V_{j,t}(\pi) p(J_t = j) \tag{10}$$

in the sum represents the contribution to the overall value $V(\pi)$ from histories which are similar to prototype j at time t, effectively stratifying the value by types of situations. Note that we can compute $V_{j,t}(\mu)$ in a similar way to compare the estimated values of π and μ from a prototype perspective.

We may express $V_{j,t}(\pi)$ as a weighted expectation under the behavior policy μ, with importance weights W,

$$V_{j,t}(\pi) \coloneqq \mathbb{E}_\mu \left[\frac{p(J_t = j \mid H_t = h_t)}{p_\pi(J_t = j)} W R_T \right],$$

where $p_\pi(J_t)$ is found by importance-weighted marginalization over H_t (see derivation in Appendix A.1). We use this strategy to estimate $V_{j,t}(\pi)$ from finite samples.

4 EXPERIMENTS

In Section 4.1, we illustrate our method by examining an example of sepsis management. Using patient data from the MIMIC-III database [Johnson et al., 2016], we evaluate a replication of the so-called AI Clinician implemented by Komorowski et al. [2018]. The data are coded as multidimensional time series with a discrete time step of 4 hours.

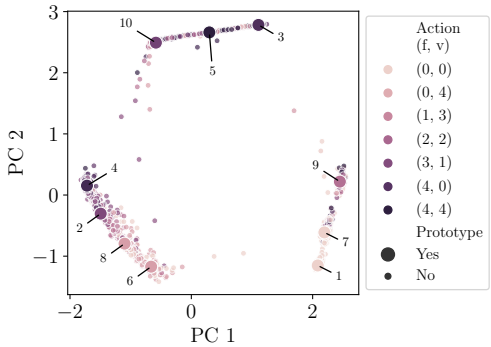

Figure 2: A PCA plot of encoded training data, colored w.r.t. the action (dose of IV fluids and vasopressors) taken by the physicians. The prototypes are numbered 1–10. Note that prototype learning affects the structure of the encoding space. Post-hoc clustering of the latent space of a model trained without prototypes gives a substantially worse approximation of the behavior policy, see Table 1.

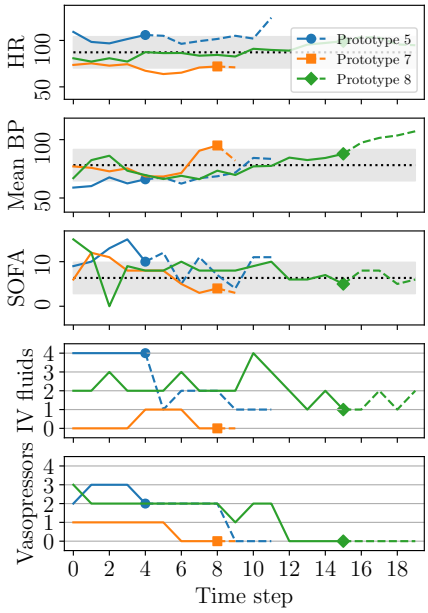

Figure 3: Vital signs and SOFA score plotted against time for three different prototype patients (upper three panels). The dashed black lines show the data average of each feature and the shaded areas mark ± 3 standard deviations. The lower two panels show the actions taken by physicians. The time index of each prototype subsequence is marked with filled marker; for example, prototype 5 is the subsequence ending at time step 4 of the corresponding patient history.

In short, AI Clinician is learned by clustering the data into 750 states, discretizing the combinations of intravenous (IV) fluids (f) and vasopressors (v) into 25 possible actions $(f, v) \in \{0, 1, 2, 3, 4\}^2$, and solving the corresponding Markov decision process using value iteration with rewards $r^i = \pm 100$ based on the survival of the patients. This process is repeated 500 times, each time with a new train-test split, and the policy that performs best on the test set is taken as the target policy, π_{AIC}. We use the split associated with π_{AIC} in our experiments.

In Section 4.2, we inspect the prototype model in more detail. We study the trade-off between transparency and bias and compare the model to several baseline estimators: a logistic regression classifier (LR), a random forest classifier (RF), a feedforward neural network (FNN), a recurrent neural network (RNN) and a model based on post-hoc clustering of encoded histories. By utilizing a sepsis simulator, we also investigate the bias induced by prototypes as a function of the sequence length.

4.1 DEMONSTRATING THE FRAMEWORK

To demonstrate the benefit of learning prototypes in OPE, we estimate the behavior policy μ, i.e., the policy followed by clinicians in the MIMIC-III data, using a prototype model with $n = 10$ prototypes, $q = 2$ prediction prototypes and an RNN encoder (see Appendix B for details). As an overview of the relationship between prototypes, a PCA plot of encoded training data is shown in Figure 2. The latent prototypes are numbered 1–10 and the colors indicate action chosen by μ in the data. Note that the figure is intended to help the reader to orient him/herself in this section; we do not rely on this projection in itself.

