Contextual Bandit Comparison

Eelis Virtanen
Center for Data Science
New York University
New York, NY 10012
ev933@nyu.edu

Mrinal Jain
Center for Data Science
New York University
New York, NY 10012
mj2377@nyu.edu

Stephen Roy
Center for Data Science
New York University
New York, NY 10012
stephen.i.roy@nyu.edu

Tim Connor
Center for Data Science
New York University
New York, NY 10012
tfc276@nyu.com

Abstract

This paper aims to reproduce and extend the multi-armed bandit policy findings by Cortes [3]. In particular, this paper looks at the specific situations where rewards are Bernoulli distributed and fit separately using a binary classifier for each bandit arm. This paper shows that the results achieved in the initial paper did not hold. Specifically, the recommended ContextualAdaptiveGreedy algorithm was not the best performer in our trials. Furthermore, our extensions show significant performance improvements through careful hyperparameter tuning and oracle/imputation algorithm selection.

1 Introduction

Contextual multi-armed bandits aim to solve the problem of choosing a single, mutually exclusive action among competing bandits (actions) based upon a given context (set of features). Each bandit provides a reward, and the goal is to maximize the expected rewards while only observing the reward for the chosen action.

This is a typical problem within reinforcement learning - we need to choose between exploiting bandits that we know give high rewards versus exploring new bandits that may provide even higher rewards than the ones we have already explored for a given context.

This work aims first to replicate and then extend specific contextual multi-armed policies introduced in recent years. In particular, we will be investigating a paper by Cortes [3] which extended standard multi-armed bandit algorithms (MAB) without context to situations where a context was also available. The author shows that specific algorithms, such as the ContextualAdaptiveGreed algorithm, are able to achieve excellent performance. We take a two-pronged approach to validate the robustness of the findings in [3]. First, we compare our own implementation of the best performing algorithm [1] against his own. Second, we extend the paper's findings by experimenting extensively with hyperparameter selection and other design choices to evaluate their impact on the results.

[1] ContextualAdaptiveGreedy with a Beta Prior

2 Problem Definition

2.1 Terminology

The typical multi-armed bandit is defined as follows: we start with an agent who wants to choose action a^t among k bandits in order to maximize their expected reward $\mathbf{E}(R)$ in each round t. The agent observes multiple rounds. At the beginning of each round, the world samples a context X of dimensionality m and a reward vector r_k^t which provides a reward for each bandit. We want the agent to learn is a policy that will allow them to choose the actions that give the highest rewards. Intuitively, we can imagine that initially, the agent has little idea about which action to choose, but over time they will learn which action gives a high reward for a given context. Hence, the historical context, action and reward tuples will form the basis for a more intelligent way of deciding subsequent actions as the rounds proceed.

A supervised machine learning imputation model (referred to as an oracle) fits rewards and covariates for each arm separately. Hence, if we have 10 arms, we would fit 10 different models. However, machine learning models do not work if we have only observed one type of label for a given model. Consequently, smoothing is a common choice to tackle potential cold-start problems. In this context, smoothing means shifting probability mass from the labels we observe to the labels we have not yet observed. However, for bandit situations with many arms, smoothing can take too much probability mass away. In these cases, an initial Beta prior can be used before switching to a contextual bandit policy as an alternative to smoothing, thus drawing samples from a Beta distribution for each arm and choosing the highest one as the action.

2.2 Technical Details

Various types of multi-armed bandits exist. They differ primarily along the dimensions of time, discounting (we place less weight on earlier rounds), online versus batch training, and the type of reward (can be binary, continuous, etc.). The paper we are investigating used the following bandit scenario, which we will use as well:

- Bernoulli distributed rewards in $\{0, 1\}$.
- Long-term horizon. In practice, the size of the dataset (which provides the number of rounds) determines the time horizon.
- No discounting – performance is measured by maximizing the cumulative rewards over all rounds.
- Each oracle is fit after 50 rounds using full batches.
- The data was shuffled 10 times with the average performance as the final result.

2.3 Project Goals

1. Validate the top-performing model from [3].
2. Reproduce the findings for various contextual bandit algorithms defined in [3].
3. Extend the findings through hyperparameter tuning and oracle selection evaluation.
4. Identify promising areas to improve or extend this project with future work.

3 Experimental Evaluation

3.1 Data

The data used consists of multi-label classification datasets from the Extreme Classification Repository (Bhatia et al. [1]). Each label has been converted into a bandit and a binary reward is given depending on whether one of the correct labels is predicted.

	Obs.	Feats.	Labels	Lab./obs.	Obs/lab.	Most common
Bibtex	7,395	1,836	159	2.4	111.71	14.09%
Delicious	16,105	500	983	19.02	311.61	40.33%
Mediamill	43,907	120	101	4.383	1902.16	77.14%
Mediamill Reduced	43,907	120	96	2.07	945.1	17.56%

Table 1: Overview of the data used to validate Cortes [3] results.

3.1.1 BibTeX

The Bibtex dataset resulted from the pre-processing applied by Katakis et al. [4] on data for automated tag recommendation. The data has 1836 features that correspond to various details of the bibtex items (like journal, bibtex title, abstract) and 159 labels that were the tags assigned by the users. Notably, there was no dominant label in this dataset, and the rewards distribution suggested a more balanced scenario.

