Active Tree Search

Robert Lieck **Marc Toussaint**
Machine Learning and Robotics Lab
University of Stuttgart
prename.surname@ipvs.uni-stuttgart.de

Abstract

Monte-Carlo tree search is based on contiguous rollouts. Since not all samples within a rollout necessarily provide relevant information, contiguous rollouts may be wasteful as compared to sampling selected transitions. In this paper, we describe an active learning approach that can be used to select single transition within the tree for sampling with the goal of maximizing information gain. We show that this approach can be used to enhance purely rollout-based MCTS by actively sampling single transitions in addition to performing contiguous rollouts. We demonstrate that our method outperforms classical MCTS in a prototypical domain and discuss the interplay of the active learning component with the classical rollout-based sampling strategy.

Introduction

Monte-Carlo tree search (MCTS) has become a standard planning method that has been successfully applied in various domains, ranging from computer Go to large-scale POMDPs (Silver et al. 2016; Browne et al. 2012). An appealing property of MCTS is that it is sufficient to be able to simulate transitions in the environment. Planning is then performed by simulating contiguous rollouts from the root node. When collecting new samples, an important concern is to improve the estimates of the transition and reward function. For this concern contiguous rollouts may be wasteful because not all samples along a rollout necessarily provide relevant information. For instance, transitions that are (close to) deterministic do not require as many samples for a good estimate as transitions with high stochasticity. We therefore suggest an alternative to rollout-based sampling by formulating an active learning measure (Settles 2009) that can be used to select single transitions for sampling *anywhere in the tree*. In order to make computation of the involved expectations tractable we derive an efficiently approximation based on reverse accumulation of the objective gradient through the tree. In a prototypical domain, we demonstrate that combining our active learning measure with rollout-based sampling outperforms classical MCTS. We also discuss the interplay between active samples and rollout-based samples providing deeper insight into the different concerns to be addressed and giving directions for further research.

Our main contributions are as follows

- We formulate an active learning measure for selecting single transition to be sampled.
- We derive an efficient approximation of our measure for practical application.
- Provide an enhanced MCTS algorithm by combining our active learning measure with rollout-based samples.
- We empirically show that our enhanced method outperforms classical MCTS in a prototypical domain.
- We discuss the characteristics, possible shortcomings, and possible extensions of our method.

In the remainder of this paper we will first discuss related work on MCTS and active learning, then present our active learning measure and show how to approximate it efficiently, and finally present our empirical evaluations and discuss the characteristics of our method.

Related Work

Monte-Carlo Tree Search

Monte-Carlo tree search (MCTS, Browne et al. 2012) comes in a number of flavors that mainly differ in three respects (Keller and Helmert 2013): (1) the *tree-policy* that is used for selecting actions (2) the *value heuristic* that is used for initializing leaf nodes and (3) the *backup method* that is used for propagating information back to the root node.

In this work we focus on sampling transitions within the tree, which is the task of the tree-policy in conventional MCTS. The tree-policy has to balance *exploration* and *exploitation*. That is, it has to choose actions that help improving the estimates of the action values (exploration) but it also has to choose actions with a high value in order to focus subsequent sampling and expansion of the tree on relevant regions of the state space (exploitation). These two concerns are somewhat conflicting and the attempt to address them separately has led to alternative scheduling schemes for the tree-policy (Feldman and Domshlak 2012).

However, to our knowledge, there is no work on departing from a rollout-based scheme for sampling transitions, which is exactly what we suggest in this paper. While we still use a tree-policy to select leaf nodes for expansion we additionally sample single transitions within the tree with the goal of maximizing information gain.

In a prototypical domain we demonstrate that this combined method outperforms purely rollout-based MCTS

Active Learning

The goal of active learning (Settles 2009) generally is to select samples optimally for learning a property of interest. For MCTS the multi-armed bandit problem (MAB, Berry and Fristedt 1985) is of particular interest as most research on improving the tree-policy is based on MABs. To transfer results from MABs to MCTS, action selection in each decision node is treated as a separate MAB with non-stationary reward distribution. The overall problem of sampling transitions within the tree is thus split up into a series of simpler problems. The approach we suggest in this paper is different in that we do not break down the problem into a series of MABs but instead formulate the problem of choosing a new transition to be sampled *anywhere in the tree* as a single active learning problem.

A common objective for active learning, especially when formulated in the framework of *optimal experimental design* (Chaloner and Verdinelli 1995), is to minimize the uncertainty of the distribution of interest as measured by the entropy or the variance. Our objective in this work is to minimize the state-value variance at the root node by sampling transitions that maximize its expected change. A major contribution of this paper is an efficient approximation of this objective by propagating its gradient through the tree.

Active Tree Search

We will first formally state the problem of *sampling-based planning* and then present our active learning approach to solve it.

In sampling-based planning the planner can repeatedly use a black-box simulator for sampling transitions, that is, query state-action pairs $(s, a) \in \mathcal{S} \times \mathcal{A}$ and observe the resulting state and reward $(s', \rho) \in \mathcal{S} \times \mathbb{R}$ in response. The value Q of action a in state s under policy π is defined as

$$Q^{(\pi)}_{(s,a)} = \sum_{s' \in \mathcal{S}} p_{(s'|s,a)} \left(r_{(s,a,s')} + \gamma \sum_{a' \in \mathcal{A}} \pi_{(a'|s')} \, Q_{(s',a')} \right) \quad (1)$$

with

$$p_{(s'|s,a)} : \mathcal{S} \times \mathcal{A} \rightsquigarrow \mathcal{S} \qquad \text{(transition function)} \quad (2)$$

$$r_{(s,a,s')} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R} \qquad \text{(expected reward)} \quad (3)$$

$$\pi_{(a'|s')} : \mathcal{S} \rightsquigarrow \mathcal{A} \qquad \text{(policy)} \quad (4)$$

$$\gamma \in [0, 1] \qquad \text{(discount factor)} , \quad (5)$$

where $\cdot \rightsquigarrow \cdot$ denotes a stochastic mapping. The action-value Q corresponds to the *expected discounted return* an agent is going to receive when following policy π (see e.g. Sutton and Barto 1998). To the planner, the transition function, p, and the expected reward, r, are unknown and only available via the black-box simulator by sampling concrete transitions. The discount factor, γ, is fixed and known. The objective for the planner is to identify the optimal action

$$a^* = \text{argmax}_a Q^{(\pi^*)}_{(s_0,a)} \quad (6)$$

for the current state s_0, where π^* is the optimal policy that deterministically chooses the optimal action in any state. The basic idea of MCTS is to estimate $Q^{(\pi^*)}_{(s_0,a)}$ by repeatedly sampling trajectories that *rollout* the recursive structure of (1) starting at s_0.

