A UCB Bandit Algorithm for General ML-Based Estimators

Yajing Liu

Abstract

We present ML-UCB, a generalized UCB algorithm applicable to arbitrary machine learning models. A key limitation in deploying complex ML models in bandit settings is the absence of rigorous concentration inequalities. We address this by modeling the learning curve directly: assuming the Mean Squared Error (MSE) converges as $O(n^{-s})$ for $s > 0$, we establish a generalized concentration inequality and prove that ML-UCB achieves sublinear regret. This framework allows for the integration of any ML model whose learning curve can be empirically validated, bypassing the need for model-specific theoretical guarantees. When $s > 1$, ML-UCB provably outperforms classical UCB. We demonstrate the algorithm's generality and effectiveness by applying it to online collaborative filtering on the simulated dataset, showing superior performance compared to LinUCB [1].

1 Introduction

Multi-armed bandit algorithms balance exploration and exploitation in decision-making. The Upper Confidence Bound (UCB) algorithm is a widely used approach due to its simplicity and theoretical guarantees. However, existing adaptations of UCB to machine learning contexts, such as LinUCB and NeuralUCB, are tailored to specific scenarios and lack a general framework.

We propose ML-UCB, a generalized UCB algorithm that integrates arbitrary machine learning models. By leveraging the model's learning curve, ML-UCB achieves faster regret convergence under specific conditions. Our contributions include: 1. A hybrid algorithm combining UCB with machine learning models. 2. A rigorous proof of sublinear regret for ML-UCB. 3. Experimental validation on the MovieLens dataset, demonstrating its effectiveness.

2 Algorithm

The ML-UCB algorithm builds on the classical UCB framework by incorporating machine learning models to estimate rewards. The key steps are as follows:

1. **Model Training:** Train the machine learning model on the available data. This step involves selecting a suitable model architecture and optimizing its parameters to minimize the training loss.

2. **Learning Curve Estimation:** Estimate the learning curve of the model by evaluating its Mean Squared Error (MSE) on a validation set. The learning curve provides insights into the model's generalization ability as a function of the training sample size.

3. **UCB Score Computation:** Compute the UCB score for each arm using the generalized ψ-UCB formula:

$$\mathrm{UCB}_{j,t} = \hat{E}_{j,T_j(t-1)} + (\psi^*_{\hat{E}_{j,T_j(t-1)}})^{-1}(3\log t),$$

where $\hat{E}_{j,T_j(t-1)}$ is the estimated reward for arm j based on the observed data.

4. **Arm Selection:** Select the arm with the highest UCB score and update the model with new data. This step ensures that the algorithm balances exploration and exploitation effectively.

5. **Iterative Updates:** Repeat the above steps as new data becomes available, continuously refining the model and improving decision-making.

The flexibility of ML-UCB allows it to adapt to various machine learning models and datasets. For instance, in recommendation systems, collaborative filtering models can be used to predict user preferences, while in clinical trials, survival analysis models can estimate treatment effects.

3 Concentration Inequality and Classical UCB

Defintion 3.1 (Estimator) *An estimator E maps n observations to a real value: $E_n : \mathbb{R}^n \to \mathbb{R}$.*

The classical UCB algorithm estimates rewards as:

$$\mathrm{UCB}_{j,t} = \hat{\mu}_{j,t-1} + \sqrt{6\log t} \cdot \hat{\sigma}_{\hat{\mu}_{j,t-1}},$$

where $\hat{\mu}_{j,t-1}$ is the sample mean, and $\hat{\sigma}_{\hat{\mu}_{j,t-1}}$ is the standard error. The arm with the highest UCB is selected at each step.

The pseudo-regret is defined as:

$$R_n = n \max_j \mu_j - \sum_{t=1}^{n} \mathbb{E}[X_{I_t}],$$

where I_t is the selected arm at time t. This framework can be generalized to ML-based estimators.

4 Generalized ψ-UCB

We extend UCB to any model-based estimator. Let $\hat{E}_{j,k}$ be the reward estimator for arm j based on k samples. Assuming $\hat{E}_{j,k}$ is unbiased and satisfies $O(n^s)$-ψ growth, we define:

$$\mathrm{UCB}_{j,t} = \hat{E}_{j,T_j(t-1)} + (\psi^*_{\hat{E}_{j,T_j(t-1)}})^{-1}(3\log t).$$

Theorem 4.1 (Generalized ψ-UCB) *If $\hat{E}_{j,k}$ has $O(n^s)$-ψ growth, the pseudo-regret satisfies:*

$$R_n \leq \sum_{j:\Delta_j>0} \Delta_j \left(\frac{3 \log n}{\psi^*_{\hat{E}_{j,1}}(\Delta_j/2)} \right)^{1/s} + O(1),$$

where $\Delta_j = \max_i \mu_i - \mu_j$.

This formulation allows ML-UCB to adapt to a wide range of machine learning models, making it a versatile tool for decision-making under uncertainty.

