Posterior-Driven Actor-Critic Framework for Active Hypothesis Testing

***Abstract*—We propose a readily adaptable, general purpose framework for learning active hypothesis testing policies across a wide variety of problem settings. Our PostAC framework trains a sequential, adaptive policy which tracks a Bayesian posterior over hypotheses and, at each time-step, uses a deep neural network to calculate the policy's actions as a function of this posterior. We first lay out an actor-critic algorithm that efficiently trains these DNN policies and then apply our algorithm to two problems: channel coding with feedback, where we match the performance of an analytically derived state-of-the-art coding scheme, and the problem of coherent state discrimination for optical communication, where we show state-of-the-art performance in the low power regime.**

***Index Terms*—Active hypothesis testing, deep learning, quantum state discrimination**

I. Introduction

In this work we lay out a posterior belief driven actor-critic algorithm, `PostAC`, which is the first general purpose framework for training deep neural network (DNN) policies for a wide variety of active hypothesis testing problems. While previous works have used deep learning to assist in the design of testing policies for specific hypothesis testing problems, these are generally bespoke solutions which require significant subject matter expertise to implement, are specifically adapted to certain action spaces [1] or observation distributions, or use DNNs as a sub-component [2] of a larger problem-specific policy design. Furthermore, many of these solutions rely on recursive neural networks to capture the state dynamics; as such, the knowledge of observation kernel are not explicitly taken into account [3]. In contrast, we create a flexible framework which can be applied to train a hypothesis testing policy for both fixed length and sequential tests, adaptive and non-adaptive tests, continuous and discrete action spaces, and both i.i.d. and dynamic environments.

We work in a Bayesian setting, where as the test progresses we maintain a belief over the hypotheses which we update via Bayes' rule upon collection of every new observation. This belief forms a sufficient statistic to inform the choice of action at each time-step [4] so we treat it as our *state*, in the reinforcement learning sense: at every time-step we choose an action as a function of the current belief. A similar approach was explored in [5] and bears some resemblance to belief value iteration for partially observable Markov decision processes (POMDPs) [6]. However, these approaches are not designed to handle the complexities of large, continuous action spaces and struggle with effectively exploring the continuous state space of belief vectors.

Our approach here enhances a deep reinforcement learning approach with the observation that Bayes' rule provides a known, differentiable state transition function in our setting. We leverage this to design a policy gradient algorithm wherein the policy network gradient updates contain information regarding both the cost associated with a given action–the likelihood of eventually making an incorrect prediction–but also explicit information about how the choice of action controls state transitions. Within this design the practitioner must only to specify appropriate neural network architectures for the policy and Q-function which take as input the current belief vector and time and produce a choice of action or an estimate of the cost to go, respectively. Customized gradient updates leverage the differentiability and structure of Bayes' rule–along with some other optional design choices which can improve performance–allow `PostAC` to learn effective policies with significantly less data and training instability than off-the-shelf deep RL algorithms. This ease of use, along with the broad class of hypothesis testing problem settings `PostAC` can accommodate, lend it great potential utility.

In the remainder of the paper, we will first describe the various settings considered within the world of active hypothesis testing and specify dynamic programming formulations of their respective objectives. We then lay out an actor-critic algorithm in which the posterior dynamic guides an adaptation of the well-known Deep Deterministic Policy Gradient (DDPG) [7] algorithm to train a policy that optimizes these objectives. Finally we consider a pair of applications: channel coding with feedback and quantum state discrimination. The channel coding results serve as a sanity check, showing that our algorithm can learn a policy comparable to the state-of-the-art, analytically driven solution. Notably, the state-of-the-art algorithm is the result of a long sequence of developments [8]–[12], whereas the `PostAC` policy can be trained with minimal subject matter expertise. In Section V we use `PostAC` to learn a sequential measurement strategy for quantum state discrimination which illustrates the primary strength of `PostAC`: its ability to learn from data in large, complicated problem settings that defy analytical solutions.

II. Background

A. Active Hypothesis Testing

In hypothesis testing an investigator is provided with a set of distributions–the hypotheses–and sampling access to an environment governed by one of the distributions. They are then generally tasked with identifying the correct hypothesis, but there are many variations on this problem. Wald [13]

first considered the *sequential* hypothesis test where, at each time-step, the tester is allowed to choose to terminate the test and declare a hypothesis or to continue the test and draw an additional sample. This allows the tester to adapt the number of samples required to the difficulty of a given problem instance.

Chernoff [14] further extended this setting to *active* testing. Here the tester has some measure of control over the distribution which the samples are drawn from. Their goal is to then to strategically choose actions so that the resulting observations are likely to be effective at discriminating between the hypotheses, increasing the sample efficiency of the test. The final defining characteristic of the test is adaptivity, we say a test is *adaptive* if the tester is allowed to condition their choice of action at each time-step based on the previously acquired samples. The performance of these tests are generally characterized by two competing quantities: the probability of error and the number of samples required. A thorough analysis and comparison of the benefits conveyed by these different capabilities can be found in [15].

B. Notation

We will consider tests over a finite, but possibly large, number of hypotheses which we identify with the set $[M] := 1, \ldots, M$. We work in a Bayesian setting, maintaining a belief, $\boldsymbol{\rho} \in \Delta_{M-1}$ over the hypotheses. $\mathcal{O}$ will denote a finite observation space and $\mathcal{A}$ the (possibly continuous) action space. Measurements will produce observations according to the user-specified *observation kernels* $q_i^a(\cdot)$ on $\mathcal{O}$, where $q_i^a(o)$ gives the probability of seeing observation o after taking action a when hypothesis i is true.