In contrast to rollout-based MCTS, the idea of *active tree search*, suggested in this paper, is to sample single transitions in a way that most rapidly decreases the uncertainty of the estimates of $Q^{(\pi^*)}_{(s_0,a)}$. To this end we will (a) quantify the uncertainty and (b) compute the expected change of uncertainty for all possible state-action queries in order to choose the one with highest information gain.

For solving the first task we extend our backup function such that it propagates not only the value but also its variance back to the root node while remaining fully differentiable. If the value is normal distributed, the variance is a monotonic function of the entropy, the most common information theoretic measure of uncertainty. While not being crucial for our method, the normality assumption is supported by the central limit theorem (the value is a sum of random variables – albeit not strictly independent ones) and the principle of maximum entropy (for fixed variance). For the second problem of computing the expected change of uncertainty, a brute force approach seems intractable because the expectations cannot be computed in closed form and instead we would need to perform dynamic programming updates throughout the whole tree for all possible state-action-outcome combinations. We therefore approximate the objective by first computing its gradient with respect to all nodes, which can be done by a single reverse accumulation pass through the tree, and then multiplying it with expected changes that we compute locally for each node.

The next two sections describe our approach in detail. For better readability we will drop the superscript denoting the policy ($Q^{(\pi)} \to Q$) and the explicit indication of the spaces in summations in what follows, as both is clear from the context.

Propagating the Variance

In the action value $Q_{(s,a)}$ defined in (1) we interpret Q, p, and r as random variables and denote their mean and variance by $\widehat{\cdot}$ and $\widetilde{\cdot}$, respectively. The mean and variance of p and r can be estimated directly from the sampled transitions. The mean and variance of Q are (see Appendix for the full derivation)

$$\widehat{Q}_{(s,a)} = \sum_{s'} \widehat{p}_{(s'|s,a)} \left(\widehat{r}_{(s,a,s')} + \gamma \sum_{a'} \pi_{(a'|s')} \, \widehat{Q}_{(s',a')} \right) \quad (7)$$

$$\widetilde{Q}_{(s,a)} = \sum_{s'} \left(\widehat{p}^2_{(s'|s,a)} + \widetilde{p}_{(s'|s,a)}\right)\left[\widetilde{r}_{(s,a,s')} + \gamma^2 \sum_{a'} \pi^2_{(a'|s')}\, \widetilde{Q}_{(s',a')}\right] + \dots$$
$$\dots + \sum_{s',s''} \widetilde{p}_{(s'/s''|s,a)}\left[\widehat{r}_{(s,a,s')} + \gamma \sum_{a'} \pi_{(a'|s')} \widehat{Q}_{(s',a')}\right]\left[\widehat{r}_{(s,a,s'')} + \gamma \sum_{a''} \pi_{(a''|s'')} \widehat{Q}_{(s'',a'')}\right], \tag{8}$$

where we assumed independence of variables belonging to different states. $\widetilde{Q}$ quantifies the uncertainty *per action*. To quantify the overall uncertainty at the root node we therefore use the state-value V of state s, defined as

$$V_s = \sum_a \pi_{(a|s)} Q_{(s,a)}\,, \tag{9}$$

and compute its variance (cf. (31) in the Appendix)

$$\widetilde{V}_s = \sum_a \pi^2_{(a|s)} \widetilde{Q}_{(s,a)}\,. \tag{10}$$

Computing Expectations

Let $\mathcal{O}(\boldsymbol{\xi})$ be the objective that we want to minimize, which depends on a set of variables $\boldsymbol{\xi}$, D be a set of known samples determining the values of $\boldsymbol{\xi}$ as $\boldsymbol{\xi}[D]$, and (x, y) be a new query-outcome sample where x is to be chosen such that the expected value of $\mathcal{O}$ becomes minimal. Further let $\boldsymbol{\xi}_x \subseteq \boldsymbol{\xi}$ be the subset of variables whose values may change when querying x. The optimal query x^* then is

$$x^* = \operatorname*{argmin}_x \mathbb{E}\big[\!\big[\mathcal{O}\big(\boldsymbol{\xi}[D, x, y]\big)\big]\!\big]_{y|x} \tag{11}$$
$$= \operatorname*{argmin}_x \mathbb{E}\big[\!\big[\mathcal{O}\big(\boldsymbol{\xi}[D, x, y]\big) - \mathcal{O}\big(\boldsymbol{\xi}[D]\big)\big]\!\big]_{y|x} \tag{12}$$
$$\approx \operatorname*{argmin}_x \mathbb{E}\Big[\!\Big[\frac{\partial \mathcal{O}\big(\boldsymbol{\xi}[D]\big)}{\partial \boldsymbol{\xi}_x}\big(\boldsymbol{\xi}_x[D, x, y] - \boldsymbol{\xi}_x[D]\big)\Big]\!\Big]_{y|x} \tag{13}$$
$$= \operatorname*{argmin}_x \frac{\partial \mathcal{O}\big(\boldsymbol{\xi}[D]\big)}{\partial \boldsymbol{\xi}_x} \mathbb{E}\big[\!\big[\boldsymbol{\xi}_x[D, x, y]\big]\!\big]_{y|x}\,, \tag{14}$$

where in (13) we assumed the change of $\mathcal{O}$ to be approximately linear in the variables $\boldsymbol{\xi}_x$ affected by query x.