We can interpret the prototypes by visualizing trajectories of the corresponding patients. In Figure 3, we take a closer look at prototypes 5, 7 and 8, which represent each of the major clusters in Figure 2. By plotting three key features—heart rate (HR), mean blood pressure (BP) and SOFA score—and the treatment variables against time, we immediately get a sense of which type of patients the prototypes represent. For example, the patient corresponding to prototype 5 has high heart rate, low blood pressure and high SOFA score—signs of severe sepsis—and receives an aggressive treatment.[3] The prototype 7 patient, who has lower heart rate, higher blood pressure and lower SOFA score, receives low doses of IV fluids and vasopressors.

We understand that trajectories that are most similar to prototype 7 in the latent space belong to patients who are currently relatively healthy, which they are more likely to be at an early stage of the course. Interestingly, we observe that all encoded histories until time $t = 0$ are most similar to either prototype 7 or prototype 9. Is it therefore relevant to ask the question: If we were to follow the AI Clinician instead of the behavior policy, how would the treatment strategy change in an initial stage? That is, how do μ and π_{AIC} differ

[3] SOFA is an abbreviation for sequential organ failure assessment.

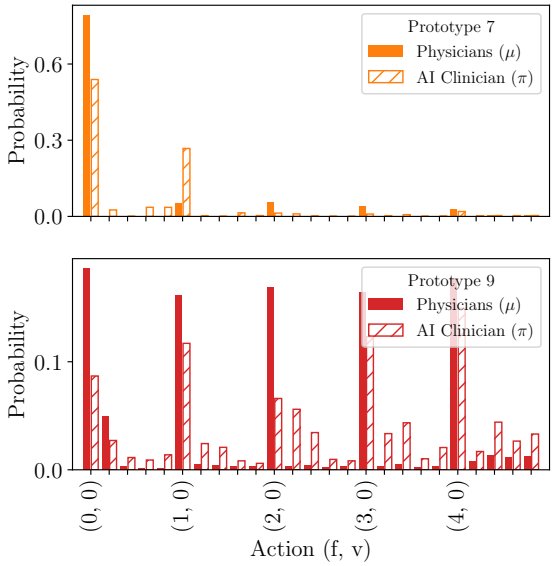

Figure 4: Action distribution under μ and π_{AIC} for prototype 7 and 9. For μ the probabilities are computed according to (7). Since π_{AIC} is deterministic, we normalize the distribution of actions suggested by π_{AIC} for input histories that are most similar to respective prototype in the latent space.

for prototypes 7 and 9?

We can answer this question by comparing the distributions of actions taken under μ and π_{AIC} for these prototypes, see Figure 4. For prototype 7, we see that the most likely treatment under both μ and π_{AIC} is to not give any IV fluids or vasopressors. However, under π_{AIC}, there is also a relatively high probability of increasing the dose of IV fluids—a rare action under μ. The differences are even greater for prototype 9, where π_{AIC} has a nonzero probability of giving an aggressive treatment with combinations of IV fluids and vasopressors. Under μ, these treatments have almost zero probability. A domain expert can reason about the validity of π_{AIC}: given characteristics of the patient corresponding to prototype 9, would it be medically sound to treat this type of patient as suggested by π_{AIC} in Figure 4?

From an OPE perspective, the initial differences between μ and π_{AIC} makes it difficult to accurately estimate $V(\pi_{\text{AIC}})$. A known problem with importance sampling is that the variance of the weights w_i can grow exponentially with the sequence length Liu et al. [2018]. Here, the assumption of overlap is potentially partially violated already at the first time step, and regardless of model of the behavior policy, we observe high variance in the IS weights (ranging from $\ll 1$ to the order of 10^3) and an extremely small effective sample size (< 10). To reduce variance, we instead evaluate a different target policy where we follow π_{AIC} in the first time step and then follow μ until the end of the sequence. For comparison, we do the same with a zero-drug policy π_0 which suggests leaving patients untreated.

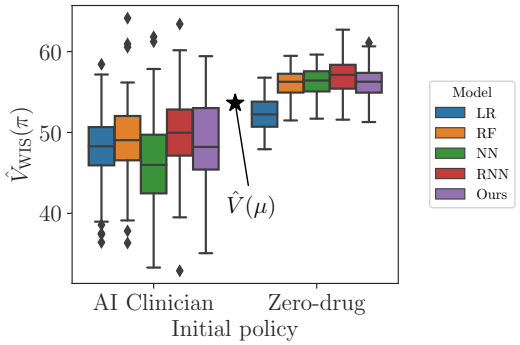

Figure 5: Bootstrap estimates of the value of the target policy of following the AI Clinician and the zero-drug policy, respectively, for one time step and then following the behavior policy. The estimated value of the behavior policy μ, $\hat{V}(\mu)$, is included as a reference.

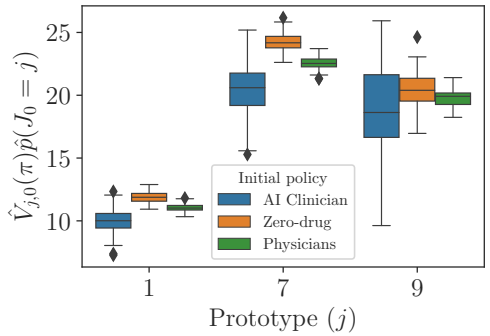

Figure 6: Bootstrap estimates of $V_{j,0}(\pi)p(J_0 = j)$ for prototype 1, 7 and 9. As described in Section 3.3, $V_{j,t}(\pi)p(J_t = j)$ is the the value of prototype j at time t multiplied with the marginal probability of being assigned to prototype j at time t.