C. Problem Formulation

Sequential and fixed-length tests provide significantly different challenges in optimal policy design. Sequential tests, in a loose sense, can make decisions based primarily on their current level of confidence as there is always the option to terminate a test early or extend it indefinitely depending on the information gathered so far. In contrast, an optimal fixed-length policy must also be cognizant of the horizon, the remaining number of samples in the test. This poses significant challenges in both policy design and theoretical analysis of policies for this setting: indeed optimal control for adaptive, fixed-length tests with 3 or more hypotheses is unknown even in the error-exponent sense [16]. Despite this, `PostAC` learns effective policies for both settings with only minor modifications. We lay out both settings now.

We first consider a a fixed-length, adaptive test, where the tester knows at the outset that they must make a decision after some fixed, known number of observations. Here our objective is to minimize the probability of error at the conclusion of the test, which can be expressed recursively as

$$J_{FL}(\boldsymbol{\rho}, t) = \begin{cases} 1 - \max_j \rho(j) & t = N \\ \min_{a \in \mathcal{A}} \sum_{j=1}^{M} \rho(j) \sum_{o \in \mathcal{O}} q_j^a(o) J_{FL}\Big(\Phi(\boldsymbol{\rho}; o, a), t{+}1\Big) & t < N. \end{cases} \tag{1}$$

Where $\Phi(\boldsymbol{\rho}; o, a)$ performs the appropriate Bayesian update:

$$\Phi(\boldsymbol{\rho}; o, a) = \left(\frac{\rho(i) q_i^a(o)}{\sum_{j=1}^{M} \rho(j) q_j^a(o)} \right)_{i=1}^{M} \tag{2}$$

The sequential setting allows the tester, at each time-step and based on all observations made so far, to decide whether to terminate the test and declare a hypothesis or to make another observation. The number of samples in this setting is not fixed so we must balance between two objectives: minimization of the probability of declaring an incorrect hypothesis and minimization of the sample complexity of the test. If $L > 0$ defines a penalty for misprediction and $c : \mathcal{A} \to \mathbb{R}_+$ gives the cost of drawing a new sample after taking some action, then we can again express this objective recursively:

$$J_{Seq}(\boldsymbol{\rho}) = \min \begin{cases} L(1 - \max_j \rho(j)) \\ \min_{a \in \mathcal{A}} \; c(a) + \sum_{j=1}^{M} \rho(j) \sum_{o \in \mathcal{O}} q_j^a(o) J_{Seq}\Big(\Phi(\boldsymbol{\rho}; o, a)\Big). \end{cases} \tag{3}$$

Our goal is to design a policy $\pi : \Delta_{M-1} \to \mathcal{A}$ which, at each time-step, takes in the current belief, $\boldsymbol{\rho}$, as a summary of all accumulated information and prescribes the next action the tester should take so that the resulting sequence of actions, over the full course of the test, minimizes Equation (3). We first focus on learning a policy for the sequential setting, the action of which is illustrated in Figure 1. The fixed length setting is similar with minor modifications that will be described below.

We assume that the observation kernels, $q_j^a(\cdot)$, are known so Equation (3) is fully specified and, in principle, the optimal sequence of actions can be found or approximated by traditional dynamic programming techniques. But many of the hypothesis testing applications involve tests of long, or indefinite, length; large observation or hypothesis spaces; and continuous action spaces. Those based on value iteration generally cannot handle continuous state and action spaces and are expensive to scale to longer test lengths. So we instead pursue data-driven *policy optimization* approach.

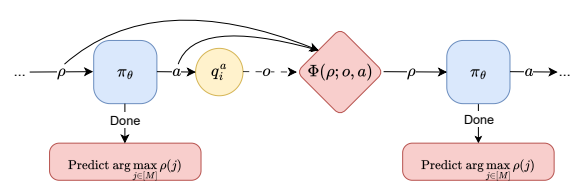

Fig. 1. Sequential `PostAC` execution: at each time-step the policy network, π_θ, may decide to terminate the test and predict the hypothesis with the largest current belief or to continue the test, in which case it chooses an action, draws an observation from the resulting observation kernel, and updates its belief via Bayes' rule.

D. Deep Q-Learning

In designing our algorithm we primarily draw from the Deep Deterministic Policy Gradient (DDPG) [7] algorithm. DDPG is an actor-critic algorithm [17] designed to handle continuous action spaces. It works by combining a *critic*, learned by deep Q-learning [18] and an *actor* learned with policy gradients.

The Q-function is a central concept in reinforcement learning wherein $Q(s,a)$ characterizes the expected return upon taking action a in state s. Q-learning [19] learns the Q-function via updates based on a Bellman equation

$$Q^*(s,a) = c(s,a) + \gamma \sum_{s'} p(s'|s,a) \max_{a' \in \mathcal{A}} Q^*(s',a'). \quad (4)$$

This recursive relationship prescribes an family of algorithms which iteratively improve estimates of the Q-function: given a current estimate of the Q-function and state s, take an action a, observe a cost $c(s,a)$ and subsequent state s', and update $Q(s,a)$ based on the cost and the Q-function evaluated at the subsequent states. This iterative approach is complicated in continuous state spaces. Deep Q-learning addresses this challenge by using a DNN to estimate the Q-function.

E. DDPG

While deep Q-learning can manage continuous state spaces, the max over actions in Equation (4) remains a complicating factor when working with continuous action spaces. To address this, DDPG uses two DNNs, Q_ϕ and π_θ, as function approximators for the Q-function and the policy where ϕ and θ are the learnable parameters of the respective networks. To learn the policy network's parameters, DDPG performs gradient descent on the estimated Q-function itself, evaluated at the action prescribed by the policy. This is possible since both the Q-function and policy are are calculated by differentiable neural networks and the action is assumed to be continuous.

