Partial Wasserstein Covering

Keisuke Kawano, Satoshi Koide, Keisuke Otaki

Toyota Central R&D Labs., Inc.
{kawano, koide, otaki}@mosk.tytlabs.co.jp

Abstract

We consider a general task called *partial Wasserstein covering* with the goal of providing information on what patterns are not being taken into account in a dataset (e.g., dataset used during development) compared with another dataset(e.g., dataset obtained from actual applications). We model this task as a discrete optimization problem with *partial Wasserstein divergence* as an objective function. Although this problem is NP-hard, we prove that it satisfies the submodular property, allowing us to use a greedy algorithm with a 0.63 approximation. However, the greedy algorithm is still inefficient because it requires solving linear programming for each objective function evaluation. To overcome this inefficiency, we propose *quasi-greedy* algorithms that consist of a series of acceleration techniques, such as sensitivity analysis based on strong duality and the so-called C-transform in the optimal transport field. Experimentally, we demonstrate that we can efficiently fill in the gaps between the two datasets and find missing scene in real driving scenes datasets.

1 Introduction

A major challenge in real-world machine learning applications is coping with mismatches between the data distribution obtained in real-world applications and those used for development. Regions in the real-world data distribution that are not well supported in the development data distribution (i.e., regions with low relative densities) result in potential risks such as a lack of evaluation or high generalization error, which in turn leads to low product quality. Our motivation is to provide developers with information on what patterns are not being taken into account when developing products by selecting some of the (usually *unlabeled*) real-world data distribution, also referred to as *application dataset*,[1] to fill in the gaps in the *development dataset*. Note that the term "development" includes choosing models to use, designing subroutines for fail safe, and training/testing models.

Our research question is formulated as follows. To resolve the lack of data density in development datasets, how and using which metric can we select data from application datasets with a limited amount of data for developers to understand? One reasonable approach is to define

[1]More precisely, we call a finite set of data sampled from the real-world data distribution *application dataset* in this study.
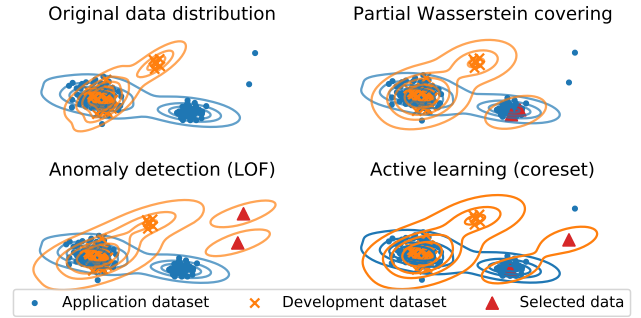

Figure 1: Concept of PWC. PWC extracts some data from an unlabeled application dataset by minimizing the partial Wasserstein divergence between the application dataset and the union of the selected data and a development dataset. PWC focuses on regions in which development data are lacking compared with the application dataset, whereas anomaly detection extracts irregular data, and active learning selects to improve accuracy.

the discrepancy between the data distributions and select data to minimize this discrepancy. The Wasserstein distance has attracted significant attention as a metric for data distributions (Arjovsky, Chintala, and Bottou 2017; Alvarez-Melis and Fusi 2020). However, the Wasserstein distance is not capable of representing the *asymmetric* relationship between application datasets and development datasets, i.e., the parts that are over-included during development increase the Wasserstein distance.

In this paper, we propose *partial Wasserstein covering (PWC)* that selects a limited amount of data from the application dataset by minimizing the partial Wasserstein divergence (Bonneel and Coeurjolly 2019) between the application dataset and the union of the development dataset and the selected data. PWC, as illustrated in Fig. 1, selects data from areas with fewer development data than application data in the data distributions (lower-right area in the blue distribution in Fig. 1) while ignoring areas with sufficient development data (upper-middle of the orange distribution). We also highlight the data selected through an anomaly detection method, LOF (Breunig et al. 2000), where irregular data (upper-right points) were selected, but the major density difference was ignored. Furthermore, we show the selection

obtained using an active learning method, coreset with a k-center (Sener and Savarese 2018), where the data are chosen to improve the accuracy rather than fill the gap in terms of the distribution mismatch.

Our main contributions are summarized as follows.

- We propose PWC that extracts data that are lacking in a development dataset from an application dataset by minimizing the partial Wasserstein divergence between an application dataset and the development dataset.

- We prove that PWC is a maximization problem involving a *submodular* function whose inputs are the set of selected data. This allows us to use a greedy algorithm with a guaranteed approximation ratio.

- Additionally, we propose fast heuristics based on sensitivity analysis and the Sinkhorn derivative for an accelerated computation.

- Experimentally, we demonstrate that compared with baselines, PWC extracts data lacking in the development data distribution from the application distribution more efficiently.

The remainder of this paper is organized as follows. Section 2 briefly introduces the partial Wasserstein divergence, linear programming (LP), and submodular functions. In Sect. 3, we detail the proposed PWC, fast algorithms for approximating the optimization problem, and some theoretical results. Section 4 presents a literature review of related work. We demonstrated the PWC in some numerical experiments in Sect. 5. In Sect. 6, we present our conclusions.

2 Preliminaries

In this section, we introduce the notations used throughout this paper. We then describe partial Wasserstein divergences, sensitivity analysis for LP, and submodular functions.

Notations Vectors and matrices are denoted in bold (e.g., $\mathbf{x}$ and $\mathbf{A}$), where x_i and A_{ij} denote the ith and (i,j)th elements, respectively. To clarify the elements, we use notations such as $\mathbf{x} = (f(i))_i$ and $\mathbf{A} = (g(i,j))_{ij}$, where f and g specify the element values depending on their subscripts. $\langle \cdot, \cdot \rangle$ denotes the inner product of matrices or vectors. $\mathbb{1}_n \in \mathbb{R}^n$ is a vector with all elements equal to one. For a natural number n, we define $[\![n]\!] := \{1, \cdots, n\}$. For a finite set V, we denote its power set as 2^V and its cardinality as $|V|$. The L_2-norm is denoted as $\| \cdot \|$. The delta function is denoted as $\delta(\cdot)$. $\mathbb{R}_+$ is a set of real positive numbers, and $[a, b] \subset \mathbb{R}$ denotes a closed interval. $I[\cdot]$ denotes an indicator function (zero for false and one for true).

Partial Wasserstein divergence In this paper, we consider the *partial Wasserstein divergence* (Figalli 2010; Bonneel and Coeurjolly 2019; Chapel, Alaya, and Gasso 2020) as an objective function. Partial Wasserstein divergence is designed to measure the discrepancy between two distributions with *different total masses* by considering variations in optimal transport. Throughout this paper, datasets are modeled as empirical distributions that are represented by mixtures of delta functions without necessarily having the same total mass. Suppose two empirical distributions (or

datasets) X and Y with probability masses $\mathbf{a} \in \mathbb{R}_+^m$ and $\mathbf{b} \in \mathbb{R}_+^n$, which are denoted as $X = \sum_{i=1}^m a_i \delta(\mathbf{x}^{(i)})$ and $Y = \sum_{j=1}^n b_j \delta(\mathbf{y}^{(j)})$. Without loss of generality, the total mass of Y is greater than or equal to that of X (i.e., $\sum_{i=1}^m a_i = 1$ and $\sum_{j=1}^n b_j \geq 1$).