Figure 5 shows 100 bootstrap estimates of the policy values using five different estimators of μ: the baselines LR, RF, FNN and RNN, and our prototype model. For all models except the RNN baseline and the prototype model, we make the Markov assumption and model $p(A \mid H)$ using only the last context-action pair of the history. We note that the results are consistent across estimators. In comparison with the estimated value of μ, $\hat{V}(\mu)$, the results indicate that it could be beneficial to avoid giving drugs to patients at the initial time step, while it seems less favorable to follow π_{AIC}.

The prototypes allow us to break down the result and answer the question: If $\hat{V}(\pi) \neq \hat{V}(\mu)$, where does π gain or lose performance? In Figure 6, we show 100 bootstrap estimates of the contribution to the overall value, see (10), for prototypes 1, 7 and 9 at time $t = 0$. These prototypes define the bottom right cluster of Figure 2 where all encoded histories until time $t = 0$ belong. As expected, prototype 7 and 9 contribute the most to the overall value. Note that the

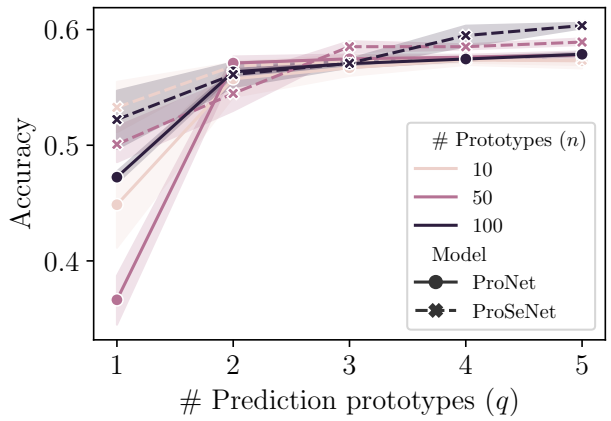

Figure 7: Accuracy on the sepsis test data for ProNet and ProSeNet using a varying number of (prediction) prototypes (q) n. The setting with $n = 10$ and $q = 2$ works well here.

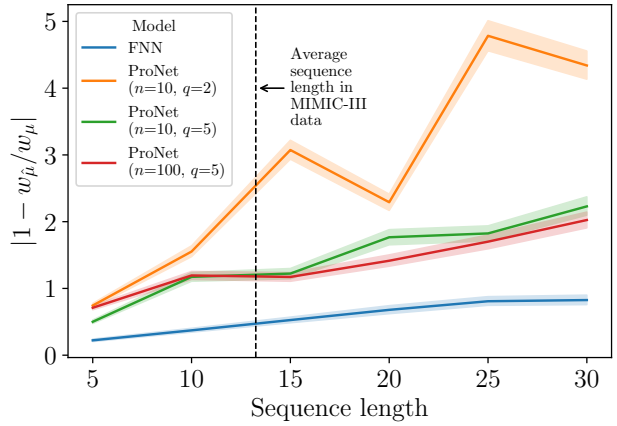

Figure 8: The relative error of estimated importance weights for increasing sequence lengths in the sepsis simulator. Ideally, if $\hat{\mu}$ is a good estimator of μ, the ratio $w_{\hat{\mu}}/w_{\mu}$ should be equal to 1. We see that approximating μ with prototype models gives rise to a bias in relation to modeling μ with a plain FNN.

difference in variance between these prototypes is explained by Figure 4 where π_{AIC} and μ differ more for prototype 9 than for prototype 7. Also note that the trend in Figure 5 is repeated here: following the zero-drug policy at the initial time step is generally better than following the AI Clinician.

4.2 PERFORMANCE OF THE PROTOTYPE MODEL

While increasing transparency, the use of prototypes imposes restrictions on the model, possibly increasing the approximation error. In Figure 7, we show the accuracy of two different prototype models—ProNet and ProSeNet—in approximating $p_\mu(A \mid H_t)$ on the sepsis test data for a varying number of prototypes n and prediction prototypes q (see Section 3.2). As encoder, the models use an FNN (ProNet) and an RNN (ProSeNet), respectively. The sequential model, making use of the entire history H_t, performs the best, especially for $q = 1$ and $q \geq 4$. Interestingly, the effect of increasing the number of prototypes from 10 to 50 or even 100 is small. Using only two prediction prototypes works well for this dataset.

In Table 1, we compare the prototype models with $n = 10$ and $q = 2$ to the baseline estimators in approximating $p_\mu(A \mid H_t)$. Here, we report accuracy, the area under the ROC curve (AUC) and the static calibration error (SCE) [Nixon et al., 2019], a multiclass extension of the expected calibration error. The prototype models are superior to the (regularized) LR but they perform slightly worse than the black-box models RF, FNN and RNN. However, as we see in Figure 7, with increased number of prototypes, ProSeNet has the capacity to approach the performance of these models, at least in terms of accuracy. Finally, we note that the prototype models are superior to a model where post-hoc clustering of the RNN encodings are used to identify "prototypes", showing the power of learning prototypes in a supervised manner. We refer to Appendix B.1 for details.