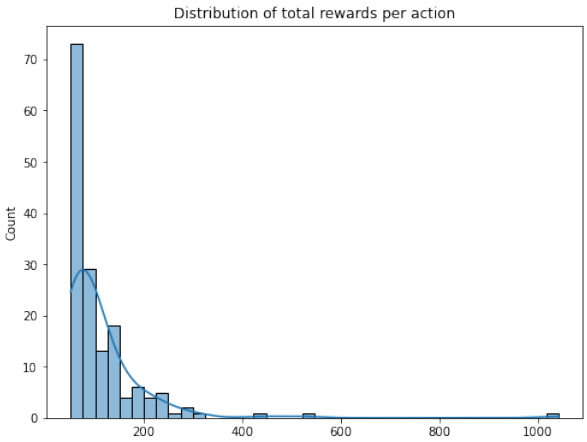
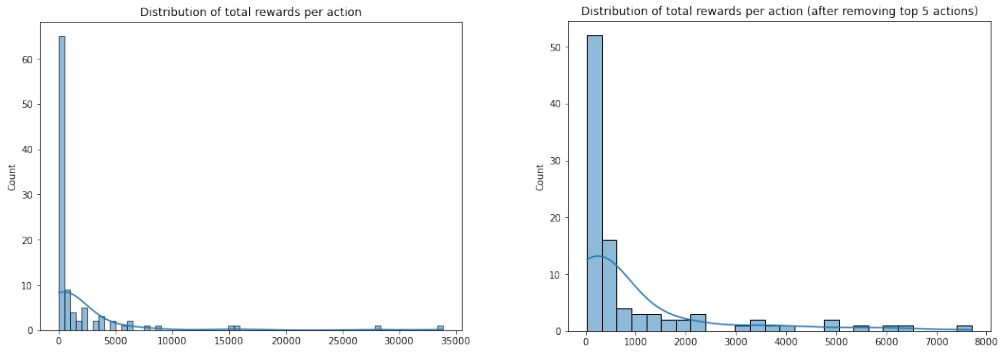


Figure 1: BibTeX data with a relatively balanced reward distribution (with only a few actions having more than 500 associated observations)

3.1.2 Mediamill

The Mediamill dataset originates from the 2006 Snoek et al. [5] video annotation challenge. Mediamill is unique among the datasets tested because the majority of its observations belong to a few dominant arms.

Although not formally published, Cortes [3] adapted the Mediamill-Reduced dataset from the original by dropping the 5 most common labels. This processing dramatically altered the distribution of the dataset, allowing for very different simulation scenarios.



(a) Full: most observations belong to few actions. (b) Reduced: more balanced observation spread.

Figure 2: MediaMill and MediaMill Reduced Datasets

3.1.3 Delicious

Tsoumakas et al. [6] extracted this dataset from its titular social bookmarking website, containing textual data of the web pages and their associated tags. After applying extensive filtering, they ended up with a 983 tags that correspond to the labels in the dataset. Then, the authors used feature ranking methods to obtain a (reduced) vocabulary of 500 words from the respective web pages, denoting the features.

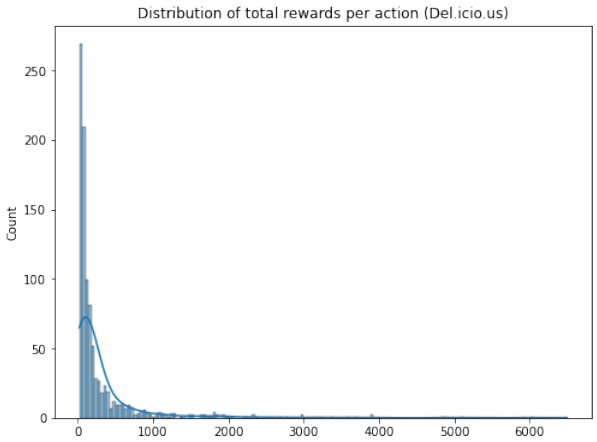


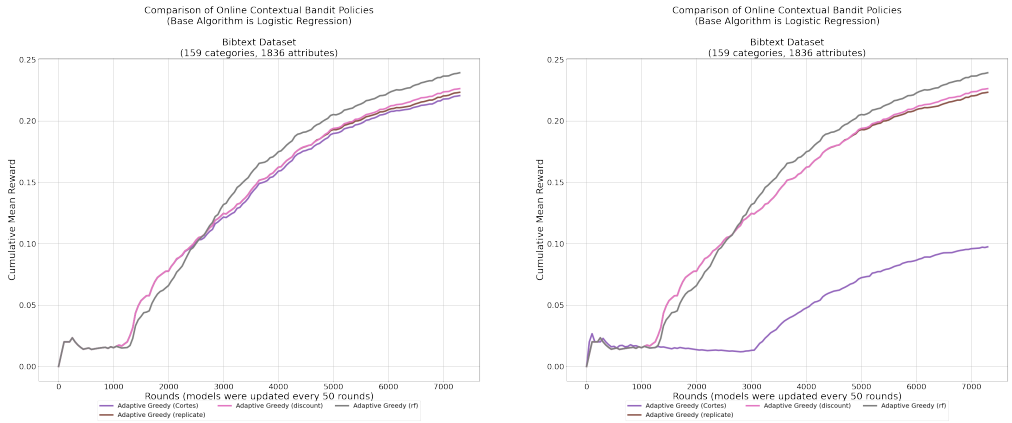
Figure 3: The observations were distributed quite evenly, except for some actions, that were associated to almost 30% of the observations, skewing the overall distribution.

3.2 Replication

We attempted to replicate the findings from Cortes [3], both by manually reimplementing the ContextualAdaptiveGreedy algorithm identified as the overall best-performing policy and by using the python package maintained by the author (contextualbandits, Cortes [2]).

3.2.1 ContextualAdaptiveGreedy Validation

Reimplementing the ContextualAdaptiveGreedy algorithm yielded nearly identical results to the python-only version available in Cortes [2]. However, by default, an optimized cython implementation is used by the current version of the package. This produces inconsistent results, as seen in the below figure, where the purple (bottom) line indicates the algorithm implemented by the package. As of writing this paper, it is still uncertain whether the inconsistent behavior results from the cython optimizations or other package dependencies.