DDPG uses two other key features common in RL algorithms. It accumulates a *replay buffer*, $\mathcal{D}$, of transitions by, at each time-step, sampling the state of the environment, acting according to the current policy, and observing the resulting transition. And it maintains *target networks* π_{θ^T} and Q_{ϕ^T} which are delayed moving averages of the actor and critic networks that yield more stable Bellman updates.

III. PostAC Algorithm

Active hypothesis testing differs, in both challenges and resources, from the standard RL problem in a few significant ways that motivate our algorithm design. The primary difference is that we have both a known state transition distribution and, conditioned on our choice of action and sampled observation, we also have a differentiable state transition function given by Bayes' rule. Careful use of these facts is key to sufficient exploration in our algorithm.

Our problem also has *sparse costs*: the misprediction error is only available at the end of the test. This challenge is complicated by the fact that greedy strategies are often sub-optimal for hypothesis testing problems. For instance: in problems with very many hypotheses it's most efficient to first take actions which may eliminate large swaths of hypotheses, only transitioning to a greedy one vs all strategy when a sufficiently likely hypothesis arises [20]. This suggests that standard approaches like reward shaping [21], which introduces hand-crafted local cost signals, may be misleading and that extensive exploration is necessary.

Gradient Calculations Given an action a and an observation o our state transition function is given by Bayes' rule. Moreover, we have known observation distributions $q_i^a(\cdot)$. With these two pieces of information, we perform both the Q function and policy updates averaged over all possible one-step transitions. At each time-step we sample a transition $(\boldsymbol{\rho}, a, c, d)$, consisting of a belief, action, cost, and termination decision, from our buffer of previously observed transitions. We then update the Q-function as shown in Algorithm 1.

Algorithm 1: QUpdate

Input: State $\boldsymbol{\rho}$, action (a,d), parameters (ϕ, ϕ^T, θ^T), lr η_Q, decay γ

$y \leftarrow 0$
for $o \in \mathcal{O}$ **do**
  $\boldsymbol{\rho}' = \Phi(\boldsymbol{\rho}; a, o)$
  $y_o = \gamma(1-d) Q_{\phi^T}\big(\boldsymbol{\rho}', \pi_{\theta^T}(\boldsymbol{\rho}')\big)$
  $y = y + \sum_{i=1}^{M} \rho(i) q_i^a(o) y_o$
end
$\phi \leftarrow \phi + \eta_Q \nabla_\phi \big(Q(s,a) - (c(\boldsymbol{\rho}, a) + y)\big)^2$

The standard approach here is to store whole transitions–a state, the action taken, and the subsequent observed state–in the buffer and use the sampled transitions to stochastically approximate the expectation in Equation (4). We sample only a state and action from our data buffer and then calculate the exact expectation over all possible observations and the target Q-function evaluated at the ensuing states and the action chosen by the target network evaluated on those states.

We take a similar approach in the the policy update, shown in Algorithm 2, where we again sample a state, $\boldsymbol{\rho}$ from our replay buffer, but now generate the action and termination decision with our current policy π_θ. If it chooses to terminate the test ($d = 1$) the loss is given by the current misprediction penalty. Otherwise we do the same sort of averaged look-ahead as in the Q-function update, where the loss is now our Q-function evaluated at a subsequent state and our policy's prescribed action at that state, averaged across every possible subsequent state.

Algorithm 2: PolicyUpdate

Input: State $\boldsymbol{\rho}$, parameters (ϕ, θ), lr η_P, penalty L

$\hat{Q}(\boldsymbol{\rho}, \pi_\theta) \leftarrow 0$
$(a,d) \leftarrow \pi_\theta(\boldsymbol{\rho})$
for $o \in \mathcal{O}$ **do**
  $\boldsymbol{\rho}' = \Phi(\boldsymbol{\rho}; a, o)$
  $\hat{Q}(\boldsymbol{\rho}, \pi_\theta) =$
  $\hat{Q}(\boldsymbol{\rho}, \pi_\theta) + \sum_{i=1}^{M} \rho(i) q_i^a(o) Q_\phi\big(\boldsymbol{\rho}', \pi_\theta(\boldsymbol{\rho}')\big)$
end
$\theta \leftarrow \theta + \eta_P \nabla_\theta \big[d \cdot L \cdot (1.0 - \max_{i \in [M]} \rho(i)) + (1 - d)\big(c(s,a) + \hat{Q}(\boldsymbol{\rho}, \pi_\theta)\big)\big]$

These modifications to the gradient updates are the defining characteristic of PostAC. Standard RL approaches must

estimate the expectation over subsequent states by sampling transitions as the state transition distributions are generally unknown. But, by leveraging the additional structure in the hypothesis testing problem to construct these modified gradients, we are able to incorporate direct information about how the policy action affects not only the expected value, through the Q-function, but the state evolution. This proved to be crucial in ensuring sufficient, effective exploration in `PostAC`.

Discrete Actions While our algorithm natively works with continuous action spaces, we also implement it for several problems with discrete action spaces. In these settings we use the common DL technique of treating the outputs of the network as parameters of a discrete distribution over the action space. In the policy update step we run the gradient directly through the parameters the network outputs, but for both the target actions in the Q-function updates and the trajectories added to the replay buffer we sample actions from the prescribed distribution in order to get a more accurate characterization of the end performance of the policy.

Critic Pre-training We are able to manage the challenge of sparse costs, in part, with another unique feature of our problem setting. The misprediction component of the cost, given by $1.0 - \max_{j \in [M]} \rho(j)$ is also a deterministic, (almost-everywhere) differentiable function of the state, $\boldsymbol{\rho}$. Moreover, the terminal states are themselves well-defined: in the sequential case they occur when algorithm chooses to terminate the test and in the fixed-length case they occur at the final time-step. We use this structure to pre-train our Q network by randomly generating beliefs $\boldsymbol{\rho}$ and actions a and calculating the ensuing Bayes update at the penultimate test step. This helps compensate for the sparse costs in our setting as it encodes in the network the general notion that beliefs near the edge of the simplex are high value while those near the center are low value.

