Gold Seeker: Information Gain from Policy Distributions for Goal-oriented Vision-and-Langauge Reasoning

Ehsan Abbasnejad[1], Iman Abbasnejad[2], Qi Wu[1], Javen Shi[1], Anton van den Hengel[1]
[1]{ehsan.abbasnejad,qi.wu01,javen.shi,anton.vandenhengel}@adelaide.edu.au
[2]i.abbasnejad@fugro.com
[1]Australian Institute for Machine Learning & The University of Adelaide, Australia,
[2]Fugro Australia Marine

Abstract

As Computer Vision moves from passive analysis of pixels to active analysis of semantics, the breadth of information algorithms need to reason over has expanded significantly. One of the key challenges in this vein is the ability to identify the information required to make a decision, and select an action that will recover it. We propose a reinforcement-learning approach that maintains a distribution over its internal information, thus explicitly representing the ambiguity in what it knows, and needs to know, towards achieving its goal. Potential actions are then generated according to this distribution. For each potential action a distribution of the expected outcomes is calculated, and the value of the potential information gain assessed. The action taken is that which maximizes the potential information gain. We demonstrate this approach applied to two vision-and-language problems that have attracted significant recent interest, visual dialog and visual query generation. In both cases the method actively selects actions that will best reduce its internal uncertainty, and outperforms its competitors in achieving the goal of the challenge.

1. Introduction

The majority of problems that computer vision might be applied to greatly benefit from agents capable of actively seeking the information they need. This might be because the information required is not available at training time, or because it is too broad to be embodied in the code or weights of an algorithm. The ability to seek the information required to complete a task enables a degree of flexibility and robustness that cannot be achieved through other means.

Some of the applications that lie at the intersection of vision and language have this property, including visual dialog [12, 11], visual question answering [13, 23, 40], and vision-and-language navigation[6]. These problems require an agent (model) to acquire information on the fly to help to make decisions, because the space of all possible questions (or dialogues) encompasses more information than can be encoded in a training set. Additionally, a range of tasks have been proposed recently that use 'language generation' as a mechanism to gather information towards achieving a (non-language based) goal[47, 23, 24]. These tasks offer a particular challenge because the set of all information that might possibly be involved is inevitably very broad, which makes concrete representations difficult to employ practically.

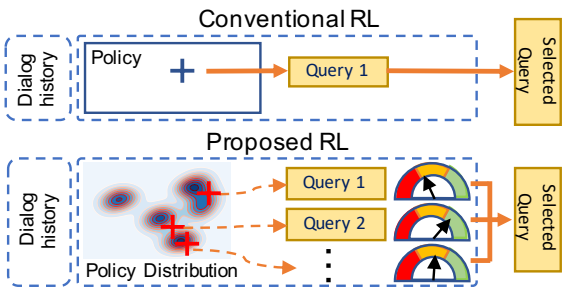

Figure 1: The role of the SEEKER in many conventional RL-based agents is to select an action based on the current (single) policy. In goal-oriented visual dialogue this means selecting the next query based on the image and the dialogue thus far. Our proposed SEEKER instead exploits a distribution of policies to generate multiple query hypotheses. The one that maximises the potential information gain is then selected. In this sense the agent is able to identify gaps in the information it holds and ask questions that will fill them.

In a visual dialog, and particularly goal-oriented visual question generation, an agent needs to understand the user request and complete a task by asking a limited number of questions. Similarly, compositional VQA (e.g. [21]) is a visual query generation problem that requires a model first to convert a natural language question to a sequence of actions (a 'program') and then obtain the answer by running

the programs on an engine. The question-to-program model represents an information 'seeker', while the broader goal is to generate an answer based on the information acquired.

Agents applicable to these tasks typically consist of four parts: a *context encoder*, an *information seeker*, a *responder* and a *goal executor*, as shown in Fig.2. The context encoder is responsible for encoding information such as images, questions, or dialog history to a feature vector. The information seeker is a model that is able to generate new queries (such as natural language questions and programs) based on the goal of the given task and its strategy. The information returned by the responder is added to the context and internal information and sent to the goal executor model to achieve the goal. The seeker model plays a crucial role in goal-oriented vision-and-language tasks, as better seeking strategies that recover more information improve the chance of the goal being achieved. Moreover, the seeker's knowledge of the value of additional information is essential in directing the seeker towards querying what is needed to achieve the goal. In this paper, we focus on exploring the **seeker** and **responder** models.

The conventional 'seeker' models in these tasks follow a sequence-to-sequence generation architecture, that is, they translate an image to a question, or translate a question to a program sequence via supervised learning. This requires large numbers of ground-truth training pairs. Reinforcement learning (RL) is thus employed in such goal-oriented vision-language tasks to mediate this problem due to the RL's ability to focus on achieving a goal through directed trial and error [13]. A policy in RL models specifies how the seeker asks for additional information. However, these methods generally suffer from two major drawbacks: (1) they maintain a single policy that translates the input sequence to the output while disregarding the strategic diversity needed. Intuitively a single policy is not enough in querying diverse information content for various goals–we need multiple strategies. In addition, (2) the RL employed in these approaches can be prohibitively inefficient since the question generation process (or query generation) does not consider its effect in directing the agent towards the goal. In fact, the agent does not have a notion of what information it needs and how it benefits in achieving its goal.

To this end, in contrast to conventional methods that use a single policy to model a vision-and-language task, we instead maintain a ***distribution of policies***. By employing a Bayesian reinforcement learning framework for learning this distribution of the seeker's policy, our model incorporates the expected gain from a query towards achieving its goal. Our framework, summarized in Fig. 1, uses recently proposed Stein Variational Gradient Descent [26] to perform an efficient update of the posterior policies. Having a distribution over seeking policies, our agent is ***capable of considering various strategies for obtaining further information***, analogous to human contemplation of various ways to ask a question. Each sample from the seeker's policy posterior represents a policy of its own, and seeks a different piece of information. This allows the agent to further contemplate the outcome of the various strategies before seeking additional information and considers the consequence towards the goal. We then formalize an approach for the agent to ***evaluate the consequence of receiving additional information*** towards achieving its goal. This consequence is the intrinsic reward the policy distribution receives to update its strategy for question generation.

