ProtoBandit: Efficient Prototype Selection via Multi-Armed Bandits

Arghya Roy Chaudhuri ARGHYAR@MICROSOFT.COM
Pratik Jawanpuria PRATIK.JAWANPURIA@MICROSOFT.COM
Bamdev Mishra BAMDEVM@MICROSOFT.COM
Microsoft India

Editors: Emtiyaz Khan and Mehmet Gönen

Abstract

In this work, we propose a multi-armed bandit based framework for identifying a compact set of informative data instances (i.e., the prototypes) that best represents a given target set. Prototypical examples of a given dataset offer interpretable insights into the underlying data distribution and assist in example-based reasoning, thereby influencing every sphere of human decision making. A key challenge is the large-scale setting, in which similarity comparison between pairs of data points needs to be done for almost all possible pairs. We propose to overcome this limitation by employing stochastic greedy search on the space of prototypical examples and multi-armed bandit approach for reducing the number of similarity comparisons. A salient feature of the proposed approach is that the total number of similarity comparisons needed is independent of the size of the target set. Empirically, we observe that our proposed approach, ProtoBandit, reduces the total number of similarity computation calls by several orders of magnitudes ($100 - 1000$ times) while obtaining solutions similar in quality to those from existing state-of-the-art approaches.

Keywords: subset selection; targeted selection; stochastic greedy; multi-armed bandits

1. Introduction

Prototypical examples are data instances that together summarizes a given dataset or an underlying data distribution (Weiser, 1982; Bien and Tibshirani, 2011b,a; Koh and Liang, 2017; Yeh et al., 2018). Such compact representation of a dataset is especially useful in this age of big-data, where the size of datasets goes beyond the capability of manual checking. Hence, prototypical samples help domain experts and data scientists by providing meaningful insights in complex domains (Gurumoorthy et al., 2019). They also provide example-based reasoning, thereby improving the interpretability of data distributions (Kim et al., 2014, 2016; Gurumoorthy et al., 2021).

A popular use-case of prototype selection is to select a test group that best represent the control group (or vice-versa). For product design (Leahy, 2013), a designer needs to know the required features for a new product directly from the targeted users. Therefore, it is important to select a small subset of users that best represent the larger customer base. In the healthcare domain, prototype selection has been employed for building training datasets (Suyal and Singh, 2021). Another interesting application of prototype selection is

in explainable AI (Linardatos et al., 2021), where we need to analyze the cause behind the output of an AI model.

Works such as (Olvera-López et al., 2008; Bien and Tibshirani, 2011b) have proposed finding prototypical elements in supervised setting where both features and label information of the data points are available. However, recent works (Gurumoorthy et al., 2019, 2021; Kim et al., 2016) have focused on the more general unsupervised setting where only feature information is available. In this paper, we focus on unsupervised prototype selection. A key challenge here is the big-data setting, where one may require to perform similarity computations between huge number of data points pairs (Zukhba, 2010). This makes the existing algorithms (Kim et al., 2016; Gurumoorthy et al., 2019, 2021) is impractical. Below, we concretize the prototype selection problem setup and discuss our contributions.

Prototype selection problem setup. Given two non-empty sets of points (source) $\mathcal{S}$ and (target) $\mathcal{T}$, a dissimilarity measure $d : \mathcal{T} \times \mathcal{S} \mapsto [0, 1]$, and a positive integer $k \leq |\mathcal{S}|$, our aim is to find a set $\mathcal{M} \subseteq \mathcal{S}$ that best represents the target set $\mathcal{T}$ such that $|\mathcal{M}| \leq k$. The set $\mathcal{M}$ is the set of prototypical elements. We call the tuple $(\mathcal{S}, \mathcal{T}, \mathbf{q}, d, k)$ a instance of the prototype selection problem, where $\mathbf{q} \in [0, 1]^{|\mathcal{T}|}$ is the weight vector associated with T. Such a weight vector can be useful in applications where not every target sample is equally important and can also be used to capture the underlying distribution of the target set (if available). For the sake of convenience and interpretability, we assume $\sum_{b\in\mathcal{T}} q_b = 1$.

Our Contributions. Gurumoorthy et al. (2021) have observed that the prototype selection problem is related to the k-medoids clustering problem for the specific case of $\mathcal{S} = \mathcal{T}$, i.e., the source set is same as the target set. Building on this, we first generalize the `BUILD` step of the partition around medoid (PAM) method (Kaufman and Rousseeuw, 1990), that solves the k-medoids clustering problem, to allow for the scenarios when $\mathcal{S} \neq \mathcal{T}$. We next prove that the `BUILD` step of the proposed generalized PAM algorithm is equivalent to the existing SPOTgreedy (Gurumoorthy et al., 2021) algorithm for prototype selection. This is interesting because the PAM algorithm can now also be understood from the lens of the optimal transport theory. Since the `BUILD` step of the proposed generalized PAM method is based on a greedy search procedure, we use it as a stepping stone to propose a sampling-based algorithm ProtoBandit for the prototype selection problem. Specifically, we introduce the following novelties in the ProtoBandit algorithm.

1. We employ random subset selection for more efficient greedy search as it reduces the search space (to find the next potential prototype) in the source set $\mathcal{S}$.

2. At each iteration, existing methods (Kim et al., 2016; Gurumoorthy et al., 2019, 2021) require similarity computations for every element of the selected subset $\mathcal{M}$ with every element in the target set $\mathcal{T}$. Thus, if the target set $\mathcal{T}$ is large, or potentially infinite, then such approaches become impractical. We circumvent this issue by employing the multi-armed bandits (MAB) based sampling technique to estimate the similarity of a source point with the target set $\mathcal{T}$. Our key technical result is that the total number of similarity computations required is independent of the target set size $|\mathcal{T}|$.

We also provide an approximation guarantee of the prototypical set obtained by the proposed ProtoBandit algorithm. In particular, let $f : 2^{\mathcal{S}} \mapsto [0, 1]$ denotes the similarity

function between an input (candidate) set $\mathcal{M}$ and the target set $\mathcal{T}$ and $\mathcal{M}^*$ be the optimal solution. Then, we prove that $f(\mathcal{M}_{ALG}) \geq \left(1 - e^{-1} - \epsilon\right) f(\mathcal{M}^*) - \nu$ with probability at least $1 - \delta$, where $\mathcal{M}_{\text{ALG}} \subset \mathcal{S}$ is output prototype set of our proposed ProtoBandit, $\epsilon, \nu \in (0, 1)$ are the approximation parameters, and $\delta \in (0, 0.05)$ is the error threshold. Empirically, we observe that our ProtoBandit reduces the total number of similarity computation calls by several orders of magnitudes ($100 - 1000$ times) while obtaining similar quality solution.