Based on the definitions above, we define the partial Wasserstein divergence as follows:[2]

$$\mathcal{PW}^2(X, Y) := \min_{\mathbf{P} \in U(\mathbf{a}, \mathbf{b})} \langle \mathbf{P}, \mathbf{C} \rangle, \text{where}$$
$$U(\mathbf{a}, \mathbf{b}) = \{\mathbf{P} \in [0, 1]^{m \times n} \mid \mathbf{P}\mathbb{1}_n = \mathbf{a}, \ \mathbf{P}^\top \mathbb{1}_m \leq \mathbf{b}\}, \quad (1)$$

where $C_{ij} := \|\mathbf{x}^{(i)} - \mathbf{y}^{(j)}\|^2$ is the transport cost between $\mathbf{x}^{(i)}$ and $\mathbf{y}^{(j)}$, and P_{ij} is the amount of mass flowing from $\mathbf{x}^{(i)}$ to $\mathbf{y}^{(j)}$ (to be optimized). Unlike the standard Wasserstein distance, the second constraint in Eq. (1) is not defined with "=", but with "$\leq$". This modification allows us to treat distributions with different total masses. The condition $\mathbf{P}^\top \mathbb{1}_m \leq \mathbf{b}$ indicates that the mass in Y does not need to be transported, whereas the condition $\mathbf{P}\mathbb{1}_n = \mathbf{a}$ indicates that all of the mass in X should be transported without excess or deficiency (just as in the original Wasserstein distance). This property is useful for the problem defined below, which treats datasets with vastly different sizes.

To compute the partial Wasserstein divergence, we must solve the minimization problem in Eq.(1). In this paper, we consider the following two methods. (i) LP using simplex method. (ii) Generalized Sinkhorn iteration with entropy regularization (with a small regularization parameter $\varepsilon > 0$) (Benamou et al. 2015; Peyré and Cuturi 2019).

As will be detailed later, an element in mass $\mathbf{b}$ varies when adding data to the development datasets. A key to our algorithm is to quickly estimate the extent to which the partial Wasserstein divergence will change when an element in $\mathbf{b}$ varies. If we compute $\mathcal{PW}^2$ using LP, we can employ a *sensitivity analysis*, which will be described in the following paragraph. If we use generalized Sinkhorn iterations, we can use automatic differentiation techniques to obtain a partial derivative with respect to b_j (see Sect. 3.3).

LP and sensitivity analysis The sensitivity analysis of LP plays an important role in our algorithm. Given a variable $\mathbf{x} \in \mathbb{R}^m$ and parameters $\mathbf{c} \in \mathbb{R}^m$, $\mathbf{d} \in \mathbb{R}^n$, and $\mathbf{A} \in \mathbb{R}^{n \times m}$, the standard form of LP can be written as follows: $\min \ \mathbf{c}^\top \mathbf{x}$, s.t. $\mathbf{Ax} \leq \mathbf{d}$, $\mathbf{x} \geq 0$. Sensitivity analysis is a framework for estimating changes in a solution when the parameters $\mathbf{c}$, $\mathbf{A}$, and $\mathbf{d}$ of the problem vary. We consider a sensitivity analysis for the right-hand side of the constraint (i.e., $\mathbf{d}$). When d_j changes as $d_j + \Delta d_j$, the optimal value changes by $y_j^* \Delta d_j$ if a change Δd_j in d_j lies within $(\underline{d}_j, \overline{d}_j)$, where $\mathbf{y}^*$ is the optimal solution of the following dual problem corresponding to the primal problem: $\max \ \mathbf{d}^\top \mathbf{y}$ s.t. $\mathbf{A}^\top \mathbf{y} \geq \mathbf{c}$, $\mathbf{y} \geq 0$. We refer readers to (Van-

---
[2]The partial optimal transport problems in the literature contain a wider problem definition than Eq.(1) as summarized in Table 1 (b) of (Bonneel and Coeurjolly 2019), but this paper employs this one-side relaxed Wasserstein divergence corresponding to Table 1 (c) in (Bonneel and Coeurjolly 2019) without loss of generality.

derbei 2015) for the details of calculating the upper bound $\bar{d}_j$ and the lower bound $\underline{d}_i$.

Submodular function   Our covering problem is modeled as a discrete optimization problem involving *submodular functions*, which are a subclass of set functions that play an important role in discrete optimization. A set function $\phi : 2^V \to \mathbb{R}$ is called a *submodular* iff $\phi(S \cup T) + \phi(S \cap T) \leq \phi(S) + \phi(T)$ $(\forall S, T \subseteq V)$. A submodular function is *monotone* iff $\phi(S) \leq \phi(T)$ for $S \subseteq T$. An important property of monotone submodular functions is that a greedy algorithm provides a $(1 - 1/e) \approx 0.63$-approximate solution to the maximization problem under a budget constraint $|S| \leq K$ (Nemhauser, Wolsey, and Fisher 1978). The *contraction* $\tilde{\phi} : 2^V \to \mathbb{R}$ of a (monotone) submodular function ϕ, which is defined as $\tilde{\phi}(S) \coloneqq \phi(S \cup T) - \phi(T)$, where $T \subseteq V$, is also a (monotone) submodular function.

3   Partial Wasserstein covering problem

3.1   Formulation

As discussed in Sect. 1, our goal is to fill in the gap between the application dataset $\mathcal{D}_{\mathrm{app}}$ by adding some data from a candidate dataset $\mathcal{D}_{\mathrm{cand}}$ to a small dataset $\mathcal{D}_{\mathrm{dev}}$. We consider $\mathcal{D}_{\mathrm{cand}} = \mathcal{D}_{\mathrm{app}}$ (i.e., we copy some data in $\mathcal{D}_{\mathrm{app}}$ and add them into D_{dev}) in the above-mentioned scenarios, but we herein consider the most general formulation. We model this task as a discrete optimization problem called the *partial Wasserstein covering problem*.