(a) python only, contextualbandits==0.1.8.5 (b) cython, contextualbandits==0.3.13

Figure 4: ContextualAdaptiveGreedy - Package Version Inconsistency

In addition to validating the algorithm performed as described in the paper, two additional models were created to test the impact of discounting and replacing the classifier algorithm. While our extension results significantly change the bandit scenario described above, they generally yielded better results for the contexts tested. Furthermore, while discounting early actions only seemed to improve performance mildly, substituting the base learner (from logistic to random forest) produced significant changes (both better and worse depending on the context selected).

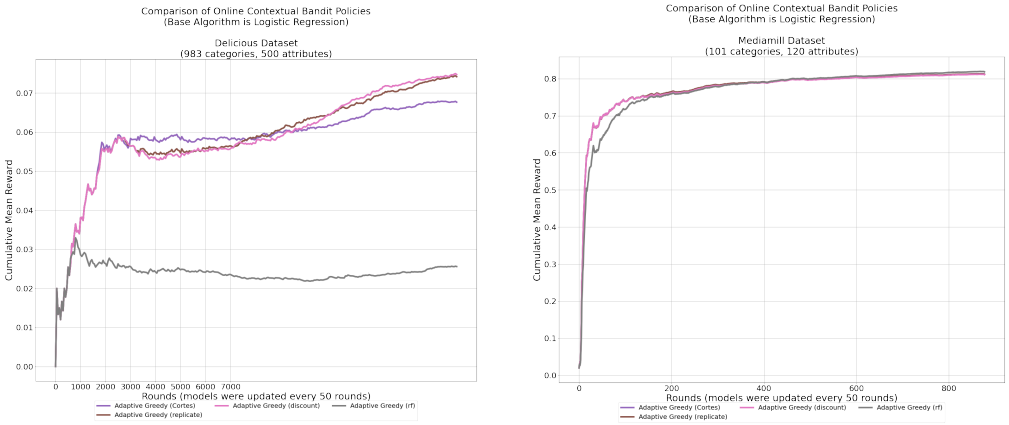


Figure 5: ContextualAdaptiveGreedy Extensions

3.2.2 Reproducibility of Paper Results

To reproduce the other algorithm results, we solely leveraged the author's package and focused on batched fitting performance. Even still, we see some major differences from the original paper. For example, Adaptive Greedy is no longer a stand-out performer on the Bibtex and Delicious datasets. Instead, the bootstrapped Upper Confidence Bound (UCB) algorithm tends to perform best. This finding is also supported by the package documentation, which offers the revised assessment: "BootstrappedUCB is the safest bet for online methods, and OffsetTree is the safest bet for off-policy methods."

While differences in model performance are expected due to randomness in both data shuffling and action selection, the magnitude of the difference in results suggests that something was systematically

different between our experiments and the paper. Our best guess is that we did not use the same parameters for the MAB-First algorithm since the paper did not give the exact configuration.[2]

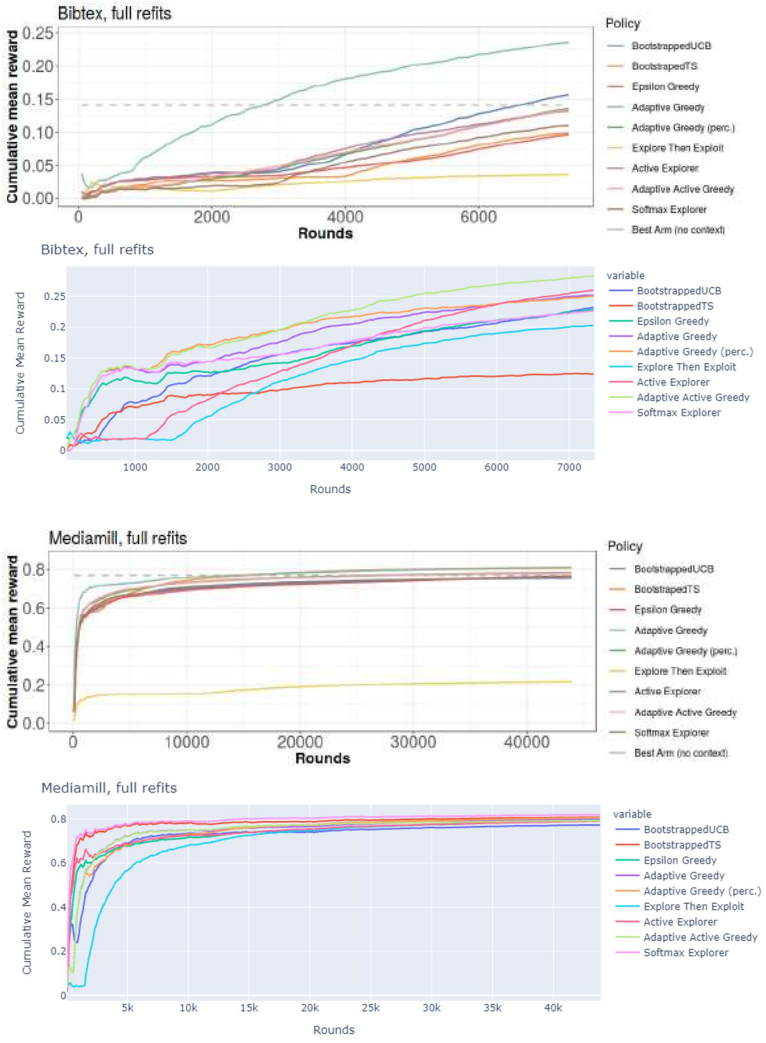


Figure 6: Replication on Bibtex & Mediamill (white charts are copied from paper, gray are replicated).