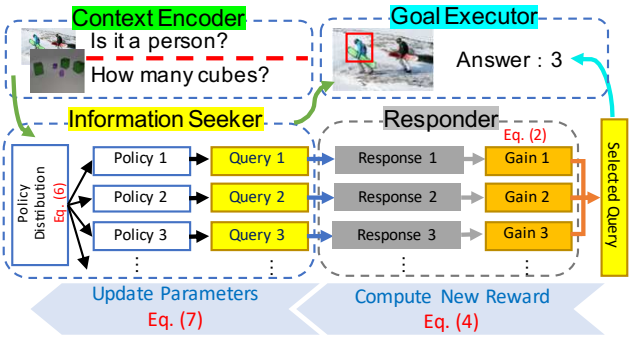

Figure 2: Information seeker maintains a distribution over the policies. Each sample (particle) from the distribution can generate a different query. For each query, responder evaluates its *gain* and the best one is chosen to be executed in EXECUTOR. Based on how well the query leads to achieving its goal, the *policy distribution* is updated.

We apply the proposed approach to two complex vision-and-language tasks, namely GuessWhat [13] and CLEVR [21], and show that it outperforms the comparative baselines and achieves the state-of-art results.

2. Related work

Goal-oriented Visual Dialog Das *et al*. [12] proposed a visual dialogue task that requires an agent to engage in conversation with a human, centred on the content of a given image. They further (in [11]) propose the use of reinforcement learning in two tasks for visual dialog. de Vries *et al*. in [13] propose a Guess-What game dataset, where one person asks questions about an image to guess which object has been selected, and the second person answers. This was a critical step in the development of goal-oriented visual dialogue because the objective there was not merely to continue the conversation, but rather to achieve a secondary goal (winning the game). Lee *et al*. [24] then developed an information theoretic approach which allows a questioner to ask appropriate consecutive questions in GuessWhat.

RL in Vision-and-Language problems Reinforcement learning (RL) [22, 44, 30] has been adopted in several

vision-and-language problems, including image captioning [27, 35, 36], VQA [17, 4, 10], and visual dialogue [11, 29]. Recently, some works [8, 41, 5, 3] have integrated Seq2Seq models and RL. RL has also been widely used to improve dialogue managers, which manage transitions between dialogue states [34]. However, nearly all of the methods use a single policy that translates the input sequence to an output. In our work, we instead maintain a distribution of policies.

Intrinsic rewards Intrinsic rewards refer to rewards beyond those gained from the environment in RL. These rewards are motivated by the sparse nature of environmental rewards and a need to motivate better exploration. For example, curiosity [32] is one such intrinsic reward mechanism by which agents are encouraged to visit new states. This idea has been extended to employ Bayesian methods to learn the expected improvement of the policy for taking an action [16, 19]. We use the expected gain in a vision-and-language task as an intrinsic reward to improve our model.

3. Goal-oriented Vision-Language Task

We represent our goal-oriented vision-and-language solution as having four constituent parts. The SEEKER takes as input the encoded image and context features produced by an ENCODER to generate a query to seek more information from a RESPONDER, that will generate a response. The role of RESPONDER is to model the environment in order to allow the agent to determine which query is best to ask. It takes in a question to produce an answer and its predicted score. Our RESPONDER is an extension of the 'A-BOT' in [11], in that it instead considers a distribution over possible answers and generates a corresponding upper bound on the score of the question. This is a critical distinction, and represents an approach that is far more statistically justifiable. To enable our approach we develop a synthetic RESPONDER inspired by the neuro-scientific formulation of agency that computes the translation of intentions into actions and evaluates their consequences in terms of predicted and actual experiences [9].

Typically the SEEKER learns a policy to generate queries on the basis of image and context features. The primary novelty in our proposed approach is that the SEEKER instead maintains a distribution of policies that enables multiple query hypotheses to be sampled. The RESPONDER then calculates an upper bound on the information gain for each hypothesis. The final query is that corresponding to the maximal (upper bound on the) information gain.

Formally, for each game at round t, we have a tuple $(I, C, q^{(t)})$, where I is the observed image, C is the context information at the current round[1] and $q^{(t)}$ is a query generated by the SEEKER agent. Subsequently, $q^{(t)}$ is sent to the RESPONDER that generates a response $a^{(t)}$. The RESPONDER imitates the potential response for a query and evaluates its value. After T rounds of this 'seek-answer' process, the tuple $(I, C, \{q^{(t)}\}_{t=1}^{T}, \{a^{(t)}\}_{t=1}^{T})$ is sent to the EXECUTOR who selects the target from the candidate list $O = \{o_1, o_2, ..., o_N\}$. The ground truth target is denoted as o^* and the game is success if the o^* is successfully selected by the EXECUTOR.

To be more specific, in the Guesswhat (visual dialog) setting, C is the dialog history and $q^{(t)}$ is a natural language question. Then, O is the candidate object bounding boxes. The answer $a^{(t)}$ is provided by the oracle as either Yes/No or N/A (not applicable for the cases when the question is unrelated). In the CLEVR (VQA), C is a single question asked by users and $q^{(t)}$ is a functional program, while the O is a candidate answer vocabulary. The answers in this problem are the same as the target candidates.

3.1. Reinforcement Learning

Reinforcement learning considers agents interacting with their environment by taking a sequence of actions and assessing their effect through a scalar reward. The agent's task is to learn a *policy* that maximizes the expected cumulative rewards from its interaction with the environment.