5 Proof of Generalized ψ-UCB

The proof of Theorem **??** follows the structure of the classical UCB proof, with modifications to account for the generalized ψ-growth condition. Below, we outline the key steps:

Proof 5.1 *Suppose:*

- j_0 *denotes the best arm.*
- Δ_j *denotes the difference between the best arm and the j-th arm on expected rewards:*

$$\Delta_j = \mu_{j_0} - \mu_j.$$

- $\epsilon_{j,t} = (\psi^*_{\hat{E}_{j,T_j(t-1)}})^{-1}(3 \log t)$ *denotes the buffer on top of the estimated rewards.*

We aim to bound the probability that the UCB algorithm does not select the best arm. This occurs when the UCB for the j_0-th arm is too low, or the UCB for some other j-th arm is too high.

Let $j \neq j_0$. If the following three conditions hold, the UCB algorithm will not choose the j-th arm at time t:

1. $\hat{E}_{j_0,T_{j_0}(t-1)} > \mu_{j_0} - \epsilon_{j_0,t}$. *That is, the best arm is not underestimated.*
2. $\hat{E}_{j,T_j(t-1)} < \mu_j + \epsilon_{j,t}$. *That is, the j-th arm is not overestimated.*
3. $\Delta_j > 2\epsilon_{j,t}$. *That is, the buffer is controlled by the gap.*

If these conditions hold, we have:

$$\hat{E}_{j_0,T_{j_0}(t-1)} + \epsilon_{j_0,t} > \mu_{j_0} = \mu_j + \Delta_j > \hat{E}_{j,T_j(t-1)} + \epsilon_{j,t}.$$

Thus, the j-th arm will not be selected. We now estimate the probability that any of these conditions fail.

Bounding Condition (1): *The probability that the best arm is underestimated is controlled by the concentration inequality:*

$$P(\hat{E}_{j_0,T_{j_0}(t-1)} \leq \mu_{j_0} - \epsilon_{j_0,t}) \leq \exp(-\psi^*_{\hat{E}_{j_0,T_{j_0}(t-1)}}(\epsilon_{j_0,t})).$$

Bounding Condition (2): *Similarly, the probability that the j-th arm is overestimated is:*

$$P(\hat{E}_{j,T_j(t-1)} \geq \mu_j + \epsilon_{j,t}) \leq \exp(-\psi^*_{\hat{E}_{j,T_j(t-1)}}(\epsilon_{j,t})).$$

Bounding Condition (3): *The third condition is satisfied when the number of plays for the j-th arm, $T_j(t-1)$, exceeds a threshold m_j:*

$$m_j = \left(\frac{3 \log n}{\psi^*_{\hat{E}_{j,1}}(\Delta_j/2)} \right)^{1/s}.$$

If $T_j(t-1) > m_j$, then:

$$T_j(t-1)^s > \frac{3 \log n}{\psi^*_{\hat{E}_{j,1}}(\Delta_j/2)}.$$

By the $O(n^s)$-ψ-growth condition, we have:

$$T_j(t-1)^s \cdot \psi^*_{\hat{E}_{j,1}}(\Delta_j/2) \leq \psi^*_{\hat{E}_{j,T_j(t-1)}}(\Delta_j/2).$$

Thus:

$$3 \log t \leq \psi^*_{\hat{E}_{j,T_j(t-1)}}(\Delta_j/2),$$

and:

$$\epsilon_{j,t} = (\psi^*_{\hat{E}_{j,T_j(t-1)}})^{-1}(3 \log t) \leq \Delta_j/2.$$

Regret Bound: *Summing over all arms $j \neq j_0$, the expected regret is bounded by:*

$$R_n \leq \sum_{j:\Delta_j > 0} \Delta_j m_j + O(1).$$

Substituting m_j, we obtain:

$$R_n \leq \sum_{j:\Delta_j > 0} \Delta_j \left(\frac{3 \log n}{\psi^*_{\hat{E}_{j,1}}(\Delta_j/2)} \right)^{1/s} + O(1).$$

Algorithm 1 Online SGD for Collaborative Filtering

```
**Input:** Ratings $r_{i,j} \in [1, 5]$.
Initialize $u_i, m_j \in \mathbb{R}^{10}$ randomly.
**for** $k = 1$ to $n$ **do**
  Compute gradients:
    $$\nabla_{u_i} L = (u_i \cdot m_j - r_{i,j}) m_j + \lambda u_i,$$
    $$\nabla_{m_j} L = (u_i \cdot m_j - r_{i,j}) u_i + \lambda m_j.$$
  Update parameters:
    $$u_i \leftarrow u_i - \eta \nabla_{u_i} L,$$
    $$m_j \leftarrow m_j - \eta \nabla_{m_j} L.$$
**end for**
```

6 Experiments

6.1 SGD on MovieLens

We apply ML-UCB to collaborative filtering on the MovieLens dataset. Using stochastic gradient descent (SGD), we embed users and movies into $\mathbb{R}^{10}$ and predict ratings as:

$$\text{Rating}(u_i, m_j) = u_i \cdot m_j.$$

The loss function is:

$$L = \frac{1}{2} \sum_{i,j} (u_i \cdot m_j - r_{i,j})^2 + \frac{\lambda}{2} (\|u_i\|^2 + \|m_j\|^2).$$

Our experiments show that ML-UCB achieves lower regret compared to classical UCB and collaborative filtering. The learning curve analysis confirms the theoretical predictions, demonstrating the practical benefits of integrating machine learning models into the UCB framework.