Given a dataset for development $\mathcal{D}_{\mathrm{dev}} = \{\mathbf{y}^{(j)}\}_{j=1}^{N_{\mathrm{dev}}}$, a dataset obtained from an application $\mathcal{D}_{\mathrm{app}} = \{\mathbf{x}^{(i)}\}_{i=1}^{N_{\mathrm{app}}}$, and a dataset containing candidates for selection $\mathcal{D}_{\mathrm{cand}} = \{\mathbf{s}^{(j)}\}_{j=1}^{N_{\mathrm{cand}}}$, where $N_{\mathrm{app}} \geq N_{\mathrm{dev}}$, the PWC problem is defined as the following optimization:[3]

$$\max_{\substack{S \subseteq \mathcal{D}_{\mathrm{cand}} \\ \text{s.t. } |S| \leq K}} -\mathcal{PW}^2(\mathcal{D}_{\mathrm{app}}, S \cup \mathcal{D}_{\mathrm{dev}}) + \mathcal{PW}^2(\mathcal{D}_{\mathrm{app}}, \mathcal{D}_{\mathrm{dev}}) \quad (2)$$

We select a subset S from the candidate dataset $\mathcal{D}_{\mathrm{cand}}$ under the budget constraint $|S| \leq K \; (\ll N_{\mathrm{cand}})$, and then add that subset to the development data $\mathcal{D}_{\mathrm{dev}}$ to minimize the partial Wasserstein divergence between the two datasets $\mathcal{D}_{\mathrm{app}}$ and $S \cup \mathcal{D}_{\mathrm{dev}}$. The second term is a constant with respect to S, which is included to make the objective non-negative.

In Eq. (2), we must specify the probability mass (i.e., $\mathbf{a}$ and $\mathbf{b}$ in Eq. (1)) for each data point. Here, we employ a uniform mass, which is a natural choice because we do not have prior information regarding each data point. Specifically, we set the weights to $a_i = 1/N_{\mathrm{app}}$ for $\mathcal{D}_{\mathrm{app}}$ and $b_j = 1/N_{\mathrm{dev}}$ for $S \cup \mathcal{D}_{\mathrm{dev}}$. With this choice of masses, we can easily show that $\mathcal{PW}^2(\mathcal{D}_{\mathrm{app}}, S \cup \mathcal{D}_{\mathrm{dev}}) = \mathcal{W}^2(\mathcal{D}_{\mathrm{app}}, \mathcal{D}_{\mathrm{dev}})$ when $S = \emptyset$, where $\mathcal{W}^2(\mathcal{D}_{\mathrm{app}}, \mathcal{D}_{\mathrm{dev}})$ is the Wasserstein distance. Therefore, based on the monotone property (Sect. 3.2), the objective value is non-negative for any S.

---

[3]We herein consider the partial Wasserstein divergence between the unlabled datasets, because the application and candidate datasets are usually unlabeled. If they are labeled, we can use the labels information as in (Alvarez-Melis and Fusi 2020).

The PWC problem in Eq.(2) can be written as a mixed integer linear program (MILP) as follows. Instead of using the divergence between $\mathcal{D}_{\mathrm{app}}$ and $S \cup \mathcal{D}_{\mathrm{dev}}$, we consider the divergence between $\mathcal{D}_{\mathrm{app}}$ and $\mathcal{D}_{\mathrm{cand}} \cup \mathcal{D}_{\mathrm{dev}}$. Hence, the mass $\mathbf{b}$ is an $(N_{\mathrm{cand}} + N_{\mathrm{dev}})$-dimensional vector, where the first N_{cand} elements correspond to the data in $\mathcal{D}_{\mathrm{cand}}$ and the remaining elements correspond to the data in $\mathcal{D}_{\mathrm{dev}}$. In this case, we never transport to data points in $\mathcal{D}_{\mathrm{cand}} \setminus S$, meaning we use the following mass that depends on S:

$$b_j(S) = \begin{cases} \frac{I[\mathbf{s}^{(j)} \in S]}{N_{\mathrm{dev}}}, & \text{if } 1 \leq j \leq N_{\mathrm{cand}} \\ \frac{1}{N_{\mathrm{dev}}}, & \text{if } N_{\mathrm{cand}} + 1 \leq j. \end{cases} \quad (3)$$

As a result, the problem is an MILP problem with an objective function $\langle \mathbf{P}, \mathbf{C} \rangle$ w.r.t. $S \subseteq \mathcal{D}_{\mathrm{cand}}$ and $\mathbf{P} \in U(\mathbf{a}, \mathbf{b}(S))$ with $|S| \leq K$ and $a_i = 1/N_{\mathrm{app}}$.

One may wonder why we use the partial Wasserstein divergence in Eq.(2) instead of the standard Wasserstein distance. This is because the asymmetry in the conditions of the partial Wasserstein divergence enables us to extract only the parts with a density lower than that of the application dataset, whereas the standard Wasserstein distance becomes large for parts with sufficient density in the development dataset. Furthermore, the guaranteed approximation algorithms that we describe later can be utilized only for the partial Wasserstein divergence version.

3.2   Submodularity of the PWC problem

In this section, we prove the following theorem to guarantee the approximation ratio of the proposed algorithms.

Theorem 1. *Given the datasets $X = \{\mathbf{x}^{(i)}\}_{i=1}^{N_x}$ and $Y = \{\mathbf{y}^{(j)}\}_{j=1}^{N_y}$, and a subset of a dataset $S \subseteq \{\mathbf{s}^{(j)}\}_{j=1}^{N_s}$, $\phi(S) = -\mathcal{PW}^2(X, S \cup Y) + \mathcal{PW}^2(X, Y)$ is a monotone submodular function.*

To prove Theorem 1, we reduce our problem to the partial maximum weight matching problem (Bar-Noy and Rabanca 2016), which is known to be submodular. First, we present the following lemmas. The proofs of lemmas are provided in the supplementary materials.

Lemma 1. *Let $\mathbb{Q}$ be the set of all rational numbers, and m and n be positive integers with $n \leq m$. Consider $\mathbf{A} \in \mathbb{Q}^{m \times n}$ and $\mathbf{b} \in \mathbb{Q}^m$. Then, the extreme points $\mathbf{x}^*$ of a convex polytope defined by the linear inequalities $\mathbf{A}\mathbf{x} \leq \mathbf{b}$ are also rational, meaning that $\mathbf{x}^* \in \mathbb{Q}^n$.*

Lemma 2. *Let $l, m,$ and n be positive integers, and $Z = [\![n]\!]$. Given a positive-valued m-by-n matrix $\mathbf{R} > 0$, the following set function $\psi : 2^Z \to \mathbb{R}$ is a submodular function:*

$$\psi(S) = \max_{\mathbf{P} \in U^{\leq}(\mathbb{1}_m/m, \mathbf{b}(S))} \langle \mathbf{R}, \mathbf{P} \rangle,$$
$$\text{where } b_j(S) = \frac{I[j \in S]}{l} \quad \forall j \in [\![n]\!]. \quad (4)$$

Here, $U^{\leq}(\mathbf{a}, \mathbf{b}(S))$ is a set defined by replacing the constraint $\mathbf{P}\mathbb{1}_n = \mathbf{a}$ in Eq. (1) with $\mathbf{P}\mathbb{1}_n \leq \mathbf{a}$.