[2]We used the default configuration from `https://github.com/david-cortes/contextualbandits/blob/master/example/online_contextual_bandits.ipynb`, hoping that it would match the original paper

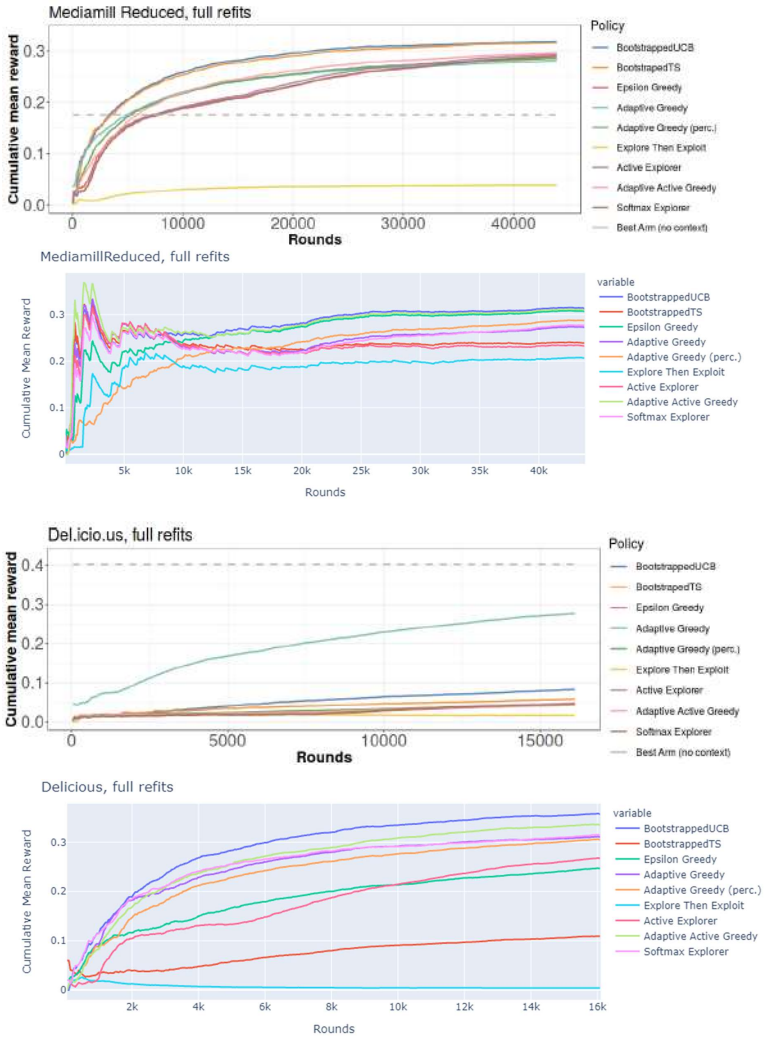


Figure 7: Replication on Mediamill Reduced & Delicious (white are copied directly from paper, gray are replicated).

3.3 Extensions

3.3.1 Discounting

One of the extensions was to use discounting, such that earlier examples get discounted more when fitting the base model for a specific bandit. The original paper chose not to use discounting since the focus was on the long-term performance, for which discounting does not necessarily lend itself to very well naturally, and choosing a discount rate requires some tuning. However, this paper chose to implement discounting anyway with the intuition that the initial examples are still probably the

least useful so we want them to be less important as seen by the fitting algorithm since the logged data will be biased by the bad initial policies. Hence, a function that penalizes early examples quite heavily but then saturates fast makes intuitive sense. This paper used $x/(x+a)$ where a larger a would result in more initial discounting.

As seen above in 4a, we see that discounting slightly improved performance under the Bibtex dataset. However, 5 shows negligible to no improvement from discounting performance on the Delicious or Mediamill datasets. This would seem to indicate discounting yields the best results when applied to datasets with more balanced observation/action distributions.

3.3.2 Computational Complexity and Experiment Design

Refitting the batch models every 50 rounds leads to an unnecessary computational burden during experimentation. It took 3.5 hours to run all 8 of the baseline models on Mediamill when refitting this frequently. We reduce this burden by refitting every time the training set grows by 7.5% or 50 rounds, whichever is longer.[3] This drastically reduces the computational burden while having a negligible impact on the performance of each algorithm. This update is significant for the sampling algorithms since they refit additional models on bootstrapped samples.[4] 8 illustrates the relative slowness of the sampling algorithms, while 8 highlights the benefits of change the refitting frequency. We note that less frequent batching may be more detrimental in other domains, especially those where arms are added/dropped or the payoff to action a is time-dependent.

		# of Batches			Runtime (Min)*			Mean Reward		
	Obs	50	7.5%	% Dif	50	7.5%	% Dif	50	7.5%	% Dif
Bibtex	7,395	147	46	-68.7%	1.2	0.2	-82.5%	0.23	0.23	2.3%
Delicious	16,105	322	57	-82.3%	4.2	0.6	-84.8%	0.25	0.33	34.3%
Mediamill	43,907	878	71	-91.9%	4.7	0.4	-90.4%	0.79	0.79	-0.5%
Mm-Reduce	43,907	878	71	-91.9%	5.0	0.4	-92.6%	0.31	0.30	-2.0%

**Runtime varies by algorithm, epsilon greedy is used for benchmarking*

Figure 8: Refitting frequency can be reduced drastically without a detrimental impact on performance

3.3.3 Hyperparameter Tuning

Cortes [3] acknowledges that one shortcoming of the paper is the lack of hyperparameter tuning. We performed a grid search on the hyperparameters for all of the batch algorithms in the original paper to address this shortcoming. The results demonstrated that hyperparameter selection has a drastic impact on the performance of the proposed contextual bandit algorithms. The impact is most pronounced and consistent for the epsilon greedy algorithm, where higher explore probabilities lead to lower cumulative reward on average.[5] The active explorer algorithm also sees a substantial impact from changing the threshold parameter, but the impact of adjustments varies across the datasets. A higher threshold leads to better performance on the Bibtex dataset, worse performance on the Mediamill-Reduced dataset, and no consistent directional impact on the Delicious or Mediamill. A few of our algorithms are quite robust to the parameters that we included in our gridsearch (e.g., see figures 15, 12 & 14). The impact of certain parameter decisions, such as the number of samples in UCB or Thompson Sampling, did not have a meaningful impact on mean performance but seemed to reduce the variance of our findings.[6]

