NEURAL THOMPSON SAMPLING

Group No. 12

Problem Motivation

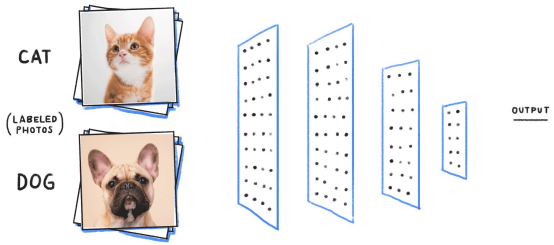
- The **stochastic multi-armed bandit** serves as a crucial model for achieving the optimal balance between **exploration** and **exploitation** in sequential decision-making scenarios.
- Our focus is on a well-researched variant of it, the **contextual bandit**.
- In each round of a contextual bandit, the agent observes a feature vector (the "**context**") for each of the K arms, pulls one of them, and in return receives a **scalar** reward.
- The goal is to **maximize** the cumulative **reward**, or **minimize** the **regret**, in a total of T rounds.

Neural Thompson Sampling

- **Thompson Sampling**, or TS is one of the most effective and widely used techniques. The basic idea is to compute the **posterior distribution** of each arm being optimal for the **present context**, and sample an arm from this distribution.
- TS is often easy to implement, and has found great success in practice.
- The paper proposes a new algorithm, **Neural Thompson Sampling** (NeuralTS), to incorporate TS exploration with neural networks.
- NeuralTS is the first **near-optimal regret bound** for neural network-based Thompson Sampling.. On performing regret analysis for the algorithm, and obtain an $\tilde{\mathcal{O}}(\tilde{d}\sqrt{T})$ **regret**, where
  - **d** is the **effective dimension**
  - p (no of parameters) = d*m + m^2 (L - 2) + m
  - **T** is the number of rounds

Setup

- To transform these classification problems into multi-armed bandits, we build a context feature vector for each arm.
- Obtain the dataset for any classification problem (input : feature vector (modified to get context vector), output label).
- Number of arms in the multiarmed bandit problem would be number of unique labels in the classification problem.
- Given an input feature vector x (dimension d) of a K-class classification problem, we build the context feature vector with dimension kd as: x_1 = (x; 0; ….; 0), x_2 = (0; x;… ; 0), ….., x_k = (0;0;… x).
- Then, the algorithm generates a set of predicted reward following Algorithm 1 and pulls the greedy arm.
- For these classification problems, if the algorithm selects a correct class by pulling the corresponding arm, it will receive a reward as 1, otherwise 0.



Algorithm

1. Initialize the covariance matrix $\text{Set } \mathbf{U}_0 = \lambda \mathbf{I}$
2. For $t = 1, \cdots, T$
   a. Get the context vectors and estimate the mean reward for all arms by forward propagation..
   $$f(\mathbf{x}_{t,k}; \boldsymbol{\theta}_{t-1})$$
   b. Compute the gradient of the mean reward w.r.t parameters $\mathbf{g}(\mathbf{x}; \boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} f(\mathbf{x}; \boldsymbol{\theta})$
   c. For each arm, use the gradients and compute -
   $$\sigma_{t,k}^2 = \lambda \, \mathbf{g}^\top(\mathbf{x}_{t,k}; \boldsymbol{\theta}_{t-1}) \, \mathbf{U}_{t-1}^{-1} \, \mathbf{g}(\mathbf{x}_{t,k}; \boldsymbol{\theta}_{t-1}) / m$$
   d. Sample the reward for each arm from the posterior distribution -
   $$\widetilde{r}_{t,k} \sim \mathcal{N}(f(\mathbf{x}_{t,k}; \boldsymbol{\theta}_{t-1}), \nu^2 \sigma_{t,k}^2)$$
   e. Pull the arm corresponding to maximum sampled reward.
   $$a_t = \text{argmax}_a \, \widetilde{r}_{t,a}$$

Algorithms

f. Calculate the reward corresponding to that arm.

g. Calculate the regret. Add it to the cumulative regret. It measures total mistakes made by the algorithm over the t iteration. Its graph is plotted against t in the next slide.