Consider a vision-and-language task where the agent generates a query $q^{(t)} \in \mathcal{Q}$ at each time step t given the state $\mathbf{s}^{(t)}$. Each $\mathbf{s}^{(t)}$ encompasses the history of the dialog (including past query-answer pairs) and the input image. Upon receiving an answer $a^{(t)} \in \mathcal{A}$ for the query, the agent then observes a new state $\mathbf{s}^{(t+1)}$ and receives a scalar reward $r(\mathbf{s}^{(t)}, q^{(t)}) \in R$. The goal of the reinforcement learning in this task is to find a querying policy $\pi(q^{(t)}|\mathbf{s}^{(t)}, \boldsymbol{\theta})$ given the state $\mathbf{s}^{(t)}$ to maximize an expected return:

$$J(\pi) = \mathbb{E}_{\mathbf{s}_0, q_0, \ldots \sim \pi}[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}^{(t)}, q^{(t)})],$$

where $0 \leq \gamma^t \leq 1$ is a discount factor. State variable $\mathbf{s}^{(t)}$ is generally considerred to encompass all the information needed for the agent to take an action (in our application, generate a query). The expected return J depends on π because $q^{(t)} \sim \pi(q^{(t)}|\mathbf{s}^{(t)}, \boldsymbol{\theta})$ drawn from the policy (distribution) π (*i.e.* $\pi(\boldsymbol{\theta}|C, I, C)$. The state $\mathbf{s}^{(t+1)} \sim P(\mathbf{s}^{(t+1)}|\mathbf{s}^{(t)}, q^{(t)})$ is generated by the seeker's environmental dynamics which are unknown. In policy gradient algorithms [45] such as the well-known REINFORCE [46], the gradient is estimated by samples from the policy $\pi(q|\mathbf{s}, \boldsymbol{\theta})$. Specifically, REINFORCE uses the following approximator of the policy gradient:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \approx \sum_{t=0}^{\infty} \nabla_{\boldsymbol{\theta}} \log \pi(q^{(t)}|\mathbf{s}^{(t)}, \boldsymbol{\theta}) r(\mathbf{s}^{(t)}, q^{(t)}),$$

This gradient is computed based on a single rollout trajectory, where $r(\mathbf{s}^{(t)}, q^{(t)}) = \sum_{i=0}^{\infty} \gamma^i r(\mathbf{s}^{(t+i)}, q^{(t+i)})$ is the

[1] The dependency on t is dropped for clarity.

accumulated return from time step t.

4. Information SEEKER and the RESPONDER

As discussed, our approach maintains a distribution of policies, π and updates it using the RESPONDER and EXECUTOR. In a nutshell, our approach takes the following steps for training an information seeking agent:

1. Conditioned on the history and context, the distribution of the query is:

$$\pi(q^{(t)}|\mathbf{s}^{(t)}, C, I) = \int \pi(q^{(t)}|\mathbf{s}^{(t)}, \boldsymbol{\theta})\pi(\boldsymbol{\theta}|C, I)d\boldsymbol{\theta},$$

where we can sample to generate a query, *i.e.*

$$q_i^{(t)} \sim \pi(q^{(t)}|\mathbf{s}^{(t)}, \boldsymbol{\theta}_i), \quad \boldsymbol{\theta}_i \sim \pi(\boldsymbol{\theta}|C, I) \quad \text{for} \quad i = 1, \ldots, n; \tag{1}$$

where n is the number of query samples simulating the alternative queries that could be made.

2. Our RESPONDER models belief over the potential answers and calculates the gain for each query $q_i^{(t)}$. Since ultimately we need to choose one query, we choose the one with highest gain and incorporate it into the reward for the RL (see Section 4.1);

3. The SEEKER models belief over the policy space rather than maintaining only a single policy (hence we can sample multiple parameters from its distribution in Eq. (1)). The posterior $\pi(\boldsymbol{\theta}|\{q^{(t)}\}_t^T, \{a^{(t)}\}_t^T, o, C, I)$ considering the outcome of executing the query (potentially at multiple rounds) and a prior is formulated as part of the RL framework. Here, $a^{(t)}$ is the correct answer obtained from the environment. For example in case of GuessWhat game, it is the answer obtained from the oracle. (see Section 4.2);

4. The SEEKER updates its belief over the distribution of the policies by incorporating the feedback from the environment. This update has to ensure the posterior for the parameters of the SEEKER remains valid (see Section 4.3).

4.1. Query Gain and the RESPONDER

In our approach the agent keeps a model (RESPONDER) of the environment to be able to predict what might most valuably be asked. The agent uses this model to imitate the behavior of the goal EXECUTOR and anticipate its potential response. Utilizing this model, the agent generates queries with answers that bring it closer to achieving its goal. In particular, we define the *gain* from state $\mathbf{s}^{(t)}$ is,

$$\mathcal{G}_{\boldsymbol{\omega}}(\mathbf{s}^{(t)}, q^{(t)}) = \mathbb{E}_a[u(p(a|\mathbf{s}^{(t)}, q^{(t)}, C, I; \boldsymbol{\omega}))] \tag{2}$$

where u is a scoring function for the answers and $\boldsymbol{\omega}$ is the set of parameters of the RESPONDER. Here, $p(a|\mathbf{s}^{(t)}, q^{(t)}, C, I; \boldsymbol{\omega})$ is the probability of the answer for a given query in the RESPONDER. This gains effectively evaluates the score of an answer. Particularly we find $\boldsymbol{\omega}$ such that the expected goal under this policy is maximized. Intuitively, the agent queries $q^{(t)}$ at time t only if it believes the answer $a^{(t)}$ it receives ultimately maximizes the gain in achieving its goal at state $\mathbf{s}^{(t)}$. For instance, in the GuessWhat game the RESPONDER takes in the history of the dialog and the current question and evaluates how good it is for achiveing the goal (*i.e.* guessing the correct object).

In order to integrate this measure into an RL framework, we use this gain in the reward. This reward is collected by choosing the best query according to its gain in Eq. (2), i.e. $\max_{q^{(t)}} \mathcal{G}_{\boldsymbol{\omega}}(\mathbf{s}^{(t)}, q^{(t)})$ (although in practice with probability ϵ we sample an alternative query from the SEEKER's distribution π to encourage exploration). Moreover, inspired by curiosity-driven and information maximizing exploration [16, 32], we incorporate this gain as an intrinsic motivation to consider the gain, i.e.

$$r^{\text{new}}(\mathbf{s}^{(t)}, q^{(t)}) = r(\mathbf{s}^{(t)}, q^{(t)}) + \eta\mathcal{G}_{\boldsymbol{\omega}}(\mathbf{s}^{(t)}, q^{(t)}) \tag{3}$$

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\pi(\mathbf{s},q|\boldsymbol{\theta})}[\sum_{t=0}^{\infty} \gamma^t r^{\text{new}}(\mathbf{s}^{(t)}, q^{(t)})], \tag{4}$$

for $\eta \geq 0$ that controls the intrinsic reward. In this new reward, an agent's anticipation of the answer is taken into account when updating the policy. When the seeker knows the answer and its gain is small, the parameters are not changed significantly. In other words, there is no need for further changes to the questions where the answer is known. On the other hand, when the agent anticipates a large gain from the answer and receives a large reward, the policy has to be adjusted by a larger change in the parameters. Similarly, if the agent expects a large gain and is not rewarded, there has to be significant update in the policy.