4.2.1 Bias Due to Increased Sequence Length

If the use of prototypes introduces a bias in the estimated propensity, it is natural to ask what it means for the sequential setting, where multiple propensities are multiplied together to form the importance weights. To quantity this effect, we consider the synthetic environment of sepsis management provided by Oberst and Sontag [2019]. By sampling a large amount of data from the environment, we estimate the true parameters of the underlying Markov decision process. We then learn an optimal behavior policy using policy iteration. We refer to Appendix B.2 for details.

We collect trajectories of the behavior policy of various lengths, from 5 to 30 time steps. For each trajectory length, we use the data to (a) estimate the behavior policy μ using an FNN and different prototype models, respectively; (b) learn a target policy π using policy iteration; and (c) estimate the value of π using both the true behavior policy μ and its estimators $\hat{\mu}$. Note that any difference in the estimated values stems from a difference in the importance weights w. In Figure 8, we plot the relative error of estimated weights against the trajectory length for four estimators of μ.[4] Averaging over 100 iterations of the sampling and learning process, we see that the ratio $w_{\hat{\mu}}/w_{\mu}$ generally differs from 1 for all estimators and that modeling with prototypes induces larger bias than using an FNN. For longer sequences, the number of prediction prototypes q becomes critical.

[4]Note that the probabilities under π cancel when considering the ratio of the weights.

Table 1: A summary of performance on the sepsis test data for different estimators of the behavior policy $p_\mu(A \mid H_t)$. For ProNet and ProSeNet, $n = 10$ and $q = 2$. The 95 percent confidence intervals are calculated from 1000 bootstraps.

Model	Accuracy (↑)	SCE (↓)	AUC (↑)
LR	0.38 (0.38, 0.39)	0.0112 (0.0110, 0.0115)	0.88 (0.88, 0.88)
RF	0.62 (0.61, 0.62)	0.0037 (0.0034, 0.0039)	0.93 (0.93, 0.93)
Post-hoc clustering	0.44 (0.44, 0.45)	0.0097 (0.0096, 0.0101)	0.86 (0.85, 0.86)
FNN	0.61 (0.61, 0.61)	0.0041 (0.0039, 0.0044)	0.93 (0.92, 0.93)
ProNet ($n = 10$, $q = 2$)	0.56 (0.55, 0.56)	0.0069 (0.0067, 0.0072)	0.90 (0.90, 0.90)
RNN	0.62 (0.62, 0.63)	0.0056 (0.0053, 0.0058)	0.94 (0.94, 0.94)
ProSeNet ($n = 10$, $q = 2$)	0.57 (0.57, 0.58)	0.0057 (0.0054, 0.0059)	0.91 (0.91, 0.91)

Finally, we quantify the absolute effect prototypes have on the value estimate for sequences of length 15, which is close to the average sequence length in the MIMIC-III data. We compute the true value of π by running it in the simulator and compare this value to weighted IS estimates using the estimators in Figure 8. We observe final rewards $r^i = \pm 1$ if a simulated patient is discharged or dies; otherwise $r^i = 0$. Averaging over 100 iterations, the estimated value has an absolute difference from the true value that amounts to 0.39 for the FNN (standard deviation 0.27), 0.46 (0.33) for ProNet with $n = 10$ and $q = 2$, 0.50 (0.34) for ProNet with $n = 10$ and $q = 5$ and 0.41 (0.31) for ProNet with $n = 100$ and $q = 5$. These results should be compared to the average value of π (0.06 (0.10)) and the WIS estimate using the true behavior policy (0.38 (0.27)).

5 RELATED WORK

Issues with importance sampling methods for OPE are well known. Several works aim at describing issues related to high variance [Gottesman et al., 2019], or mitigating them using methodological advances [Precup, 2000, Thomas and Brunskill, 2016, Jiang and Li, 2016, Schneeweiss et al., 2009, Swaminathan and Joachims, 2015]. Others aim to use weights to identify a new study population for which the policy's value can be more efficiently estimated [Li et al., 2018a, Fogarty et al., 2016]. Oberst et al. [2020] emphasize the value of interpretability in this endeavour to communicate the generalizability of the estimate. Our method is compatible with all three approaches, allowing for transparent descriptions of variance issues, identifying new study populations and for use as plug-in estimates.

Interpretability is a an important component of learning systems deployed in increasingly critical functions [Rudin, 2019, Lipton, 2018]. Rule-based estimators, such as rule list [Wang and Rudin, 2015] and decision trees, are often favored for their short descriptions but generalize poorly to sequential inputs which are the focus of this work. Gottesman et al. [2020] proposed an approach for interpretable OPE which highlights transitions in data whose removal would have a large impact on the estimate. This approach is related to ours but answers a different set of questions.

Evaluating policies using direct sample-to-sample comparison has a long tradition in policy evaluation through the use of matching estimators of causal effects, see e.g., [Rosenbaum and Rubin, 1983, Rubin, 2006, Kallus, 2020]. While favored for its transparency, this approach is typically only used to compare two deterministic policies such as "treat all" or "treat none". Matching often relies either on specifying a similarity function in advance or on an estimate of the behavior policy. In high-dimensional settings, this often leads to bias or lost interpretability. Our approach aims to combine the transparency of matching estimators with the flexibility of representation learning methods.