[3] After 1000 rounds, the next update takes place after 1000*0.075=75 rounds

[4] In Cortes [3] 10 samples are taken per batch period

[5] Higher explore probability can lead to better performance in earlier rounds

[6] Due to computational limitations, we did not run multiple samples to validate this theory

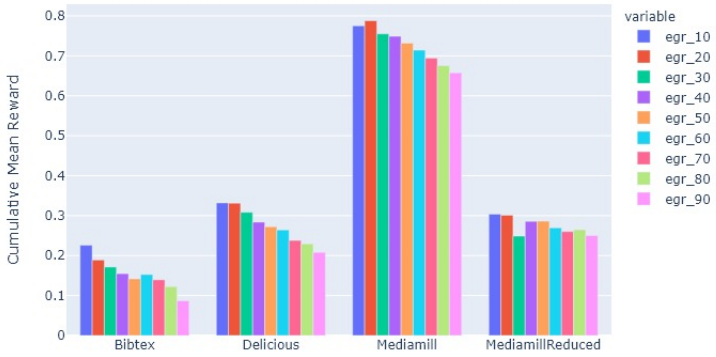


Figure 9: Adjusting explore probability in the Epsilon Greedy algorithm (Baseline=20%)

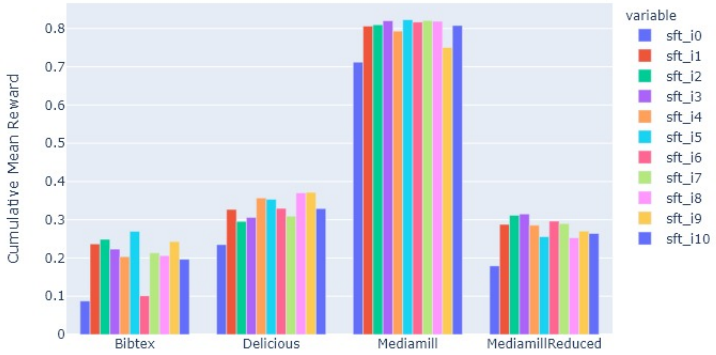


Figure 10: Adjusting the multiplier in the Softmax Explorer algorithm (Baseline=1)

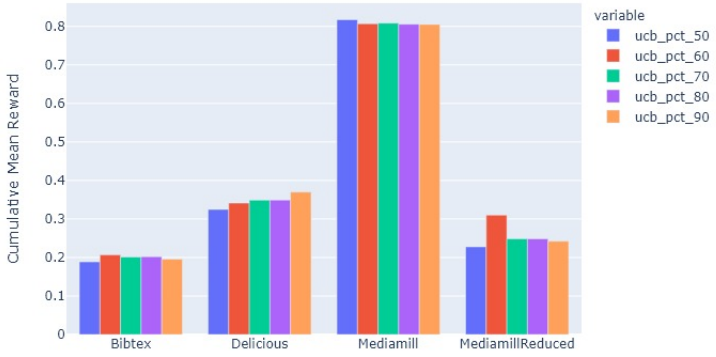


Figure 11: Adjusting the percentile cut-off in the Bootstrapped UCB Algorithm (Baseline=80%)

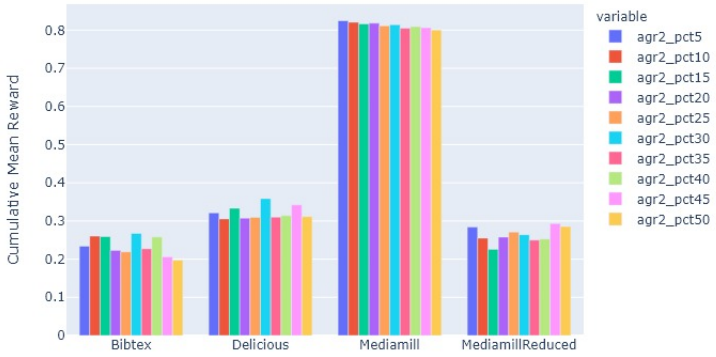


Figure 12: Adjusting the percentile cut-off in the Adaptive Greedy (v2) Algorithm (Baseline=30%)

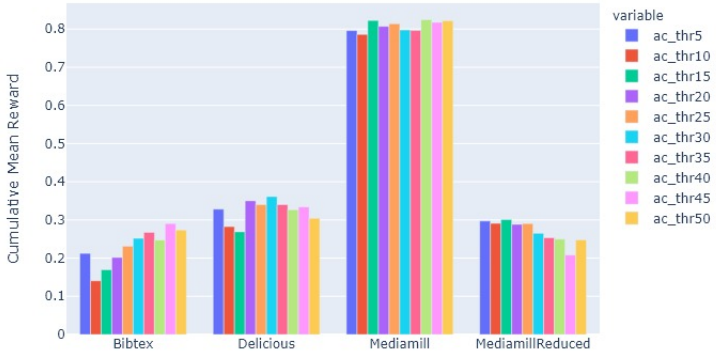


Figure 13: Adjusting explore probability in the Active Explorer algorithm (Baseline=15%)