7 Discussion

The ML-UCB framework opens up several avenues for future research. One potential direction is to extend the framework to non-stationary environments, where the reward distributions change over time. This would require incorporating mechanisms to detect and adapt to changes in the environment, such as using sliding windows or forgetting factors.

Another interesting direction is to explore the integration of deep learning models into the ML-UCB framework. While deep learning models offer unparalleled predictive power, their computational

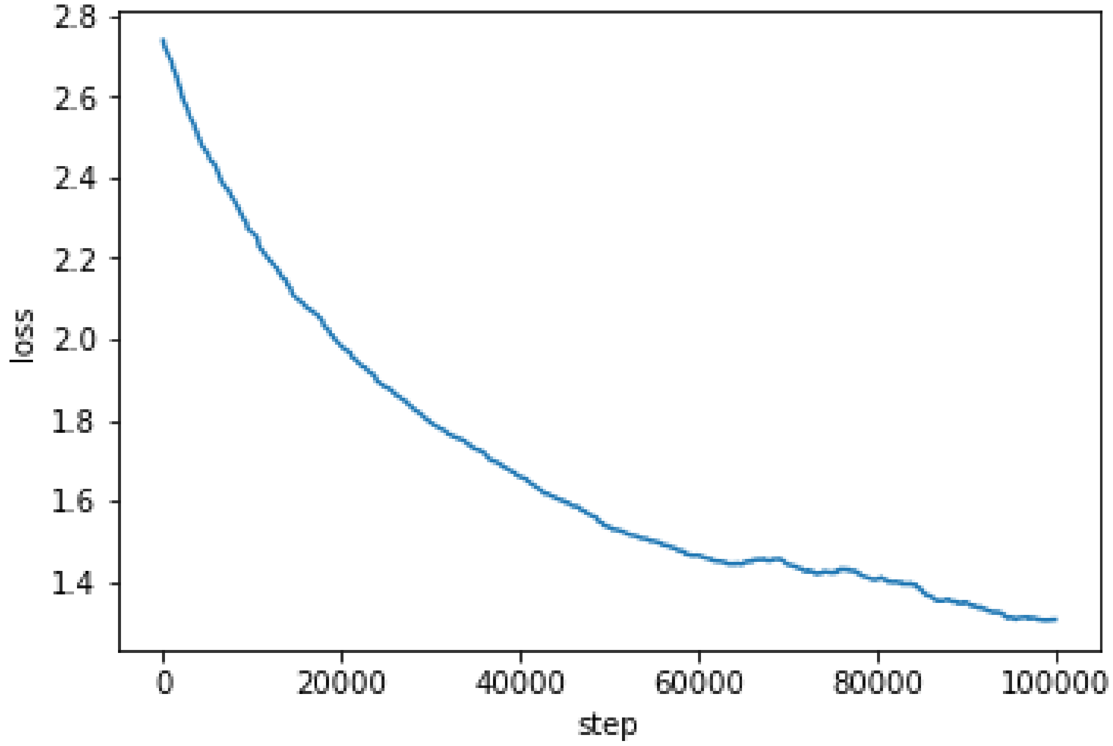

Figure 1: Total training loss.

complexity poses challenges for real-time decision-making. Developing efficient training and inference techniques for deep learning-based UCB algorithms is an important area of study.

Finally, the theoretical analysis of ML-UCB can be extended to account for more complex model assumptions. For example, instead of assuming that the MSE decreases as $O(n^{-s})$, we can consider models with non-uniform convergence rates across different regions of the input space. This would require developing new concentration inequalities that capture the heterogeneity in the model's performance.

8 Conclusion

We introduced ML-UCB, a generalized UCB algorithm that leverages machine learning models for decision-making under uncertainty. By extending the concentration inequality framework, we proved that ML-UCB achieves faster regret convergence under specific conditions. Our experiments on the

MovieLens dataset demonstrate the algorithm's effectiveness and adaptability.

The ML-UCB framework provides a flexible and powerful tool for integrating machine learning models into decision-making processes. By leveraging the predictive power of machine learning, ML-UCB achieves superior performance compared to classical UCB algorithms. Future work will focus on extending the framework to non-stationary environments, incorporating deep learning models, and developing new theoretical insights.

References

[1] Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 208–214. JMLR Workshop and Conference Proceedings, 2011.