Data Symmetries In hypothesis testing problems there often exist symmetries among the hypotheses which imply that the optimal policy should change in some structured, predictable way given certain permutations of the posterior. Formally this can be expressed as a pair of functions, a permutation $\sigma : [M] \to [M]$ and an associated transformation $f_\sigma : \mathcal{A} \to \mathcal{A}$, which leave the observation kernels unchanged: $q_i^a = q_{\sigma(i)}^{f_\sigma(a)}$. We will give examples of these relationships as we work through different applications of our algorithm.

These symmetries simplify policy design as they restrict the space of potentially optimal actions. DL is well suited to take advantage of this as the ability to enforce equivariances in neural networks is a rich field of study which includes convolutional networks for translation invariance in image processing, DeepSets [22] for set function, various symmetries corresponding to physical constraints [23], and the standard practice of data augmentation.

A. Fixed Length Testing

In fixed length tests the policy must take into account not only the current belief state, but also how many samples remain in the test as there is no benefit to ending the test early and no ability to extend the test if insufficient evidence has been collected at the final time-step. Our approach to the fixed length setting follows the same principles as the sequential test laid out above with minor modifications to address these additional difficulties. The first is that the time-step is included in the state of the problem and given to the policy neural network as part of the input at each time-step. Then while training the critic we proceed "backwards" through time: first taking a gradient step on a batch of transitions from timestep N, then from timestep $N-1$, and so on. Since only the final time-step receives an explicit cost signal and the update of the Q network is a function of the Q network evaluated at subsequent time-steps, this has the effect of propagating the cost signal back through the test.

B. `PostAC`

Algorithm 3: `PostAC`

Input: Parameters (ϕ, θ), action noise P_a, target lag τ

Pretrain(ϕ)
while *Policy not converged* **do**
  `/* Add new transition under current policy to replay buffer */`
  Observe current state $\boldsymbol{\rho}$
  $a \leftarrow \pi_\theta(\boldsymbol{\rho}) + \epsilon,\ \epsilon \sim P_a$
  $\mathcal{D} \leftarrow (\boldsymbol{\rho}, a)$
  $\boldsymbol{\rho}, a \sim \mathcal{D}$
  QUpdate($\boldsymbol{\rho}, a, \phi, \phi^T, \theta^T$)
  PolicyUpdate($\boldsymbol{\rho}, \phi, \theta$)
  `/* Update target networks */`
  $\phi^T \leftarrow (1-\tau)\phi^T + \tau\phi,\ \theta^T \leftarrow (1-\tau)\theta^T + \tau\theta$
end

With these design choices in place full pseudo-code for `PostAC` is given in Algorithm 3. As discussed above, to implement this algorithm for any particular hypothesis testing problem requires very few additional design choices. The observation kernels, q_i^a, which define the problem must be specified and appropriate neural network architectures must be chosen for Q_ϕ and π_θ, but in our applications thus far we've found that small, fully connected networks trained with learning rate tuning suffices to learn good policies.

C. Related Works

The closest comparison to our work is [5] which applies deep Q-learning [24] to policy design for active hypothesis testing with discrete action spaces. This work takes a more traditional reinforcement learning approach in its use of reward shaping [21] to address the challenges with sparse rewards discussed above. In particular, it greedily maximizes the expected change in log-likelihood of the posterior at every time-step.

Another common, related course of research is the application of recurrent neural networks (RNNs) to active hypothesis testing, controlled sensing [25], POMDPs [26], [27], and other related problems. These works generally consider settings less

precisely specified than ours, as in the POMDPs or in problems with unknown transition dynamics or unknown observation kernels as in [3]. With less structure imposed on the types of problems they handle, these approaches rely on the hidden state of the RNN to learn an information state and its function from data. In contrast, we dictate the use of the posterior belief and its update by Bayes' rule and build this into the training process so that the policy is tailored to it. This allows us to bypass the challenges often associated with training RNNs [28] and facilitates application of our algorithm to larger, more complex hypothesis testing problems.

While our approach is designed to be a flexible, plug-and-play approach to designing policies for any active hypothesis testing problem, DL has also been applied to specific instances of hypothesis tests. This is often done in a bespoke manner: either to solve a specific sub-problem or with constraints supplied by problem-specific side knowledge. Some such applications include mmWave beam alignment [1], topics within channel coding like [2] and [29], and group recommendations [30]. Application of the `PostAC` framework to these settings and comparison to the hand-crafted solutions is an important direction for future work.

IV. Channel Coding with Feedback

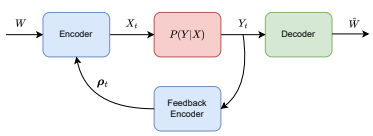

Fig. 2. Channel coding with the feedback encoded in the posterior $\boldsymbol{\rho}$.

Channel coding [31, Chapter 8] is one of the fundamental problems of information theory. It concerns the amount of information which can be transmitted over a noisy channel and how to effectively achieve this communication. Formally we must take a message, W, and encode it into a sequence of symbols, X^n. This sequence of symbols is sent through the channel with conditional probability distribution $P(Y|X)$ and the receiver, on the other end of the channel, observes a sequence of symbols $Y^n \sim P(Y^n|X^n)$ and decodes them to obtain a putative message $\hat{W}$. Success is measured by the probability that the correct message is decoded: $\mathbb{P}[W = \hat{W}]$.