2. Related work

Prototype selection. The problem of prototype selection has been mostly explored (Bien and Tibshirani, 2011b; Crammer et al., 2002; Wohlhart et al., 2013; Wei et al., 2015) in the supervised learning setups where the label of the data points are available. Recent prototype selection approaches such as MMD-Critic (Kim et al., 2016), ProtoDash (Gurumoorthy et al., 2019), and SPOTgreedy (Gurumoorthy et al., 2021), aim at obtaining the set of prototypical elements $\mathcal{M} \subset \mathcal{S}$ such that $\mathcal{M}$'s underlying distribution is close to that of the target $\mathcal{T}$. Such approaches are suitable for unsupervised setting as they typically assume that only the similarity (or distance) between pairs (a, b), where $a \in \mathcal{S}$ and $b \in \mathcal{T}$, are available. MMD-Critic and ProtoDash employ the MMD distance to capture the similarity between two distributions while SPOTgreedy uses the Wasserstein distance, i.e., the optimal transport framework. We describe the SPOTgreedy algorithm in detail in Appendix **??** of our extended version (Chaudhuri et al., 2022). The above three approaches are based on greedy search and a common bottleneck in them is that, at every iteration of greedy search, they require $\mathcal{O}(|\mathcal{T}|)$ similarity computations, which becomes impractical as the target set $\mathcal{T}$ becomes large. To this end, we propose a multi-armed bandits based approach to alleviate this concern.

PAM algorithm. Prototype selection when the source and target sets are identical may specifically be viewed as identifying important data points in the given set, i.e., data summarisation. An intuitive way of unsupervised data summarisation is to apply a centroid-based clustering technique and choose the cluster centers as the representative of the whole dataset. A popular approach to accomplish this is k-medoids clustering using the partition around medoid (PAM) method (Kaufman and Rousseeuw, 1990). PAM applies an exhaustive greedy search over the whole set through two main steps: `BUILD` and `SWAP`. During the `BUILD` step, PAM selects the medoids, and during the `SWAP` step it improves upon the already chosen medoids by replacing them with the new ones. Recently, Tiwari et al. (2020) have proposed a randomized algorithm within the multi-armed bandit framework for k-medoids clustering, albeit without approximation guarantees.

3. ProtoBandit: An Efficient Prototype Selection Algorithm

In this section, we first generalize the PAM algorithm (Kaufman and Rousseeuw, 1990) to the prototype selection problem (i.e., the sets $\mathcal{S}$ and $\mathcal{T}$ are different). We then build on the generalized PAM algorithm and propose ProtoBandit, which makes use of approximate greedy search and multi-armed bandit frameworks.

3.1. Generalized PAM

We propose two modifications in the existing k-medoids clustering algorithm, PAM, to generalize it to the prototype selection setting. We detail the `BUILD` step of the generalized PAM algorithm in Algorithm 1. The full generalized PAM algorithm is in Appendix A of the extended version of this paper (Chaudhuri et al., 2022).

The first modification enables the algorithm to choose medoids from a set that is different from the set of points to be clustered. Therefore, unlike the original PAM algorithm, Algorithm 1 can choose set of medoids from $\mathcal{S}$ to cluster the points in $\mathcal{T}$, even if $\mathcal{S} \neq \mathcal{T}$. The second modification lies in choosing the number of elements r while updating the set of chosen points $\mathcal{M}$ during the `BUILD` step, which allows for additional flexibility and efficiency. It should, however, be noted that setting r to unity implies that medoids from the source set $\mathcal{S}$ are selected in a (strict) sequential manner, which has both theoretical and qualitative benefits. Henceforth, in our experiments, we consider $r = 1$ setting unless specified otherwise.

Algorithm 1 `BUILD` step for the generalized PAM algorithm

Input : Problem instance given by $(\mathcal{S}, \mathcal{T}, \mathbf{q}, d, k)$, and a positive integers $r \in \{1, \cdots, k\}$.
Output: A set $\mathcal{M}$, such that $\mathcal{M} = k$
Initialization: For each $j \in \mathcal{T}$, set $D_j = \infty$.
In this step we apply greedy strategy to choose an initial set of k prototypes. $\mathcal{M} = \emptyset$
while $|\mathcal{M}| < k$ **do**
  Define gain vector g with entries

$$g_i = \sum_{j \in \mathcal{T}} q_j \max\{D_j - d(j,i), 0\}, \ \forall i \in \mathcal{S} \setminus \mathcal{M}. \tag{1}$$

  if $\forall i \in \mathcal{S}, g_i = 0$ **then** `// no meaningful prototype to add`
    $Q \stackrel{\text{def}}{=}$ Choose r elements from $\mathcal{S} \setminus \mathcal{M}$ at random
  else
    $Q \stackrel{\text{def}}{=}$ Set of indices of top r largest *non-zero* elements in $g = \{g_i\}_{i=1}^{|\{\mathcal{S} \setminus \mathcal{M}\}|}$
  end
  $\mathcal{M} = \mathcal{M} \cup Q$, and $\forall j \in \mathcal{T}$ update D_j.
end

Below, we show the equivalence between the `BUILD` step of the proposed generalized PAM algorithm (in Algorithm 1) and the SPOTgreedy algorithm (Gurumoorthy et al., 2021). To this end, we define D_j to be the dissimilarity between a point j and the closest object in $\mathcal{M}$, i.e.,

$$D_j = \min_{i \in \mathcal{M}} \{d(j,i)\} \ \text{[wherein dissimilarity measure } d : \mathcal{T} \times \mathcal{S} \mapsto [0,1]]\,. \tag{2}$$

Lemma 1 *Given a problem instance $(\mathcal{S}, \mathcal{T}, \mathbf{q}, d, k)$, for a fixed value of the input parameter $r \in \{1, \cdots, k\}$, the k points identified by SPOTgreedy is identical with the set of selected point by Algorithm 1.*

The proof of Lemma 1 is given in Appendix B of the extended version of this paper (Chaudhuri et al., 2022). The objective of this lemma is to show that given a similarity matrix $Z \in [0,1]^{m \times n}$ at each iteration Algorithm 1 chooses a prototype by maximizing the following function:

$$f(\mathcal{M}) = \sum_{j \in \mathcal{T}} q_j \max_{i \in \mathcal{M}} Z_{ji} \text{ [for } \mathcal{M} \subset \mathcal{S}], \tag{3}$$

where Z is a $|\mathcal{T}| \times |\mathcal{S}|$ similarity matrix between the target and the source sets, based on the given dissimilarity measure d. We compute similarity between j-th data point in $\mathcal{T}$ and i-th data point in $\mathcal{S}$ as

$$Z_{ji} = C - d(j,i), \tag{4}$$

where C is a constant such that $C \geq \max_{i \in \mathcal{S}, j \in \mathcal{T}} \{d(j,i)\}$ and $C \geq 1$.

