Sequential Decision-Making under Uncertainty
(Active Learning, Bayesian Optimization, Bandits)

Piyush Rai

Topics in Probabilistic Modeling and Inference (CS698X)

April 15, 2019
Outline for today

- An overview of three problems
  - Bayesian Active Learning
  - Bayesian Optimization
  - Multi-armed Bandits and Contextual Bandits

- All of these can be framed as sequential decision-making problems

- Leveraging uncertainty is the key in all these problems!
Bayesian Active Learning
(Bayesian) Active Learning

- Supervised Learning needs labeled data which is expensive to obtain
- The typical "passive" approach: Get plenty of labeled data \( \{(x_n, y_n)\}_{n=1}^{N} \) and learn \( f : x \rightarrow y \)
- The "active" approach: The learner asks for most useful training examples and learns iteratively
  1. Start with an initial model \( f_0 \) learned using an initial training set \( D_0 \)
  2. For iteration \( t = 1, \ldots, T \)
     1. Use current model \( f_{t-1} \) to identify the "hardest" input(s) \( x_n \) and get their true label(s) \( y_n \)
     2. Augment training data \( D_t = D_{t-1} \cup (x_n, y_n) \) and retrain to get the new model \( f_t \)
     3. Stop if exhausted the budget for getting labeled data, else continue with next iteration

- How to identify the hardest inputs?
- Estimate of uncertainty in \( f \) helps in that (and Bayesian methods provide that naturally)
(Bayesian) Active Learning

- Some popular (Bayesian) ways to identify the most useful (hardest) inputs $x$
  - Inputs $x$ whose posterior predictive distribution has the largest uncertainty/entropy
    \[
    \mathbb{H}(y|x, D_{t-1}) = -\sum_{k=1}^{K} p(y = k|x, D_{t-1}) \log p(y = k|x, D_{t-1})
    \]
  - Inputs $x$ including which the current model $f_{t-1}$’s expected uncertainty reduces maximally†
    \[
    \mathbb{H}[f_{t-1}|D_{t-1}] - \mathbb{E}_{y \sim p(y|x, D_{t-1})} \mathbb{H}[f_{t-1}|D_{t-1}, x, y]
    \]
- The criteria (such as the above ones) for choosing $x$ are typically known as “acquisition function”
- Bayesian Active Learning‡ shown to be very successful for many deep learning models that require lots of labeled data when learned “passively”

† Bayesian Active Learning for Classification and Preference Learning (Houlsby et al, 2011), ‡ Deep Bayesian Active Learning with Image Data (Gal et al, 2017)
Bayesian Optimization
Bayesian Optimization: The Basic Formulation

Consider finding the optima $x^*$ (say minima) of a function $f(x)$

Caveat: We don’t know the form of the function; can’t get its gradient, Hessian, etc

Suppose we can only query the function’s values at certain points (i.e., only black-box access)

Several applications, such as drug design, hyperparameter optimization, etc.
Bayesian Optimization

- We would like to locate the minima using the queries we made

- We would like to do so while making as few queries as possible

- Reason: Each query may be costly

- The cost may be time, money, or both
Bayesian Optimization

- Suppose we are allowed to make the queries sequentially.
- Queries so far can help us guess what the function looks like (by solving a regression problem).

![DIAGRAM]

Above: dotted = true $f(x)$, solid green = current estimate of $f(x)$, shaded = uncertainty in $f(x)$

- BO use past queries and the function’s estimate+uncertainty to decide where to query next.
- Somewhat similar to Active Learning but BO also uses past queries to decide where to query next.
So basically, Bayesian Optimization requires two ingredients

- A **regression model** to learn the function given the current set of query-value pairs \( \{ x_n, f(x_n) \}_{n=1}^{N} \)

- An **acquisition function** \( A(x) \) that tells us the utility of any future point \( x \)