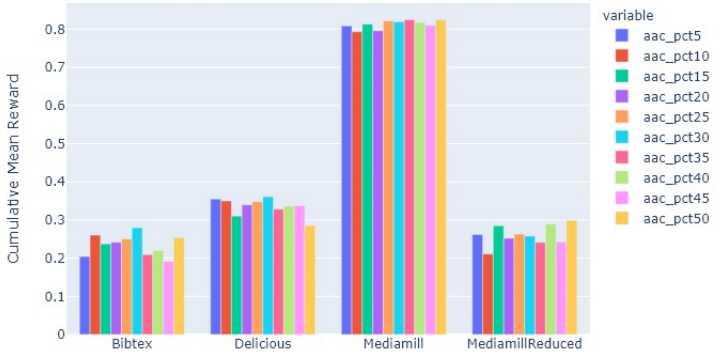


Figure 14: Adjusting the percentile cut-off in the Active Adaptive Greedy algorithm (Baseline=30%)

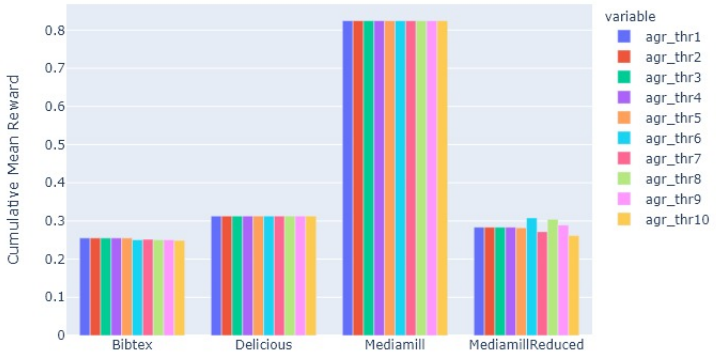


Figure 15: Adjusting the threshold in the Adaptive Greedy (Baseline=$\frac{1}{2\sqrt{\#ofactions}}$, we experiment with $\frac{n}{\#ofactions}$, where n is listed in the legend labels

3.3.4 Base Models

Each contextual bandit algorithm leverages predictions from a classification oracle to determine which actions to take. The original paper, as well as all previous hyperparameter exhibits, uses logistic regression as the oracle. We attempted to improve performance by using the Random Forest and Gradient Boosted Decision Trees. 3-fold validation was used to select hyperparameters for both algorithms during each refit. While this strengthens our confidence in the findings, we would advocate for a larger grid in a production setting. We only experiment with different base models on the Bibtex dataset due to the increased computational intensity of both the tree-based algorithms, and the hyperparameter tuning process. We also exclude the sampling algorithms from this section due to computational constraints. We also did not experiment with different base models in the Active Explorer and Active Adaptive Greedy, since we do not have access to the gradient of the tree-based models. We saw minor improvements for 3 of 4 algorithms when using random forest, while gradient-boosting decision trees led to an inferior result.

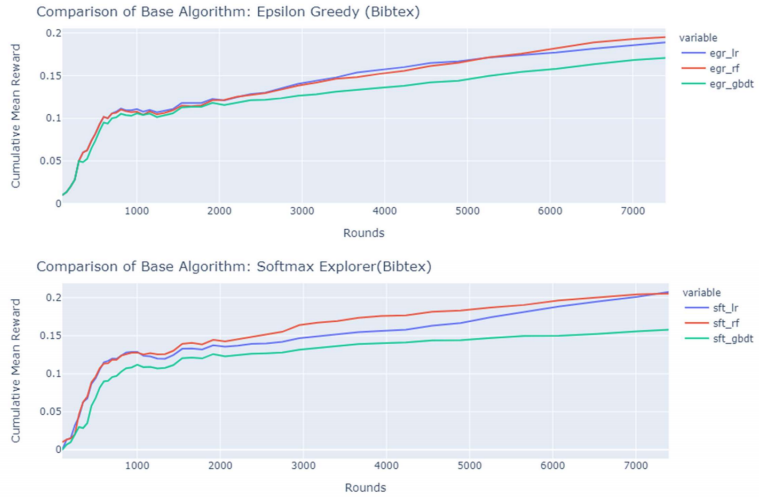


Figure 16: Comparison of different base models (e.g. oracles) part 1)

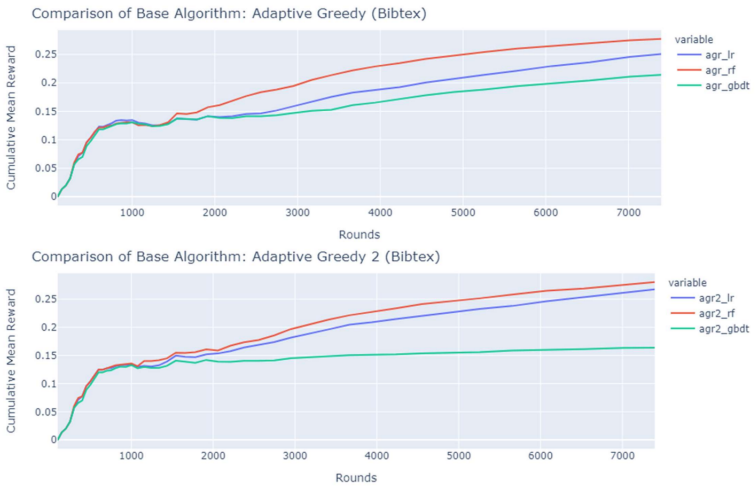


Figure 17: Comparison of different base models (e.g. oracles) part 2)

4 Conclusion

This paper shows that the results achieved in the initial paper did not hold. Specifically, the recommended ContextualAdaptiveGreedy algorithm was not the best performer in our trials. Therefore, in line with the updated contextualbandits documentation we recommend using BootstrappedUCB for the most robust performance on basic contextual settings. Furthermore, our extensions demonstrate that discounting is helpful in balanced contexts and, significant performance improvements can be gained through careful hyperparameter tuning and oracle/imputation algorithm selection.

