Tensor bandits and their applications

Horbach Maryna

HSE University, Moscow, Russia

Main contribution

- New algorithm based on tensor train decomposition was developed. It combines idea of low rank approximation and effective tensor operations. The comparative research of existing algorithms was made and TensorTrain algorithm showed competitive results.
- Approach for usage of tensor bandits in context case was studied. Context versions of algorithms were developed and tested.

Introduction

Multi-armed bandit algorithms — these are algorithms where a decision maker iteratively selects one of multiple fixed choices, with each option represented by an arm. After making a choice a reward is received, and the agent's goal is to maximize the reward sum.

Given: B — a set of n choices, T — the time period over which we maximize the reward.

- $Rd = \sum_{t=1}^{T} r_t$ – Total reward, where r_t is sampled by the environment from the reward distribution of the arm chosen at step t, which is unknown to the agent.
- $Rt = T \cdot \mu^* - Rd$ – Total regret, where $\mu^* = \max_{\text{arm} \in \text{all arms}} \mathbb{E}(\mu_{\text{arm}})$, and μ_{arm} is a random variable corresponding to the reward of an arm.

Objective: Design an algorithm that maximizes the reward (minimizes regret).

Key Algorithms:

- **Vectorized UCB (Upper Confidence Bound)**: Selects arms based on optimistic reward estimates.
- **Epoch Greedy**: Selects the optimal arm at each step by choosing the maximum reward.
- **Tensor Elimination**: Removing arms based on whether their reward estimates fall outside a specified confidence interval.

TensorTrain Algorithm

```
input: dimensions ∈ $\mathbb{N}^n$ - dimension of the reward tensor, ranks ∈ $\mathbb{N}^n$ - dimension of the tensor train decomposition, $T$ - number of steps, $T_e$ - number of exploration steps
For t in range($T_e$)
    Choose random arm
    Update average reward tensor
Tensor completion → estimated reward tensor R
Tensor train decomposition of R
For t in range($T_e$, $T$)
    Finding maximum of R (optima_tt_max)
    Choose optimal arm
    Update R
    if t % update-each == 0:
        Restore R from updated decomposition
        Tensor completion of R
        Tensor Train decomposition of R
```

Algorithm description

1. The `optima_tt_max` algorithm represents tensor elements as a distribution, selecting the k most probable at each step to find the optimal solution.
2. Updating **R** without restoring from tensor train:

Formula:

Let $A, B \in \mathbb{R}^{p_1 \times \dots \times p_n}$ be two tensors, with tensor train cores $C_{A,i} \in \mathbb{R}^{r_{A,(i-1)} \times p_i \times r_{A,i}}$ $C_{B,i} \in \mathbb{R}^{r_{B,(i-1)} \times p_i \times r_{B,i}}$, then cores $C_{D,0} \in \mathbb{R}^{1 \times p_1 \times (r_{A,1} + r_{B,1})}$, $C_{D,i} \in \mathbb{R}^{(r_{A,(i-1)} + r_{B,(i-1)}) \times p_i \times (r_{A,i} + r_{B,i})}$, $C_{D,n} \in \mathbb{R}^{(r_{A,(n-1)} + r_{B,(n-1)}) \times p_n \times 1}$ for $D = A + B$ can be calculated as

$$C_{D,0} = \begin{bmatrix} A_{A,0} & C_{B,0} \end{bmatrix},$$
$$C_{D,i} = \begin{bmatrix} C_{A,i} & 0 \\ 0 & C_{B,i} \end{bmatrix} \quad i = 1, \dots, n-1,$$
$$C_{D,n} = \begin{bmatrix} C_{A,n} & C_{B,n} \end{bmatrix}.$$