We consider the expanded setting of channel coding *with feedback*–where the transmitting party receives immediate, noiseless feedback of what symbol the receiver sees. This setting is an active hypothesis testing problem where the actions are the choice of encodings, the observations are the channel outputs Y, and decoding is done by maintaining a belief over the messages and declaring $\hat{W} = \arg\max_{j\in[M]} \rho_j$.

While feedback cannot improve the capacity of a channel [32], Burnashev [9] first showed that, when designing variable length codes to achieve a targeted probability of error, feedback can improve the error exponent. This analysis, and designs of practical coding schemes, was extended to the non-asymptotic regime in [12], [33], [34].

A. `PostAC` for Channel Coding

We first consider a binary symmetric channel with crossover probability p. At each time-step we will characterize our chosen encoding by a subset $S_t \subseteq [M]$, and for each message i, we set $X_t{=}0$ if $i \in S_t$ and $X_t{=}1$ otherwise. Thus our observation kernels are Bernoulli distributions: $q_i^S \overset{d}{=} \mathcal{B}(p)$ for $i \in S$ and $q_i^S \overset{d}{=} \mathcal{B}(1{-}p)$ for $i \notin S$. To instantiate our algorithm we must design a policy network which selects S given a belief $\boldsymbol{\rho}$ over the M messages. There are 2^M possible actions, so allowing the output of the network to be an enumeration of possible subsets is infeasible. We instead set the output dimension to be M and identify the ith output with an indicator as to whether $i \in S$.

Hypothesis Symmetry As discussed in Section II, it is often possible to take advantage of symmetries in hypothesis testing problems. This is particularly the case here as the channel coding problem is fully symmetric–the messages are interchangeable and so all that matters is the relative likelihoods among them. We leverage this by first sorting the posterior before feeding it to our policy network, then un-sorting the resulting actions: if σ_{sort} is a permutation that orders $\boldsymbol{\rho}$ then our policy is given by $\sigma_{\text{sort}}^{-1}\big(\pi_\theta\big(\sigma_{\text{sort}}(\boldsymbol{\rho})\big)\big)$.

B. Binary Symmetric Channel Results

We compare the policy learned by `PostAC` for a BSC to the SED coding scheme of [35], a theoretical upper bound on expected stopping time for SED coding [34, Theorem 7], and a lower bound on all variable length feedback codes from [36, Theorem 3]. These results show that the coding scheme learned by DAHT achieves performance comparable to state-of-the-art algorithms in this regime.

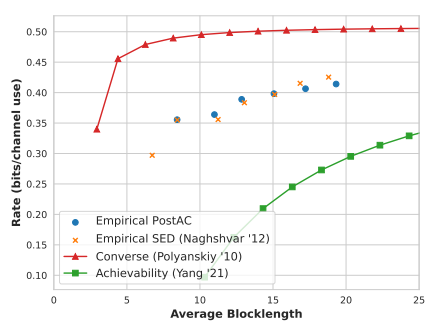

Fig. 3. Bits per channel usage for `PostAC` and SED coding schemes for a BSC with crossover $p = 0.11$ and target error probability 10^{-3}. The converse, shown in red, is derived in [33] and the green curve is a theoretical bound on the performance of the SED scheme, obtained by [34].

This result serves as a good sanity check, they verify that our framework learns policies comparable to the best known algorithms in regimes which are complex, but still somewhat susceptible to analytical tools. Moreover, design of algorithms like the SED coding scheme is the end result of a significant amount of expertise and a sequence of developments in terms of the analysis of the channel structure, how it interacts with feedback, and practical ways to turn that analysis into an

actual coding scheme. In contrast, very little domain expertise is needed to implement `PostAC` and learn a policy with comparable performance. But the primary strength of DL based approaches is that they often work well in regimes too large or complex for hand-designed approaches: we explore one such setting below.

V. Coherent State Discrimination

In this section we explore an application of the `PostAC` framework to the problem of coherent state discrimination in quantum optics. This problem can be abstracted to that of an active hypothesis test among a set of Poisson distributions, and our focus here is on the performance of `PostAC`, so we will give only a brief discussion of the physics context, but see [37], [38] for background on coherent states and [39], [40] for discussion of this problem in the context of active hypothesis testing.

Coherent states are states of the electromagnetic field which are parameterized by a complex scalar $\alpha \in \mathbb{C}$. The states are vectors in a complex Hilbert space, commonly written as a *ket*: $|\alpha\rangle$. In coherent state discrimination we are given an unknown coherent state which is one of a known set of states and our task is to design a measurement to identify the correct state with minimal probability of misprediction. We will consider the M-ary phase-shift keying (PSK) alphabet, $\left\{|\alpha e^{\frac{2\pi i k}{M}}\rangle\right\}_{k=1}^{M}$, which is a set of M coherent states with amplitudes of equal magnitude, distributed uniformly on a ring in $\mathbb{C}$.

A. Problem Formulation

In order to apply `PostAC` to this problem setting we need to define the allowed action and observation spaces and the observation kernels, the observation distributions as a function of the true, unknown coherent state and our choice of action. We follow the line of work in coherent state detection originating with the Kennedy receiver [41] which allow the tester to apply an arbitrary *displacement* to the state, which is then measured via *photon detection*.

Photon Detection In general, the number of photons observed when measuring a coherent state $|\alpha\rangle$ is given by a Poisson distribution with mean $|\alpha|^2$ [38]. However, detectors have a resolution: the maximum number of photons they're able to distinguish, which we'll denote K. It is common to consider $K = 1$, i.e. distinguishing 0 photons from 1 or more photons. This significantly simplifies the design and analysis of testing algorithms, but it does throw away information when 2 or more photons could be detected. Our DL policy allows us to accomodate more complicated observation kernels and so leverage higher photon resolutions.