Lemma 3. *If $|S| \geq l$, there exists an optimal solution $\mathbf{P}^*$ in the maximization problem of ψ satisfying $\mathbf{P}^*\mathbb{1}_n = \frac{\mathbb{1}_m}{m}$ (i.e., $\mathbf{P}^* \coloneqq \arg\max_{\mathbf{P} \in U^{\leq}(\frac{\mathbb{1}_m}{m}, \mathbf{b}(S))} \langle \mathbf{R}, \mathbf{P} \rangle = \arg\max_{\mathbf{P} \in U(\frac{\mathbb{1}_m}{m}, \mathbf{b}(S))} \langle \mathbf{R}, \mathbf{P} \rangle).$*

Proof of Theorem 1. Given $C_{\max} = \max_{i,j} C_{ij} + \gamma$, where $\gamma > 0$, the following is satisfied:

$$
\begin{aligned}
\phi(S) = &\left(-C_{\max} + \max_{\mathbf{P} \in U(\mathbf{a}, \mathbf{b}(S))} \langle \mathbf{R}, \mathbf{P} \rangle \right) \\
&- \left(-C_{\max} + \max_{\mathbf{P} \in U(\mathbf{a}, \mathbf{b}(\emptyset))} \langle \mathbf{R}, \mathbf{P} \rangle \right),
\end{aligned}
\tag{5}
$$

where $\mathbf{R} = (C_{\max} - C_{ij})_{ij} > 0$, $m = N_x$, $n = N_s + N_y$, and $l = N_y$. Here, $|S \cup Y| \geq N_y = l$ and Lemma 3 yield $\phi(S) = \psi(S \cup Y) - \psi(Y)$. Since $\phi(S)$ is a contraction of the submodular function $\psi(S)$ (Lemma 2), $\phi(S) = -\mathcal{PW}^2(X, S \cup Y) + \mathcal{PW}^2(X, Y)$ is a submodular function.

We now prove the monotonicity of ϕ. For $S \subseteq T$, we have $U(\mathbf{a}, \mathbf{b}(S)) \subseteq U(\mathbf{a}, \mathbf{b}(T))$ because $\mathbf{b}(S) \leq \mathbf{b}(T)$. This implies that $\phi(S) \leq \phi(T)$. $\qquad\square$

Finally, we prove Proposition 1 to clarify the computational intractability of the PWC problem.

Proposition 1. *The PWC problem with marginals* $\mathbf{a}$ *and* $\mathbf{b}(S)$ *is NP-hard.*

The proofs of Proposition 1 are provided in the supplementary materials.

3.3 Algorithms

MILP and greedy algorithms The PWC problem in Eq. (2) can be solved directly as an MILP problem. We refer to this method as `PW-MILP`. However, this is extremely inefficient when N_{app}, N_{dev}, and N_{cand} are large.

As a consequence of Theorem 1, a simple greedy algorithm provides a 0.63 approximation because the problem is a type of monotone submodular maximization with a budget constraint $|S| \leq K$ (Nemhauser, Wolsey, and Fisher 1978). Specifically, this greedy algorithm selects data from the candidate dataset $\mathcal{D}_{\mathrm{cand}}$ sequentially in each iteration to maximize ϕ. We consider two baseline methods, called `PW-greedy-LP` and `PW-greedy-ent`. `PW-greedy-LP` solves the LP problem using the simplex method. `PW-greedy-ent` computes the optimal $\mathbf{P}^\star$ using Sinkhorn iterations (Benamou et al. 2015).

Even for the greedy algorithms mentioned above, the computational cost is still high because we need to calculate the partial Wasserstein divergences for all candidates on $\mathcal{D}_{\mathrm{app}}$ in each iteration, yielding a computational time complexity of $O(K \cdot N_{\mathrm{cand}} C_{PW})$. Here, C_{PW} is the complexity of the partial Wasserstein divergence with the simplex method for the LP or Sinkhorn iteration.

To reduce the computational cost, we propose heuristic approaches called *quasi-greedy algorithms*. A high-level description of the quasi-greedy algorithms is provided below. In each step of greedy selection, we use a type of *sensitivity value* that allows us to *estimate* how much the partial Wasserstein divergence varies when adding candidate data, instead of performing the actual computation of divergence. Specifically, if we add a data point $\mathbf{s}^{(j)}$ to S, denoted as $T = \{\mathbf{s}^{(j)}\} \cup S$, we have $\mathbf{b}(T) = \mathbf{b}(S) + \frac{\mathbf{e}_j}{N_{\mathrm{dev}}}$, where $\mathbf{e}_j$ is a one-hot vector with the corresponding jth element activated. Hence, we can estimate the change in the divergence for data addition by computing the sensitivity with respect to $\mathbf{b}(S)$. If efficient sensitivity computation is possible, we can speed

up the greedy algorithms. We describe the concrete methods for each of the approaches, simplex method, and Sinkhorn iterations in the following paragraphs.

Quasi-greedy algorithm for the simplex method When we use the simplex method for the partial Wasserstein divergence computation, we employ a sensitivity analysis for LP. The dual problem of partial Wasserstein divergence computation is defined as follows.

$$
\begin{aligned}
\max_{\mathbf{f} \in \mathbb{R}^{N_{\mathrm{app}}}, \mathbf{g} \in \mathbb{R}^{N_{\mathrm{cand}} + N_{\mathrm{dev}}}} \quad & \langle \mathbf{f}, \mathbf{a} \rangle + \langle \mathbf{g}, \mathbf{b}(S) \rangle, \\
\text{s.t. } & g_j \leq 0, \ f_i + g_j \leq C_{ij}, \forall i, j,
\end{aligned}
\tag{6}
$$

where $\mathbf{f}$ and $\mathbf{g}$ are dual variables. Using sensitivity analysis, the changes in the partial Wasserstein divergences can be estimated as $g_j^* \Delta b_j$, where g_j^* is an optimal dual solution of Eq. (6), and Δb_j is the change in b_j. It is worth noting that Δb_j now corresponds to $I[\mathbf{s}^{(j)} \in S]$ in Eq. (3), and a smaller Eq. (6) results in a larger Eq. (2). Thus, we propose a heuristic algorithm that iteratively selects $\mathbf{s}^{(j^*)}$ satisfying $j^* = \arg\min_{j \in [\![N_{\mathrm{cand}}]\!], s^{(j)} \notin S} \ g_j^{*(t)}$ at iteration t, where $g_j^{*(t)}$ is the optimal dual solution at t. We emphasize that we have $-\frac{1}{N_{\mathrm{dev}}} g_j^* = \phi(\{\mathbf{s}^{(j)}\} \cup S) - \phi(S)$ as long as $\overline{b}_j - b_j \geq 1/N_{\mathrm{dev}}$ holds, where $\overline{b}_j$ is the upper bound obtained from the sensitivity analysis, leading to the same selection as that of the greedy algorithm. The computational complexity of the heuristic algorithm is $O(K \cdot C_{PW})$ because we can obtain $\mathbf{g}^*$ by solving the dual simplex. It should be noted that we add a small value to b_j when $b_j = 0$ to avoid undefined values in $\mathbf{g}$. We refer to this algorithm as `PW-sensitivity-LP` and summarize the overall algorithm in Algorithm 1.