Numerical Results

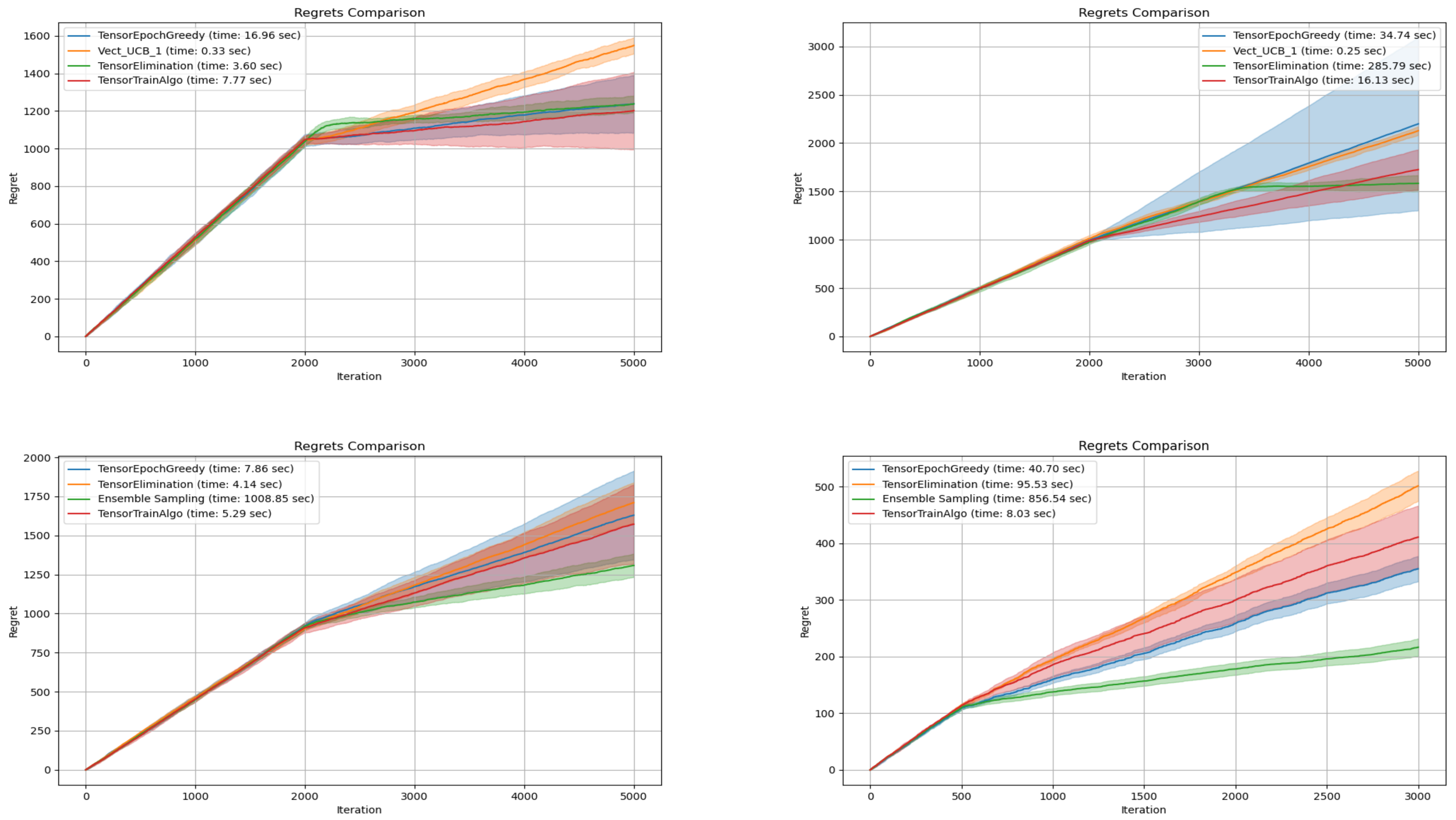

Figure 1:
First row: Five runs were conducted on reward tensors of sizes $10 \times 10 \times 10$ and $5 \times 5 \times 5$ generated from a normal distribution.
Third row 1: Contextual algorithms were run five times on reward tensors of size $5 \times 5 \times 5$, where the first dimension corresponded to the context.
Third row 2: Contextual algorithms were run five times on the SyntheticBanditDataset simulator (Open Bandit dataset), where the context dimension was 3 and the number of arms was 5.

Contextual bandits

To create contextual version of tensor bandits algorithm additional dimensions need to be added to the reward tensor to encode context. The algorithm can be outlined as follows:

- Obtain context from the environment.
- Select the part of the reward tensor corresponding to the given context.
- Choose the optimal arm using the tensor bandits algorithm.

The idea of the Ensemble Sampling algorithm is to initially set a prior distribution, which is then updated through the algorithm's actions. It builds on Thompson Sampling and utilizes Tucker decomposition

Concluding remarks

- This work investigated, implemented, and tested various existing algorithms for solving the tensor multi-armed bandit problem, leading to the development of a new algorithm called TensorTrain, which utilizes low-rank decomposition. Results showed that while TensorTrain is one of the fastest algorithms, Ensemble Sampling outperforms it on contextual task.
- One of future research directions is to explore how changes in the environment during algorithm operations affect their effectiveness, as this frequently occurs in real-world scenarios.

Contact Information

- Implementations of algorithms are available athttps://github.com/horbachmp/TensorBandits