Time-Sliced States Given a coherent state $|\alpha\rangle$ we can *slice* the state into N coherent states $|\alpha/N\rangle$–i.e. states with mean photon number $|\alpha|^2/N$. This slicing must be performed at the outset of the test, by application of a beam splitter to the initial coherent state that we're testing. This trade-off, between the length and per-sample utility of the test, is a novel question in the realm of hypothesis testing that was explored theoretically in [40]. But importantly, the degree of slicing must be chosen before beginning the test. This means that, from a hypothesis testing perspective, we're in a fixed-length testing setting.

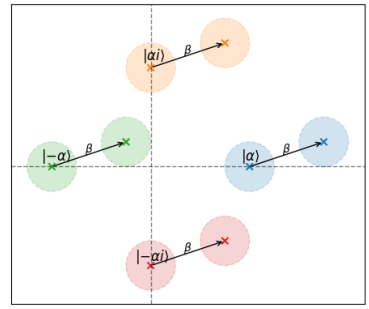

Fig. 4. Displacement β applied to the M=4 PSK constellation.

Displacement Before performing photon detection our tester will be allowed to apply an arbitrary *displacement* to the unknown state. This is an operation on a coherent state that acts as a translation of the state's amplitude in $\mathbb{C}$. So if we apply a displacement $\beta \in \mathbb{C}$, denoted by the operator $\hat{D}(\beta)$, to a coherent state $|\alpha\rangle$, we obtain the new coherent state $\hat{D}(\beta)\,|\alpha\rangle = |\alpha + \beta\rangle$. The effect of this action on the $M = 4$ PSK constellation is illustrated in Figure 4. This means that $\mathbb{C}$ is our action space and our policy must choose the displacement parameter, $\beta \in \mathbb{C}$, at each time-step.

By applying displacement β and doing photon detection with resolution K, our observation kernels are given by

$$q_\alpha^\beta(n) := \begin{cases} \dfrac{e^{|\alpha+\beta|^2}|\alpha+\beta|^{2n}}{n!} & n < K \\ \displaystyle\sum_{n=K}^{\infty} \dfrac{e^{|\alpha+\beta|^2}|\alpha+\beta|^{2n}}{n!} & n \geq K. \end{cases} \tag{5}$$

Hypothesis Symmetry The symmetry of the PSK constellation is which is more complicated than that of the channel coding problem. Here the hypotheses possess only rotational symmetry: their amplitudes are uniformly distributed on a ring of radius α in $\mathbb{C}$. If we let σ be the permutation such that $\sigma : \rho(i) \rightarrowtail \rho\big((i+1)\%M\big)$, i.e. rotates the posterior around the hypotheses, then if our policy chooses a displacement $\beta = \pi_\theta(\boldsymbol{\rho})$ we expect that $\pi_\theta\big(\sigma(\boldsymbol{\rho})\big) = e^{\frac{2\pi i}{M}}\beta$. We leverage this via data augmentation: when sampling trajectories for our data buffer we randomly apply a permutation to the belief state and the associated rotation to the prescribed action. Since our actions here are simply points in the complex plane we can easily visualize them: Figure 5 shows the distribution of our policy's displacements at a single time-step over many simulations of the test–clearly showing the expected rotational symmetry in the policy.

B. Baselines

This setting originated with the study of the M=2 case where it's known that the Dolinar receiver [42], using displacement and photon detection, achieves the theoretically optimal probability of error, the Helstrom limit [43], in the limit of infinitely fine slicing. But, for $M > 2$ it is an open question if the Helstrom limit is achievable for this setting.

Bondurant [44] extended the Dolinar receiver to the many hypothesis case with the *sequential nulling* receiver, which sequentially displaces each of the hypotheses to the vacuum, $|0\rangle$, meaning that if a photon is detected the nulled hypothesis can be rejected. [45] also applies only nulling displacements, but maintains a belief over the hypotheses and nulls the current most likely hypothesis at each time-step. This receiver is known to be error-exponent optimal and so performs well in the high power, high average photon number regime, so here we focus on the challenging low power regime. We will compare our `PostAC` policy against the Becerra receiver as well as classical heterodyne detection.

C. Policy Properties

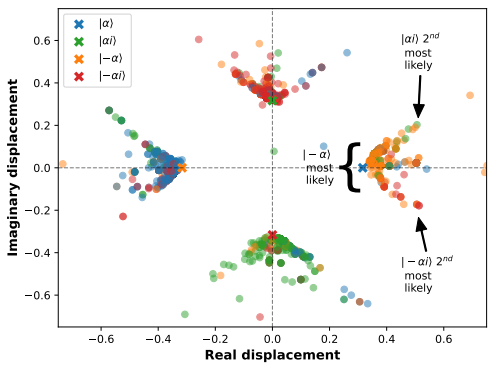

Fig. 5. Distribution of displacements from one time-step of a DL policy during inference. Each point indicates a separate simulation of a test with the colors of the points indicating the true hypothesis underlying that test and the $\times$'s indicate the hypotheses themselves.

We'll first examine the qualitative properties of the displacement policy learned by `PostAC` here. Figure 5 shows the distribution of displacements prescribed by a `PostAC`-learned policy for an $N = 10$ step,$M = 4$ hypothesis test. Each point shows the displacement applied by the DNN policy at the 9th time-step of one execution of the test and the color of the points indicates the true hypothesis underlying that instance of the test. The $\times$'s indicate the hypotheses themselves.