For choosing k prototypes, Algorithm 1 incurs $O(|\mathcal{S}||\mathcal{T}|k/r)$ similarity comparisons. This may become a computational bottleneck for large $|\mathcal{S}|$ and $|\mathcal{T}|$ values. To mitigate this, we next propose an approximate greedy-search for Algorithm 1 that is frugal in terms of the number of similarity comparisons needed.

3.2. Approximate Greedy Search

Due to simplicity and versatility of the greedy search framework, it has attracted a lot of attentions (Minoux, 1978; Wei et al., 2014; Badanidiyuru and Vondrák, 2014; Mirzasoleiman et al., 2015) for reducing the number of similarity comparisons. We choose Stochastic-Greedy (Mirzasoleiman et al., 2015) as it is one of the simplest and empirically fast (Mirzasoleiman et al., 2015) algorithms for approximate greedy search. Without approximation, at every step a greedy algorithm exhaustively searches through the set to find next the potential solution. Instead, Stochastic-Greedy considers a random subset for the same. The size of the subset is chosen depending on the approximation parameter ϵ, such that the outcome is optimal within a $(1 - e^{-1} - \epsilon)$ multiplicative factor.

Stochastic-Greedy (Mirzasoleiman et al., 2015) assumes that the objective function evaluation can be done in $O(1)$. Also, for each point it examines, the exact value of the objective function has to be computed. Let us understand its effect on the problem of prototype selection through an example. Assuming, $\mathcal{R} \subset \mathcal{S} \setminus \mathcal{M}$ be a random subset chosen by Stochastic-Greedy at the i-th iteration, it will modify the Equation 1 to the following:

$$g_i = \sum_{j \in \mathcal{T}} q_j \max\{D_j - d(j,i), 0\}, \ \forall i \in \mathcal{R}. \tag{5}$$

We note, for each point $i \in \mathcal{S}$, it needs $O(|\mathcal{T}|)$ similarity comparisons to evaluate Equation 5 exactly. This might become impractical for a large target set $\mathcal{T}$, and to get around, one can estimate the value via the Monte-Carlo sampling on $\mathcal{T}$ (according to the distribution $\mathbf{q}$). However, not all the elements in $\mathcal{R}$ are equally strong contenders for being the next prototype. Hence, it is desired to allocate more similarity comparisons to differentiate between contentious elements of $\mathcal{R}$ which are close to the local optimal, while saving them on the easily distinguishable sub-optimal elements. Therefore, an off-the-shelf use of Stochastic-Greedy is not a good choice. We mitigate this issue by identifying an approximate solution of Equation 5 via multi-armed bandits (MAB) (Berry and Fristedt, 1985) that we discuss next.

3.3. Using Multi-Armed Bandits (MAB) in Prototype Selection

Background on Multi-Armed Bandits. Multi-armed bandits (Berry and Fristedt, 1985) is a popular abstraction of sequential decision making under the uncertainty. An *arm* of a bandit represents a decision, while *pull* of an arm represents taking decision corresponding to that arm. Further, we assume each arm has a probability distribution associated with it, and when pulled, a real-valued reward is generated in i.i.d. fashion from the underlying probability-distribution. This probability distribution is called reward distribution of that particular arm, and is unknown to the experimenter. For simplicity, we assume the reward-distribution for each arm is supported on the interval $[0, 1]$. Under this scenario, it is an interesting problem to identify the best arm incurring minimal number of total samples.

To put formally, we assume $\mathcal{A}$ be the given set of n arms with μ_a being the expected mean reward of arm $a \in \mathcal{A}$. For simplicity, we assume, $\mu_{a_1} \geq \mu_{a_2} \geq \cdots \mu_{a_n} \geq 0$. As the reward distribution of the arms are not known, it can be trivially shown that there is no algorithm that incurs a finite number of samples to identify the arm with the highest expected reward with certainty. Hence, we seek for an approximate probabilistic solution. For a given tolerance $\nu \in [0, 1]$, we call an arm a to be (ν, m)-*optimal* (for $m \leq n$) if $\mu_a \geq \mu_{a_m}$. For $m = 1$, it is called problem of best-arm identification to identify an $(\nu, 1)$-optimal arm. When $1 \leq m \leq |\mathcal{A}|$, the problem generalizes to identification of the best subset of size m (Kalyanakrishnan et al., 2012), wherein the objective is to identify any m (ν, m)-optimal arms. Inheriting notations from Roy Chaudhuri and Kalyanakrishnan (2019), let us denote the set of all (ν, m)-optimal arms by $\mathcal{TOP}_m(\nu) \stackrel{\text{def}}{=} \{a : \mu_a \geq \mu_{a_m}\}$[1]. Then, an algorithm is said to solve best-subset identification problem if presented with a set of arms $\mathcal{A}$, size of the output set m, a tolerance $\nu \in (0, 1)$, and an error probability bound $\delta \in (0, 1)$, it stops with probability 1 after a finite number of steps, and output an m-sized subset $Q_{(m)} \in \mathcal{A}$, such that $\Pr\{Q_{(m)} \subset \mathcal{TOP}_m(\nu)\} \geq 1 - \delta$. The efficiency of solving a best-subset identification lies in incurring as low number of samples as possible.

Applying MAB to Approximate Equation 5. We have seen in Section 3.2 that evaluating Equation 5 is intensive in terms of the number of similarity comparisons for a large target set $\mathcal{T}$. During every iteration $\mathcal{M}$ remains fixed, so is the dissimilarity D_j between a point $j \in \mathcal{T}$ and the closest object in $\mathcal{M}$. Therefore, if we sample from $\mathcal{T}$ using the probability distribution $\mathbf{q}$, by normalization assumption of the dissimilarity measure, for every element $i \in \mathcal{S}$ we shall get a probability distribution over the $[0, 1]$, with the true mean being given by Equation 1. Hence, we can treat elements of the source set $\mathcal{S}$ as arms of a MAB instance. Thus, at each round, after we select a random subset $\mathcal{R} \subset \mathcal{S}$, the problem reduces to identification of the best subset of size r. Based on this idea, next, we propose an algorithm, and analyze its correctness, and upper bound the incurred number of similarity comparisons in the worst-case. In this paper, we confine to $r = 1$, i.e., we select one element per iteration.