In the case of active tree search, our objective is the variance of the state value $\mathcal{O} \equiv \widetilde{V}_s$ at the root node; the independent variables are $\boldsymbol{\xi} \equiv \{\widehat{r}_{(s,a,s')}, \widetilde{r}_{(s,a,s')}, \widehat{p}_{(s'|a,s)}, \widetilde{p}_{(s'|a,s)}\}$ for all possible transitions within the tree; the query $x \equiv (s, a)$ is a state-action pair; and the outcome $y \equiv (s', \rho)$ is a state-reward pair.

Under this approximation the problem of choosing a sample that minimizes the variance at the root node decomposes into (a) computing the gradient of the variance and (b) computing expected changes of the variables $\boldsymbol{\xi}$.

The gradient of the variance can be computed in the same order of complexity as the variance itself by performing reverse accumulation through the tree, starting at the root node (Bartholomew-Biggs et al. 2000). In our implementation, which will be released along with this paper, we employ Theano (Theano Development Team 2016) for computing the partial derivatives of (7) and (8) and then perform reverse accumulation throughout the tree for computing the gradient.

For computing the expected changes, note that sampling a transition from a specific node in the tree only affects variables at that node. Specifically, when sampling (s, a) all changes are accumulated into $\widehat{Q}_{(s,a)}$ and $\widetilde{Q}_{(s,a)}$ at that node. The corresponding expectations can be computed as

$$\mathbb{E}\big[\!\big[\widehat{Q}_{(s,a)}\big]\!\big] = \sum_{s'} \widehat{p}_{(s'|s,a)}\, \widehat{Q}^*_{(s,a)} \tag{15}$$

$$\mathbb{E}\big[\!\big[\widetilde{Q}_{(s,a)}\big]\!\big] = \sum_{s^*} \widehat{p}_{(s^*|s,a)} \Big\{ \widetilde{Q}^*_{(s,a)} - \frac{1}{(n+1)^3}\Big[\cdots \tag{16}$$
$$\cdots (n^2 + 3n + 1)\, \widehat{p}^{*2}_{(s^*|s,a)} + (2n+1)\, \widetilde{p}^*_{(s^*|s,a)}\Big] \widetilde{r}_{(s,a,s^*)} \Big\}$$

where the starred variables $\widehat{p}^*$, $\widetilde{p}^*$, $\widehat{Q}^*$, $\widetilde{Q}^*$ are computed by temporally adding the corresponding transition $(s, a) \to s^*$ (ignoring the reward) and updating the transition probabilities $\widehat{p}$ and $\widetilde{p}$ accordingly. The second term in (16) accounts for changes of $\widetilde{r}_{(s,a,s')}$ and $\widehat{r}_{(s,a,s')}$. We defer the full derivation including more details to the Appendix.

As the policy we use a soft-max of the upper bound

$$\pi_{(a|s)} \propto \exp \frac{1}{\tau}\Big[\widehat{Q}_{(s,a)} + C\Big(\sqrt{\widetilde{Q}_{(s,a)} + \epsilon^2} - \epsilon\Big)\Big]\,, \tag{17}$$

where the temperature τ regulates the greediness, C determines the amount of exploration/optimism, and the small constant ϵ ensures differentiability for zero variance.

Extensions and Challenges

The framework of *active tree search* presented in this paper provides a number of interesting options for further research.

- The method of automatic differentiation is not restricted to first order derivatives so that one might attempt to relax the linearity assumption in Eqs. (11–14) and go to higher order expansions of the objective.
- The effect of structural changes when sampling new transitions may be included in computing the expectations and modeled using a Dirichlet process.
- The employed priors for the transition probabilities and expected rewards have a crucial influence, especially in the early planning phase, since they determine the independent variable's values in case of little or no data. We conjecture that choosing priors more carefully in a domain specific way or inferring them from data may substantially improve the performance.
- In this paper we are concerned with sampling informative transition within the tree. Other relevant aspects that could be included into an active learning objective include selecting a leaf node for expansion and performing rollouts for value initialization.

- Our choice of the soft-max policy is by no means compulsive and it would be interesting to explore the effect of other policies.
- The same is true for our choice of the variance as objective function $\mathcal{O}$.
- We estimate the transition probabilities and the expected reward for each transition separately. Also we can compute the expectations of $\widehat{Q}_{(s,a)}$ and $\widetilde{Q}_{(s,a)}$ analytically. However, due to our factorization into gradient and expectations Monte-Carlo methods become applicable, which allow the estimation of more complex or correlated expectations.
- Employing our active learning objective saves on the number of required samples at the cost of additional computations for the gradient and the expected changes. In practice this is only useful if the sampling costs are large compared to the costs of computing our objective, for instance, if sampling involves a complex physics simulation of the environment.

Experiments

Our empirical evaluation aims at investigating multiple aspects. (1) we designed a prototypical environment (as described below) where we expect active sampling to provide an advantage over purely rollout-based sampling. (2) recall that our active learning strategy only selects samples *within the tree* while expanding the tree and performing rollouts for initializing leaf nodes are important aspects in MCTS, too. This is especially the case in the initial planning phase. We thus want to explore to what extend our method impairs performance at that stage. (3) we evaluate how efficiently our method optimizes the chosen objective and compare its performance to rollout-based sampling. (4) we are interested in how far our objective of minimizing the variance correlates with the ability to actually choose the best action. We discuss these points below, after describing the employed environment and some technical details.

Our *Highway* environment consists of n lanes of depth T. At each time step the agent moves forward one step and if it is at one of the predefined switch points it may additionally change to a neighboring lane. At switch points, with probability α the agent ends up in an arbitrary lane (left, right, or same) irrespective of its action and with probability $1-\alpha$ it ends up in the lane corresponding to its action. The agent starts in the left-most lane and the only reward it receives is when reaching a terminal state after T steps: When ending in the right-most lane it receives a reward uniformly distributed in $[0.9, 1]$ otherwise it receives a reward uniformly distributed in $[0, 0.1]$. The *Highway* instance we performed our experiments on had $n = 4$ lanes, a depth of $T = 21$, a randomness of $\alpha = 0.5$, and switch points at times $\{0, 10, 20\}$.