6 CONCLUSION

In this work, we have studied off-policy evaluation (OPE) using importance sampling (IS) in the case where the behavior policy μ is unknown and must be estimated from data. While IS is a popular OPE method, it may be difficult to assess the quality of an IS value estimate; standard practices, such as inspecting importance weights, provide only an average or a per-sample view of potential issues. To address this issue, we proposed estimating the behavior policy for IS using prototype learning to better explain patterns in policy decisions and value estimates. We demonstrated our idea using a real-world example of sepsis management. While the use of prototypes increases the approximation error, we find that prototype models have the capacity to perform similarly to plain neural networks. We hope to provide theoretical results for the approximation error in future work.

When reflecting upon the results of the sepsis study it may seem strange that it would be advantageous to never treat patients at the onset of sepsis. Even though prototypes serve as a tool for inspecting policies and value estimates, they do not answer all questions. For example, there may be variables affecting both the treatment and the outcome that are not present in the observed data. In such a case, the ignorability assumption defined in Section 2 no longer holds.

References

Richard K Crump, V Joseph Hotz, Guido W Imbens, and Oscar A Mitnik. Dealing with limited overlap in estimation of average treatment effects. *Biometrika*, 96(1): 187–199, 2009.

Colin B Fogarty, Mark E Mikkelsen, David F Gaieski, and Dylan S Small. Discrete optimization for interpretable study populations and randomization inference in an observational study of severe sepsis mortality. *Journal of the American Statistical Association*, 111(514):447–458, 2016.

Omer Gottesman, Fredrik Johansson, Joshua Meier, Jack Dent, Donghun Lee, Srivatsan Srinivasan, Linying Zhang, Yi Ding, David Wihl, Xuefeng Peng, et al. Evaluating reinforcement learning algorithms in observational health settings. *arXiv preprint arXiv:1805.12298*, 2018.

Omer Gottesman, Fredrik Johansson, Matthieu Komorowski, Aldo Faisal, David Sontag, Finale Doshi-Velez, and Leo Anthony Celi. Guidelines for reinforcement learning in healthcare. *Nature medicine*, 25(1): 16–18, 2019.

Omer Gottesman, Joseph Futoma, Yao Liu, Sonali Parbhoo, Leo Celi, Emma Brunskill, and Finale Doshi-Velez. Interpretable off-policy evaluation in reinforcement learning by highlighting influential transitions. In *International Conference on Machine Learning*, pages 3658–3667. PMLR, 2020.

Josiah Hanna, Scott Niekum, and Peter Stone. Importance sampling policy evaluation with an estimated behavior policy. In *International Conference on Machine Learning*, pages 2605–2613. PMLR, 2019.

Keisuke Hirano, Guido W Imbens, and Geert Ridder. Efficient estimation of average treatment effects using the estimated propensity score. *Econometrica*, 71(4):1161–1189, 2003.

Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 652–661. PMLR, 2016.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

Nathan Kallus. Generalized optimal matching methods for causal inference. *Journal of Machine Learning Research*, 21(62):1–54, 2020.

Matthieu Komorowski, Leo A Celi, Omar Badawi, Anthony C Gordon, and A Aldo Faisal. The artificial intelligence clinician learns optimal treatment strategies for sepsis in intensive care. *Nature medicine*, 24(11): 1716–1720, 2018.

Zhifeng Kong and Kamalika Chaudhuri. The expressive power of a class of normalizing flow models. In *International Conference on Artificial Intelligence and Statistics*, pages 3599–3609. PMLR, 2020.

Fan Li, Kari Lock Morgan, and Alan M Zaslavsky. Balancing covariates via propensity score weighting. *Journal of the American Statistical Association*, 113(521):390–400, 2018a.

Fan Li, Laine E Thomas, and Fan Li. Addressing extreme propensity scores via the overlap weights. *American journal of epidemiology*, 188(1):250–257, 2019.

Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018b.

Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018.

Qiang Liu, Lihong Li, Ziyang Tang, and Dengyong Zhou. Breaking the curse of horizon: Infinite-horizon off-policy estimation. *Advances in Neural Information Processing Systems*, 31, 2018.

Yao Ming, Panpan Xu, Huamin Qu, and Liu Ren. Interpretable and steerable sequence learning via prototypes. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 903–913, 2019.

Hongseok Namkoong, Ramtin Keramati, Steve Yadlowsky, and Emma Brunskill. Off-policy policy evaluation for sequential decisions under unobserved confounding. *arXiv preprint arXiv:2003.05623*, 2020.

Jeremy Nixon, Michael W Dusenberry, Linchuan Zhang, Ghassen Jerfel, and Dustin Tran. Measuring calibration in deep learning. In *CVPR Workshops*, volume 2–7, 2019.

Michael Oberst and David Sontag. Counterfactual off-policy evaluation with gumbel-max structural causal models. In *International Conference on Machine Learning*, pages 4881–4890. PMLR, 2019.

Michael Oberst, Fredrik Johansson, Dennis Wei, Tian Gao, Gabriel Brat, David Sontag, and Kush Varshney. Characterization of overlap in observational studies. In *International Conference on Artificial Intelligence and Statistics*, pages 788–798. PMLR, 2020.