5 Areas for Future Work

- Use of additional multi-label datasets; either from Bhatia et al. [1] or other domains where online algorithms would be benificial (e.g., stock-trading, recommendation, etc.)
- Alternative methods for priors/smoothing.
- Further testing of different oracle/imputation models.
- Compare different methods of decaying threshold.
- Use distance metric in feature space as a heuristic to guide exploration.

6 Github

https://github.com/MrinalJain17/ml-tools-project

7 Appendix

All algorithm definitions are copied from [3]

Algorithm 1 MAB-first

Inputs const. a, b, threshold m, contextual bandit policy π_k, covariates $\mathbf{x}$
Output score for arm $\hat{r}_k$

if $|\{r \in \mathcal{R}_k \mid r = 0\}| < m$ or $|\{r \in \mathcal{R}_k \mid r = 1\}| < m$ **then**
    Sample $\hat{r}_k \sim \text{Beta}(a + |\{r \in \mathcal{R}_k | r = 1\}|,\ b + |\{r \in \mathcal{R}_k | r = 0\}|)$
else
    Set $\hat{r}_k = \pi_k(x)$
return $\hat{r}_k$

Algorithm 2 Epsilon-Greedy

Inputs probability $p \in (0, 1]$, decay rate $d \in (0, 1]$, oracles $\hat{f}_{1:k}$

for each succesive round t with context $\mathbf{x}^t$ **do**
    With probability $(1 - p)$:
        Select action $a = \text{argmax}_k\, \hat{f}_k(\mathbf{x}^t)$
    Otherwise:
        Select action a uniformly at random from 1 to k
    Update $p := p \times d$
    Obtain reward r_a^t, Add observation $\{\mathbf{x}^t, r_a^t\}$ to the history for arm a
    Update oracle $\hat{f}_a$ with its new history

Algorithm 3 Explore-Then-Exploit

Inputs breakpoint t_b, oracles $\hat{f}_{1:k}$

for each succesive round t with context $\mathbf{x}^t$ **do**
    if $t < t_b$ **then**
        Select action a uniformly at random from 1 to k
    else
        Select action $a = \text{argmax}_k\, \hat{f}_k(\mathbf{x}^t)$
    Obtain reward r_a^t, Add observation $\{\mathbf{x}^t, r_a^t\}$ to the history for arm a
    Update oracle $\hat{f}_a$ with its new history

Algorithm 4 SoftmaxExplorer

Inputs oracles $\hat{f}_{1:k}$, multiplier m, inflation rate i

for each succesive round t with context $\mathbf{x}^t$ **do**
  Sample action $a \sim \text{Mult}(\text{softmax}(m \times \text{sigmoid}^{-1}(\hat{f}_1(\mathbf{x}^t), ..., \hat{f}_k(\mathbf{x}^t))))$
  Update $m := m \times i$
  Obtain reward r_a^t, Add observation $\{\mathbf{x}^t, r_a^t\}$ to the history for arm a
  Update oracle $\hat{f}_a$ with its new history

Algorithm 5 BootstrappedUCB

Inputs number of resamples m, percentile p, oracles $\hat{f}_{1:k,1:m}$

for each succesive round t with context $\mathbf{x}^t$ **do**
  for arm q in 1 to k **do**
    Set $\hat{r}_q^{ucb} = \text{Percentile}_p\{\hat{f}_{q,1}(\mathbf{x}^t), ..., \hat{f}_{q,m}(\mathbf{x}^t)\}$
  Select action $a = \text{argmax}_q \hat{r}_q^{ucb}$
  Obtain reward r_a^t, Add observation $\{\mathbf{x}^t, r_a^t\}$ to the history for arm a
  for resample s in 1 to m **do**
    Take bootstrapped resample $\mathbf{X}_s, \mathbf{r}_s$ from $\mathcal{X}_a, \mathcal{R}_a$
    Refit $\hat{f}_{a,s}$ to this resample

Algorithm 6 OnlineBoostrappedUCB

Inputs number of resamples m, percentile p, oracles $\hat{f}_{1:k,1:m}$

for each succesive round t with context $\mathbf{x}^t$ **do**
  for arm q in 1 to k **do**
    Set $\hat{r}_q^{ucb} = \text{Percentile}_p\{\hat{f}_{q,1}(\mathbf{x}^t), ..., \hat{f}_{q,m}(\mathbf{x}^t)\}$
  Select action $a = \text{argmax}_q \hat{r}_q^{ucb}$
  Obtain reward r_a^t, Add observation $\{\mathbf{x}^t, r_a^t\}$ to the history for arm a
  for resample s in 1 to m **do**
    Sample observation weight $w \sim \text{Gamma}(1, 1)$
    Update $\hat{f}_{a,s}$ with the new observation $\{\mathbf{x}^t, r_a^t\}$ with weight w

Algorithm 7 BootstrappedTS

Inputs number of resamples m, oracles $\hat{f}_{1:k,1:m}$

for each succesive round t with context $\mathbf{x}^t$ **do**
  for arm q in 1 to k **do**
    Select resample s uniformly at random from 1 to m
    Set $\hat{r}_q^{ts} = \hat{f}_{q,s}(\mathbf{x}^t)$
  Select action $a = \text{argmax}_q \hat{r}_q^{ts}$
  Obtain reward r_a^t, Add observation $\{\mathbf{x}^t, r_a^t\}$ to the history for arm a
  for resample s in 1 to m **do**
    Take bootstrapped resample $\mathbf{X}_s, \mathbf{r}_s$ from $\mathcal{X}_a, \mathcal{R}_a$
    Refit $\hat{f}_{a,s}$ to this resample