Algorithm 1: Data selection for the PWC problem with sensitivity analysis (`PW-sensitivity-LP`)

1: **Input:** $\mathcal{D}_{\mathrm{app}}, \mathcal{D}_{\mathrm{dev}}, \mathcal{D}_{\mathrm{cand}}$
2: **Output:** S
3: $S \leftarrow \{\}, t \leftarrow 0$
4: **while** $|S| < K$ **do**
5: $\quad$ Calculate $\mathcal{PW}^2(\mathcal{D}_{\mathrm{app}}, S \cup \mathcal{D}_{\mathrm{dev}})$, and obtain $g_j^{*(t)}$ from the sensitivity analysis.
6: $\quad j^* = \arg\min_{j \in [\![N_{\mathrm{cand}}]\!], \mathbf{s}^{(j)} \notin S} \ g_j^{*(t)}$
7: $\quad S \leftarrow S \cup \{\mathbf{s}^{(j^*)}\}$
8: $\quad t \leftarrow t + 1$
9: **end while**

For computational efficiency, we use the solution matrix $\mathbf{P}$ from the previous step for the initial value in the simplex algorithm in the current step. Any feasible solution $\mathbf{P} \in U(\mathbf{a}, \mathbf{b}(S))$ is also feasible because $\mathbf{P} \in U(\mathbf{a}, \mathbf{b}(T))$ for all $S \subseteq T$. The previous solutions of the dual forms $\mathbf{f}$ and $\mathbf{g}$ are also utilized for the initialization of the dual simplex and Sinkhorn iterations.

Faster heuristic using C-transform The heuristics above still require solving a large LP in each step, where the number of dual constraints is $N_{\mathrm{app}} \times (N_{\mathrm{cand}} + N_{\mathrm{dev}})$.

Here, we aim to reduce the size of the constraints in the LP to $N_{\text{app}} \times (|S| + N_{\text{dev}}) \ll N_{\text{app}} \times (N_{\text{cand}} + N_{\text{dev}})$ using C-transform (Sect.3.1 in (Peyré and Cuturi 2019)).

To derive the algorithm, we consider the dual Eq.(6). Considering the fact that $b_j(S) = 0$ for $\mathbf{s}^{(j)} \notin S$, we first solve the smaller LP problem by ignoring j such that $\mathbf{s}^{(j)} \notin S$, whose system size is $N_{\text{app}} \times (|S| + N_{\text{dev}})$. Let the optimal dual solution of this smaller LP be $(\mathbf{f}^*, \mathbf{g}^*)$, where $\mathbf{f}^* \in \mathbb{R}^{N_{\text{app}}}, \mathbf{g}^* \in \mathbb{R}^{|S|+N_{\text{dev}}}$. For each $\mathbf{s}^{(j)} \notin S$, we consider an LP in which $\mathbf{s}^{(j)}$ is added to S. Instead of solving each LP such as `PW-greedy-LP`, we approximate the optimal solution using a technique called the C-transform. More specifically, for each j such that $\mathbf{s}^{(j)} \notin S$, we only optimize g_j and fix the other variables to be the dual optimal above. This is done by $g_j^{\mathbf{C}} := \min\{0, \min_{i \in [\![N_{\text{dev}}]\!]} C_{ij} - f_i^*\}$. Note that this is the largest value of g_j, satisfying the dual constraints. This $g_j^{\mathbf{C}}$ gives the estimated increase in the objective function when $\mathbf{s}^{(j)} \notin S$ is added to S. Finally, we select the instance $j^* = \arg\min_{j \in [\![N_{\text{cand}}]\!], \mathbf{s}^{(j)} \notin S} g_j^{\mathbf{C}}$. As shown later, this heuristic experimentally works efficiently in terms of computational times. We refer to this algorithm as `PW-sensitivity-Ctrans`.

Quasi-greedy algorithm for Sinkhorn iteration When we use the generalized Sinkhorn iterations, we can easily obtain the derivative of the entropy-regularized partial Wasserstein divergence $\nabla_j := \partial \mathcal{PW}_\epsilon^2(\mathcal{D}_{\text{app}}, S \cup \mathcal{D}_{\text{dev}})/\partial b_j$, where $\mathcal{PW}_\epsilon^2$ denotes the partial Wasserstein divergence, using automatic differentiation techniques such as those provided by PyTorch (Paszke et al. 2019). Based on this derivative, we can derive a quasi-greedy algorithm for the Sinkhorn iteration as $j^* = \arg\min_{j \in [\![N_{\text{cand}}]\!], \mathbf{s}^{(j)} \notin S} \nabla_j$ (i.e., we replace Lines 6 and 7 of Algorithm 1 with this formula). Because the derivative can be obtained with the same computational complexity as the Sinkhorn iterations, the overall complexity is $O(K \cdot C_{PW})$. We refer this algorithm `PW-sensitivity-ent`.

4 Related work

Instance selection and data summarization Tasks for selecting an important portion of a large dataset are common in machine learning. *Instance selection* involves extracting a subset of a training dataset while maintaining the target task performance. Accoring to (Olvera-López et al. 2010), two types of instance selection methods have been studied: model-dependent methods (Hart 1968; Ritter 1975; Chou, Kuo, and Chang 2006; Wilson 1972; Vázquez, Sánchez, and Pla 2005) and label-dependent methods (Wilson and Martinez 2000; Brighton and Mellish 2002; Riquelme, Aguilar-Ruiz, and Toro 2003; Bezdek and Kuncheva 2001; Liu and Motoda 2002; Spillmann et al. 2006). Unlike instance selection methods, PWCs do not depend on either model or ground-truth labels. *Data summarization* involves finding representative elements in a large dataset (Ahmed 2019). Here, the submodularity plays an important role (Mitrovic et al. 2018; Lin and Bilmes 2011; Mirzasoleiman, Zadimoghaddam, and Karbasi 2016; Tschiatschek et al. 2014).

Unlike data summarization, PWC focuses on filling in gaps between datasets.

Anomaly detection Our covering problem can be considered as the task of selecting a subset of a dataset that is not included in the development dataset. In this regard, our method is related to anomaly detectors e.g., LOF (Breunig et al. 2000), one-class SVM (Schölkopf et al. 1999), and deep learning-based methods (Kim et al. 2019; An and Cho 2015). However, our goal is to extract patterns that are less included in the development dataset, but have a certain volume in the application, rather than selecting rare data that would be considered as observation noise or infrequent patterns that are not as important as frequent patterns.

Active learning The problem of selecting data from an unlabeled dataset is also considered in active learning. A typical approach usees the output of the target model e.g., the entropy-based method (Holub, Perona, and Burl 2008; Coletta et al. 2019), whereas some methods are independent of the output of the model e.g., the k-center-based (Sener and Savarese 2018), and Wasserstein distance-based approach (Shui et al. 2020). In contrast, our goal is finding unconsidered patterns during development, even if they do not contribute to improving the prediction performance by directly adding them to the training data; developers may use these unconsidered data to test the generalization performance, design subroutines to process the patterns, redevelop a new model, or alert users to not use the products in certain special situations. Furthermore, we emphasize that the submodularity and guaranteed approximation algorithms can be used only with the partial Wasserstein divergence, but not with the vanilla Wasserstein distance.