Art B. Owen. *Monte Carlo theory, methods and examples*. 2013.

Doina Precup. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, page 80, 2000.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

James Robins. A new approach to causal inference in mortality studies with a sustained exposure period—application to control of the healthy worker survivor effect. *Mathematical modelling*, 7(9-12):1393–1512, 1986.

Paul R Rosenbaum and Donald B Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

Paul R Rosenbaum et al. *Design of observational studies*, volume 10. Springer, 2010.

Donald B Rubin. *Matched sampling for causal effects*. Cambridge University Press, 2006.

Reuven Y Rubinstein and Dirk P Kroese. *Simulation and the Monte Carlo method*, volume 10. John Wiley & Sons, 2016.

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

Sebastian Schneeweiss, Jeremy A Rassen, Robert J Glynn, Jerry Avorn, Helen Mogun, and M Alan Brookhart. High-dimensional propensity score adjustment in studies of treatment effects using health care claims data. *Epidemiology (Cambridge, Mass.)*, 20(4):512, 2009.

Til Stürmer, Kenneth J Rothman, Jerry Avorn, and Robert J Glynn. Treatment effects in the presence of unmeasured confounding: dealing with observations in the tails of the propensity score distribution—a simulation study. *American journal of epidemiology*, 172(7):843–854, 2010.

Adith Swaminathan and Thorsten Joachims. Counterfactual risk minimization: Learning from logged bandit feedback. In *International Conference on Machine Learning*, pages 814–823, 2015.

Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 2139–2148. PMLR, 2016.

Philip S Thomas. *Safe reinforcement learning*. PhD thesis, University of Massachusetts Libraries, 2015.

Fulton Wang and Cynthia Rudin. Falling rule lists. In *Artificial Intelligence and Statistics*, pages 1013–1022. PMLR, 2015.

A THE PROTOTYPE MODEL

In Section 3.1, we only briefly described the objective function

$$J(\Theta) = \mathrm{NLL}(\mathcal{D};\Theta) + \lambda_d R_d(\Theta) + \lambda_c R_c(\Theta) + \lambda_e R_e(\Theta),$$

where $\mathrm{NLL}(\mathcal{D};\Theta)$ is the negative log-likelihood, $R_d(\Theta)$, $R_c(\Theta)$ and $R_e(\Theta)$ are regularization terms and λ_d, λ_c and λ_e are regularization parameters. Θ denotes the set of model parameters, i.e., the parameters of the encoding network e, the weights B, c and the prototypes $\tilde{H}$. For a given dataset $\mathcal{D} = ((h^1_{t_1}, a^1_{t_1}), \ldots, (h^m_{t_m}, a^m_{t_m}))$, drawn according to a distribution p_μ, the NLL loss of the estimate $\hat{p}_\mu$, parameterized in Θ, is defined as

$$\mathrm{NLL}(\mathcal{D};\Theta) = -\frac{1}{m}\sum_{i=1}^{m} \log\left(\hat{p}_\mu(A_t = a^i_{t_i} \mid H_t = h^i_{t_i})\right).$$

Furthermore, the regularization terms are defined as follows (see Ming et al. [2019] for further details):

- The **diversity** regularization

$$R_d(\Theta) = \sum_{i=1}^{n}\sum_{j=i+1}^{n} \max\left(0, d_{\min} - d(\tilde{z}_i, \tilde{z}_j)\right)^2,$$

  where $d(z, z') = \|z - z'\|_2$, penalizes latent prototypes that are too close to each other. The parameter $d_{\min}$ is a tunable hyperparameter in our experiments.

- The **clustering** regularization

$$R_c(\Theta) = \sum_{h\in\mathcal{D}} \min_i d(\tilde{z}_i, e(h))^2$$

  encourages the encoded histories to approach the most similar latent prototypes, which creates a clustering structure in the latent space.

- The **evidence** regularization

$$R_e(\Theta) = \sum_{i=1}^{n} \min_{h\in\mathcal{D}} d(\tilde{z}_i, e(h))^2$$

  encourages the latent prototypes to approach the encodings that are most similar to them.

A.1 PROTOTYPE VALUE

In Section 3.3, we showed how the prototypes can be used to compute the value of a policy π for prototype j at time t,

$V_{j,t}(\pi)$. Below, we derive a statistical estimator for $V_{j,t}(\pi)$ using observations under μ. First, we have

$$V_{j,t}(\pi) := \mathbb{E}_\pi\left[\sum_{t'\geq t} R_{t'} \;\middle|\; J_t = j\right]$$
$$= \mathbb{E}_\pi\left[\frac{p(J_t = j \mid H_t)}{p_\pi(J_t = j)} \sum_{t'\geq t} R_{t'}\right].$$

This equation follows from the fact that J_t is conditionally independent of all other variables given H_t. Now, with W importance weights for π and μ,

$$V_{j,t}(\pi) = \mathbb{E}_\mu\left[\frac{p(J_t = j \mid H_t)}{p_\pi(J_t = j)} W \sum_{t'\geq t} R_{t'}\right].$$

Following standard definitions,

$$p_\pi(J_t = j) = \mathbb{E}_\pi[p(J_t \mid H_t)],$$

which may be identified using importance sampling,

$$p_\pi(J_t) = \mathbb{E}_\pi[p(J_t \mid H_t)] = \mathbb{E}_\mu[p(J_t \mid H_t)W_t],$$

with $W_t = \prod_{t'=0}^{t} \frac{p_\pi(A_{t'}|H_{t'})}{p_\mu(A_{t'}|H_{t'})}$. Hence, we may estimate

$$\hat{p}_\pi(J_t = j) = \frac{1}{m}\sum_{i=1}^{m} p(J_t = j \mid H_t = h_t^i) w_t^i.$$

For trajectories i which end before t, we let $\hat{p}(J_t = j \mid H_t = h_t^i) = 0$.