In addition, for each parameter of the SEEKER and each corresponding query $q^{(t)}$, we have a different gain. As such, when the variance of this gain $\mathbb{V}_{q^{(t)}}[\mathcal{G}(\mathbf{s}^{(t)}, q^{(t)})]$ is small, all the queries are expected to have similar answer and hence are almost the same.

The advantage of this approach is twofold: (1) it helps deal with sparse rewards and (2) if a query's response carries more information by providing better gain, we encourage its positive reinforcement. This allows the agent to learn to mimic the behavior of the goal executor and generalize to unseen cases.

4.2. Information SEEKER's Belief

As discussed in Eq. (1), each query is sampled from the *seeker's policy distribution*. Each sample of the parameter $\boldsymbol{\theta}$ gives rise to a different querying policy allowing us

to model policy distribution. This distribution allows for the agent to consider alternatives, or contemplates, various query policies to improve the overall dialog performance. As such, here we consider the policy parameter $\boldsymbol{\theta}$ as a random variable (leading to random policies that we can model their distribution) and seek a distribution to optimize the expected return. We incorporate a prior distribution π_0 over the policy parameter, for instance, for when we have no answer for query-response pairs or to incorporate prior domain knowledge of parameters. The posterior in the conventional definition is

$$\pi(\boldsymbol{\theta}|C^+, I) \propto \pi(o|\{q^{(t)}\}_t^T, \{a^{(t)}\}_t^T, C, I, \boldsymbol{\theta})\pi_0(\boldsymbol{\theta}).$$

where we denote $C^+ = C \cup \{\{q^{(t)}\}_t^T, \{a^{(t)}\}_t^T, o\}$ as an augmented context with the rollout of one seeker's round (for instance a dialog round in GuessWhat). Since we need to define an additional likelihood for the goal and even then this posterior is intractable (unless major approximations and simplifying assumptions are made), we alternatively utilize the RL framework in which this posterior is used for. Specifically, we formulate the problem to find the policy distribution π under which the expected cumulative reward is maximized with additional prior regularization:

$$\max_{\pi} \left\{ \mathbb{E}_{\pi(\boldsymbol{\theta}|C^+,I)}[J(\boldsymbol{\theta})] - \alpha \mathbf{KL}(\pi \| \pi_0) \right\}, \tag{5}$$

where $\mathbf{KL}(\pi\|\pi_0) = \mathbb{E}_{\pi}[\log \pi(\boldsymbol{\theta}|C^+, I) - \log \pi_0(\boldsymbol{\theta})]$. Effectively we seek a parameter distribution that gives rise to policies that maximize the expected reward while is close to the prior. It is easy to see if we use an uninformative prior such as a uniform distribution, the second KL term is simplified to the entropy of π. Then optimization in Eq. (5) becomes $\max_{\pi} \left\{ \mathbb{E}_{\pi(\boldsymbol{\theta}|C^+,I)}[J(\boldsymbol{\theta})] + \alpha \mathbf{H}(\pi) \right\}$ which explicitly encourages exploration in the parameter space. This exploration yields diverse policies that result in varied queries.

By taking the derivative of the objective function in Eq. (5) and setting it to zero, the optimal distribution of policy parameter $\boldsymbol{\theta}$ is obtained as

$$\pi(\boldsymbol{\theta}|C^+, I) \propto \exp\left(J(\boldsymbol{\theta})/\alpha\right) \pi_0(\boldsymbol{\theta}). \tag{6}$$

In this formulation, $\pi(\boldsymbol{\theta}|C^+, I)$ is the "posterior" of the parameters $\boldsymbol{\theta}$ in the conventional Bayesian approach. Then, $\exp(J(\boldsymbol{\theta})/\alpha)$ is effectively the "likelihood" function. The coefficient α is the parameter that controls the strength of exploration in the parameter space and how far the posterior is from the prior. As $\alpha \to 0$, samples drawn from $\pi(\boldsymbol{\theta}|C^+, I)$ will be concentrated on a single policy and lead to less diverse seekers.

Remember from Eq. (4) that the "likelihood" here considers the agent's anticipation of the answer. If its reward is higher, then a larger change to the parameter is needed to allow exploitation of new knowledge about the effect of the current policy on the goal.

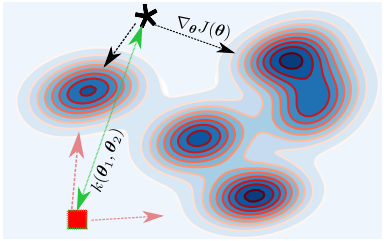

Figure 3: An illustration of the multi-modal distribution of the SEEKERs policies. Unlike conventional policy gradient methods that only explore nearest mode, our novel approach always use a number of initial points (i.e. the policy parameters) to explore multiple modes collaboratively in analogy of human contemplation of multiple strategies. We only show two initial points, a red rectangle and black asterisk, for the ease of visualization.

Similar ideas of entropy regularization has been investigated in other reinforcement learning methods [28, 38]. However, in our approach we use the regularization to obtain the posterior for the policy parameters in the information seeking framework where the gain from the RESPONDER refines the policy distribution.