Algorithm 8 OnlineBootstrappedTS

Inputs number of resamples m, oracles $\hat{f}_{1:k,1:m}$

for each succesive round t with context $\mathbf{x}^t$ **do**
  for arm q in 1 to k **do**
    Select resample s uniformly at random from 1 to m
    Set $\hat{r}_q^{ts} = \hat{f}_{q,s}(\mathbf{x}^t)$
  Select action $a = \text{argmax}_q \hat{r}_q^{ts}$
  Obtain reward r_a^t, Add observation $\{\mathbf{x}^t, r_a^t\}$ to the history for arm a
  for resample s in 1 to m **do**
    Sample observation weight $w \sim \text{Gamma}(1, 1)$
    Update $\hat{f}_{a,s}$ with the new observation $\{\mathbf{x}^t, r_a^t\}$ with weight w

Algorithm 9 ContextualAdaptiveGreedy

Inputs threshold $z \in (0,1)$, decay rate $d \in (0,1]$, oracles $\hat{f}_{1:k}$

for each succesive round t with context $\mathbf{x}^t$ **do**
  if $\max_k \hat{f}_k(\mathbf{x}^t) > z$ **then**
    Select action $a = \text{argmax}_k \hat{f}_k(\mathbf{x}^t)$
  else
    Select action a uniformly at random from 1 to k
  Update $z := z \times d$
  Obtain reward r_a^t, Add observation $\{\mathbf{x}^t, r_a^t\}$ to the history for arm a
  Update oracle $\hat{f}_a$ with its new history

Algorithm 10 ContextualAdaptiveGreedy2

Inputs window size m, initial threshold z_0, decay $d \in (0,1]$, oracles $\hat{f}_{1:k}$

for each succesive round t with context $\mathbf{x}^t$ **do**
  if $t = 1$ **then**
    Set $z = z_0$
  Set $\hat{r}_t = \max_k \hat{f}_k(\mathbf{x}^t)$
  if $\hat{r}_t > z$ **then**
    Select action $a = \text{argmax}_k \hat{f}_k(\mathbf{x}^t)$
  else
    Select action a uniformly at random from 1 to k
  if $t \geq m$ **then**
    Update $z := \text{Percentile}_p\{\hat{r}_t, \hat{r}_{t-1}, ..., \hat{r}_{t-m+1}\}$
    Update $p := p \times d$
  Obtain reward r_a^t, Add observation $\{\mathbf{x}^t, r_a^t\}$ to the history for arm a
  Update oracle $\hat{f}_a$ with its new history

Algorithm 11 ActiveExplorer

Inputs probability p, oracles $\hat{f}_{1:k}$, gradient functions for oracles $g_{1:k}(\mathbf{x}, r)$

for each succesive round t with context $\mathbf{x}^t$ **do**
  With probability p:
    Select action $a = \text{argmax}_k \hat{f}_k(\mathbf{x}^t)$
  Otherwise:
    for arm q in 1 to k **do**
      Set $z_q = (1 - \hat{f}_q(\mathbf{x}^t))\|g_q(\mathbf{x}^t, 0)\| + \hat{f}_q(\mathbf{x}^t)\|g_q(\mathbf{x}^t, 1)\|$
    Select action $a = \text{argmax}_k z_k$
  Obtain reward r_a^t, Add observation $\{\mathbf{x}^t, r_a^t\}$ to the history for arm a
  Update oracle $\hat{f}_a$ with its new history, along with $\hat{g}_a$

Algorithm 12 ActiveAdaptiveGreedy

Inputs window size m, initial threshold z_0, decay rate $d \in (0,1]$, oracles $\hat{f}_{1:k}$, gradient functions for oracles $g_{1:k}(\mathbf{x}, r)$

for each succesive round t with context $\mathbf{x}^t$ **do**
  if $t = 1$ **then**
    Set $z = z_0$
  Set $\hat{r}_t = \max_k \hat{f}_k(\mathbf{x}^t)$
  if $\hat{r}_t > z$ **then**
    Select action $a = \text{argmax}_k \hat{f}_k(\mathbf{x}^t)$
  else
    for arm q in 1 to k **do**
      Set $z_q = (1 - \hat{f}_q(\mathbf{x}^t))\|g_q(\mathbf{x}^t, 0)\| + \hat{f}_q(\mathbf{x}^t)\|g_q(\mathbf{x}^t, 1)\|$
    Select action $a = \text{argmax}_k z_k$
  if $t \geq m$ **then**
    Update $z := \text{Percentile}_p\{\hat{r}_t, \hat{r}_{t-1}, ..., \hat{r}_{t-m+1}\}$
  Obtain reward r_a^t, Add observation $\{\mathbf{x}^t, r_a^t\}$ to the history for arm a
  Update oracle $\hat{f}_a$ with its new history, along with $\hat{g}_a$

References

[1] K. Bhatia, K. Dahiya, H. Jain, P. Kar, A. Mittal, Y. Prabhu, and M. Varma. The extreme classification repository: Multi-label datasets and code, 2016. URL `http://manikvarma.org/downloads/XC/XMLRepository.html`.

[2] D. Cortes. contextualbandits, 2019. URL `https://contextual-bandits.readthedocs.io/en/latest/`.

[3] D. Cortes. Adapting multi-armed bandits policies to contextual bandits scenarios. *arXiv:1811.04383 [cs, stat]*, Nov. 2019. URL `http://arxiv.org/abs/1811.04383`. arXiv: 1811.04383.

[4] I. Katakis, G. Tsoumakas, and I. Vlahavas. Multilabel text classification for automated tag suggestion. In *Proceedings of the ECML/PKDD*, volume 18, page 5. Citeseer, 2008.

[5] C. G. Snoek, M. Worring, J. C. Van Gemert, J.-M. Geusebroek, and A. W. Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *Proceedings of the 14th ACM international conference on Multimedia*, pages 421–430, 2006.

[6] G. Tsoumakas, I. Katakis, and I. Vlahavas. Effective and efficient multilabel classification in domains with large number of labels. In *Proc. ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08)*, volume 21, pages 53–59, 2008.