The best-arm identification has attracted a lot of attention over the years (Even-Dar et al., 2002, 2006; Karnin et al., 2013; Jamieson et al., 2014). Among an array of algorithms, Median Elimination (Even-Dar et al., 2002) is well-known due to its simplicity, and for in-

1. Given a set of arms $\mathcal{A}$, and an $\nu \in (0, 1]$, there might be more than m arms in $\mathcal{TOP}_m(\nu)$.

curring a number of samples that is within a constant factor of the lower bound (Mannor and Tsitsiklis, 2004). Recently, Hassidim et al. (2020) have proposed the approximate best arm (ABA) algorithm for identification of an $(\nu, 1)$-optimal arm, and have proven its efficiency in sample-complexity compared to the existing algorithms like Median Elimination (Even-Dar et al., 2002). We restate Theorem 1 from Hassidim et al. (2020) on upperbounding the sample-complexity of ABA.

Lemma 2 (Restatement of (Hassidim et al., 2020, Theorem 1)) *Suppose, given a set of arms $\mathcal{A}$, the underlying reward distribution of its each arm is supported on $[0, 1]$. Then, given an $\nu \in (0, 1)$, and $\delta \in (0, 0.5)$, ABA initialized with $\alpha = 1 - e^{-1}$ returns an $(\nu, 1)$-optimal arm with probability at least $1 - \delta$ incurring no more than $18\frac{|\mathcal{A}|}{\nu^2} \ln \frac{1}{\delta}$ samples.*

Now, we are ready to introduce an MAB-based prototype selection algorithm, ProtoBandit, and present its analysis in the next section.

3.4. Efficient Prototype Selection by Stochastic Greedy Search and Multi-Armed Bandits

We propose a meta algorithm ProtoBandit in Algorithm 2 that at each iteration selects a random subset $\mathcal{R} \subset \mathcal{S}$ (like Stochastic-Greedy (Mirzasoleiman et al., 2015)) to reduce the search space, and then applies ABA to identify the best element. We emulate pulling of an arm via sampling the target set $\mathcal{T}$ according to the distribution $\mathbf{q}$ specified by the problem instance. To this end, we define the method Pull in Algorithm 3, that is internally used by the ABA inside ProtoBandit. We note, due to normalization assumption, the objective function $f(\cdot)$ defined in Equation 3 must lie in the range $[0, 1]$. In other words, if for all $i \in \mathcal{S}$, and $j \in \mathcal{T}$, then we can take $C = 1$ to make $Z_{i,j} \in [0, 1]$. Therefore, for any $\mathcal{M} \subseteq \mathcal{S}$, $f(\mathcal{M}) \in [0, 1]$. Subsequently, the assumptions to apply ABA remain valid.

Algorithm 2 ProtoBandit: Randomized Greedy Prototype Selection with MAB

Input : Problem instance $(\mathcal{S}, \mathcal{T}, \mathbf{q}, d, k)$, tolerance $\epsilon \in (0, 1)$, $\nu \in (0, 1 - \epsilon - 1/e)$, and acceptable error probability $\delta \in (0, 0.05)$.
Output: Set of prototype $\mathcal{M} \subset \mathcal{S}$, such that $|\mathcal{M}| = k$
Assert : $\forall (y, x) \in \mathcal{T} \times \mathcal{S}, d(y, x) \in [0, 1]$.
Define a similarity function $sim : \mathcal{T} \times \mathcal{S} \mapsto [0, 1]$ as $sim \stackrel{\text{def}}{=} 1 - d(y, x), \forall (y, x) \in \mathcal{T} \times \mathcal{S}$.
Set $\mathcal{M} = \emptyset$
while $\mathcal{M} \leq k$ **do**
  Select a subset $\mathcal{R} \subset \mathcal{S} \setminus \mathcal{M}$ of size $|\mathcal{R}| = \left\lceil \frac{|\mathcal{S}|}{k} \log \frac{1}{\epsilon} \right\rceil$, using the sampling distribution $\mathbf{q}$ with replacement.
  Apply ABA on $\mathcal{R}$, with tolerance $\frac{\nu}{1-1/e-\epsilon}$, error threshold $\frac{\delta}{k}$, and return an arm a_{out}.
  $\mathcal{M} = \mathcal{M} \cup \{a_{\text{out}}\}$.
end

Theorem 3 (Upper bound on the number of similarity comparisons) *Given a problem instance $(\mathcal{S}, \mathcal{T}, \mathbf{q}, d, k)$, $\mathcal{M}^*$ is the optimal solution, and for a tolerance $\epsilon \in (0, 1)$,*

Algorithm 3 Pull

Input : Sampling domain $\mathcal{T}$, sampling distribution $\mathbf{q}$, chosen set of prototypes $\mathcal{M} \subset \mathcal{S}$, an element $i \in \mathcal{S} \setminus \mathcal{M}$, similarity function $sim : \mathcal{T} \times \mathcal{S} \mapsto [0, 1]$,.

Output: Contribution value of i.

Sample an element $j \in \mathcal{T}$ according to sampling distribution $\mathbf{q}$.

Return $\max\{D_j - d(j, a), 0\}$, wherein D_j is defined in Equation 2.

$\nu \in (0, 1 - e^{-1} - \epsilon)$, *and acceptable error probability* $\delta \in (0, 0.05)$, *the number of similarity comparisons incurred by ProtoBandit is in* $O\left(k^2|\mathcal{S}|\left(\frac{\nu}{1-\epsilon-1/e}\right)^{-2}\log\frac{1}{\epsilon}\log\frac{k}{\delta}\right)$.

Proof As Lemma 2 presents, treating $\mathcal{R}$ as the set of arms, and letting $\nu_0 = \nu/(1-e^{-1}-\epsilon)$, ABA incurs at most $\frac{18|\mathcal{R}|}{\nu_0^2}\log\frac{1}{\delta}$ samples to identify an $(\nu_0, 1)$-optimal arm from $\mathcal{R}$. Further, at iteration i, there are $(i-1)$ elements in the set $\mathcal{M}$. Hence, for each sampled element from $\mathcal{T}$, it takes $O(|\mathcal{M}|)$ or $i-1$ similarity comparison to compute D_j (given by Equation 2). As there are k such calls to ABA, the total number of similarity comparisons is given by

$$\sum_{i=1}^{k}(i-1)\frac{|\mathcal{S}|}{k}\log\frac{1}{\epsilon}\times 18\nu_0^{-2}\log\frac{1}{\delta} = \sum_{i=1}^{k}(i-1)\frac{|\mathcal{S}|}{k}\log\frac{1}{\epsilon}\times 18\nu_0^{-2}\log\frac{1}{\delta} \quad \left[\because \mathcal{R} = \frac{|\mathcal{S}|}{k}\log\frac{1}{\epsilon}\right]$$
$$< 36k^2|\mathcal{S}|\left(\frac{\nu}{1-\epsilon-1/e}\right)^{-2}\log\frac{1}{\epsilon}\log\frac{k}{\delta}.$$

■

To derive the approximation guarantee for ProtoBandit, we inherit the analysis of Stochastic-Greedy from Mirzasoleiman et al. (2015, Theorem 1) and lower bound the probability of choosing an element from the optimal set in Lemma 4. Then, we apply this result to prove the approximation ratio.