**Note**: Function evaluations given to us may be noisy, i.e., \( f(x) + \epsilon \)

**Important**: The regression model must also have estimate of function’s uncertainty, e.g.,

- Gaussian Process, Bayesian Neural Network, or any nonlinear regression model with uncertainty
Some Acquisition Functions: Probability of Improvement (PI)

- Given the set of previous query points $X$ and corresponding function values $f$, suppose
  \[ f' = \min f \]

- Suppose $f$ denotes the function's value at some new point $x$

- We have an improvement if $f \leq f'$ (recall we are doing minimization)

- The posterior predictive for $x$ is $p(f|x) = \mathcal{N}(f|\mu(x), \sigma^2(x))$

- We can therefore define the probability of improvement based acquisition function
  \[
  A_{PI}(x) = p(f \leq f') = \int_{-\infty}^{f'} \mathcal{N}(f|\mu(x), \sigma^2(x)) df = \Phi \left( \frac{f' - \mu(x)}{\sigma(x)} \right)
  \]

- Point with the highest probability of improvement is selected as the next query point

- Note that BO involves optimizing the acquisition function to find the next query point $x$ (this optimization to be cheaper than the original problem)
Some Acquisition Functions: Expected Improvement (EI)

- PI doesn’t take into account the amount of improvement
- Expected Improvement (EI) takes this into account and is defined as

\[
A_{EI}(x) = \mathbb{E}[f' - f] = \int_{-\infty}^{f'} (f' - f)N(f|\mu(x), \sigma^2(x))df
\]

\[
= (f' - \mu(x))\Phi\left(\frac{f' - \mu(x)}{\sigma(x)}\right) + \sigma(x)N\left(\frac{f' - \mu(x)}{\sigma(x)}; 0, 1\right)
\]

- Point with the highest expected improvement is selected as the next query point
- Trade-off b/w exploitation and exploration
  - Prefer points with low predictive mean (exploitation) and large predictive variance (exploration)
Some Acquisition Functions: Lower Confidence Bound (LCB)

- Another acquisition function that takes into account exploitation vs exploration
- Used when the regression model is a Gaussian Process (GP)
- Assuming the posterior predictive for a new point $x$ to be $\mathcal{N}(\mu(x), \sigma^2(x))$, the LCB is
  \[ A_{LCB}(x) = \mu(x) - \kappa \sigma(x) \]
- $\kappa$ is a parameter to trade-off b/w mean and variance
- Point with the smallest LCB is selected as the next query point
- Strong theoretical results; under certain conditions, the iterative application of this acquisition function will converge to the true global optima of $f$ (Srinivas et al. 2010)
- Note: When solving maximization problems, the analogous quantity is the “Upper Confidence Bound” (UCB), and point with largest UCB is chosen
  \[ A_{UCB}(x) = \mu(x) + \kappa \sigma(x) \]
Bayesian Optimization: Some Challenges/Open Problems

- Learning the regression model for the function
  - GPs can be expensive as $N$ grows
    - Bayesian neural networks can be an more efficient alternative to GPs (Snoek et al, 2015)
  - Hyperparams of the regression model itself (e.g., GP cov. function, Bayesian NN hyperparam)

- High-dimensional Bayesian Optimization (optimizing functions of many variables)
  - Number of function evaluations required would be quite large in high dimensions
  - Lot of recent work on this (e.g., based on dimensionality reduction)

- Multitask Bayesian Optimization (optimizing several related functions)
  - Can leverage ideas from multitask learning
Bayesian Optimization: Further Resources

- Some survey papers:
  - A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning (Brochu et al., 2010)
  - Taking the Human Out of the Loop: A Review of Bayesian Optimization (Shahriari et al., 2015)