4.3. SEEKER's Posterior Update

A conventional method to utilise the posterior in Eq. (6) is Markov Chain Monte Carlo (MCMC) sampling. However, MCMC methods are computationally expensive, suffer from slow convergence and have high-variance due to stochastic nature of estimating $J(\boldsymbol{\theta})$. Since estimating $J(\boldsymbol{\theta})$ by itself is a computationally demanding task and may vary significantly for each policy, we look for an efficient alternative. Thus, rather than $J(\boldsymbol{\theta})$, we use the gradient information $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ that points to the direction for seeker's policy change using the *Stein variational gradient descent* (SVGD) for Bayesian inference [25, 26, 2, 1]. SVGD is a nonparametric variational inference algorithm that leverages efficient deterministic dynamics to transport a set of particles $\{\boldsymbol{\theta}_i\}_{i=1}^n$ to approximate given target posterior distributions $\pi(\boldsymbol{\theta}|C^+, I)$. Unlike traditional variational inference methods, SVGD does not confine the approximation within a parametric families, which means the SEEKER's policy does not need to be approximated by another. Further, SVGD converges faster than MCMC due to the deterministic updates that efficiently leverage gradient information of the SEEKER's policy posterior. This inference is efficiently performed by iteratively updating multiple "particles" $\{\boldsymbol{\theta}_i\}$ as $\boldsymbol{\theta}_i = \boldsymbol{\theta}_i + \epsilon_\theta \psi^*(\boldsymbol{\theta}_i)$, where ϵ_θ is a step size and $\psi(\cdot)$ is a function in the unit ball of a reproducing kernel Hilbert space (RKHS). Here, ψ^* is chosen as the solution to minimizing KL divergence between the particles and the target distribution. It was shown that this function has a closed form empirical estimate [26]:

$$\hat{\psi}(\boldsymbol{\theta}_i) = \frac{1}{n} \sum_{j=1}^{n} [\nabla_{\boldsymbol{\theta}_j} \log \pi(\boldsymbol{\theta}_j|C^+, I) k(\boldsymbol{\theta}_j, \boldsymbol{\theta}_i) + \nabla_{\boldsymbol{\theta}_j} k(\boldsymbol{\theta}_j, \boldsymbol{\theta}_i)]. \tag{7}$$

Algorithm 1 Seeker

Input: Learning rate $\epsilon_\theta, \epsilon_\omega$, kernel $k(\boldsymbol{\theta}, \boldsymbol{\theta}')$, initial policy particles $\{\boldsymbol{\theta}_i\}$, context history C, image I.
for iteration $t = 0, 1, .., T$ **do**
  for particle $i = 1, \ldots, n$ **do**
    Sample $q \sim \pi(q|\mathbf{s}^{(t)}, C, I; \boldsymbol{\theta}_i)$
    Compute $\mathcal{G}(\mathbf{s}^{(t)}, q)$ from Eq. (2).
  end for
  Select $q^{(t)}$ with maximum gain
  $a^{(t)} = \arg\max_a p(a|\mathbf{s}^{(t)}, q^{(t)}, C, I; \boldsymbol{\omega})$
  $\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} + \epsilon_\omega \nabla_{\boldsymbol{\omega}} \log\Big(p(o|a^{(t)}, \mathbf{s}^{(t)}; \boldsymbol{\omega})\Big)$ ▷ score
  Compute $\nabla_{\boldsymbol{\theta}_i} J(\boldsymbol{\theta}_i)$ in Eq. (4). ▷ from Eq. (3)
  for particle $i = 0, 1, .., n$ **do**
    $J_{\text{new}}(\boldsymbol{\theta}_j) = \frac{1}{\alpha} J(\boldsymbol{\theta}_j) + \log \pi_0(\boldsymbol{\theta}_j)$
    $\Delta\boldsymbol{\theta}_i \leftarrow \frac{1}{n} \sum_{j=1}^{n} \Big[\nabla_{\boldsymbol{\theta}_j} J_{\text{new}}(\boldsymbol{\theta}_j) k(\boldsymbol{\theta}_j, \boldsymbol{\theta}_i) + \nabla_{\boldsymbol{\theta}_j} k(\boldsymbol{\theta}_j, \boldsymbol{\theta}_i)\Big]$
    $\boldsymbol{\theta}_i \leftarrow \boldsymbol{\theta}_i + \epsilon \Delta\boldsymbol{\theta}_i$ ▷ update the policy
  end for
end for

where k is the the positive definite kernel associated with the RKHS space. In this update rule $\hat{\psi}$, the first term involves the gradient $\nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}|C^+, I)$ which moves the seeker's policy particles $\boldsymbol{\theta}_i$ towards the high probability regions by sharing information across similar particles. The second term $\nabla_{\boldsymbol{\theta}_j} k(\boldsymbol{\theta}_j, \boldsymbol{\theta}_i)$ utilizes the curvature of the parameter space to push the particles away from each other, which leads to diversification of the seeker's policies.

An example of the landscape of the policies is shown in Fig. 3. Each initial sample from the policy distribution can move towards one of the modes of a highly multi-modal distribution. These moves are governed by the gradient of the policy that in our case consists of the agent's belief about the answer and its consequence once its response is known. In addition, kernel k controls the distance between the parameters to deter from collapsing to a single point in multi-modal distribution. It is intuitive from the figure that a better gradient from the rewards by incorporating the answers and considering the distribution of policies improves the performance of the seeker by guiding the parameter updates.

It is noteworthy to mention that even though we only receive the reward for one sample of $q^{(t)}$ taken from one particle (from Eq. 2), due to our formalization the posterior is adjusted for all particles allowing the feedback to be propagated.

It is important to note that diversification in the parameter space allows for an accurate modeling of a highly multi-modal policy space. Otherwise, the policy distribution collapses to a single point which is the same as the conventional maximum a posteriori (MAP) estimate. This MAP estimate only considers a single policy that in the highly complex task of vision-language is inadequate.

5. Experiments

To evaluate the performance of the SEEKER we conducted experiments on two different goal-oriented vision-and-language datasets: GuessWhat [13] and CLEVR [21]. The former is a visual dialog task while the later is a compositional visual question answering task. In both experiments we pre-train the networks using the supervised model and refine using reinforcement learning, as is common practice in this area [11, 13]. Policies are generated by sampling from the policy posterior $\boldsymbol{\theta} \sim \pi(\boldsymbol{\theta}|C^+, I)$ and generate the query with the highest gain measured by the RESPONDER. Our approach outperforms the previous state-of-art in both cases. Note that our approach is architecture neutral and as such we expect using better representations to even improve performance further.

5.1. GuessWhat

GuessWhat [13] is a classical goal-oriented visual dialog game. In each game, a random object in the scene is assigned to the answerer, but hidden from the questioner (our SEEKER). The questioner can ask a series of yes/no questions to locate the object. The list of objects is also hidden from the questioner during the question-answer rounds. Once the questioner has gathered enough information, the guesser (our EXECUTOR) can start to guess. If a guess is correct the game is successfully concluded. The extrinsic reward is "one" at the end of a dialog (i.e. series of question-answers) when the predicted object matches the true object the oracle chose.