Lemma 4 *Given a current solution* $\mathcal{M}$, *the expected-gain of ProtoBandit in one iteration is at least* $\frac{1-\epsilon}{k}\sum_{a\in\mathcal{M}^*\setminus\mathcal{M}}\left(\Delta(a|\mathcal{M}) - \frac{\nu}{1-\epsilon-1/e}\right)$.

Proof To prove the lemma, let us first lower bound the probability of $\mathcal{R}\cap(\mathcal{M}^*\setminus\mathcal{M}) \neq \emptyset$. Letting $|\mathcal{R}| = r$, we note,

$$\begin{aligned}
&\Pr\{\mathcal{R}\cap(\mathcal{M}^*\setminus\mathcal{M}) = \emptyset\} \\
&= \left(1 - \frac{|\mathcal{M}^*\setminus\mathcal{M}|}{|\mathcal{S}\setminus\mathcal{M}|}\right)^{|\mathcal{R}|} \leq \exp\left(-|\mathcal{R}|\frac{|\mathcal{M}^*\setminus\mathcal{M}|}{|\mathcal{S}\setminus\mathcal{M}|}\right) \\
&\leq \exp\left(-\frac{|\mathcal{R}|}{|\mathcal{S}|}|\mathcal{M}^*\setminus\mathcal{M}|\right) = \exp\left(-k\frac{|\mathcal{R}|}{|\mathcal{S}|}\frac{|\mathcal{M}^*\setminus\mathcal{M}|}{k}\right). \qquad (6)
\end{aligned}$$

Now, using the concavity of $exp(-\frac{rk}{|\mathcal{S}|}x)$ with respect to x, and given that $|\mathcal{M}^* \setminus \mathcal{M}| \in \{0, 1, \cdots, k\}$, we can re-write Equation 6 as

$$\begin{aligned}
&\implies \Pr\{\mathcal{R} \cap (\mathcal{M}^* \setminus \mathcal{M}) \neq \emptyset\} \\
&\geq \left(1 - \exp\left(-k\frac{|\mathcal{R}|}{|\mathcal{S}|}\right)\right)\frac{|\mathcal{M}^* \setminus \mathcal{M}|}{k}, \\
&= \left(1 - \exp\left(-\left\lceil\frac{|\mathcal{S}|}{k}\log\frac{1}{\epsilon}\right\rceil\frac{k}{|\mathcal{S}|}\right)\right)\frac{|\mathcal{M}^* \setminus \mathcal{M}|}{k} \left[\because |\mathcal{R}| = \left\lceil\frac{|\mathcal{S}|}{k}\log\frac{1}{\epsilon}\right\rceil \text{ in Algorithm 2}\right], \\
&\geq \left(1 - \log\frac{1}{\epsilon}\right)\frac{|\mathcal{M}^* \setminus \mathcal{M}|}{k} \geq (1-\epsilon)\frac{|\mathcal{M}^* \setminus \mathcal{M}|}{k}. \qquad (7)
\end{aligned}$$

Now, we lower bound the value of $\Delta(a|\mathcal{M})$ by lower-bounding the approximation offered by ABA. Suppose, at some iteration $\mathcal{M}$ be the already chosen set of prototypes. Also, let $\hat{a} \stackrel{\text{def}}{=} \arg\max_{a \in \mathcal{R}} \Delta(a|\mathcal{M})$ be the the locally optimal solution, and a_{out} be the output of ABA inside a single iteration of ProtoBandit. Assuming $\mathcal{R} \cap (\mathcal{M}^* \setminus \mathcal{M}) \neq \emptyset$, we note, the marginal contribution of $\hat{a}$ is at least as the marginal contribution of any element of $\mathcal{R} \cap (\mathcal{M}^* \setminus \mathcal{M})$. Therefore, $\Delta(\hat{a}|\mathcal{M}) \geq \max_{\mathcal{R} \cap (\mathcal{M}^* \setminus \mathcal{M})} \Delta(\hat{a}|\mathcal{M})$.

$$\mathbb{E}[\Delta(\hat{a}|\mathcal{M})|\mathcal{M}] \geq \Pr\{\mathcal{R} \cap (\mathcal{M}^* \setminus \mathcal{M}) \neq \emptyset\} \cdot \max_{a \in \mathcal{R} \cap (\mathcal{M}^* \setminus \mathcal{M})} \Delta(\hat{a}|\mathcal{M}) \qquad (8)$$

We note, $\Delta(\hat{a}|\mathcal{M})$ is at least as much as the contribution of an element uniformly chosen at random from $\mathcal{R} \cap (\mathcal{M}^* \setminus \mathcal{M})$. Again as $\mathcal{R}$ is chosen uniformly at random from $|\mathcal{S} \setminus \mathcal{M}|$, each element of $\mathcal{M}^* \setminus \mathcal{M}$ is equally likely to belong to $\mathcal{R}$. Hence, choosing an element uniformly at random from $\mathcal{R} \cap (\mathcal{M}^* \setminus \mathcal{M})$ is equivalent to choosing it uniformly at random from $\mathcal{M}^* \setminus \mathcal{M}$. Putting together, from Equation 8 we get

$$\mathbb{E}[\Delta(\hat{a}|\mathcal{M})|\mathcal{M}] \geq \Pr\{\mathcal{R} \cap (\mathcal{M}^* \setminus \mathcal{M}) \neq \emptyset\}\frac{1}{|\mathcal{M}^* \setminus \mathcal{M}|}\sum_{a \in \mathcal{M}^* \setminus \mathcal{M}} \Delta(a|\mathcal{M}). \qquad (9)$$