One common complication from the use of DNNs is that they function as black boxes: they may work well, but make decisions in a way that is incomprehensible to humans [46]. This is not the case here as the results are very intuitive and can be viewed as a natural extension of the nulling receivers of [44], [45]. The `PostAC` displacements also feature the nulling displacements: when the true hypothesis is $|\alpha\rangle$, the prescribed displacements concentrate near $|-\alpha\rangle$. But the `PostAC` policy allows arbitrary, adaptive displacements, resulting in a more sophisticated policy: we can see additional structure in the wedge shape that spreads out from the nulling displacements. These deviations correspond to the *second* most likely hypothesis according to the current belief, $\boldsymbol{\rho}$. I.e. the policy aims to null the most likely hypothesis while *increasing* the expected photon count of the second most likely hypothesis. This creates a situation where observing no photons is continued confirmation of the current prediction, but observing a photon is strong evidence for the second most likely hypothesis.

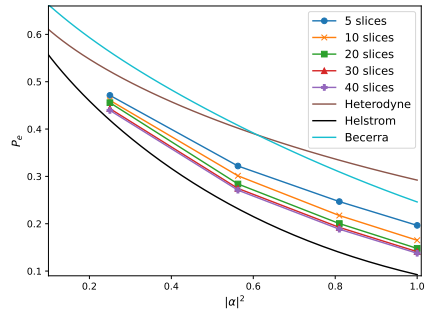

Fig. 6. Probability of error as a function of mean photon number for tests of differing lengths.

D. Slicing Dependence

Here we compare the performance of the `PostAC` policy to our baselines and examine the trade-off in slicing the state more finely to obtain additional samples. In the $M = 2$ case the Dolinar receiver achieves the optimal probability of error only in the limit of infinite slices and, generally, in hypothesis testing more samples are better. But in this setting slicing the state more thinly to obtain more samples results in observation kernels with lower mean photon number.

Figure 6 shows the results of a test of $M = 4$ coherent states with the `PostAC` policies trained for $N \in \{5, 10, 20, 30, 40\}$ slices and compared against the Helstrom limit, classical heterodyne detection, and the Bayesian sequential nulling receiver of [45]. We can see that, in the low photon regime, the `PostAC` policies all significantly outperform the baselines. Moreover, there is a clear pattern of improvement for tests on increasing numbers of samples, but this pattern shows steadily diminishing returns. For $M > 2$ it is unknown if the Helstrom limit is attainable via displacement and direct detection–this offers evidence that it may not be.

VI. Discussion

We've laid out the first general-purpose, readily adaptable method for learning DNN policies for active hypothesis testing. With minimal user specification, `PostAC` can learn policies for fixed-length or sequential tests, i.i.d. or dynamic environments, and arbitrary action spaces. We established that the policy learned by `PostAC` in the relatively well understood setting of channel coding matches state-of-the-art performance and, in the more complicated setting of coherent state discrimination, we showed that the learned policy achieved state-of-the-art performance in the low photon regime. Moreover, it does so in a clearly interpretable way which can serve as motivation for the development of more sophisticated, analytically optimized policies.

REFERENCES

[1] D. Tandler, S. Doerner, M. Gauger, and S. ten Brink, "Deep reinforcement learning for mmwave initial beam alignment," in *WSA & SCC 2023; 26th International ITG Workshop on Smart Antennas and 13th Conference on Systems, Communications, and Coding*. VDE, 2023, pp. 1–6.

[2] F. Carpi, C. Häger, M. Martalò, R. Raheli, and H. D. Pfister, "Reinforcement learning for channel coding: Learned bit-flipping decoding," in *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2019, pp. 922–929.

[3] G. Stamatelis and N. Kalouptsidis, "Active hypothesis testing in unknown environments using recurrent neural networks and model free reinforcement learning," in *2023 31st European Signal Processing Conference (EUSIPCO)*. IEEE, 2023, pp. 1380–1384.

[4] K. J. Astrom, "Optimal control of markov decision processes with incomplete state estimation," *J. Math. Anal. Applic.*, vol. 10, pp. 174–205, 1965.

[5] D. Kartik, E. Sabir, U. Mitra, and P. Natarajan, "Policy design for active sequential hypothesis testing using deep learning," in *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2018, pp. 741–748.

[6] R. D. Smallwood and E. J. Sondik, "The optimal control of partially observable markov processes over a finite horizon," *Operations research*, vol. 21, no. 5, pp. 1071–1088, 1973.

[7] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[8] M. Horstein, "Sequential transmission using noiseless feedback," *IEEE Transactions on Information Theory*, vol. 9, no. 3, pp. 136–143, 2003.

[9] M. V. Burnashev, "Data transmission over a discrete channel with feedback. random transmission time," *Problemy peredachi informatsii*, vol. 12, no. 4, pp. 10–30, 1976.

[10] H. Yamamoto and K. Itoh, "Asymptotic performance of a modified schalkwijk-barron scheme for channels with noiseless feedback (corresp.)," *IEEE Transactions on Information Theory*, vol. 25, no. 6, pp. 729–733, 2003.

[11] O. Shayevitz and M. Wigger, "On the capacity of the discrete memoryless broadcast channel with feedback," *IEEE Transactions on Information Theory*, vol. 59, no. 3, pp. 1329–1345, 2012.

[12] M. Naghshvar, T. Javidi, and M. Wigger, "Extrinsic jensen–shannon divergence: Applications to variable-length coding," *IEEE Transactions on Information Theory*, vol. 61, no. 4, pp. 2148–2164, 2015.

[13] A. Wald, "Sequential Tests of Statistical Hypotheses," *The Annals of Mathematical Statistics*, vol. 16, no. 2, pp. 117 – 186, 1945. [Online]. Available: https://doi.org/10.1214/aoms/1177731118

[14] H. Chernoff, "Sequential Design of Experiments," *The Annals of Mathematical Statistics*, vol. 30, no. 3, pp. 755 – 770, 1959. [Online]. Available: https://doi.org/10.1214/aoms/1177706205

[15] M. Naghshvar and T. Javidi, "Sequentiality and adaptivity gains in active hypothesis testing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 5, pp. 768–782, 2013.