- Some open source software libraries

<table>
<thead>
<tr>
<th>SOFTWARE</th>
<th>REGR. MODEL</th>
<th>ACQ. FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPEARMINT</td>
<td>GAUSSIAN PROCESS</td>
<td>EXP. IMPROV</td>
</tr>
<tr>
<td>MOE</td>
<td>GAUSSIAN PROCESS</td>
<td>EXP. IMPROV</td>
</tr>
<tr>
<td>HYPEROPT</td>
<td>TREE PARZEN EST.</td>
<td>EXP. IMPROV</td>
</tr>
<tr>
<td>SMAC</td>
<td>RANDOM FOREST</td>
<td>EXP. IMPROV</td>
</tr>
</tbody>
</table>

- Another one: SIGOPT (https://sigopt.com/research)
Multi-armed Bandits
and Contextual Bandits
Multi-armed Bandits (MAB)

- Suppose we have $K$ options/actions (a.k.a. “arms”) to choose from.
- Each arm $a \in \{1, \ldots, K\}$ has a binary reward $r_a$ with (unknown) reward probability $p(r_a = 1) = \mu_a$.
- Player’s goal is to play “optimally” based on player’s current estimates of $\mu_1, \ldots, \mu_K$.

Assume the best arm/action at time $t$ is $a_t^* = \text{argmax}_{a \in \{1, \ldots, K\}} \mu_a$ (but unknown to the player).

A measure of how well the player has played is the cumulative regret

$$R(T) = \sum_{t=1}^{T} (\mu_{a_t^*} - \mu_{a_t})$$

The problem setting called “Bandit” since we only get to know the reward for the chosen arm (don’t get to know the rewards we would have got upon choosing other arms).
Multi-armed Bandits: Exploration vs Exploitation

What should be our strategy to choose the next arm?

Suppose we have used the past action-reward history \( \{a_t, r_t\}_{t=1}^T \) and estimated \( \hat{\mu}_1, \ldots, \hat{\mu}_K \)

- **Exploitation**: Choose the optimal arm, \( a_{T+1} = \arg\max_{a \in \{1, \ldots, K\}} \hat{\mu}_a \) (greedy policy)
- **Exploitation + Exploration**: Also explore other sub-optimal arms with some probability, e.g.,
  - **\( \epsilon \)-Greedy**: With prob. \( 1 - \epsilon \), choose optimal arm, with prob. \( \epsilon \) randomly choose an arm
  - **Upper Confidence Bound (UCB)**: \( a_{T+1} = \arg\max_{a \in \{1, \ldots, K\}} [\hat{\mu}_a + \sigma_a] \) where \( \sigma_a \) is a confidence bound
  - **Thompson Sampling**: Estimate each arm’s reward distribution and sample an arm randomly, e.g.,

\[
\mu_a \sim \text{Beta}(\alpha + N_{a,1}, \beta + N_{a,0})
\]

\( a_{T+1} = \arg\max_{a \in \{1, \ldots, K\}} \mu_a \) (reward dist. of arm \( a \), assuming binary rewards)
Contextual Bandits

- Similar to MAB but here, at time $t$, we have a set of context vectors $D_t \subset \mathbb{R}^D$
- What the context vectors are depends on the application, e.g.,
  - For a web-advertisement problem, context vectors can be based on user-website interactions
  - In matrix fact. based learning of reco-sys, context vectors can be user/item latent factors
- At time $t$, a context $x_t \in D_t$ (e.g., website) is chosen and system receives a reward
  \[ r_t = f(x_t) + \epsilon_t \quad (f \text{ is unknown to the user}) \]
- The reward could be “ad clicked or not”, “time-spent”, “amount of purchases”, etc
- The best context at time $t$: $x^*_t = \arg\max_{x \in D_t} f(x)$ (unknown to the user)
- Just like the MAB case, performance is measured in terms of regret. For linear model
  \[ R(T) = \sum_{t=1}^{T} (\theta^\top x^*_t - \theta^\top x_t) \quad (\text{assuming } f \text{ is linear model}) \]
  where $x^*_t$ is the best context at time $t$ and $x_t$ is the chosen context

---

† Bandits and Recommender Systems (Mary et al, 2016), Efficient Thompson Sampling for Online Matrix-Factorization Recommendation (Kawale et al, 2015),

Topics in Prob. Modeling & Inference - CS698X (Piyush Rai, IITK)

Sequential Decision-Making under Uncertainty
Contextual Bandits (Contd.)