In ProtoBandit, the arm a_{out} returned by the subroutine ABA is not necessarily identical to $\hat{a}$, but $\left(\frac{\nu}{1-\epsilon-1/e}, 1\right)$-optimal (in $\mathcal{R}$) with probability at least $1 - \delta/k$. Therefore, letting $\nu_0 \stackrel{\text{def}}{=} \frac{\nu}{1-1/e-\epsilon}$, $(\Delta(\hat{a}|\mathcal{M}) - \Delta(a_{\text{out}}|\mathcal{M})) \leq \nu_0$ holds with probability at least $1 - \delta/k$. Now, assuming a_{out} is $(\nu_0, 1)$-optimal, and following the argument we used to build Equations 8 and 9, we can write

$$\begin{aligned}
&\mathbb{E}[\Delta(a_{\text{out}}|\mathcal{M})|\mathcal{M}] \\
&\geq \Pr\{\mathcal{R} \cap (\mathcal{M}^* \setminus \mathcal{M}) \neq \emptyset\}\left(\max_{a \in \mathcal{R} \cap (\mathcal{M}^* \setminus \mathcal{M})} \Delta(\hat{a}|\mathcal{M}) - \nu_0\right) \text{ [w. p. } \geq 1 - \frac{\delta}{k}], \qquad (10) \\
&\geq \Pr\{\mathcal{R} \cap (\mathcal{M}^* \setminus \mathcal{M}) \neq \emptyset\}\left(\frac{1}{|\mathcal{M}^* \setminus \mathcal{M}|}\sum_{a \in \mathcal{M}^* \setminus \mathcal{M}} \Delta(a|\mathcal{M}) - \nu_0\right), \\
&\geq \Pr\{\mathcal{R} \cap (\mathcal{M}^* \setminus \mathcal{M}) \neq \emptyset\}\frac{1}{|\mathcal{M}^* \setminus \mathcal{M}|}\sum_{a \in \mathcal{M}^* \setminus \mathcal{M}} (\Delta(a|\mathcal{M}) - \nu_0),
\end{aligned}$$

$$\geq (1-\epsilon)\frac{|\mathcal{M}^* \setminus \mathcal{M}|}{k}\frac{1}{|\mathcal{M}^* \setminus \mathcal{M}|}\sum_{a\in\mathcal{M}^*\setminus\mathcal{M}}(\Delta(a|\mathcal{M})-\nu_0) \text{ [using Equation 7]},$$

$$= \frac{1-\epsilon}{k}\sum_{a\in\mathcal{M}^*\setminus\mathcal{M}}(\Delta(a|\mathcal{M})-\nu_0). \tag{11}$$

■

Theorem 5 (Correctness of ProtoBandit) *Suppose, given a problem instace $(\mathcal{S},\mathcal{T},\mathbf{q},d,k)$, $\mathcal{M}^*$ is the optimal solution, and for a tolerance $\epsilon\in(0,1)$, $\nu\in(0,1-e^{-1}-\epsilon)$, and acceptable error probability $\delta\in(0,0.05)$, let $\mathcal{M}_{ALG}$ be the output by ProtoBandit. Then, $f(\mathcal{M}_{ALG})\geq\left(1-e^{-1}-\epsilon\right)f(\mathcal{M}^*)-\nu$ with probability at least $1-\delta$, where $f(\cdot)$ is defined in Equation 3.*

Proof We use Lemma 4 to prove Theorem 5. Let $\mathcal{M}^i=\{a^1_{\text{out}},\cdots,a^i_{\text{out}}\}$ be the solution obtained by ProtoBandit at the end of i-th iteration, and $\nu_0=\frac{\nu}{1-\epsilon-1/e}$. Then,

$$\begin{aligned}
\mathbb{E}[f(\mathcal{M}^{i+1})-f(\mathcal{M}^i)|\mathcal{M}^i] &= \mathbb{E}[\Delta(a^{i+1}_{\text{out}}|\mathcal{M}^i)|\mathcal{M}^i],\\
&\geq \frac{1-\epsilon}{k}\sum_{a\in\mathcal{M}^*\setminus\mathcal{M}}\left(\Delta(a|\mathcal{M}^i)-\nu_0\right) \text{ [using Lemma 4]},\\
&\geq \frac{1-\epsilon}{k}\left(\Delta(\mathcal{M}^*|\mathcal{M}^i)-\nu_0\right) \text{ [by submodularity of } f(\cdot)],\\
&\geq \frac{1-\epsilon}{k}(f(\mathcal{M}^*)-f(\mathcal{M}^i)-\nu_0).
\end{aligned}$$

We note, $f(\mathcal{M}^i)$ is a random variable, due to the randomness introduced by random sampling in the source set $\mathcal{S}$ to select $\mathcal{R}$, and also by the MAB subroutine ABA. By taking expectation over $f(\mathcal{M}^i)$,

$$\mathbb{E}[f(\mathcal{M}^{i+1})-f(\mathcal{M}^i)]\geq\frac{1-\epsilon}{k}\mathbb{E}[f(\mathcal{M}^*)-f(\mathcal{M}^i)-\nu_0].$$

Now, by induction,

$$\begin{aligned}
\mathbb{E}[f(\mathcal{M}^k)] &\geq \left(1-\left(\frac{1-\epsilon}{k}\right)^k\right)(f(\mathcal{M}^*)-\nu_0),\\
&\geq \left(1-\left(\exp\left(-(1-\epsilon)\right)\right)\right)(f(\mathcal{M}^*)-\nu_0),\\
&\geq \left(1-e^{-1}-\epsilon\right)(f(\mathcal{M}^*)-\nu_0)=\left(1-e^{-1}-\epsilon\right)f(\mathcal{M}^*)-\nu.
\end{aligned}\tag{12}$$

The last step to prove Theorem 5 is proving the correctness of Equation 12. We note, Equation 12 is valid only if Equation 11 holds for every iteration from 1 to k. Further, the correctness of Equation 11 is dependent on the Equation 10. Now, we note, the arm a_{out} returned by ABA may not be $(\nu_0,1)$-optimal with probability at most δ/k, and Equation 10 fails to hold. However, the probability at in any of the k iterations, a_{out} is not $(\nu_0,1)$-optimal is at most $\sum_{i=1}^{k}(\delta/k)=\delta$. Hence, Equation 12 holds with probability at least $1-\delta$. This completes the proof of Theorem 5. ■

3.5. Practical Considerations for Implementing ProtoBandit

It is important to note that the number of samples incurred by ABA (Hassidim et al., 2020) is problem-independent, that is given any fixed-sized set of arms $\mathcal{A}$, for a fixed $\epsilon, \delta \in (0, 0.05)$ the number of samples incurred will be the same (as given by Lemma 2). Hence, it is a non-adaptive algorithm as the number of incurred samples does not depend on the means of the arms. However, in practice, it is common to handle set arms whose mean are not very close to each other, and hence, it is more efficient to use adaptive algorithms like KL-LUCB (Kaufmann and Kalyanakrishnan, 2013) that take advantage of the difference between mean of the arms. At each iteration, KL-LUCB judicially selects two arms to sample and the algorithm stops if the confidence-intervals of the arms does not meet some criterion. We note that despite KL-LUCB being more sample-efficient than non-adaptive algorithms, complying with it still requires a number of samples that might be too high for a medium-sized datasets. However, in practice, seldom we encounter such pathological cases, and hence, we can use a heuristic to make the algorithm incur lesser number of similarity comparisons by stopping early. We make use of the early-stop heuristic inside KL-LUCB for our experiments. The early stopping makes use of an optimistic threshold that is easier to meet than the stopping criterion set by KL-LUCB.