h. Train the neural network based on the obtained reward as follows:

h.1 Loss in the t^{th} iteration is defined as follows:

h.2 Formula:

$$\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) = \sum_{i=1}^{t} [f(\mathbf{x}_{i,a_i}; \boldsymbol{\theta}) - r_{i,a_i}]^2/2 + m\lambda \|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|_2^2/2.$$

h.3 Obtain the optimal parameters θ that minimize the above loss using gradient descent. (Backward Propagation)

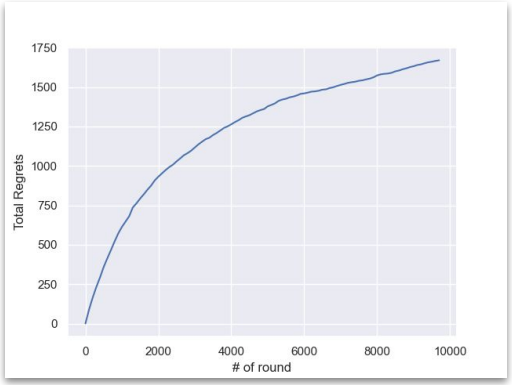
i. Update U using following formula:

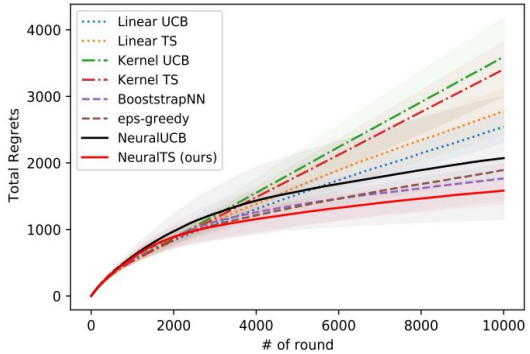
i.1 Formula:

$$\mathbf{U}_t = \mathbf{U}_{t-1} + \mathbf{g}(\mathbf{x}_{t,a_t}; \boldsymbol{\theta}_t)\mathbf{g}(\mathbf{x}_{t,a_t}; \boldsymbol{\theta}_t)^\top/m$$

