Fully Dynamic Algorithms for Chamfer Distance[*]

Gramoz Goranci
Faculty of Computer Science
University of Vienna, Austria
`gramoz.goranci@univie.ac.at`

Shaofeng H.-C. Jiang
School of Computer Science
Peking University, China
`shaofeng.jiang@pku.edu.cn`

Peter Kiss
Faculty of Computer Science
University of Vienna, Austria
`peter.kiss@univie.ac.at`

Eva Szilagyi
Faculty of Computer Science
UniVie Doctoral School Computer Science DoCS
University of Vienna, Austria
`eva.szilagyi@univie.ac.at`

Qiaoyuan Yang
School of Computer Science
Peking University, China
`qiaoyuanyang@stu.pku.edu.cn`

Abstract

We study the problem of computing Chamfer distance in the fully dynamic setting, where two set of points $A, B \subset \mathbb{R}^d$, each of size up to n, dynamically evolve through point insertions or deletions and the goal is to efficiently maintain an approximation to $\mathrm{dist}_{\mathrm{CH}}(A, B) = \sum_{a \in A} \min_{b \in B} \mathrm{dist}(a, b)$, where dist is a distance measure. Chamfer distance is a widely used dissimilarity metric for point clouds, with many practical applications that require repeated evaluation on dynamically changing datasets, e.g., when used as a loss function in machine learning. In this paper, we present the first dynamic algorithm for maintaining an approximation of the Chamfer distance under the ℓ_p norm for $p \in \{1, 2\}$. Our algorithm reduces to approximate nearest neighbor (ANN) search with little overhead. Plugging in standard ANN bounds, we obtain $(1+\epsilon)$-approximation in $\tilde{O}(\epsilon^{-d})$ update time and $O(1/\epsilon)$-approximation in $\tilde{O}(dn^{\epsilon^2}\epsilon^{-4})$ update time. We evaluate our method on real-world datasets and demonstrate that it performs competitively against natural baselines.

1 Introduction

We consider the problem of computing the Chamfer distance, a popular dissimarlity metric between point clouds. Given two sets of points $A, B \subset \mathbb{R}^d$, each of size up to n, the Chamfer distance of A from B with respect to a distance measure $\mathrm{dist} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ is defined as

$$\mathrm{dist}_{\mathrm{CH}}(A, B) = \sum_{a \in A} \min_{b \in B} \mathrm{dist}(a, b).$$

The Chamfer distance is typically defined with respect to distance measures such as Manhattan and Euclidean metrics. It has found a wide range of applications in various domains, including machine learning [KSKW15, WCL+19], computer vision [AS03, LJG05, FSG17, JSQJ18], and geometric

---

[*] Authors are listed in the alphabetical order.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

computing [HSS$^+$24]. Due to its strong empirical performance, the Chamfer distance is often used as a computationally efficient alternative to the more demanding Earth-Mover's distance (EMD) [KSKW15, AM19].

In many practical applications, the Chamfer distance is repeatedly calculated on evolving datasets. A notable example is cloud completion and up-scaling, where the models aim to reconstruct missing regions or enhance the resolution of 3D point clouds. In such tasks, the Chamfer distance is commonly used as a loss function during training and it must be evaluated continously as the model's predictions evolve [LYH$^+$23, LLZ$^+$24, WPZ$^+$21]. Two other mainstream use cases include (1) object reconstruction from video sequences, where the objective is to represent objects as point clouds based on observations from a moving camera [RLT$^+$20, TMPF22, HHT$^+$23]; and (2) medical imaging, where it is used to track anatomical structures (such as heart motion on ultrasound images) over time [VHK94, HB91, MDJT14].

Motivated by these applications, we pose the following fundamental question: *Can the Chamfer distance be maintained under dynamically evolving point sets?* More concretely, consider two input sets A an B that undergo point insertions or deletions, referred to as *updates*. The goal is to design a dynamic algorithm that *efficiently* supports these updates while maintaining an estimate that approximates the Chamfer distance up to a small relative error. A naive solution to handle such updates is to recompute the Chamfer distance from scratch after each update. However, this is computationally prohibitive: the best-known static algorithms require either (1) $O(n^2 \cdot d)$ time for exact computation or (2) $O(nd \log n\epsilon^{-2})$ [BIJ$^+$23], for a $(1+\epsilon)$-approximation when the underlying metric is an ℓ_p norm for $p \in \{1,2\}$. In summary, even for low-dimensional datasets, these off-the-shelf static algorithms cannot go beyond the linear time barrier for handling updates.