We employed a mixing strategy where the ratio of active samples and rollout samples is kept constant. That is, after each rollout the strategy *active*$[\chi]$ continues drawing active samples until they make up an χ-fraction of all samples (e.g. 10% of all samples for $\chi = 0.1$). We compare *active*$[0.2]$, *active*$[0.1]$, *active*$[0.01]$, and *active*$[0]$, where the last one corresponds to purely rollout-based MCTS. For the backups we used a discount of $\gamma = 1$. In the policy (17) we used $(\tau, C, \epsilon) = (0.5, 1, 10^{-3})$ for active samples and $(\tau, C, \epsilon) = (10^{-2}, 0, 10^{-3})$ for rollout samples, where the latter practically corresponds to the greedy policy. States without a proper value initialization (i.e. states only ever reached by an active sample) were excluded from computations for rollout samples. As the tree-policy for rollouts we used UCB1 (Auer, Cesa-Bianchi, and Fischer 2002; Kocsis and Szepesvri 2006)

$$a^* = \operatorname*{argmax}_a \widehat{Q}_{(s,a)} + 2\, C_p \sqrt{\frac{2 \log n_s}{n_{(s,a)}}} \tag{18}$$

with $C_p = \sqrt{2}$. Note that by using UCB1 as tree-policy in conjunction with the greedy policy for backups (see above), *active*$[0]$ corresponds to the classical UCT algorithm (Kocsis and Szepesvri 2006) with full Bellman updates (Keller and Helmert 2013). Value initialization was done by continuing the rollout with uniform policy until reaching a terminal state. For active samples, we used the *absolute value* of the expected change of the objective as scoring function, which counteracts the risk of getting stuck in local optima (Kulick, Lieck, and Toussaint 2016). In case of equal scores, the least-sampled transition was chosen (with random tie-breaking).

In Figs. 1–3 we show the results of about 2600 runs for each method (between 2659 and 2819). The error bands correspond to one standard deviation of the mean estimator at that point. Note that due to the fixed depth of the *Highway* environment active samples and rollout samples follow exactly the same pattern for different runs of the same method. This allows a detailed analysis on a per-sample basis.

Fig. 1 shows the performance of the different methods as measured by their probability of choosing the best action. There are two salient aspects that confirm our points (1) and (2) from above. First, note that towards the end of the planning process all methods that draw active samples are on the same level, while the purely rollout-based method falls significantly behind. At that stage the tree is fully expanded so that any samples, including rollout samples, are drawn within the tree. Our active learning approach is tailored for this scenario and the experiments show its superiority for that case. On the other hand, in the early planing phase (from 500 to 3000 samples) the methods with a higher fraction of active samples fall behind pure rollout-based sampling and *active*$[0.01]$. Their lag corresponds approximately to the amount of active samples they draw. This suggests that active samples do not significantly contribute to the performance at the early planning stage, which goes well with the intuition that at this stage expanding the tree has a greater significance. Note, however, that they do contribute to optimizing the objective (as discussed below). Also note that all active methods fully compensate at a later stage. This suggests that active samples drawn during the initial planning phase build up a latent potential that only shows during the final planning phase. Finally, we observe that the *active*$[0.01]$ method does at no point fall behind pure rollout-

based sampling, yet it shows a significant advantage at the final stage.

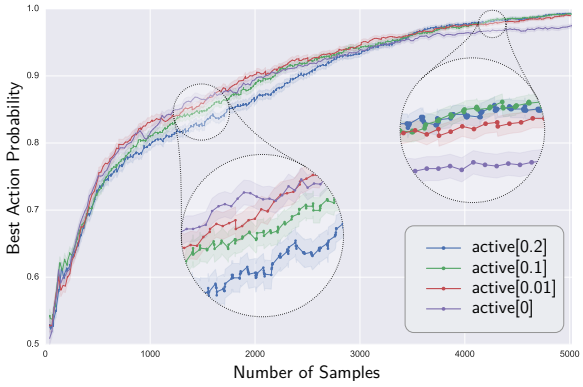

Figure 1: Probability of selecting the best action at the root node as a function of the number of sampled transitions.

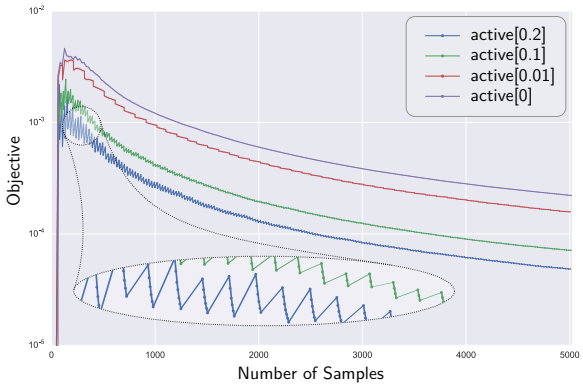

Figure 2: Objective value as a function of the number of sampled transitions. Note the logarithmic scale.

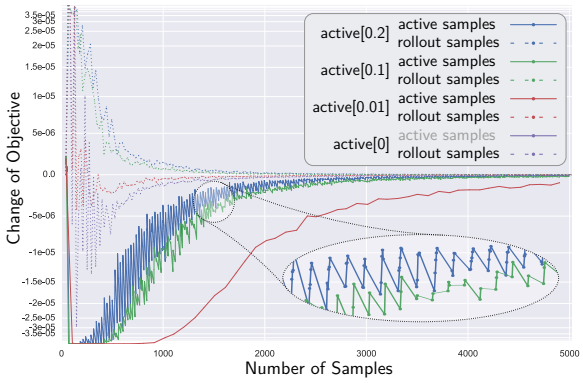

Figure 3: Change of the objective value as a function of the number of sampled transitions, distinguishing active and rollout samples. For rollouts we divided the change by the number of samples that were drawn. Note that we used a non-linear scaling for better visualization.

In order to understand where the advantage of active samples comes from, note that the *Highway* environment is constructed such that a rollout has to pass through segments where it cannot gain valuable information because transitions are deterministic. The *Highway* environment is deliberately kept extreme in this respect but an alternation between more and less predictable passages is common in other domains, too. In contrast to rollouts, our active learning approach can selectively pick switch points for sampling, thereby reducing uncertainty more efficiently. We were also able to construct an adversarial environment where any transition provides roughly the same amount of information so that selectively picking single transitions is futile. In that situation rollouts collect the maximum amount of information possible and *active*[0.01], while being at the level with pure rollout-based sampling, cannot display its merits in the final planning phase. We omit a more detailed discussion here because it does not provide additional insights.