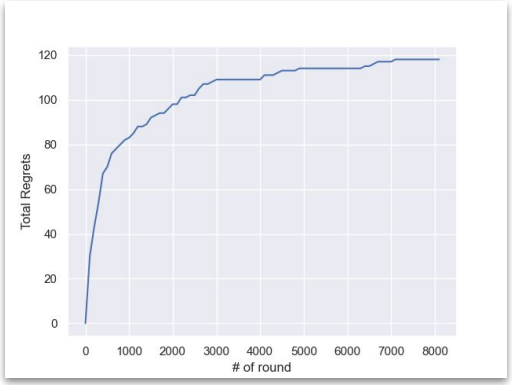
Main results

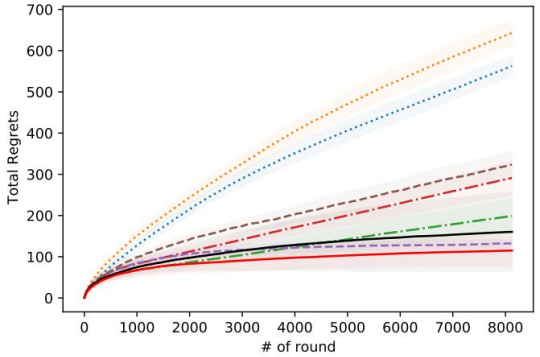
MNIST Dataset





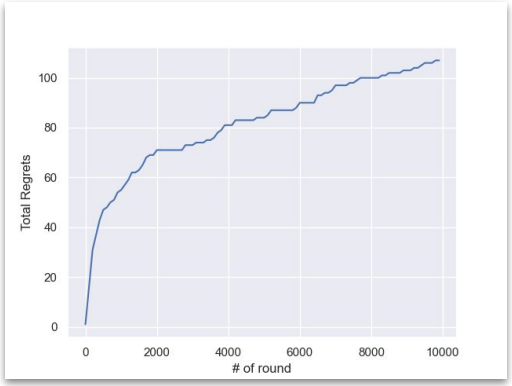
Mushroom Dataset

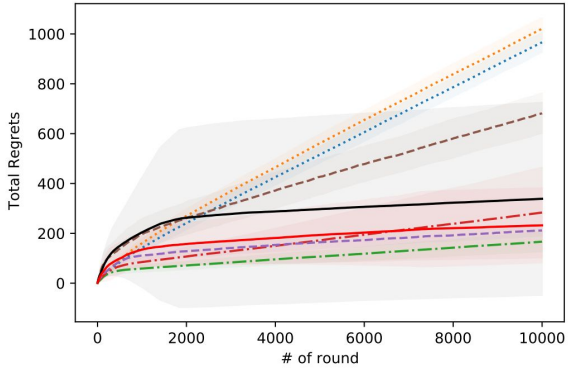




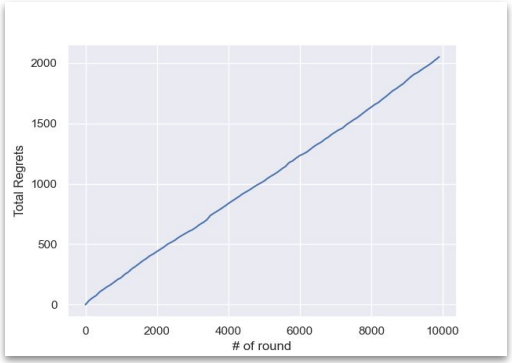
Main results

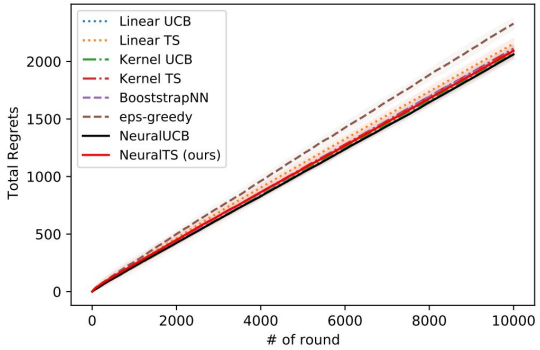
Shuttle Dataset





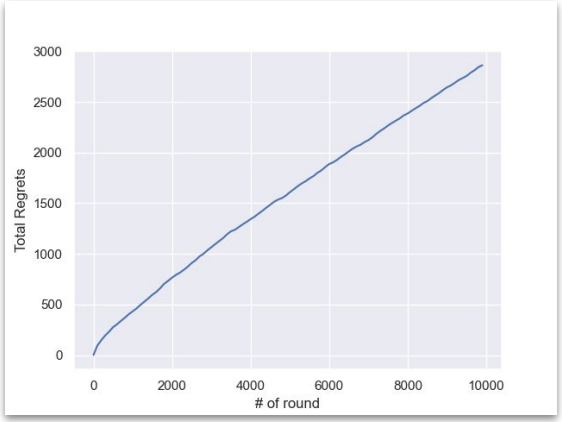
Adult Dataset

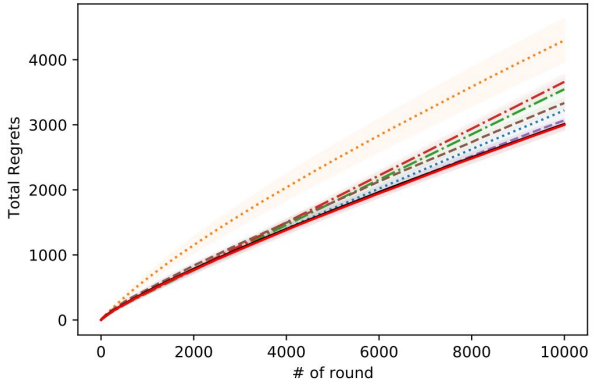




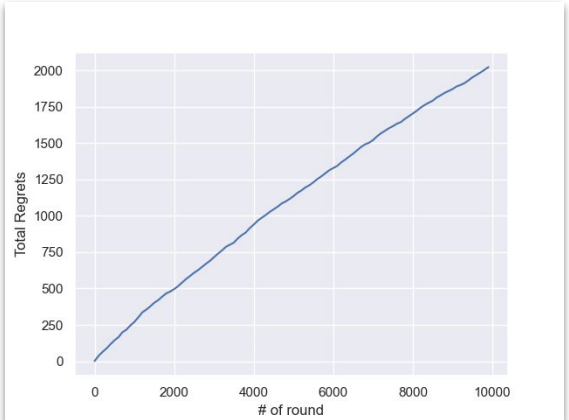
Main results

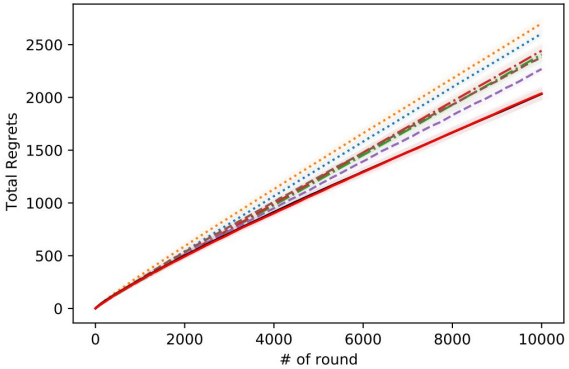
Covertype Dataset





MagicTelescope Dataset





Future Research directions

- NeuralTS needs to perform **multiple gradient descent steps** to train the neural network in each round.
  - analyze the case where NeuralTS only performs one gradient descent step in each round,
  - trade-off between optimization precision and regret minimization.
- Second, when the number of arms is finite, $\tilde{\mathcal{O}}(\sqrt{dT})$ regret has been established for parametric bandits with linear and generalized linear reward functions. It is an open problem how to adapt NeuralTS to achieve the same rate.

Thank You!

extras

- Novelty:
  - differs from NeuralLinear by considering weight uncertainty in all layers