A.2 IS THERE A GOOD PROTOTYPE MODEL?

In Section 3.1, we asked the question: Assuming that adjusting for the history H_t is sufficient for unbiased policy evaluation, do there exist prototype histories $\tilde{H}$, an encoding e and a similarity function s such that evaluation using the prototype model is accurate? Here follows an example when this is provably the case.

Consider the history at the first time step, $H_0 = X_0 \in \mathbb{R}^d$. Assume that for each action a, the distribution of histories in which action a is taken is isotropic Gaussian, $(H_0 \mid A_0 = a) \sim \mathcal{N}(\mu_a, \gamma^2)$. Then, the joint distribution of (H_0, A_0) is a Gaussian mixture model (GMM) with components identified by the actions, $a = 1, \ldots, k$. We have by the definition of the GMM that

$$p(A_0 = a \mid H_0 = h) \propto e^{-\frac{\|h-\mu_a\|_2^2}{\gamma^2}} = s(\mu_a, h),$$

where s is defined as in (3). As a result, with the component means $\tilde{X} = [\mu_1, ..., \mu_k]$ as prototypes and $S(\tilde{X}, h) = [s(\mu_1, h), ..., s(\mu_k, h)]^\intercal$, the behavior policy is given by $p(A_0 \mid H_0 = h) = S(\tilde{X}, h)/(\sum_a s(\tilde{x}_a, h))$, which matches (4) with $B = 1/(\sum_a s(\tilde{x}_a, h))$ and $c = 0$, up to the application of the softmax function. Furthermore, the prototype assignment probability in (8) is equal to the behavior policy. We state a generalization below.

Theorem 1. *Assume that there exists a bijective, differentiable encoding function $e : \mathcal{H} \to \mathcal{Z}$ such that $\forall t : (e(H_t), A_t) \sim$ GMM with stationary component means $\{\mu_a\}_{a=1}^k$ and variance γ^2. Then, with prototypes $\tilde{H} = [e^{-1}(\mu_1), \ldots, e^{-1}(\mu_k)]^\intercal$, and S as defined above,*

$$\forall t : p(A_t \mid H_t = h) \propto S(e(\tilde{H}), e(h)).$$

Proof. Due to bijectivity, $p(A_t \mid H_t = h) = p(A_t \mid e(H_t) = e(h))$. The final result follows from the same argument as for the special case of H_0 above. □

Theorem 1 shows that there are indeed problems for which a prototype model that exactly describes the behavior policy *exists*. This also implies that there exists a prototype estimate of the value $V(\pi)$ which is unbiased. However, it does not give guarantees for recovering such a model from data, or that the training set contains samples which act well as prototypes. Learning encoding functions e which satisfy the conditions of Theorem 1 has been studied in the context of normalizing flows [Kong and Chaudhuri, 2020, Rezende and Mohamed, 2015].

B EXPERIMENTAL DETAILS

The prototype approach for OPE was evaluated on an real-world sepsis data extracted from the MIMIC-III database [Johnson et al., 2016]. In addition, a synthetic environment for sepsis management, provided by Oberst and Sontag [2019], was used to study the bias induced by prototypes as a function of the trajectory length. In this section, we give details about the experiments. To produce the results presented in this paper, we needed about 750 core-hours of computational time. The neural networks were implemented in PyTorch and trained on GPU (Nvidia Tesla T4). Other models were implemented using scikit-learn.

B.1 USING DATA FROM MIMIC-III

We extracted the dataset of patients suffering from sepsis from the MIMIC-III database using the code provided by Komorowski et al. [2018].[5] This dataset contains the features listed in Supplementary Table 2 in Komorowski et al. [2018] as well as the total fluid intake and the total urine output for each patient. We learned the target policy, the so-called AI Clinician, using the Matlab code provided by Komorowski et al. [2018]. To evaluate the 500 candidate policies, we used only the MIMIC-III test data and not data from the eICU Research Institute Database. We used the train-test split associated with the best performing candidate in our experiments.

[5]The code is available at `https://github.com/matthieukomorowski/AI_Clinician`.

We trained the estimators of the behavior policy using a subset of the available features: heart rate, systolic blood pressure, diastolic blood pressure, mean blood pressure, shock index, hemoglobin, BUN, creatine, urine output over 4 hours, pH, base excess, bicarbonate, lactate, PaO_2/FiO_2 ratio, age, Elixhauser index and SOFA score. In addition, we included the treatment dose of vasopressors and IV fluids, respectively, over the previous 4 hours. These values were set to 0 at the initial time steps.

