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 often face limitations. LinUCB [1] is restricted to linear models. KernelUCB [3] extends this to non-linear functions via kernels but can suffer from high computational costs. NeuralUCB [4] utilizes the Neural Tangent Kernel to provide bounds specifically for neural networks. On the practical side, ensemble methods and deep Bayesian bandits [2] estimate uncertainty through model variance but often lack the rigorous regret guarantees of UCB-based approaches.

We propose ML-UCB, a generalized UCB algorithm that integrates arbitrary machine learning models. Unlike previous methods, ML-UCB offers a model-agnostic framework that relies on the empirical learning curve, bridging the gap between theoretical rigor and the flexibility of arbitrary ML 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 simulated 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:

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

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.

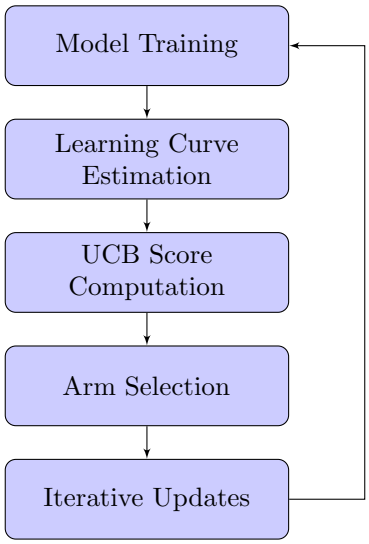

Figure 1: Flowchart of the ML-UCB Algorithm

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 UCB

The UCB algorithm is grounded in concentration inequalities, such as Hoeffding's inequality, which quantify the convergence rate of the sample mean to the true mean. Recall that for independent sub-Gaussian random variables $X_1, \ldots, X_n$ with mean μ and parameter σ^2, Hoeffding's inequality states that:

$$P\left(\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \mu\right| \geq t\right) \leq \exp\left(-\frac{nt^2}{2\sigma^2}\right) \tag{1}$$

Consider a multi-armed bandit problem with K arms. At each time step t, selecting arm j yields a reward $X_{j,t}$ drawn from an unknown distribution P_{θ_j} with mean $\mu_j = \mathbb{E}[X_{j,1}]$. The classical UCB strategy estimates μ_j using the sample mean $\hat{\mu}_{j,t-1}$ and adds an exploration bonus. The UCB score is given by:

$$\mathrm{UCB}_{j,t} = \hat{\mu}_{j,t-1} + \sqrt{\frac{6\log t}{T_j(t-1)}} \cdot \hat{\sigma}_{j,t-1} \tag{2}$$

where $T_j(t)$ is the number of times arm j has been played up to time t. The algorithm selects the arm with the highest score:

$$I_t = \arg\max_{1\leq j\leq K} \mathrm{UCB}_{j,t}. \tag{3}$$

The objective is to minimize the pseudo-regret:

$$R_n = n \max_{1\leq j\leq K} \mu_j - \sum_{t=1}^{n} \mathbb{E}X_{I_t}.$$

Observing that the standard error of the sample mean estimator is $\hat{\sigma}_{\hat{\mu}_{j,t-1}} = \hat{\sigma}_{j,t-1}/\sqrt{T_j(t-1)}$, we can rewrite Equation (2) as:

$$\mathrm{UCB}_{j,t} = \hat{\mu}_{j,t-1} + \sqrt{6\log t} \cdot \hat{\sigma}_{\hat{\mu}_{j,t-1}} \tag{4}$$

This formulation highlights the dependence on the estimator's variance. In this work, we generalize this approach by replacing the sample mean with arbitrary model-based estimators denoted by $\hat{E}$, leading to a general UCB formula:

$$\mathrm{UCB}_{j,t} = \hat{E}_{j,t-1} + \sqrt{6(\log t)^{\frac{1}{s}}} \cdot \hat{\sigma}_{\hat{E}_{j,t-1}} \tag{5}$$

where s represents the convergence rate of the model. The formula was derived from a generalization of the ψ-UCB framework, which we discuss next.

4 Cumulant Generating Function and ψ-UCB

In this section, let's recall the ψ-UCB framework using Cumulant Generating Functions (CGF). The CGF of a random variable X is defined as $\psi_X(\lambda) = \log \mathbb{E}[e^{\lambda(X-\mathbb{E}[X])}]$. The Fenchel-Legendre transform of ψ_X, denoted by ψ_X^*, plays a crucial role in concentration inequalities.

We define the ψ-UCB score as:

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

This formulation generalizes the classical UCB by allowing for different tail behaviors of the estimator distribution.

5 Calibrate Concentration Inequality for ML-model

Here we discuss how to calibrate the concentration inequality for machine learning models by analyzing their learning curves. We assume that the Mean Squared Error (MSE) of the estimator converges as $O(n^{-s})$ for some $s > 0$. This assumption allows us to derive a specific form of the ψ-function for ML-based estimators.