Facility location problem (FLP) and k-median The FLP and related k-*median* (ReVelle and Swain 1970) are similar to our problem in the sense that we select data $S \subseteq \mathcal{D}_{\text{cand}}$ to minimize an objective function. FLP is a mathematical model for finding a desirable location for facilities (e.g., stations, schools, and hospitals). While FLP models facility opening costs, our data covering problem does not consider data selection costs, making it more similar to k-median problems. However, unlike the k-median, our covering problem allows relaxed transportation for distribution, which is known as *Kantorovich relaxation* in the optimal transport field (Peyré and Cuturi 2019). Hence, our approach using partial Wasserstein divergence enables us to model more flexible assignments among distributions beyond naïve assignments among data.

5 Experiments

In our experiments, we considered a scenario in which we wished to select some data in the application dataset $\mathcal{D}_{\text{app}}$ for the PWC problem (i.e., $\mathcal{D}_{\text{app}} = \mathcal{D}_{\text{cand}}$). For `PW-*-LP` and `PW-sensitivity-Ctrans`, we used IBM ILOG CPLEX 20.01 (IBM ILOG CPLEX Optimization Studio 2013) for the dual simplex algorithm, where the sensitivity analysis for LP is also available. For `PW-*-ent`, we computed the entropic regularized version of the partial Wasser-
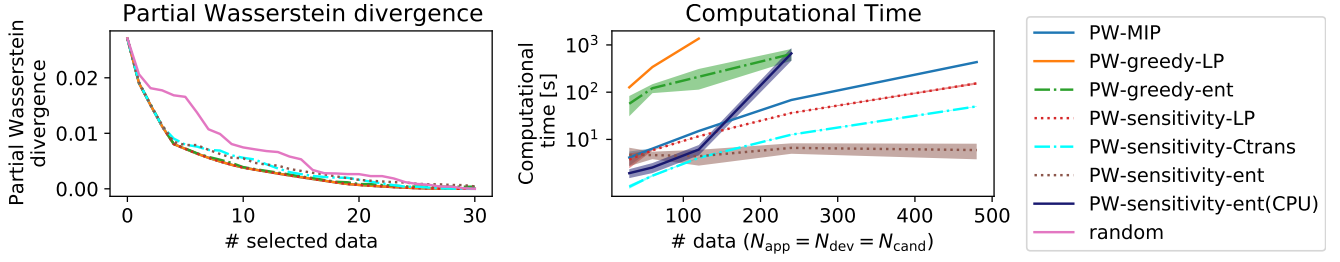
Figure 2: Partial Wasserstein divergence $\mathcal{PW}^2(\mathcal{D}_{\text{app}}, S \cup \mathcal{D}_{\text{dev}})$ when data are sequentially selected and added to S (left). Computational time, where colored areas indicate standard deviations (right).
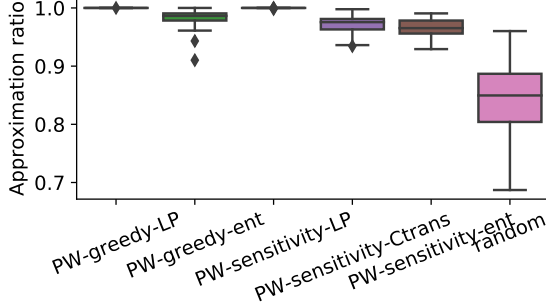

Figure 3: Approximation ratios evaluated empirically for $K = 15$ and 50 random instances.
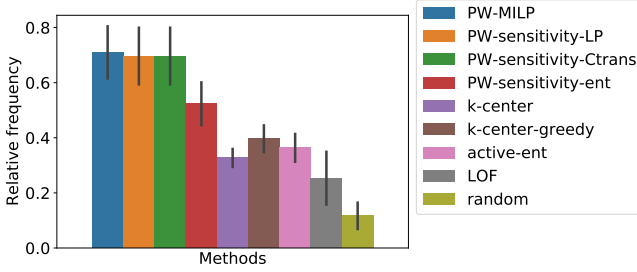

Figure 4: Relative frequency of missing category in selected data. The black bars denote the standard deviations.

stein divergence using PyTorch (Paszke et al. 2019) on a GPU. We computed the sensitivity using automatic differentiation. We set the coefficient for entropy regularization $\epsilon = 0.01$ and terminated the Sinkhorn iterations when the divergence did not change by at least $\max_{i,j} C_{ij} \times 10^{-12}$ compared with the previous step, or the number of iterations reached 50,000. All experiments were conducted with an Intel® Xeon® Gold 6142 CPU and an NVIDIA® TITAN RTX™ GPU.

5.1 Comparison of algorithms

First, we compare our algorithms, namely, the greedy-based PW-greedy-*, sensitivity-based PW-sensitivity-*, and random selection for the baseline. For the evaluation, we generated two-dimensional random values following a normal distribution for $\mathcal{D}_{\text{app}}$ and $\mathcal{D}_{\text{dev}}$, respectively. Figure 2 (left) presents the partial Wasserstein divergences be-

tween $\mathcal{D}_{\text{app}}$ and $S \cup \mathcal{D}_{\text{dev}}$ while varying the number of the selected data when $N_{\text{app}} = N_{\text{dev}} = 30$ and Fig. 2 (right) presents the computational times when $K = 30$ and $N_{\text{app}} = N_{\text{dev}} = N_{\text{cand}}$ varied. As shown in Fig. 2 (left), PW-greedy-LP, and PW-sensitivity-LP select exactly the same data as the global optimal solution obtained by PW-MILP. As shown in Fig. 2 (right), considering the logarithmic scale, PW-sensitivity-* significantly reduces the computational time compared with PW-MILP, while the naïve greedy algorithms do not scale. In particular, PW-sensitivity-ent can be calculated quickly as long as the GPU memory is sufficient, whereas its CPU version (PW-sensitivity-ent(CPU)) is significantly slower. PW-sensitivity-Ctrans is the fastest among the methods without GPUs. It is 8.67 and 3.07 times faster than PW-MILP and PW-sensitivity-LP, respectively. For quantitative evaluation, we define the *empirical approximation ratio* as $\phi(S_K)/\phi(S_K^*)$, where S_K^* is the optimal solution obtained by PW-MILP when $|S| = K$. Figure 3 shows the empirical approximation ratio ($N_{\text{app}} = N_{\text{dev}} = 30, K = 15$) for 50 different random seeds. Both PW-*-LP achieve approximation ratios close to one, whereas PW-*-ent and PW-sensitivity-Ctrans have slightly lower approximation ratios[4].

5.2 Finding a missing category

For the quantitative evaluation of the missing pattern findings, we herein consider a scenario in which a category (i.e., label) is less included in the development dataset than in the application dataset[5]. We employ subsets of the MNIST dataset (LeCun and Cortes 2010), where the development dataset contains 0 labels at a rate of 0.5%, whereas all labels are included in the application data in equal ratios (i.e., 10%). We randomly sampled 500 images from the validation split for each $\mathcal{D}_{\text{app}}$ and $\mathcal{D}_{\text{dev}}$. We employ the squared L^2 distance on the pixel space as the ground metric for our method. For baseline methods, we employed LOF (Breunig et al. 2000) novelty detection provided by scikit-learn (Pedregosa et al. 2011) with default parameters. For the LOF, we selected top-K data according to the anomaly scores.