We estimated the value of the policies performing weighted importance sampling of the test set trajectories. For all estimators of the behavior policy, we excluded a small number of sequences whose IS weight exceeded 100.

B.1.1 Model Parameters

We considered two types of encoders for the prototype framework: a feedforward neural network (FNN) and a recurrent neural network (RNN). We named these models ProNet (FNN encoder) and ProSeNet (RNN encoder), respectively. Both encoders had two hidden layers, each of size 64, and we used ReLu and tanh, respectively, as activation function. We compared the prototype models to two neural network baselines—a vanilla FNN and a vanilla RNN—with architectures identical to the encoders. A linear layer was added to the baselines to obtain predictions of the desired shape.

We trained all neural networks over 400 epochs, using a batch size of 64 for RNN and ProSeNet, and 1024 for FNN and ProNet. For optimization, the Adam algorithm was used with default parameters, learning rate 0.001 and weight decay 0.001. The NLL was used as loss function for FNN and RNN. For ProNet and ProSeNet, we selected parameters of the diversity regularization $(d_{\text{min}}, \lambda_d)$ by performing 3-fold cross-validation over a grid of points in the parameter space $\{1, 2, 3, 4, 5\} \times \{0.00001, 0.0001, 0.001, 0.01, 0.1\}$. These parameters were optimized for each combination of prototypes n and prediction prototypes q in our experiments. The parameters λ_c and λ_e were set to 0.001, and we performed the projection step, see (6), every fifth epoch.

We also compared the prototype models to a logistic regression (LR) classifier and a random forest (RF) classifier. For LR and RF, we searched for optimal models using 3-fold cross-validation, considering the following parameter values:

- LR: regularization: $\{\text{L1}, \text{L2}\}$; regularization strength: 10 numbers spaced evenly on a log-scale from 1×10^{-4} to 1×10^{4};
- RF: maximum tree depth: $\{5, 10, 15, 20, \text{None}\}$.

Finally, we compared the prototype models to the a model based on post-hoc clustering of encodings. More specifically, we performed K-means clustering to cluster the encodings of the converged RNN model into 10 clusters. We used the encodings that had the shortest Euclidean distance to the cluster centroids as "prototypes" and trained a logistic regression model on the similarity vector between encodings and "prototypes".

All models were calibrated using sigmoid calibration on a held-out validation set (25 % of the training data).

B.2 USING THE SEPSIS SIMULATOR

To estimate the bias induced by prototypes as a function of trajectory length, we utilized the sepsis simulator provided by Oberst and Sontag [2019].[6] We used the full state representation consisting of five discrete variables—a binary diabetes indicator and four ordinal-valued vital signs (heart rate, systolic blood pressure, blood glucose level and blood oxygen level)—as well as the previously administered treatments. There is a probability of 0.2 that a randomly initialized patient has diabetes. Furthermore, there are three binary treatment variables—antibiotics, vasopressors and ventilation—resulting in an 8-dimensional action space. A simulated patient is discharged if the patient has normal vitals and is not given any treatments; discharge results in a positive reward. Death is associated with a negative reward and occurs if at least three vitals are abnormal. If neither discharge nor death has occurred at the end of the sequence, the reward is zero. We refer to the code for details about, for example, the levels of the vitals and the transition probabilities of the Markov decision process (MDP).

For the experiment presented in Figure 8, we used the notebook `learn_mdp_parameters` provided by Oberst and Sontag [2019] to estimate the true parameters of the MPD. We then learned an optimal behavior policy using policy iteration. The policy was softened so that all actions had a nonzero probability of being chosen in each state. We generated trajectories of varying length from this policy to estimate the effect of bias. Some trajectories ended prematurely due to discharge or death and we therefore ensured that the number of collected state-action pairs were roughly the same for all sequence lengths. Specifically, for each sequence length, we generated 20 000 state-action pairs for training and sigmoid calibration of the estimators $\hat{\mu}$, respectively. The training set was also used to estimate the MDP parameters and learn a target policy π using policy iteration. Also this policy was softened to avoid zero probabilities. We generated an additional evaluation set of the same size as the training and calibration sets. We repeated the process of data collection and learning 100 times. To produce Figure 8, we removed a small fraction of samples for which the weight ratio exceeded 10^3.

[6] The simulator is publicly available at `https://github.com/clinicalml/gumbel-max-scm/tree/sim-v2`.

B.2.1 Model Parameters

Since we used the full state representation, we made the Markov assumption and estimated the behavior policy using a vanilla FNN and FNN-based prototype models—ProNet models—with different settings for the number of (prediction) prototypes. The FNN-coder had two layers, each of size 64, and ReLu as activation function. The FNN baseline used the same architecture with an additional linear layer to obtain predictions of the right shape

We trained all models over 30 epochs, using a batch size of 128. For optimization, the Adam algorithm was used with default parameters, learning rate 0.001 and weight decay 0.001. We selected parameters of the diversity regularization $(d_{\text{min}}, \lambda_d)$ by performing 3-fold cross-validation over a grid of points in the parameter space $\{1, 2, 3, 4, 5\} \times \{0.00001, 0.0001, 0.001, 0.01, 0.1\}$. The parameters λ_c and λ_e were set to 0.001, and we performed the projection step every fifth epoch.