Fig. 2 shows the objective value as a function of the number of samples. These results clearly demonstrate that our active sampling strategy optimizes the objective more efficiently than rollout samples. This becomes apparent on the global time scale where methods with a higher fraction of active samples converge more quickly. But this can also be see on the per-sample basis as showcased in the inset: Active samples successively minimize the objective while rollouts lead to an increase (except for *active*[0.01]).

The interplay of active samples and rollout samples becomes more apparent when plotting the change of the objective for both types separately, as done in Fig. 3. Note that we plot the change *per sample* so that, for instance, active samples in *active*[0.01] have the strongest impact among the active sampling methods yet the convergence (see Fig. 2) is slowest because the they make up only 1% of all samples. We also see (confer the inset in Fig. 3) that successive active samples have a decreasing impact on the objective. As it seems, there is a competition between active samples and rollout samples (and even between successive active samples) on collecting variance-reducing information. On that score, our active learning approach proves superior to rollout-sampling as it is designed specifically for that purpose. In the final planing phase this coincides with a superior ability of identifying the best action, which suggests that reducing the variance is a suitable objective at that stage. However, we also note that this is not the case in the initial planning phase. This raises the question what an appropriate objective for sampling transitions in the early planing stage is.

Conclusion

In this paper we applied the idea of active learning to the problem of sampling transitions in Monte-Carlo tree search. We formulated an active learning objective that minimizes uncertainty of the state value at the root node by collecting samples that maximize information gain. We also derived an approximation that allows computing our objective efficiently. In empirical evaluations on a prototypical environment we showed that combining rollout-samples with actively sampling single transitions within the tree outperforms purely rollout-based tree search.

Appendix

Backup Equations

The mean and variance of the transition probabilities p and the expected reward r are computed (assuming a Dirichlet distribution for p) as

$$\widehat{p}_{(s'|s,a)} = \frac{n_{(s'|s,a)}}{n_{(s,a)}} \tag{19}$$

$$\widetilde{p}_{(s'|s,a)} = \frac{\widehat{p}_{(s'|s,a)}\left(1-\widehat{p}_{(s'|s,a)}\right)}{n_{(s,a)}+1} \tag{20}$$

$$\widetilde{p}_{(s'/s''|s,a)} = -\frac{\widehat{p}_{(s'|s,a)}\,\widehat{p}_{(s''|s,a)}}{n_{(s,a)}+1} \quad s' \neq s'' \tag{21}$$

$$\widehat{r}_{(s,a,s')} = \frac{\Sigma_{\rho_{(s,a,s')}}}{n_{(s'|s,a)}} \tag{22}$$

$$\widetilde{r}_{(s,a,s')} = \frac{\Sigma_{\rho^2_{(s,a,s')}}}{n^2_{(s'|s,a)}} - \frac{\widehat{r}^2_{(s,a,s')}}{n_{(s'|s,a)}}\,, \tag{23}$$

where $n_{(s'|s,a)}$ is the number of transitions that ended in s', $n_{(s,a)} = \sum_{s'} n_{(s'|s,a)}$ is the number of all transitions, $\Sigma_{\rho_{(s,a,s')}}$ is the sum of rewards for transitions that ended in s', and $\Sigma_{\rho^2_{(s,a,s')}}$ is the corresponding sum of squared rewards. Note that in order to get meaningful values also for a single sample, we use the sample variance for the reward instead of the bias-corrected estimator of the population variance. As an alternative one could use a Bayesian approach including a prior for the reward distribution.

For the state value V and the action value Q there are some commonly made implicit assumptions that we explicitly state here to avoid confusion. The first assumption is that states visited in the future are independent from the current state, which allows to go from the definition of V and Q in (25) and (27) below directly to their expected value in (26) and (28). The second assumption is that the expected reward $r_{(s,a,s')}$, the state value V_s, and the action value $Q_{(s,a)}$ are independent whenever s, a or s' differs. As a consequence their covariance is zero, which simplifies some of the sums occurring below. The third assumption is that the policy π is not a random variable, even though in most cases it actually depends on the action values. These assumptions are implicit when performing vanilla MCTS and we will take them for granted from now on. In the following calculations we will repeatedly use the fact that for two random variables x and y

$$\mathbb{E}[\![xy]\!] = \widehat{x}\,\widehat{y} + \mathrm{Cov}[x,y] \tag{24}$$

and that the covariance $\mathrm{Cov}[x,y]$ is zero if x and y are independent.

Expected Value of V and Q

$$V_s = \sum_a \pi_{(a|s)} \underbrace{\sum_{s'} p_{(s'|s,a)}\left(r_{(s,a,s')} + \gamma\, V_{s'}\right)}_{Q_{(s,a)}} \tag{25}$$

$$Q_{(s,a)} = \sum_{s'} p_{(s'|s,a)}\left(r_{(s,a,s')} + \gamma \underbrace{\sum_{a'} \pi_{(a'|s')}\, Q_{(s',a')}}_{V_{s'}}\right) \tag{26}$$

$$\widehat{V}_s = \sum_a \pi_{(a|s)} \sum_{s'} \widehat{p}_{(s'|s,a)}\left(\widehat{r}_{(s,a,s')} + \gamma\, \widehat{V}_{s'}\right) \tag{27}$$

$$\widehat{Q}_{(s,a)} = \sum_{s'} \widehat{p}_{(s'|s,a)}\left(\widehat{r}_{(s,a,s')} + \gamma \sum_{a'} \pi_{(a'|s')}\, \widehat{Q}_{(s',a')}\right) \tag{28}$$