In this paper, we obtain the first dynamic algorithm for maintaining an estimate to the Chamfer distance under the ℓ_p norm for $p \in \{1,2\}$, which significantly outperforms the linear-time update barrier. Our problem reduces to nearest neighbour (NN) oracles: given parameters $\alpha > 0, \tau \geq 1$, there is a data structure that maintains a dynamic point-set $B \subset \mathbb{R}^d$ and supports in τ time the following opertions (i) insert/delete a point in B and (ii) given a point x, return an $(1+\Theta(\alpha))$-approximate nearest neighbour of x in B. We call such a data structure an $(1+\Theta(\alpha))$-approximate NN oracle with time parameter τ. Plugging in known bounds for NN oracles, our algorithm achieves constant (or even $(1+\epsilon)$) approximation and supports very fast updates across different parameter regimes. The guarantees of our algorithmic reduction are summarized in the theorem below.

Theorem 1.1. *Let A, B be two set of points from $\mathbb{R}^d$, with $|A|, |B| \leq n$, and let $\epsilon \in (0,1)$ $\alpha > 0$, and $\tau \geq 1$ be parameters. Assume that there is an $(1+\Theta(\alpha))$-approximate NN oracle with time parameter τ. Then there is a dynamic algorithm that supports insertions and deletions of points to A and B in $\tilde{O}(\tau)$ worst-case time per update, and when queried, with high probability, it returns a $(1+\alpha+\epsilon)$-approximation to $\mathrm{dist}_{\mathrm{CH}}(A,B)$ in $\tilde{O}((d+\tau)\epsilon^{-2}\max\{1,\alpha^2\})$ time, when the underlying metric is the Manhattan (ℓ_1) metric.*

The state-of-the-art trade-offs for $(1+\Theta(\alpha))$-approximate NN oracles differ between low and high dimensions; (a) for low dimensions, the trade-off is $\alpha = \epsilon$ with query/update time $\tau = \tilde{O}(\epsilon^{-d})$ [AMN$^+$98], while (b) for high dimensions, we have $\alpha = O(1/\epsilon)$ with $\tau = \tilde{O}(dn^{\epsilon^2})$ [AR15]. Substituting these bounds in Theorem 1.1, we obtain dynamic algorithms for the Chamfer distance achieving $(1+\epsilon)$-approximation in $\tilde{O}(\epsilon^{-d})$ update time and an $O(1/\epsilon)$-approximation in $\tilde{O}(dn^{\epsilon^2}\epsilon^{-4})$ update time. [2] Moreover, using known embeddings from ℓ_2 into ℓ_1, Theorem 1.1 readily extends to the setting when the underlying metric is Euclidean (ℓ_2) (see Appendix A for details).

Our dynamic algorithms maintain an *estimate* to the Chamfer distance $\mathrm{dist}_{\mathrm{CH}}(A,B)$ with provable approximation ratios. It is natural to ask whether it is possible to maintain the underlying *assignment* $g : A \to B$ that attains these approximation ratios, i.e., $\sum_{a\in A} \mathrm{dist}(a, g(a)) \leq (1+\epsilon)\cdot \mathrm{dist}_{\mathrm{CH}}(A,B)$. Since reporting the assignment itself takes $\Theta(n)$ time, reporting the *changes* in the assignment due to an update, also known as *recourse*, would be desirable. Unfortunately, it turns out that any dynamic algorithm that maintains an α-approximate assignment between A and B must have at least $\Omega(n)$ recourse, and thus also $\Omega(n)$ update time (see Lemma B.1). In fact, for any constant $\delta > 0$, [BIJ$^+$23]

---

[2] The stated bounds on the update time are in fact stronger in that they achieve a better dependency on the dimension d. This is because Theorem 1.1 presents only a simplified version of our main result; the precise trade-offs are given in Theorem 3.1.

show that even in the *static* setting, reporting a $(1+\epsilon)$-approximate assignment requires $\Omega(n^{2-\delta})$ time under the hitting set conjecture [Wil18].

Experiments. We implement our algorithm and validate its performance over four real datasets covering both high and low dimensions, as well as their noisy versions with injected outlier points. In all datasets, our algorithm achieves less than 10% error using only hundreds of samples, even against injected outliers, in time up to magnitudes better than a naive dynamic algorithm. We also discover that a simple uniform sampling baseline is competitive for these real datasets, but its performance degrade significantly when outliers are present, where our algorithm has a clear advantage.

1.1 Technical Contribution

We employ an importance sampling framework to estimate $\operatorname{dist}_{\mathrm{CH}}(A,B)$, and our main contribution is a dynamic data structure that maintains an importance sampler. This importance sampling framework was introduced in [BIJ$^+$23] in a static setting, where the key idea is to compute a coarse $O(\log n)$-approximate assignment $\widehat{g}$ from A to B, and then sample with probability proportional to the distance $\operatorname{dist}(a,\widehat{g}(a))$. A standard importance sampling argument shows that an average of $\tilde{O}(1)$ samples suffices for $(1+\epsilon)$-approximation to $\operatorname{dist}_{\mathrm{CH}}(A,B)$.