Implementation Details In our model, the information seeker is a set of 10 recurrent neural networks (RNNs) that represent the particles from the likelihood in Eq. (6). We use LSTM [15] cells in these RNNs for which the parameters are updated according to Eq. (7) to simulate the posterior. The hidden representations of these LSTM networks (with size 1024) correspond to the state in the reward function. The image representation is obtained using VGG [39]. The concatenation of the image and history features are given to each particle in the SEEKER for question generation where each word is sampled conditioned on its previous word. We use $u(\cdot) = \exp(\cdot)$ to operate as the score function for computing gain in Eq. (2), and reward in Eq. (4).

We set $\eta = 0.1 \times \frac{\text{epoch}_{\max} - \text{epoch}}{\text{epoch}_{\max}}$ to encourage the policies to explore more in the initial stages. In addition, $\alpha = 0.001$. We use the median trick from [26] to compute the RBF-kernel's hyper-parameter which essentially ensures $\sum_j k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) \approx 1$.

Overall Results We compare two cases, labeled *New Object* and *New Image*. In the former the object sought is new, but the image has been seen previously. In the latter the image is also previously unseen. We report the prediction accuracy for the guessed objects. It is clear that the accuracies

Model	New Object	New Image
Supervised-S [13]	41.6	39.2
Supervised-G [13]	43.5	40.8
RL-S [40]	56.5	58.5
RL-G [40]	60.3	58.4
Tempered [47]	62.6	-
Tempered-Seq2Seq [47]	63.5	-
Tempered-MemoryNet [47]	68.3	-
Ours (no intrinsic reward)	63.3	-
Ours	64.2	62.1
Ours+MemoryNet (Single)	70.1	67.9
Ours+MemoryNet	**74.4**	**72.1**

Table 1: Accuracy in identifying the goal object in the Guess-What dataset (higher is better). The "S" indicator is for sampling for words method vs "G" which is greedy. Ours+MemoryNet is the method with modified RESPONDER that employs Memory network and Attention. Further, (Single) indicates training our method with a single particle.

are generally higher for the new objects as they are obtained from the already seen images.

The results are summarized in Table 1. As shown, using the conventional REINFORCE [40] by either sampling each word (RL-S) or greedily selecting one (RL-G) improves the performance compared to the supervised baseline significantly. Since our approach explore and exploits the space of policies for question generation better, it achieves better performance. Furthermore, this performance is improved when a better goal seeker or RESPONDER model is employed. Better RESPONDER leads to more realistic intrinsic rewards that corresponds to true gains and guide the policy distribution to a better posterior. For instance, employing a Memory network [43] within the RESPONDER improves its performance that in turn is reflected in the quality of the questions and consequently agent's ability to achieve goals more accurately. Note that even in this case the single particle experiment has improved since the rewards are more accurate to evaluate the question-answer relations.

5.2. CLEVR

CLEVR [21] is a synthetically generated dataset containing 700K (image, question, answer, program) tuples. Images are 3D-rendered objects of various shapes, materials, colors, and sizes. Questions are compositional in nature and range from counting questions to comparison questions and can be 40+ words long. An answers is a word from a set of 28 choices. For each image and question, a program consists of step-by-step instructions, on how to answer the question. During the test, the programs are not given, which need to be generated conditioned on the input question. The extrinsic reward is "one" when the generated program yields the correct answer.

Model	Overall	Count	Compare Numbers	Exist	Query Attribute	Compare Attribute
NMN [7]	72.1	52.5	72.7	79.3	79.0	78.0
N2NMN [17]	88.8	68.5	84.9	85.7	90.0	88.8
Human [21]	92.6	86.7	86.4	96.6	95.0	96.0
LSTM+RN [37]	95.5	90.1	93.6	97.8	97.1	97.9
PG+EE (9k) [20]	88.6	79.7	79.7	89.7	92.6	96.0
PG+EE (18k) [20]	95.4	90.1	96.2	95.3	97.3	97.9
PG+EE (700k) [20]	96.9	92.7	98.6	97.1	98.1	98.9
FiLM [33]	97.6	94.5	93.8	99.2	99.2	99.0
DDRprog [42]	98.3	96.5	98.4	98.8	99.1	99.0
MAC [18]	98.9	97.2	99.4	99.5	99.3	99.5
TbD-net [31]	98.7	96.8	99.1	98.9	99.4	99.2
TbD-net++ [31]	99.1	97.6	99.4	99.2	99.5	99.6
Ours+G+entropy (9k)	91.4	86.4	93.6	89.8	93.2	96.2
Ours+G+entropy (18k)	95.6	93.3	96.8	95.4	97.8	98.1
Ours+G+entropy (700k)	97.4	96.8	98.1	98.2	96.2	98.1
Ours+D+entropy (9k)	94.7	92.2	95.6	93.2	95.1	97.7
Ours+D+entropy (18k)	96.6	94.6	96.1	95.6	98.1	98.6
Ours+D+entropy (700k)	98.3	98.1	99.1	97.1	98.6	98.8
Ours+G+exp (9k)	91.8	87.5	93.7	90.2	93.1	96.5
Ours+G+exp (18k)	96.3	93.3	96.8	95.4	97.8	98.1
Ours+G+exp (700k)	98.0	96.2	98.6	98.0	98.0	99.0
Ours+D+exp (9k)	95.2	91.5	96.7	93.8	95.7	98.7
Ours+D+exp (18k)	97.1	94.5	98.2	96.1	98.3	98.6
Ours+D+exp (700k)	98.9	97.8	99.2	98.9	99.5	99.3
Ours+D+exp++ (700k)	**99.2**	**97.8**	**99.5**	**99.4**	**99.6**	**99.6**

Table 2: Performance comparison of state-of-the-art models on the CLEVR dataset. "Ours+G+entropy" is our seeker when used with the generic architecture and entropic gain; "Ours+D+entropy" is the same except for using designed architecture. Similarly, "Ours+G+exp" is generic architecture with u_{exp}; and, "Ours+D+exp" is its designed counterpart. We achieve state of the art performance, especially using smaller ground-truth programs. The '++' indicator shows a model was trained using higher-resolution 28×28 feature maps rather than 14×14.