Variance of V and Q

$$\widetilde{V}_s = \mathbb{E}\Big[\Big[\sum_a \pi_{(a|s)}\, Q_{(s,a)}\Big]^2\Big] - \widehat{V}_s^2 \tag{29}$$

$$= \mathbb{E}\Big[\sum_{a,a'} \pi_{(a|s)}\, \pi_{(a'|s)}\, Q_{(s,a)}\, Q_{(s,a')}\Big] - \widehat{V}_s^2 \tag{30}$$

$$= \sum_a \pi^2_{(a|s)}\, \widetilde{Q}_{(s,a)}\,. \tag{31}$$

$$\widetilde{Q}_{(s,a)} = \mathbb{E}\Big[\Big[\sum_{s'} p_{(s'|s,a)}\left(r_{(s,a,s')} + \gamma\, V_{s'}\right)\Big]^2\Big] - \widehat{Q}^2_{(s,a)} \tag{32}$$

$$= \mathbb{E}\Big[\sum_{s',s''} p_{(s'|s,a)}\, p_{(s''|s,a)}\left[r_{(s,a,s')}\, r_{(s,a,s'')} + \gamma^2\, V_{s'}\, V_{s''} + \gamma\, r_{(s,a,s')}\, V_{s''} + \gamma\, r_{(s,a,s'')}\, V_{s'}\right]\Big] - \widehat{Q}^2_{(s,a)} \tag{33}$$

$$= \sum_{s'} \left(\widehat{p}^2_{(s'|s,a)} + \widetilde{p}_{(s'|s,a)}\right)\left[\widetilde{r}_{(s,a,s')} + \gamma^2\,\widetilde{V}_{s'}\right] + \dots$$
$$\dots + \sum_{s',s''} \widetilde{p}_{(s'/s''|s,a)}\left[\widehat{r}_{(s,a,s')}\,\widehat{r}_{(s,a,s'')} + \gamma^2\,\widehat{V}_{s'}\,\widehat{V}_{s''} + \gamma\,\widehat{r}_{(s,a,s')}\,\widehat{V}_{s''} + \gamma\,\widehat{r}_{(s,a,s'')}\,\widehat{V}_{s'}\right] + \dots \tag{34}$$
$$\dots + \sum_{s',s''} \widehat{p}_{(s'|s,a)}\,\widehat{p}_{(s''|s,a)}\left[\widehat{r}_{(s,a,s')}\,\widehat{r}_{(s,a,s'')} + \gamma^2\,\widehat{V}_{s'}\,\widehat{V}_{s''} + \gamma\,\widehat{r}_{(s,a,s')}\,\widehat{V}_{s''} + \gamma\,\widehat{r}_{(s,a,s'')}\,\widehat{V}_{s'}\right] - \widehat{Q}^2_{(s,a)}$$

$$= \sum_{s'} \left(\widehat{p}^2_{(s'|s,a)} + \widetilde{p}_{(s'|s,a)}\right)\left[\widetilde{r}_{(s,a,s')} + \gamma^2\,\widetilde{V}_{s'}\right] + \sum_{s',s''} \widetilde{p}_{(s'/s''|s,a)}\left[\widehat{r}_{(s,a,s')} + \gamma\,\widehat{V}_{s'}\right]\left[\widehat{r}_{(s,a,s'')} + \gamma\,\widehat{V}_{s''}\right], \tag{35}$$

$$= \sum_{s'} \left(\widehat{p}^2_{(s'|s,a)} + \widetilde{p}_{(s'|s,a)}\right)\left[\widetilde{r}_{(s,a,s')} + \gamma^2 \sum_{a'} \pi^2_{(a'|s')}\,\widetilde{Q}_{(s',a')}\right] + \dots$$
$$\dots + \sum_{s',s''} \widetilde{p}_{(s'/s''|s,a)}\left[\widehat{r}_{(s,a,s')} + \gamma \sum_{a'} \pi_{(a'|s')}\widehat{Q}_{(s',a')}\right]\left[\widehat{r}_{(s,a,s'')} + \gamma \sum_{a''} \pi_{(a''|s'')}\widehat{Q}_{(s'',a'')}\right], \tag{36}$$

where the third line in (34) cancels out.

Expected Change of $\widehat{Q}$ and $\widetilde{Q}$

$\widehat{Q}$ and $\widetilde{Q}$ are expected to change given a new sample $(s,a) \to (s^*, \rho^*)$. The expected probability of observing s^* is $\widehat{p}_{(s^*|s,a)}$, which allows us to compute the corresponding expectation explicitly by summing over possible outcomes s^*. The observed reward ρ^* influences the mean and variance of the expected reward. Note that mean and variance of ρ^* relate to the mean and variance of the expected reward as

$$\widehat{\rho}^* = \widehat{r} \qquad \text{and} \qquad \widetilde{\rho}^* = n\,\widetilde{r}\,, \tag{37}$$

where n is the number of transitions observed so far. We denote values computed after adding the new sample with an asterisk. The expected values of $\widehat{r}^*$, $\widehat{r}^{*2}$ and $\widetilde{r}^*$, which occur below, are

$$\mathbb{E}[\![\,\widehat{r}^*\,]\!] = \mathbb{E}\Big[\!\Big[\frac{1}{n+1}\left[\rho^* + n\widehat{r}\right]\Big]\!\Big] = \widehat{r} \tag{38}$$
$$\mathbb{E}[\![\,\widehat{r}^{*2}]\!] = \mathbb{E}\Big[\!\Big[\frac{1}{(n+1)^2}\left[\rho^* + n\widehat{r}\right]^2\Big]\!\Big] \tag{39}$$
$$= \frac{1}{(n+1)^2}\left[\mathbb{E}[\![\rho^{*2}]\!] + 2n\mathbb{E}[\![\rho^*]\!]\widehat{r} + n^2\widehat{r}^2\right] \tag{40}$$
$$= \frac{1}{(n+1)^2}\left[\widehat{r}^2 + n\widetilde{r} + 2n\widehat{r}^2 + n^2\widehat{r}^2\right] \tag{41}$$
$$= \widehat{r}^2 + \frac{n}{(n+1)^2}\,\widetilde{r} \tag{42}$$
$$\mathbb{E}[\![\,\widetilde{r}^*]\!] = \mathbb{E}\Big[\!\Big[\frac{\rho^{*2} + n^2\widetilde{r} + n\widehat{r}^2}{(n+1)^2} - \frac{\widehat{r}^{*2}}{n+1}\Big]\!\Big] \tag{43}$$
$$= \frac{\widehat{r}^2 + n\widetilde{r} + n^2\widetilde{r} + n\widehat{r}^2}{(n+1)^2} - \frac{\widehat{r}^2}{n+1} - \frac{n\widetilde{r}}{(n+1)^3} \tag{44}$$
$$= \frac{\widehat{r}^2 + n\widetilde{r} + n^2\widetilde{r} + n\widehat{r}^2}{(n+1)^2} - \frac{\widehat{r}^2}{n+1} - \frac{n\widetilde{r}}{(n+1)^3} \tag{45}$$
$$= \frac{n^2(n+2)}{(n+1)^3}\,\widetilde{r}\,, \tag{46}$$