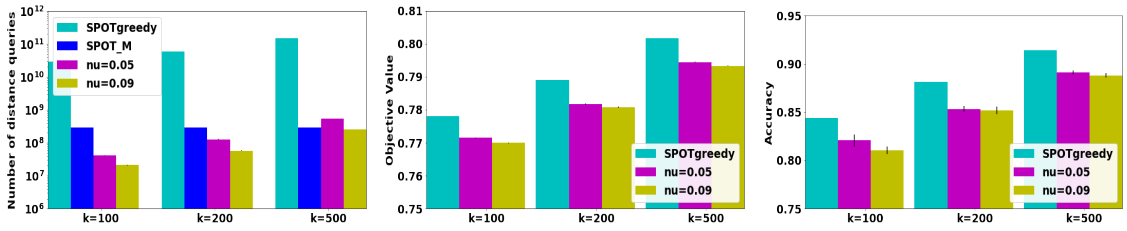
4. Experiments

In this section, we show the benefit of ProtoBandit over SPOTgreedy (Gurumoorthy et al., 2021) in obtaining a good trade-off between the number of distance queries needed and the generalization performance of the obtained prototypes.

We consider the MNIST (LeCun et al., 1998) dataset, which is a collection of 60K images of hand-written digits from 0 to 9, each of size 28×28 pixels. There are two subsets mnist_train, and mnist_test consisting of 50K and 10K images, respectively. Following Gurumoorthy et al. (2021), we sample 5K points uniformly at random from mnist_test and create the source set $\mathcal{S}$. We create the target set $\mathcal{T}$ from mnist_test as follows. First we note, the population of label 5 is 5421, and it is the least among all the labels. Therefore, to create a target set with skew θ% we take all the elements of label 5 from mnist_train and we uniformly sample its remaining elements leading to a $\mathcal{T}$ of size $(542100/\theta)$. Thus, for $\theta = 10$, the target set $\mathcal{T}$ is completely balanced, while for $\theta = 100$ it consists elements only from label 5. We conduct different experiments by varying $\theta \in \{10, 20, 50, 70, 100\}$, and $k \in \{100, 200, 500\}$. For ProtoBandit, we have used $\epsilon \in \{0.2, 0.4\}$, and $\nu \in \{0.05, 0.09\}$. The results are averaged over ten randomized runs. We use the Euclidean distance as the pair-wise dissimilarity measure d between points. We normalize the distances to lie in the range $[0, 1]$.

In our experiments, we consider the setting in which both SPOTgreedy and ProtoBandit select only one element at each iteration, i.e., $r = 1$. We also ensure that both ProtoBandit and SPOTgreedy compute the similarity comparisons on the fly (i.e., they do not memorize the computed similarity comparisons between data points).

The results are shown in Figure 1. For $\mathcal{T}$ with 10% skew (i.e., $\theta = 10$), Figure 1(a) shows that ProtoBandit (for $\epsilon = 0.2$) incurs as little as 1/100-th of the number of similarity comparisons made by SPOTgreedy; however as Figure 1(a) shows the drop in objective value is within only 2%.



(a) (b) (c)

Figure 1: Comparison of SPOTgreedy and ProtoBandit (with $\epsilon = 0.2$) on $\mathcal{T}$ with skew = 10% in terms of (a) number of similarity comparisons; (b) objective value; and (c) generalization performance. SPOT_M in Figure 1(a) is an implementation of SPOTgreedy with all the $|\mathcal{S} \times \mathcal{T}|$ pairwise-similarity values memorized. Our proposed ProtoBandit with different ν values obtain good accuracy at much lower number of similarity comparisons for SPOTgreedy. Compared to SPOT_M, which memorizes all the $|\mathcal{S} \times \mathcal{T}|$ pair-wise similarity values, we see the benefit of ProtoBandit (that never memorizes the similarity values).

We note that the number of distance computations for SPOTgreedy is minimum if it is allowed to memorize all the $|\mathcal{S} \times \mathcal{T}|$ pair-wise similarity values. Let us call this implementation of SPOTgreedy where it is allowed to memorize all these values as SPOT_M. We also compare ProtoBandit with SPOT_M. Figure 1(a) shows the number of incurred distance queries by the proposed ProtoBandit is far less as compared to SPOT_M.

In practice, it is very common that value of the objective is not the final thing that an experimenter seeks. A more sensible way to compare the results is via the accuracy of the selected prototypes. Thanks to Lemma 1, we can now inherit the barycentric projection method from the theory of optimal transport. Following Gurumoorthy et al. (2021), the accuracy of the selected prototypes is measured by using the barycentric mapping for both SPOTgreedy and ProtoBandit. As depicted by Figure 1(c), ProtoBandit achieves a very close accuracy compared to SPOTgreedy. Detailed comparisons between these two algorithms for different values of skew (θ), ϵ, and ν are in Appendix C of the extended version of this paper (Chaudhuri et al., 2022).

In Appendix D of the extended version of this paper (Chaudhuri et al., 2022), we present experiments on an additional dataset.

5. Conclusion and Future Work

In this paper, we have proposed a novel unsupervised algorithm ProtoBandit for the prototype selection problem, which offers a good trade-off between the number of similarity comparisons needed and good accuracy of prototypes obtained. The key idea is to use a generalized version of the `BUILD` step of the popular clustering method PAM (Rdusseeun and Kaufman, 1987) for the general prototype selection problem. We use two modifications: first, use of random subset selection (to reduce the search space), and second, using multi-armed bandits based identification of an approximately best candidate. This allows

the proposed algorithm ProtoBandit to have an upper bound on the number of similarity comparisons needed. Additionally, we show that the proposed algorithm ProtoBandit is backed by theory of optimal transport by proving its equivalence with the existing SPOTgreedy (Gurumoorthy et al., 2021) algorithm. Through experiments we show the benefit of ProtoBandit over SPOTgreedy, where ProtoBandit achieves an accuracy similar to SPOTgreedy but using much smaller number of incurred similarity comparisons.