  - from other neural network-based TS implementation by sampling the estimated reward from the posterior (as opposed to sampling parameters).
- regret analysis => O(d root (T)) regret
- several benchmarks tested
- mean => NN approximator
- variance => built on neural target features of corresponding NN
- What is CMAB? (slide 1)
- What is U?
  - Covariance matrix of parameters

Algorithm

- At round t (out of T), the agent observes K contextual vectors of dimenson kxd. Then the agent selects an arm and receives a reward . Our goat is to minimize the following psudeo regret.
- is the optimal arm at round t the has the maximum expected reward. To estimate the unknown reward given a contextual vector x, we use a fully connected neural network f(x;θ) of depth L >= 2, defined recursively by :

$$R_T = \mathbb{E}\left[\sum_{t=1}^{T}(r_{t,a_t^*} - r_{t,a_t})\right]$$

Algorithm

- Loss function
- Given an input feature vector x (dimension d) of a K-class classification problem, we build the context feature vector with dimension kd as: x_1 = (x; 0; ….; 0), x_2 = (0; x;… ; 0), …., x_k = (0;0;… x).

$$\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) = \sum_{i=1}^{t} [f(\mathbf{x}_{i,a_i}; \boldsymbol{\theta}) - r_{i,a_i}]^2/2 + m\lambda\|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|_2^2/2.$$