- The goal is to learn the underlying reward function $f$ to minimize the regret
- Assuming a linear model, i.e., $r_t = \theta^\top x_t + \epsilon_t$, estimating $\theta$ is like solving linear regression
- If $r_t$ is not real-valued (e.g., binary), we can use a GLM to estimate $\theta$
- Nonlinear reward functions can also be handled using GP or deep neural nets

$$r_t = f(x_t) + \epsilon_t$$

- For selecting the next context, can use exploitation or exploitation+exploration as in MAB
  - Greedy, $\epsilon$-Greedy, UCB, Thompson Sampling†, etc
  - For exploration, we need an estimate of the uncertainty in $f$ (e.g., using $f$’s posterior, or some other uncertainty estimate)

Thompson Sampling for Contextual Bandits

- Assume a linear model, i.e., \( r_t = \theta^\top x_t + \epsilon_t \) with \( \epsilon_t \sim \mathcal{N}(0, \beta^{-1}) \)

- Assuming Gaussian prior \( p(\theta) = \mathcal{N}(0, I) \), the posterior over \( \theta \) (assuming hyperparams known)

\[
p(\theta|\{x_t, r_t\}_{t=1}^T) = \mathcal{N}(\mu_T, \Sigma_T)
\]

where \( \mu_T \) and \( \Sigma_T \) are the mean and covariance of the Gaussian posterior

- Thompson Sampling chooses the next context \( x_{T+1} \) as follows

\[
\tilde{\theta} \sim \mathcal{N}(\mu_T, \Sigma_T) \quad \text{(sample a } \theta \text{ randomly)}
\]

\[
x_{T+1} = \arg\max_{x \in \mathcal{D}_{T+1}} \tilde{\theta}^\top x \quad \text{(select the context that gives the largest reward given the sampled } \tilde{\theta} \text{)}
\]

- Posterior is updated by including the new observation \( (x_{T+1}, r_{T+1}) \) where reward \( r_{T+1} = \tilde{\theta}^\top x_{T+1} \)

- TS can also be applied for contextual bandits with nonlinear reward functions†

---

† Deep Bayesian Bandits Showdown: An Empirical Comparison of Bayesian Deep Networks for Thompson Sampling (Riquelme et al, 2018)
Conclusion

- Bayesian methods can be very useful for sequential decision-making under uncertainty
  - Bayesian Active Learning can be very useful when labeled data is costly
  - Bayesian Optimization is widely used nowadays in “automated” ML (e.g., hyperparameter tuning) and various other expensive “discovery” problems
  - Bandit algos are used in recommendation systems, web-advertising, reinforcement learning, etc (with or without contexts)

<table>
<thead>
<tr>
<th>Example</th>
<th>Context</th>
<th>Action</th>
<th>Reward</th>
</tr>
</thead>
<tbody>
<tr>
<td>News site</td>
<td>User location</td>
<td>an article to display</td>
<td>1 if clicked, 0 otherwise makes sense even w/o context</td>
</tr>
<tr>
<td>Dynamic pricing</td>
<td>Buyer’s profile</td>
<td>a price $p$</td>
<td>$p$ if sale, 0 otherwise $\Rightarrow$ bandits or contextual bandits</td>
</tr>
<tr>
<td>Health advice</td>
<td>User health profile</td>
<td>what to recommend</td>
<td>1 if adopted, 0 otherwise Context is essential $\Rightarrow$ contextual bandits</td>
</tr>
<tr>
<td>Chatbot</td>
<td>Stage of conversation</td>
<td>what to say</td>
<td>1 if task completed, 0 otherwise Context is essential, depends on the past actions $\Rightarrow$ reinforcement learning</td>
</tr>
</tbody>
</table>

(Table credit: Alekh Agarwal)