where (43) follows from (23). To simplify notation below, we define

$$\alpha_{(s,a,s'=s^*)} = \begin{cases} \frac{n^2(n+2)}{(n+1)^3} - 1 & \text{if } s' = s'' \\ 0 & \text{else} \end{cases} \tag{47}$$
$$\beta_{(s,a,s'=s''=s^*)} = \begin{cases} \frac{n}{(n+1)^2} & \text{if } s' = s'' = s^* \\ 0 & \text{else}\,. \end{cases} \tag{48}$$

This allows us to compute the expected change of $\widehat{Q}$ and $\widetilde{Q}$ as

$$\mathbb{E}[\![\widehat{Q}_{(s,a)}]\!] = \mathbb{E}\Big[\!\Big[\sum_{s'} \widehat{p}^*_{(s'|s,a)}\left(\widehat{r}^*_{(s,a,s')} + \gamma \sum_{a'} \pi_{(a'|s')}\,\widehat{Q}_{(s',a')}\right)\Big]\!\Big]_{s^*,\rho^*} \tag{49}$$
$$= \sum_{s^*} \widehat{p}_{(s^*|s,a)}\mathbb{E}\Big[\!\Big[\sum_{s'} \widehat{p}^*_{(s'|s,a)}\left(\widehat{r}^*_{(s,a,s')} + \gamma \sum_{a'} \pi_{(a'|s')}\,\widehat{Q}_{(s',a')}\right)\Big]\!\Big]_{\rho^*} \tag{50}$$
$$= \sum_{s^*} \widehat{p}_{(s^*|s,a)} \sum_{s'} \widehat{p}^*_{(s'|s,a)}\left(\mathbb{E}[\![\widehat{r}^*_{(s,a,s')}]\!]_{\rho^*} + \gamma \sum_{a'} \pi_{(a'|s')}\,\widehat{Q}_{(s',a')}\right) \tag{51}$$
$$= \sum_{s^*} \widehat{p}_{(s^*|s,a)} \sum_{s'} \widehat{p}^*_{(s'|s,a)}\left(\widehat{r}_{(s,a,s')} + \gamma \sum_{a'} \pi_{(a'|s')}\,\widehat{Q}_{(s',a')}\right) \tag{52}$$

$$= \sum_{s^*} \widehat{p}_{(s^*|s,a)}\, \widehat{Q}^*_{(s,a)} \tag{53}$$

$$\mathbb{E}[\![\widetilde{Q}_{(s,a)}]\!] = \mathbb{E}\Big[\!\Big[\sum_{s'} \big(\widehat{p}^{*2}_{(s'|s,a)} + \widetilde{p}^*_{(s'|s,a)}\big)\big[\widetilde{r}^*_{(s,a,s')} + \gamma^2 \sum_{a'} \pi^2_{(a'|s')}\, \widetilde{Q}_{(s',a')}\big] + \dots$$
$$\dots + \sum_{s',s''} \widetilde{p}^*_{(s'/s''|s,a)}\big[\widehat{r}^*_{(s,a,s')} + \gamma \sum_{a'} \pi_{(a'|s')}\widehat{Q}_{(s',a')}\big]\big[\widehat{r}^*_{(s,a,s'')} + \gamma \sum_{a''} \pi_{(a''|s'')}\widehat{Q}_{(s'',a'')}\big]\Big]\!\Big]_{s^*,\rho^*} \tag{54}$$

$$= \sum_{s^*} \widehat{p}_{(s^*|s,a)}\, \mathbb{E}\Big[\!\Big[\sum_{s'} \big(\widehat{p}^{*2}_{(s'|s,a)} + \widetilde{p}^*_{(s'|s,a)}\big)\big[\widetilde{r}^*_{(s,a,s')} + \gamma^2 \sum_{a'} \pi^2_{(a'|s')}\, \widetilde{Q}_{(s',a')}\big] + \dots$$
$$\dots + \sum_{s',s''} \widetilde{p}^*_{(s'/s''|s,a)}\big[\widehat{r}^*_{(s,a,s')} + \gamma \sum_{a'} \pi_{(a'|s')}\widehat{Q}_{(s',a')}\big]\big[\widehat{r}^*_{(s,a,s'')} + \gamma \sum_{a''} \pi_{(a''|s'')}\widehat{Q}_{(s'',a'')}\big]\Big]\!\Big]_{\rho^*} \tag{55}$$

$$= \sum_{s^*} \widehat{p}_{(s^*|s,a)}\Big\{ \sum_{s'} \big(\widehat{p}^{*2}_{(s'|s,a)} + \widetilde{p}^*_{(s'|s,a)}\big)\Big[\mathbb{E}[\![\widetilde{r}^*_{(s,a,s')}]\!]_{\rho^*} + \gamma^2 \sum_{a'} \pi^2_{(a'|s')}\, \widetilde{Q}_{(s',a')}\Big] + \dots$$
$$\dots + \sum_{s',s''} \widetilde{p}^*_{(s'/s''|s,a)}\Big[\mathbb{E}[\![\widehat{r}^*_{(s,a,s')}\widehat{r}^*_{(s,a,s'')}]\!]_{\rho^*} + \mathbb{E}[\![\widehat{r}^*_{(s,a,s')}]\!]_{\rho^*}\gamma \sum_{a''} \pi_{(a''|s'')}\widehat{Q}_{(s'',a'')} + \dots \tag{56}$$
$$\dots + \mathbb{E}[\![\widehat{r}^*_{(s,a,s'')}]\!]_{\rho^*}\gamma \sum_{a'} \pi_{(a'|s')}\widehat{Q}_{(s',a')} + \gamma \sum_{a'} \pi_{(a'|s')}\widehat{Q}_{(s',a')}\gamma \sum_{a''} \pi_{(a''|s'')}\widehat{Q}_{(s'',a'')}\Big]\Big\}$$