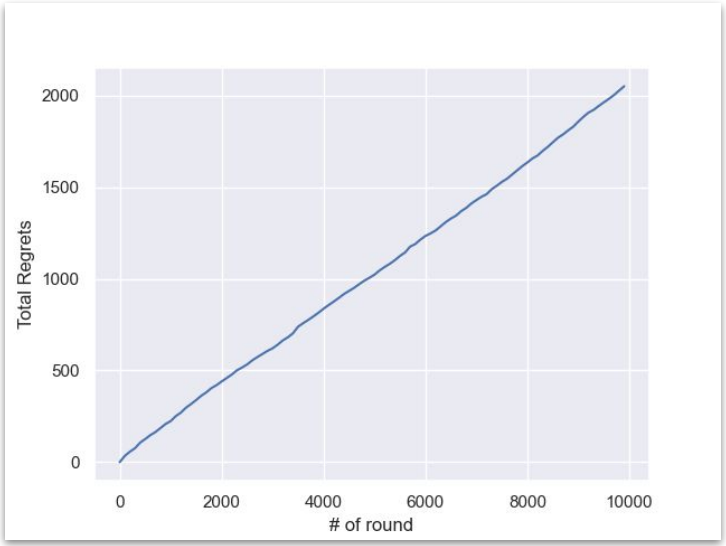
Algorithm (contd)

for $t = 1, \cdots, T$ **do**
  for $k = 1, \cdots, K$ **do**
    $\sigma^2_{t,k} = \lambda \, \mathbf{g}^\top(\mathbf{x}_{t,k}; \boldsymbol{\theta}_{t-1}) \, \mathbf{U}^{-1}_{t-1} \, \mathbf{g}(\mathbf{x}_{t,k}; \boldsymbol{\theta}_{t-1})/m$
    Sample estimated reward $\widetilde{r}_{t,k} \sim \mathcal{N}(f(\mathbf{x}_{t,k}; \boldsymbol{\theta}_{t-1}), \nu^2 \sigma^2_{t,k})$
  end for
  Pull arm a_t and receive reward r_{t,a_t}, where $a_t = \operatorname{argmax}_a \widetilde{r}_{t,a}$
  Set $\boldsymbol{\theta}_t$ to be the output of gradient descent for solving (2.3)
  $\mathbf{U}_t = \mathbf{U}_{t-1} + \mathbf{g}(\mathbf{x}_{t,a_t}; \boldsymbol{\theta}_t)\mathbf{g}(\mathbf{x}_{t,a_t}; \boldsymbol{\theta}_t)^\top/m$
end for

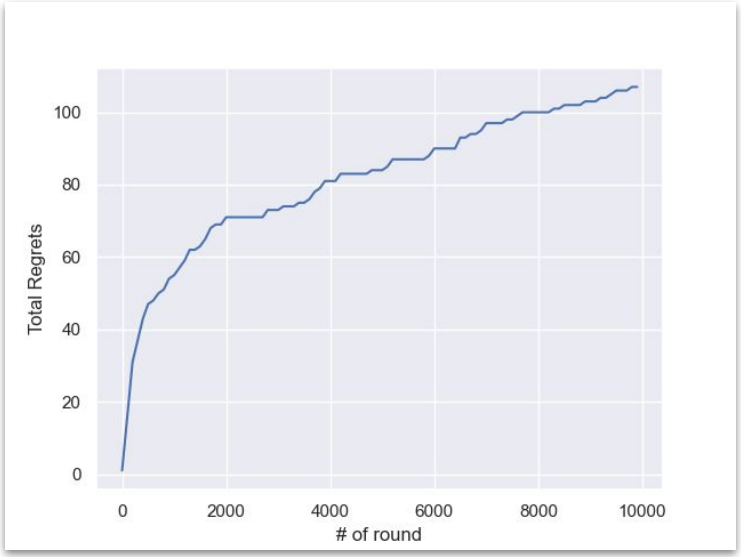
- At each round t, sample reward for each arm from N(f(x(t,k),θ), σ).
- Then select the arm with maximum reward. Receive the reward of that arm.
- Update the parameter θ of the neural network using this reward.
- Update the distribution of the arms.