Implementation Details We follow the experimental setup of [20, 21] in which uses a ResNet [14] to encode the given images and a standard LSTM [15] to generate programs in the context encoder. We use 10 particles in Algorithm 1 to model the policy distribution using samples from the pre-trained model with added noise so that they correspond to different initial policies. For more efficient implementation, we use two sets of shared parameters for the encoder in the underlying Seq2Seq model and use independent parameters for the LSTM decoder. This parameter sharing also ensures there are common latent representations that particles learn. We use our information SEEKER model in Section 4 to generate samples or programs for each question and consider the consequence of that program using the RESPONDER internally to choose one. Once a program is generated, it is then executed by the goal EXECUTOR to obtain the feedback and compute the corresponding rewards. The computed reward is then used to update the policy distributions as discussed. We use the Adam optimization method with learning rate set to 10^{-5} to update

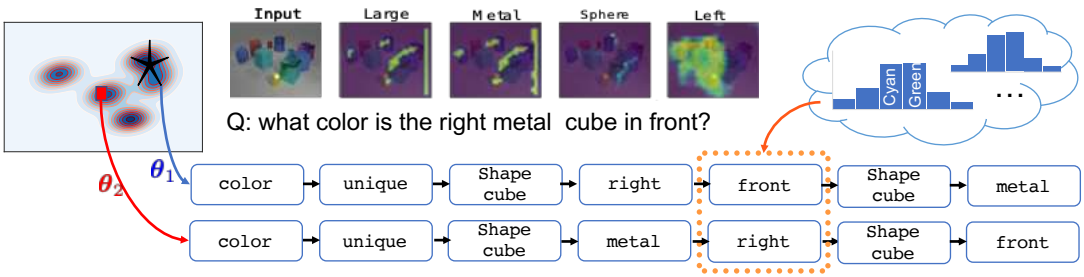

Figure 4: An example of a question and the programs generated using samples from the posterior in CLEVR. Samples from the policy distribution take the input image and the question and generate its corresponding programs. As observed, these two samples produce different program sequences which enable to explore multiple distributions over the goal (final answer) shown on top in the cloud. Expected score of each question $\mathcal{G}_{\omega}(\mathbf{s}^{(t)}, q^{(t)})$ gives us an indication of which one is better to ask.

both the SEEKER and the RESPONDER's parameters. The testing procedure thus takes an image and question pair, produces a program, then the goal executor produces an answer. The goal EXECUTOR then evaluates the quality of the generated program. We set $\alpha = 0.01$ and η similar to the GuessWhat experiment.

Overall Results For the RESPONDER and the EXECUTOR, we consider two alternative baselines: (G)eneric similar to [20] where each module follows a generic architecture; and, (D)esigned similar to [31] where each module is specifically designed based on the desired operation.

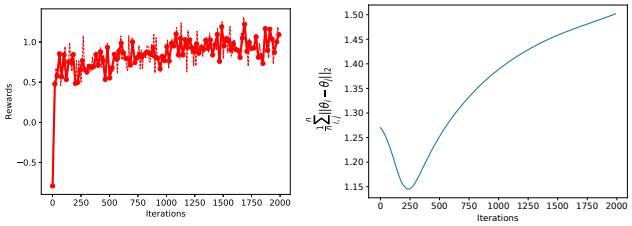

Figure 5: Average reward for the agent at each iteration; and, average distance between particles in the posterior for CLEVR.

We report the accuracy of the goal executor. Since the later case provides a better representation on each module, we expect it to perform better. Further, we use two functions $u_{\text{entropy}}(\cdot) = \log(\cdot)$ (corresponding to the information-theoretic notion of gain in the expectation) and $u_{\text{exp}}(\cdot) = \exp(\cdot)$ to operate on the output of the RESPONDER's score to compute the gain and ultimately the new reward in Eq. (2) and (4). The results in Tab. 2 show that our approach outperforms the baselines almost to the maximum extent possible. In particular, our approach almost achieves the same performance as that of [20] with half the programs used for training with the same neural architecture. Moreover, the choice of u affects the policies found, for instance using u_{entropy} generally leads to outperforming in the "count" function. Thus, since u_{exp} has a smaller range and is smoother, it provides a more uniform penalty for mistakes in all modules during training leading to generally better performance. As shown in Figure 4, each sample from the policy generates a different program. In addition, we are able to utilize the attention mechanism in the model to *reason* about where in the image the information seeker focuses.

Fig. 5 plots the average reward at each iteration, and the average distance between the particles in the policies. If the problem was indeed unimodal (as conventional methods assume), all the particles would collapse to a single point indicated by a zero average distance. However, as is observed, while the distance between the particles decreases in early stages, they soon increase indicating convergence to independent modes. Our context encoder, unlike [23], is a pixel level model that does not extract objects explicitly from the given image. To be fair, we only consider methods that are directly comparable.

6. Conclusion

The ability to identify the information needed to support a conclusion, and the actions required to obtain it, is a critical capability if agents are to move beyond carrying out low-level prescribed tasks towards achieving flexible high semantic level goals. The method we describe is capable of reasoning about the information it holds, and the information it will need to achieve its goal, in order to identify the action that will best enable it to fill the gap between the two. Our approach thus actively seeks the information it needs to achieve its goal on the basis of a model of the uncertainty in its own understanding.

If we are to enable agents that actively work towards a high-level goal the capability our approach demonstrates will be critical. In particular, agents need to be able to consider alternative policies for achieving a goal and their corresponding uncertainty, evaluate the outcome of executing those policies and the information it gains.

Acknowledgment: This material is based on research sponsored by Air Force Research Laboratory and DARPA under agreement number FA8750-19-2-0501. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

References

[1] Ehsan Abbasnejad, Anthony R. Dick, and Anton van den Hengel. Infinite variational autoencoder for semi-supervised learning. In *CVPR*, pages 781–790. IEEE Computer Society, 2017.

[2] Ehsan Abbasnejad, Justin Domke, and Scott Sanner. Loss-calibrated monte carlo action selection. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[3] Ehsan Abbasnejad, Qinfeng Shi, Anton van den Hengel, and Lingqiao Liu. A generative adversarial density estimator. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[4] Ehsan Abbasnejad, Damien Teney, Amin Parvaneh, Qinfeng Shi, Anton van den Hengel, and Lingqiao Liu. Counterfactual vision and language learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[5] Ehsan Abbasnejad, Qi Wu, Qinfeng Shi, and Anton van den Hengel. What's to know? uncertainty as a guide to asking goal-oriented questions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[6] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, 2018.