$$= \sum_{s^*} \widehat{p}_{(s^*|s,a)}\Big\{ \sum_{s'} \big(\widehat{p}^{*2}_{(s'|s,a)} + \widetilde{p}^*_{(s'|s,a)}\big)\Big[(\alpha_{(s,a,s'=s^*)} + 1)\,\widetilde{r}_{(s,a,s')} + \gamma^2 \sum_{a'} \pi^2_{(a'|s')}\, \widetilde{Q}_{(s',a')}\Big] + \dots$$
$$\dots + \sum_{s',s''} \widetilde{p}^*_{(s'/s''|s,a)}\Big[\widehat{r}_{(s,a,s')}\widehat{r}_{(s,a,s'')} + \beta_{(s,a,s'=s''=s^*)}\widetilde{r}_{(s,a,s^*)} + \widehat{r}_{(s,a,s')}\gamma \sum_{a''} \pi_{(a''|s'')}\widehat{Q}_{(s'',a'')} + \dots \tag{57}$$
$$\dots + \widehat{r}_{(s,a,s'')}\gamma \sum_{a'} \pi_{(a'|s')}\widehat{Q}_{(s',a')} + \gamma \sum_{a'} \pi_{(a'|s')}\widehat{Q}_{(s',a')}\gamma \sum_{a''} \pi_{(a''|s'')}\widehat{Q}_{(s'',a'')}\Big]\Big\}$$

$$= \sum_{s^*} \widehat{p}_{(s^*|s,a)}\Big\{ \sum_{s'} \big(\widehat{p}^{*2}_{(s'|s,a)} + \widetilde{p}^*_{(s'|s,a)}\big)\Big[\widetilde{r}_{(s,a,s')} + \gamma^2 \sum_{a'} \pi^2_{(a'|s')}\, \widetilde{Q}_{(s',a')}\Big] + \dots$$
$$\dots + \sum_{s',s''} \widetilde{p}^*_{(s'/s''|s,a)}\big[\widehat{r}_{(s,a,s')} + \gamma \sum_{a'} \pi_{(a'|s')}\widehat{Q}_{(s',a')}\big]\big[\widehat{r}_{(s,a,s'')} + \gamma \sum_{a''} \pi_{(a''|s'')}\widehat{Q}_{(s'',a'')}\big] + \dots \tag{58}$$
$$\dots + \big(\widehat{p}^{*2}_{(s^*|s,a)} + \widetilde{p}^*_{(s^*|s,a)}\big)\widetilde{r}_{(s,a,s^*)}\alpha_{(s,a,s^*)} + \widetilde{p}^*_{(s^*|s,a)}\widetilde{r}_{(s,a,s^*)}\beta_{(s,a,s^*)}\Big\}$$

$$= \sum_{s^*} \widehat{p}_{(s^*|s,a)}\Big\{ \widetilde{Q}^*_{(s,a)} + \Big[\big(\widehat{p}^{*2}_{(s^*|s,a)} + \widetilde{p}^*_{(s^*|s,a)}\big)\big(\frac{n^2(n+2)}{(n+1)^3} - 1\big) + \widetilde{p}^*_{(s^*|s,a)}\frac{n}{(n+1)^2}\Big]\widetilde{r}_{(s,a,s^*)}\Big\} \tag{59}$$

$$= \sum_{s^*} \widehat{p}_{(s^*|s,a)}\Big\{ \widetilde{Q}^*_{(s,a)} - \frac{1}{(n+1)^3}\Big[(n^2+3n+1)\,\widehat{p}^{*2}_{(s^*|s,a)} + (2n+1)\,\widetilde{p}^*_{(s^*|s,a)}\Big]\widetilde{r}_{(s,a,s^*)}\Big\}\,, \tag{60}$$

where the starred variables $\widehat{p}^*$, $\widetilde{p}^*$, $\widehat{Q}^*$, $\widetilde{Q}^*$ are computed by temporally adding the corresponding transition $(s,a) \to s^*$ (ignoring the reward) and updating the transition probabilities $\widehat{p}$ and $\widetilde{p}$ accordingly.

References

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning* 47(2-3):235–256.

Bartholomew-Biggs, M.; Brown, S.; Christianson, B.; and Dixon, L. 2000. Automatic differentiation of algorithms. *Journal of Computational and Applied Mathematics* 124(1):171–190.

Berry, D. A., and Fristedt, B. 1985. *Bandit Problems: Sequential Allocation of Experiments (Monographs on Statistics and Applied Probability)*. Springer.

Browne, C. B.; Powley, E.; Whitehouse, D.; Lucas, S. M.; Cowling, P. I.; Rohlfshagen, P.; Tavener, S.; Perez, D.; Samothrakis, S.; and Colton, S. 2012. A Survey of Monte Carlo Tree Search Methods. *Computational Intelligence and AI in Games, IEEE Transactions on* 4(1):1–43.

Chaloner, K., and Verdinelli, I. 1995. Bayesian experimental design: A review. *Statistical Science* 273–304.

Feldman, Z., and Domshlak, C. 2012. Simple regret optimization in online planning for markov decision processes. *arXiv preprint arXiv:1206.3382*.

Keller, T., and Helmert, M. 2013. Trial-Based Heuristic Tree Search for Finite Horizon MDPs. In *ICAPS*.

Kocsis, L., and Szepesvri, C. 2006. Bandit based monte-carlo planning. In *Machine Learning: ECML 2006*. Springer. 282–293.

Kulick, J.; Lieck, R.; and Toussaint, M. 2016. Cross-Entropy as a Criterion for Robust Interactive Learning of Latent Properties. In *NIPS Workshop on the Future of Interactive Learning Machines*.

Settles, B. 2009. Active Learning Literature Survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; van den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; and others. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature* 529(7587):484–489.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning)*. The MIT Press.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints* abs/1605.02688.