Main results

Adult

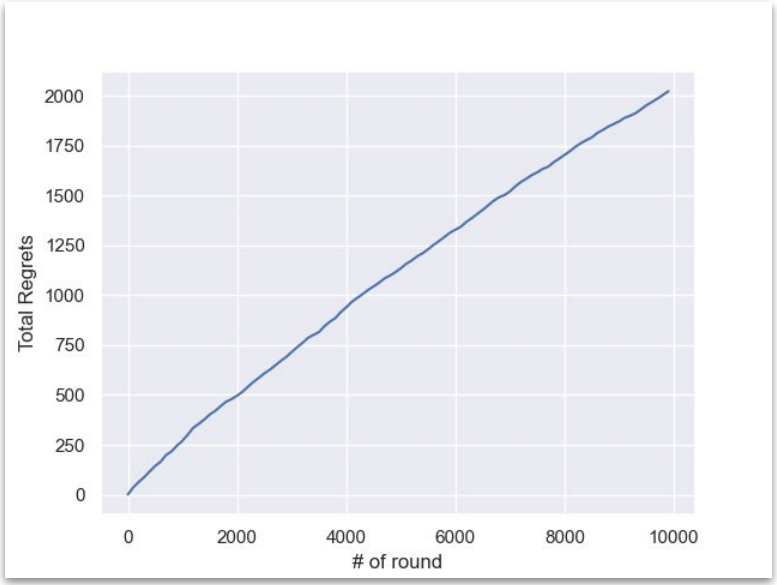


Shuttle

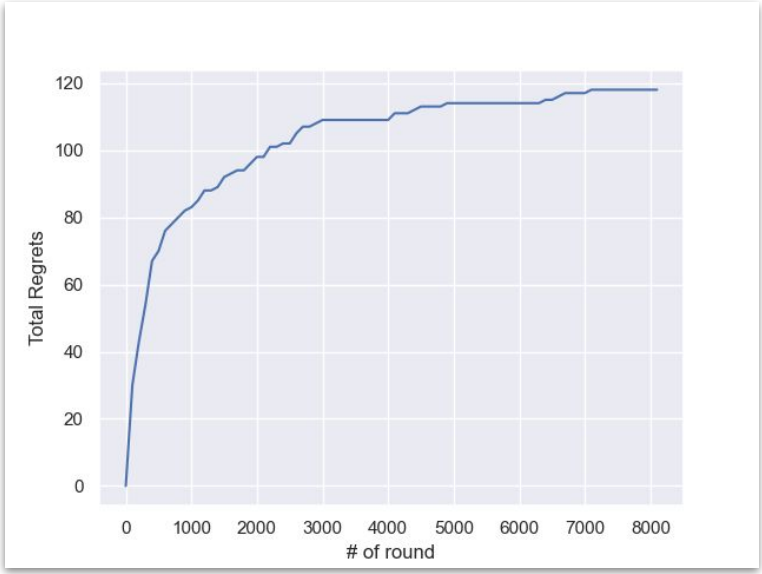


Main results continued…

Magic telescope

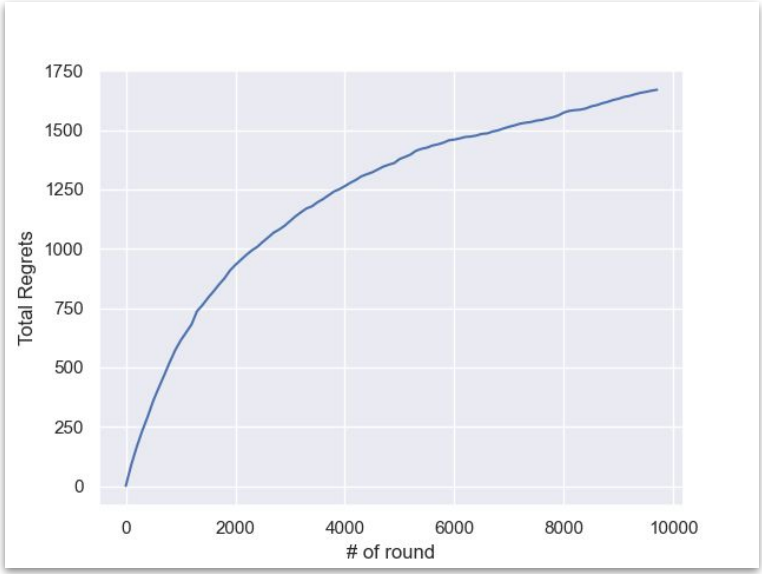


Mushroom



Main results continued…

MNIST



Covertype