Recently, progress has been made in developing MAB approaches in a distributed environment for a class of problems (Mahadik et al., 2020; Li et al., 2016). It would be interesting to see whether such approaches can be applied to ProtoBandit. Furthermore, as the proposed algorithm ProtoBandit is derived from the `BUILD` step of the generalized PAM method, it would be interesting to see whether the `SWAP` step too can be derived from the theory of optimal transport. This remains a future research direction.

References

Ashwinkumar Badanidiyuru and Jan Vondrák. Fast algorithms for maximizing submodular functions. In *ACM-SIAM Symposium on Discrete Algorithms*, SODA '14, page 1497–1514. Society for Industrial and Applied Mathematics, 2014.

Donald A. Berry and Bert Fristedt. *Bandit Problems: Sequential Allocation of Experiments*. Chapman & Hall, 1985.

Jacob Bien and Robert Tibshirani. Hierarchical clustering with prototypes via minimax linkage. *Journal of the American Statistical Association*, 106(495):1075–1084, 2011a.

Jacob Bien and Robert Tibshirani. Prototype selection for interpretable classification. *Ann. Appl. Stat.*, 5(4):2403–2424, 2011b.

Arghya Roy Chaudhuri, Pratik Jawanpuria, and Bamdev Mishra. ProtoBandit: Efficient prototype selection via multi-armed bandits. Technical report, arXiv preprint arXiv:2210.01860, 2022.

Koby Crammer, Ran Gilad-Bachrach, Amir Navot, and Naftali Tishby. Margin analysis of the lvq algorithm. In *NeurIPS*, 2002.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. PAC bounds for multi-armed bandit and Markov Decision Processes. In *Proc. COLT 2002*, pages 255–270. Springer, 2002.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *J. Mach. Learn. Res.*, 7:1079–1105, 2006.

Karthik S. Gurumoorthy, Amit Dhurandhar, Guillermo Cecchi, and Charu Aggarwal. Efficient data representation by selecting prototypes with importance weights. In *IEEE ICDM*, pages 260–269, 2019.

Karthik S. Gurumoorthy, Pratik Jawanpuria, and Bamdev Mishra. SPOT: A framework for selection of prototypes using optimal transport. In *Applied Data Science Track* -

European Conference, ECML PKDD 2021, volume 12978 of *Lecture Notes in Computer Science*, pages 535–551. Springer, 2021.

Avinatan Hassidim, Ron Kupfer, and Yaron Singer. An optimal elimination algorithm for learning a best arm. In *Advances in Neural Information Processing Systems*, volume 33, pages 10788–10798. Curran Associates, Inc., 2020.

Kevin G. Jamieson, Matthew Malloy, Robert Nowak, and Sébastien Bubeck. lil' UCB : An optimal exploration algorithm for multi-armed bandits. In *Proc. COLT 2014*, pages 423–439, 2014.

Shivaram Kalyanakrishnan, Ambuj Tewari, Peter Auer, and Peter Stone. PAC subset selection in stochastic multi-armed bandits. In *Proc. ICML 2012*, pages 655–662. Omnipress, 2012.

Zohar Karnin, Tomer Koren, and Oren Somekh. Almost optimal exploration in multi-armed bandits. In *Proc. ICML 2013*, volume 28, pages 1238–1246. PMLR, 2013.

Leonard Kaufman and Peter J. Rousseeuw. *Partitioning Around Medoids (Program PAM)*, chapter 2, pages 68–125. John Wiley & Sons, Ltd, 1990. ISBN 9780470316801.

Emilie Kaufmann and Shivaram Kalyanakrishnan. Information complexity in bandit subset selection. In *Proc. COLT 2013*, volume 30, pages 228–251. JMLR, 2013.

Been Kim, Cynthia Rudin, and Julie Shah. The Bayesian case model: A generative approach for case-based reasoning and prototype classification. In *NeurIPS*, 2014.

Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In *NeurIPS*, 2016.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *ICML*, 2017.

James Leahy. Targeted consumer involvement: An integral part of successful new product development. *Research-Technology Management*, 56, 07 2013.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

Shuai Li, Alexandros Karatzoglou, and Claudio Gentile. Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 539–548, 2016.

Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. Explainable ai: A review of machine learning interpretability methods. *Entropy*, 23(1), 2021. ISSN 1099-4300.

Kanak Mahadik, Qingyun Wu, Shuai Li, and Amit Sabne. Fast distributed bandits for online recommendation systems. In *Proceedings of the 34th ACM International Conference on Supercomputing*, ICS '20. Association for Computing Machinery, 2020. ISBN 9781450379830.

Shie Mannor and John N. Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *JMLR*, 5:623–648, 2004.

Michel Minoux. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization techniques*, pages 234–243. Springer, 1978.

Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrák, and Andreas Krause. Lazier than lazy greedy. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, page 1812–1818. AAAI Press, 2015.

J Arturo Olvera-López, J Ariel Carrasco-Ochoa, and J Fco Martínez-Trinidad. Prototype selection via prototype relevance. In *Progress in Pattern Recognition, Image Analysis and Applications*, pages 153–160. Springer Berlin Heidelberg, 2008.

L. Rdusseeun and P Kaufman. Clustering by means of medoids. *Statistical Data Analysis Based on the L1 Norm and Related Methods*, pages 405–416, 1987.

Arghya Roy Chaudhuri and Shivaram Kalyanakrishnan. PAC identification of many good arms in stochastic multi-armed bandits. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 991–1000. PMLR, 09–15 Jun 2019.

Himanshu Suyal and Avtar Singh. *Improving multi-label classification in prototype selection scenario*, chapter 6, pages 103–119. John Wiley & Sons, Ltd, 2021. ISBN 9781119818717.

Mo Tiwari, Martin J Zhang, James Mayclin, Sebastian Thrun, Chris Piech, and Ilan Shomorony. Banditpam: Almost linear time k-medoids clustering via multi-armed bandits. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 10211–10222. Curran Associates, Inc., 2020.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. Fast multi-stage submodular maximization. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, page II–1494–II–1502. JMLR.org, 2014.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *ICML*, 2015.

Mark Weiser. Programmers use slices when debugging. *Comm. ACM*, 25(7):446–452, 1982.

Paul Wohlhart, Martin Kostinger, Michael Donoser, Peter M Roth, and Horst Bischof. Optimizing 1-nearest prototype classifiers. In *CVPR*, 2013.

Chih-Kuan Yeh, Joon Kim, Ian En-Hsu Yen, and Pradeep K Ravikumar. Representer point selection for explaining deep neural networks. In *NeurIPS*, 2018.

A. V. Zukhba. NP-completeness of the problem of prototype selection in the nearest neighbor method. *Pattern Recognition and Image Analysis*, 20:484–494, 12 2010.