---

[4]An important feature of PW-*-ent is that they can be executed without using any commercial solvers.

[5]Note that in real scenarios, missing patterns do not always correspond to labels.

We also employed active learning approaches based on entropies of the prediction (Holub, Perona, and Burl 2008; Coletta et al. 2019) and coreset with the k-center and k-center greedy (Sener and Savarese 2018; Bachem, Lucic, and Krause 2017). For the entropy-based method, we train a LeNet-like neural network using the training split, and then select K data with high entropies of the predictions from the application dataset. LOF and coreset algorithms are conducted on the pixel space.

We conducted 10 trials using different random seeds for the proposed algorithms and baselines, except for the `PW-greedy-*` algorithms, because they took over 24h. Figure 4 presents a histogram of the selected labels when $K = 30$, where x-axis corresponds to labels of selected data (i.e., 0 or not 0) and y-axis shows relative frequencies of selected data. The proposed methods (i.e., `PW-*`) extract more data corresponding to the label 0 (0.71 for `PW-MILP`) than the baselines. We can conclude that the proposed methods successfully selected data from the candidate dataset $\mathcal{D}_{cand}(= \mathcal{D}_{app})$ to fill in the missing areas in the distribution.

5.3 Missing scene extraction in driving datasets

Finally, we demonstrate PWC in a realistic scenario, using driving scene images. We adopted two datasets, BDD100K (Yu et al. 2020) and KITTI (Object Detection Evaluation 2012) (Geiger et al. 2013) as the application and development datasets, respectively. The major difference between these two datasets is that KITTI ($\mathcal{D}_{dev}$) contains only daytime images, whereas BDD100k ($\mathcal{D}_{app}$) contains both daytime and nighttime images. To reduce computational time, we randomly selected 1,500 data points for the development dataset from the test split of the KITTI dataset and 3,000 data points for the application dataset from the test split of BDD100k. To compute the transport costs between images, we calculated the squared L_2 norms between the feature vectors extracted by a pretrained ResNet with 50 layers (He et al. 2016) obtained from Torchvision (Marcel and Rodriguez 2010). Before inputting the images into the ResNet, each image was resized to a height of 224 pixels and then center-cropped to a width of 224, followed by normalization. As baseline methods, we adopted the LOF (Breunig et al. 2000) and coreset with the k-center greedy (Sener and Savarese 2018). Figure 5 presents the obtained top-3 images. One can see that PWC (`PW-sensitivity-LP`) selects the major pattern (i.e., nighttime scenes) that is not included in the development data, whereas LOF and coreset mainly extracts specific rare scenes (e.g., a truck crossing a street or out-of-focus scene). The coreset selects a single image of nighttime scenes; however, we emphasize that providing multiple images is essential for the developers to understand what patterns in the image (e.g., nighttime, type of cars, or roadside) are less included in the image.

The above results indicate that the PWC problem enables us to accelerate updating ML-based systems when the distributions of application and development data are different because our method does not focus on isolated anomalies but major missing patterns in the application data, as shown in Fig. 1 and Fig. 5. The ML workflow using our method can efficiently find such patterns and allow developers to incre-
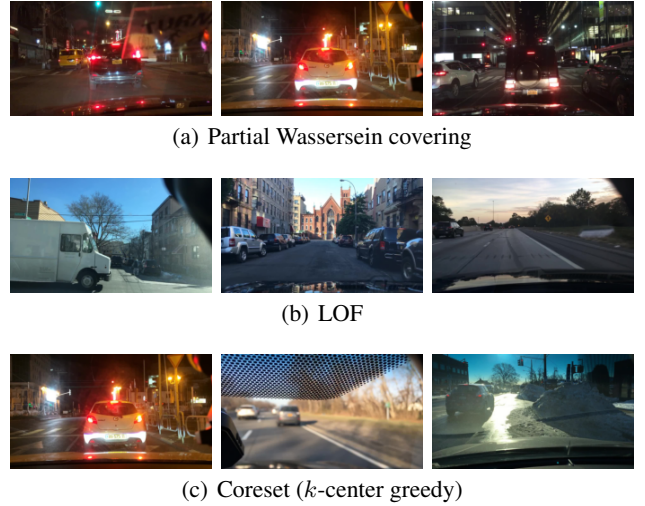

(a) Partial Wassersein covering

(b) LOF

(c) Coreset (k-center greedy)

Figure 5: Covering results when the application dataset (BDD100k) contains nighttime scenes, but the development dataset (KITTI) does not. PWC (`PW-sensitivity-LP`) extracts the major difference between the two datasets (i.e., nighttime images), whereas LOF and coreset (k-center greedy) mainly extract some rare cases.

mentally evaluate and update the ML-based systems (e.g., test the generalization performance, redevelop a new model, and designing a special subroutine for the pattern). Identifying patterns that are not taken into account during development and addressing them individually can improve the reliability and generalization ability of ML-based systems, such as image recognition in autonomous vehicles.

6 Conclusion

In this paper, we proposed the PWC, which fills in the gaps between development datasets and application datasets based on partial Wasserstein divergence. We also proved the submodularity of the PWC, leading to a greedy algorithm with the guaranteed approximation ratio. In addition, we proposed quasi-greedy algorithms based on sensitivity analysis and derivatives of Sinkhorn iterations. Experimentally, we demonstrated that the proposed method covers the major areas of development datasets with densities lower than those of the corresponding areas in application datasets. The main limitation of the PWC is scalability; the space complexity of the dual simplex or Sinkhorn iteration is at least $O(N_{app} \cdot N_{dev})$, which might require approximation, e.g., stochastic optimizations (Aude et al. 2016), slicing (Bonneel and Coeurjolly 2019) and neural networks (Xie et al. 2019)). As a potential risk, if infrequent patterns in application datasets are as important as frequent patterns, our proposed method may ignore some important cases, and it may be desirable to use our covering with methods focusing on fairness.

References

Ahmed, M. 2019. Data summarization: a survey. *Knowledge and Information Systems*, 58(2): 249–273.

Alvarez-Melis, D.; and Fusi, N. 2020. Geometric Dataset Distances via Optimal Transport. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M. F.; and Lin, H., eds., *Advances in Neural Information Processing Systems 33*, volume 33, 21428–21439. Curran Associates, Inc.

An, J.; and Cho, S. 2015. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2(1): 1–18.

Arjovsky, M.; Chintala, S.; and Bottou, L. 2017. Wasserstein Generative Adversarial Networks. In Precup, D.; and Teh, Y. W., eds., *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, 214–223. PMLR.

Aude, G.; Cuturi, M.; Peyré, G.; and Bach, F. 2016. Stochastic optimization for large-scale optimal transport. *arXiv preprint arXiv:1605.08527*.