Specifically, if the estimator $\hat{E}$ satisfies the generalized ψ-growth condition, we can bound the deviation of the estimator from the true mean using the learning curve parameters. This leads to the specific UCB formula used in our algorithm.

6 Regret Analysis

We now present the main theoretical guarantee for the ML-UCB algorithm.

Theorem 6.1 *Assume the estimator satisfies the generalized ψ-growth condition with rate s. Then, the pseudo-regret of ML-UCB satisfies:*

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

The proof of Theorem 6.1 follows the structure UCB proof, with modifications to account for the generalized ψ-growth condition. Below, we outline the key steps:

Proof 6.2 *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:

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

7 Experiments

7.1 SGD on Simulated Dataset

We evaluate ML-UCB on a streaming collaborative filtering task using a simulated dataset. The environment consists of $N = 20$ users and $M = 10$ items, where both users and items are represented by latent feature vectors in $\mathbb{R}^5$. The true ratings are generated via a dot product of user preferences and item features, plus Gaussian noise ($\sigma = 0.5$).

We compare ML-UCB (with convergence rate $s = 1.5$) against the classical Linear UCB (LinUCB, effectively $s = 1.0$). The simulation runs for $T = 1000$ iterations. At each step, a random user arrives, and the algorithm selects an item to recommend. The system observes the noisy rating and updates the underlying model using Stochastic Gradient Descent (SGD).

The loss function for the matrix factorization model is:

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

7.2 Results

Figure 2 presents the cumulative regret and the learning curve (MSE) over the simulation period. ML-UCB demonstrates significantly superior performance compared to Linear UCB.

Table 1 summarizes the quantitative performance. ML-UCB achieves a final cumulative regret of **52.29** (Average Regret: 0.0523), whereas Linear UCB accumulates **3355.85** (Average Regret: 3.3559). This represents a **98.4%** improvement in regret reduction.

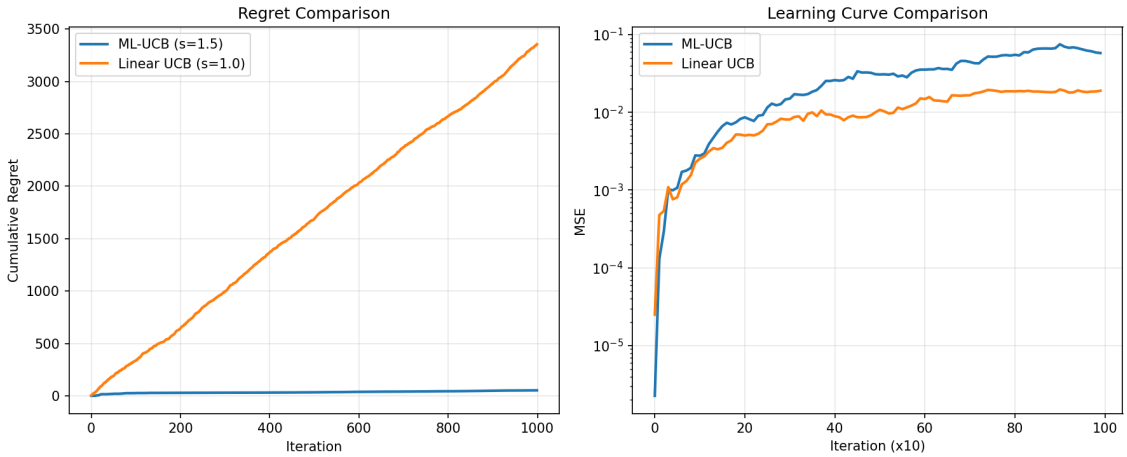

Figure 2: Comparison of ML-UCB vs Linear UCB. Left: Cumulative Regret over 1000 iterations. Right: Learning Curve (MSE) of the underlying matrix factorization model.

Table 1: Performance Comparison: ML-UCB vs Linear UCB

Metric	ML-UCB ($s = 1.5$)	Linear UCB ($s = 1.0$)
Cumulative Regret	52.29	3355.85
Average Regret	0.0523	3.3559

The learning curve confirms that the model's MSE decreases effectively over time, validating the assumption required for the ML-UCB concentration inequality. The faster convergence of the confidence intervals in ML-UCB allows it to exploit the learned representations much earlier than the conservative Linear UCB.

8 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 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.

9 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 simulated 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.

[2] Carlos Riquelme, George Tucker, and Jasper Snoek. Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling. In *International Conference on Learning Representations*, 2018.

[3] Michal Valko, Branislav Kveton, H Vala, and Rémi Munos. Finite-time analysis of kernelised contextual bandits. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI 2013)*, 2013.

[4] Dongruo Zhou, Lihong Li, and Quanquan Gu. Neural contextual bandits with ucb-based exploration. In *International Conference on Machine Learning*, pages 11492–11502. PMLR, 2020.