[7] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. *CVPR*, pages 39–48, 2016.

[8] Nabiha Asghar, Pascal Poupart, Jiang Xin, and Hang Li. Online sequence-to-sequence reinforcement learning for open-domain conversational agents. *arXiv preprint arXiv:1612.03929*, 2016.

[9] Valérian Chambon, Nura Sidarus, and Patrick Haggard. From action intentions to action effects: how does the sense of agency come about? *Frontiers in Human Neuroscience*, 8:320, 2014.

[10] Anton van den Hengel Damien Teney, Ehsan Abbasnejad. Unshuffling data for improved generalization. *arXiv preprint arXiv:2002.11894*, 2020.

[11] A. Das, S. Kottur, J. M. F. Moura, S. Lee, and D. Batra. Learning cooperative visual dialog agents with deep reinforcement learning. In *ICCV*, pages 2970–2979, 2017.

[12] Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José MF Moura, Devi Parikh, and Dhruv Batra. Visual dialog. In *ICCV*, 2017.

[13] Harm de Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron C. Courville. Guesswhat?! visual object discovery through multi-modal dialogue. In *CVPR*, 2017.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[15] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. 9:1735–80, 12 1997.

[16] Rein Houthooft, Xi Chen, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1109–1117. Curran Associates, Inc., 2016.

[17] Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. Learning to reason: End-to-end module networks for visual question answering. *arXiv preprint arXiv:1704.05526*, 2017.

[18] Drew Arad Hudson and Christopher D. Manning. Compositional attention networks for machine reasoning. In *ICLR*, 2018.

[19] Laurent Itti and Pierre F. Baldi. Bayesian surprise attracts human attention. In Y. Weiss, B. Schölkopf, and J. C. Platt, editors, *NIPS*, pages 547–554. 2006.

[20] J. Johnson, B. Hariharan, L. v. Maaten, J. Hoffman, L. Fei-Fei, C. L. Zitnick, and R. Girshick. Inferring and executing programs for visual reasoning. In *ICCV*, volume 00, pages 3008–3017, Oct. 2018.

[21] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, 2017.

[22] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *J. Arti. Intell. Research*, 4:237–285, 1996.

[23] Sang-Woo Lee, Yu-Jung Heo, and Byoung-Tak Zhang. Answerer in questioner's mind for goal-oriented visual dialogue. *CoRR*, abs/1802.03881, 2018.

[24] Sang-Woo Lee, Yu-Jung Heo, and Byoung-Tak Zhang. Answerer in questioner's mind for goal-oriented visual dialogue. *arXiv preprint arXiv:1802.03881*, 2018.

[25] Qiang Liu. Stein variational gradient descent as gradient flow. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *NIPS*, pages 3115–3123. Curran Associates, Inc., 2017.

[26] Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2378–2386. Curran Associates, Inc., 2016.

[27] Siqi Liu, Zhenhai Zhu, Ning Ye, Sergio Guadarrama, and Kevin Murphy. Optimization of image description metrics using policy gradient methods. *arXiv preprint arXiv:1612.00370*, 2016.

[28] Yang Liu, Prajit Ramachandran, Qiang Liu, and Jian Peng. Stein variational policy gradient. *arXiv preprint arXiv:1704.02399*, 2017.

[29] Jiasen Lu, Anitha Kannan, Jianwei Yang, Devi Parikh, and Dhruv Batra. Best of both worlds: Transferring knowledge from discriminative learning to a generative visual dialog model. *arXiv preprint arXiv:1706.01554*, 2017.

[30] Anthony Manchin, Ehsan Abbasnejad, and Anton van den Hengel. Reinforcement learning with attention that works: A self-supervised approach. In Tom Gedeon, Kok Wai Wong, and Minho Lee, editors, *Neural Information Processing*, pages 223–230, Cham, 2019. Springer International Publishing.

[31] D. Mascharka, P. Tran, R. Soklaski, and A. Majumdar. Transparency by Design: Closing the Gap Between Performance and Interpretability in Visual Reasoning. *ArXiv e-prints*, March 2018.

[32] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*, 2017.

[33] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018.

[34] Olivier Pietquin, Matthieu Geist, Senthilkumar Chandramohan, and Hervé Frezza-Buet. Sample-efficient batch reinforcement learning for dialogue management optimization. *TSLP*, 7(3):7, 2011.

[35] Zhou Ren, Xiaoyu Wang, Ning Zhang, Xutao Lv, and Li-Jia Li. Deep reinforcement learning-based image captioning with embedding reward. *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2017.

[36] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. *arXiv preprint arXiv:1612.00563*, 2016.

[37] Adam Santoro, David Raposo, David G. T. Barrett, Mateusz Malinowski, Razvan Pascanu, Peter W. Battaglia, and Timothy P. Lillicrap. A simple neural network module for relational reasoning. In *NIPS*, 2017.

[38] John Schulman, Sergey Levine, Philipp Moritz, Michael Jordan, and Pieter Abbeel. Trust region policy optimization. ICML'15, pages 1889–1897, 2015.

[39] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[40] Florian Strub, Harm De Vries, Jeremie Mary, Bilal Piot, Aaron Courville, and Olivier Pietquin. End-to-end optimization of goal-driven and visually grounded dialogue systems. *arXiv preprint arXiv:1703.05423*, 2017.

[41] Pei-Hao Su, Milica Gasic, Nikola Mrksic, Lina Rojas-Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve Young. Continuously learning neural dialogue management. *arXiv preprint arXiv:1606.02689*, 2016.

[42] Joseph Suarez, Justin Johnson, and Fei-Fei Li. DDRprog: A CLEVR differentiable dynamic reasoning programmer, 2018.

[43] Sainbayar Sukhbaatar, arthur szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *NIPS*. 2015.

[44] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[45] Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. NIPS'99, Cambridge, MA, USA, 1999. MIT Press.

[46] Ronald Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *ML*, 8, May 1992.

[47] Rui Zhao and Volker Tresp. Improving goal-oriented visual dialog agents via advanced recurrent nets with tempered policy gradient. In *IJCAI*, 2018.