[16] S. Nitinawarat, G. K. Atia, and V. V. Veeravalli, "Controlled sensing for multihypothesis testing," *IEEE Transactions on automatic control*, vol. 58, no. 10, pp. 2451–2464, 2013.

[17] V. Konda and J. Tsitsiklis, "Actor-critic algorithms," *Advances in neural information processing systems*, vol. 12, 1999.

[18] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[19] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3, pp. 279–292, 1992.

[20] M. Naghshvar and T. Javidi, "Active sequential hypothesis testing," *Annals of Statistics*, 2013.

[21] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Icml*, vol. 99. Citeseer, 1999, pp. 278–287.

[22] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, "Deep sets," *Advances in neural information processing systems*, vol. 30, 2017.

[23] M. Mattheakis, P. Protopapas, D. Sondak, M. Di Giovanni, and E. Kaxiras, "Physical symmetries embedded in neural networks," *arXiv preprint arXiv:1904.08991*, 2019.

[24] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[25] S.-H. Choi and S.-J. Yoo, "Rnn based optimal sensing schedule control for wireless sensor networks," in *2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, 2021, pp. 150–154.

[26] S. Carr, N. Jansen, and U. Topcu, "Verifiable rnn-based policies for pomdps under temporal logic constraints," *arXiv preprint arXiv:2002.05615*, 2020.

[27] ——, "Task-aware verifiable rnn-based policies for partially observable markov decision processes," *Journal of Artificial Intelligence Research*, vol. 72, pp. 819–847, 2021.

[28] F. Bonassi, M. Farina, J. Xie, and R. Scattolini, "On recurrent neural networks for learning-based control: recent results and ideas for future developments," *Journal of Process Control*, vol. 114, pp. 92–104, 2022.

[29] H. Kim, Y. Jiang, S. Kannan, S. Oh, and P. Viswanath, "Deepcode: Feedback codes via deep learning," *Advances in neural information processing systems*, vol. 31, 2018.

[30] M. Stratigi, E. Pitoura, and K. Stefanidis, "Squirrel: A framework for sequential group recommendations through reinforcement learning," *Information Systems*, vol. 112, p. 102128, 2023.

[31] T. M. Cover, *Elements of information theory*. John Wiley & Sons, 1999.

[32] C. Shannon, "The zero error capacity of a noisy channel," *IRE transactions on information theory*, vol. 2, no. 3, pp. 8–19, 2003.

[33] Y. Polyanskiy, H. V. Poor, and S. Verdú, "Variable-length coding with feedback in the non-asymptotic regime," in *2010 IEEE International Symposium on Information Theory*. IEEE, 2010, pp. 231–235.

[34] H. Yang, M. Pan, A. Antonini, and R. D. Wesel, "Sequential transmission over binary asymmetric channels with feedback," *IEEE Transactions on Information Theory*, vol. 68, no. 11, pp. 7023–7042, 2022.

[35] M. Naghshvar, M. Wigger, and T. Javidi, "Optimal reliability over a class of binary-input channels with feedback," in *2012 IEEE Information Theory Workshop*, 2012, pp. 391–395.

[36] Y. Polyanskiy, H. V. Poor, and S. Verdu, "Feedback in the non-asymptotic regime," *IEEE Transactions on Information Theory*, vol. 57, no. 8, pp. 4903–4925, 2011.

[37] R. J. Glauber, "Coherent and incoherent states of the radiation field," *Physical Review*, vol. 131, no. 6, p. 2766, 1963.

[38] C. C. Gerry and P. L. Knight, *Introductory quantum optics*. Cambridge university press, 2023.

[39] H. W. Chung, S. Guha, and L. Zheng, "Capacity of optical communications over a lossy bosonic channel with a receiver employing the most general coherent electro-optic feedback control," *Physical Review A*, vol. 96, no. 1, p. 012320, 2017.

[40] Y.-F. Lo and M. R. Bloch, "Active hypothesis testing for quantum detection of phase-shift keying coherent states," *arXiv preprint arXiv:2501.16218*, 2025.

[41] R. S. Kennedy, "A near-optimum receiver for the binary coherent state quantum channel," *Research Laboratory of Electronics, MIT, Quarterly Progress Report*, vol. 108, pp. 219–225, 1973.

[42] S. J. Dolinar, "An optimum receiver for the binary coherent state quantum channel," *Research Laboratory of Electronics, MIT, Quarterly Progress Report*, vol. 11, pp. 115–120, 1973.

[43] C. W. Helstrom, "Quantum detection and estimation theory," *Journal of Statistical Physics*, vol. 1, pp. 231–252, 1969.

[44] R. S. Bondurant, "Near-quantum optimum receivers for the phase-quadrature coherent-state channel," *Opt. Lett.*, vol. 18, no. 22, pp. 1896–1898, Nov 1993. [Online]. Available: https://opg.optica.org/ol/abstract.cfm?URI=ol-18-22-1896

[45] F. Becerra, J. Fan, G. Baumgartner, J. Goldhar, J. Kosloski, and A. Migdall, "Experimental demonstration of a receiver beating the standard quantum limit for multiple nonorthogonal state discrimination," *Nature Photonics*, vol. 7, no. 2, pp. 147–152, 2013.

[46] Y. Zhang, P. Tiňo, A. Leonardis, and K. Tang, "A survey on neural network interpretability," *IEEE transactions on emerging topics in computational intelligence*, vol. 5, no. 5, pp. 726–742, 2021.