Bachem, O.; Lucic, M.; and Krause, A. 2017. Practical coreset constructions for machine learning. *arXiv preprint arXiv:1703.06476*.

Bar-Noy, A.; and Rabanca, G. 2016. Tight approximation bounds for the seminar assignment problem. In *International Workshop on Approximation and Online Algorithms*, 170–182. Springer.

Benamou, J.-D.; Carlier, G.; Cuturi, M.; Nenna, L.; and Peyré, G. 2015. Iterative Bregman projections for regularized transportation problems. *SIAM Journal on Scientific Computing*, 37(2): A1111–A1138.

Bezdek, J. C.; and Kuncheva, L. I. 2001. Nearest prototype classifier designs: An experimental study. *International Journal of Intelligent Systems*, 16(12): 1445–1473.

Bonneel, N.; and Coeurjolly, D. 2019. SPOT: sliced partial optimal transport. *ACM Transactions on Graphics (TOG)*, 38(4): 1–13.

Breunig, M. M.; Kriegel, H.-P.; Ng, R. T.; and Sander, J. 2000. LOF: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, 93–104.

Brighton, H.; and Mellish, C. 2002. Advances in instance selection for instance-based learning algorithms. *Data Mining and Knowledge Discovery*, 6(2): 153–172.

Chapel, L.; Alaya, M.; and Gasso, G. 2020. Partial optimal transport with applications on positive-unlabeled learning. In *Advances in Neural Information Processing Systems 33*, 2903–2913.

Chou, C.-H.; Kuo, B.-H.; and Chang, F. 2006. The generalized condensed nearest neighbor rule as a data reduction method. In *Proceedings of the 18th International Conference on Pattern Recognition*, volume 2, 556–559. IEEE.

Coletta, L. F.; Ponti, M.; Hruschka, E. R.; Acharya, A.; and Ghosh, J. 2019. Combining clustering and active learning for the detection and learning of new image classes. *Neurocomputing*, 358: 150–165.

Figalli, A. 2010. The optimal partial transport problem. *Archive for rational mechanics and analysis*, 195(2): 533–560.

Geiger, A.; Lenz, P.; Stiller, C.; and Urtasun, R. 2013. Vision meets Robotics: The KITTI Dataset. *International Journal of Robotics Research*.

Hart, P. 1968. The condensed nearest neighbor rule (corresp.). *IEEE transactions on information theory*, 14(3): 515–516.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Holub, A.; Perona, P.; and Burl, M. C. 2008. Entropy-based active learning for object recognition. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 1–8. IEEE.

IBM ILOG CPLEX Optimization Studio. 2013. CPLEX User's manual. *Vers*, 12: 207–258.

Kim, K. H.; Shim, S.; Lim, Y.; Jeon, J.; Choi, J.; Kim, B.; and Yoon, A. S. 2019. Rapp: Novelty detection with reconstruction along projection pathway. In *International Conference on Learning Representations*.

LeCun, Y.; and Cortes, C. 2010. MNIST handwritten digit database.

Lin, H.; and Bilmes, J. 2011. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 510–520.

Liu, H.; and Motoda, H. 2002. On issues of instance selection. *Data Mining and Knowledge Discovery*, 6(2): 115–130.

Marcel, S.; and Rodriguez, Y. 2010. Torchvision the machine-vision package of torch. In *Proceedings of the 18th ACM International Conference on Multimedia*, 1485–1488.

Mirzasoleiman, B.; Zadimoghaddam, M.; and Karbasi, A. 2016. Fast Distributed Submodular Cover: Public-Private Data Summarization. In *Advances in Neural Information Processing Systems 29*, 3594–3602.

Mitrovic, M.; Kazemi, E.; Zadimoghaddam, M.; and Karbasi, A. 2018. Data summarization at scale: A two-stage submodular approach. In *Proceedings of the 35th International Conference on Machine Learning*, 3596–3605. PMLR.

Nemhauser, G. L.; Wolsey, L. A.; and Fisher, M. L. 1978. An analysis of approximations for maximizing submodular set functions—I. *Mathematical programming*, 14(1): 265–294.

Olvera-López, J. A.; Carrasco-Ochoa, J. A.; Martínez-Trinidad, J. F.; and Kittler, J. 2010. A review of instance selection methods. *Artificial Intelligence Review*, 34(2): 133–143.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Köpf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative

Style, High-Performance Deep Learning Library. In Wallach, H. M.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E. B.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32*, 8024–8035.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; and Duchesnay, E. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830.

Peyré, G.; and Cuturi, M. 2019. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6): 355–607.

ReVelle, C. S.; and Swain, R. W. 1970. Central facilities location. *Geographical analysis*, 2(1): 30–42.

Riquelme, J. C.; Aguilar-Ruiz, J. S.; and Toro, M. 2003. Finding representative patterns with ordered projections. *Pattern Recognition*, 36(4): 1009–1018.

Ritter, G. 1975. An Algorithm for a selective nearest neighbor decision rule. *IEEE Trans. Information Theory*, 21(11): 665–669.

Schölkopf, B.; Williamson, R. C.; Smola, A. J.; Shawe-Taylor, J.; Platt, J. C.; et al. 1999. Support vector method for novelty detection. In *Advances in Neural Information Processing Systems 12*, volume 12, 582–588. Citeseer.

Sener, O.; and Savarese, S. 2018. Active Learning for Convolutional Neural Networks: A Core-Set Approach. In *International Conference on Learning Representations*.

Shui, C.; Zhou, F.; Gagné, C.; and Wang, B. 2020. Deep active learning: Unified and principled method for query and training. In *International Conference on Artificial Intelligence and Statistics*, 1308–1318. PMLR.

Spillmann, B.; Neuhaus, M.; Bunke, H.; Pekalska, E.; and Duin, R. P. 2006. Transforming strings to vector spaces using prototype selection. In *Joint IAPR international workshops on statistical techniques in pattern recognition (SPR) and structural and syntactic pattern recognition (SSPR)*, 287–296. Springer.

Tschiatschek, S.; Iyer, R. K.; Wei, H.; and Bilmes, J. A. 2014. Learning mixtures of submodular functions for image collection summarization. In *Advances in Neural Information Processing Systems 27*, 1413–1421.

Vanderbei, R. J. 2015. *Linear programming*, volume 3. Springer.

Vázquez, F.; Sánchez, J. S.; and Pla, F. 2005. A stochastic approach to Wilson's editing algorithm. In *Iberian conference on pattern recognition and image analysis*, 35–42. Springer.

Wilson, D. L. 1972. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, (3): 408–421.

Wilson, D. R.; and Martinez, T. R. 2000. Reduction techniques for instance-based learning algorithms. *Machine Learning*, 38(3): 257–286.

Xie, Y.; Chen, M.; Jiang, H.; Zhao, T.; and Zha, H. 2019. On scalable and efficient computation of large scale optimal transport. In *International Conference on Machine Learning*, 6882–6892. PMLR.

Yu, F.; Chen, H.; Wang, X.; Xian, W.; Chen, Y.; Liu, F.; Madhavan, V.; and Darrell, T. 2020. BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*.