However, it is difficult to *explicitly* maintain the approximate assignment $\widehat{g}$ in the dynamic setting, since the recourse/change of $\widehat{g}$ can already be very significant per update (let alone the running time). To resolve this issue, our dynamic sampler obtains these sampling guarantees with an *implicit* representation of distance estimates $\hat{\operatorname{dist}}_a : a \in A$ such that $\hat{\operatorname{dist}}_a$ is an $O(\log^2 n)$ approximation to $\min_{b\in B}\operatorname{dist}(a,b)$ for $a\in A$ in $\tilde{O}(d)$ update time. This sampler only generates a sample $\widehat{a}\in A$, and the final estimator for the importance sampling is computed through $O(\log^2 n\cdot\epsilon^{-2}\max\{1,\alpha^2\}) = \tilde{O}(\epsilon^{-2}\max\{1,\alpha^2\})$ queries to an $(1+\Theta(\alpha))$-approximate nearest neighbor oracle. This eventually leads to a $(1+\alpha+\epsilon)$-approximation to $\operatorname{dist}_{\mathrm{CH}}(A,B)$,

Our dynamic sampler (similarly to the static algorithm of [BIJ$^+$23]) relies on a family of partitions of the input plane $\mathbb{R}^d$ into a series of nested cells that exponentially decrease in size on lower levels (also known as randomly-shifted quadtree in the literature). For each $a\in A$, we say that a is matched in some cell of our decomposition if it is the smallest cell that contains a and any point of B. The size of this unique sub-cell serves as an approximation to $\min_{b\in B}\operatorname{dist}(a,b)$. Our goal is to implicitly maintain some information about the unique matching cell for all $a\in A$.

Importantly, we cannot afford to explicitly maintain the matching cell for each point $a\in A$ as it could change for $\Omega(n)$ points of A due to a single update in B. Instead, for each cell we maintain how many points of A happen to be matched to B inside that sub-cell. This allows us to implement a sampler that, instead of explicitly sampling a point of A, it samples a cell in our nested decomposition based on its size and the number of points of A matched in it. Once a cell is sampled, our goal is to sample a uniformly random point of A matched in that cell. To achieve this, every cell maintains a dynamic sampler which allows it to sample one of its sub-cells in our family of partitioning with probability proportional to the number of points of A in that sub-cell. Repeating this sub-cell sampling process through the $\tilde{O}(1)$ levels of the algorithm finds a cell that contains a single point of A, which we then return.

1.2 Related Work

In the static setting, [BIJ$^+$23, FI25] present a near-optimal algorithm for estimating the Chamfer distance, running in $\tilde{O}(nd\cdot\epsilon^{-2})$ time. For comparison, our dynamic algorithm handles point updates in near-optimal time proportional to $\tilde{O}(d)$, up to the cost of invoking the nearest neighbour oracle.

Perhaps the closest problem to the dynamic Chamfer distance is the dynamic maintenance of the Earth Mover distance (EMD), for which Chamfer distance is often used as a proxy in practical applications [KSKW15, AM19]. For $d=2$, dynamic EMD is known to admit an algorithm that achieves $O(1/\epsilon)$ approximation in $O(n^{1/\epsilon})$ update time [GKP$^+$25]. In contrast, our dynamic algorithm for the Chamfer distance extends to *any* dimensions, and can even achieve an improved approximation ratio of $(1+\epsilon)$ in low dimensions. Similar to our negative result on the recourse, [GKP$^+$25] show that maintaining a mapping to the dynamic EMD problem that achieves an approximation ratio better than 2 requires at least $\Omega(n)$ time, even for 1 dimensional point sets. This highlights the difficulty of dynamically maintaining mappings underlying different proximity measures between point clouds.

Recently, there has been a growing interest in designing dynamic algorithms for fundamental problems in machine learning, thus contributing towards the grand vision of building a library of efficient data structures for key machine learning primitives. Notable progress has been made on several fronts, including dynamic algorithms for various clustering objectives such as k-center [CGS18, GHL$^+$21, BEF$^+$23, CFG$^+$24, BHMS23, LHG$^+$24, CLSW24], k-median/k-means [CHP$^+$19a, HK20, BCLP23], facility location [GHL18, CHP$^+$19b, BGJ$^+$24], correlation clustering [CLMP24], as well as dynamic matrix multiplication for structured matrices arising in machine learning applications [AvdBM25].

2 Preliminaries

Definition 2.1 (Chamfer distance)**.** Given two point-sets $A, B \subset \mathbb{R}^d$ with $\max\{|A|, |B|\} \leq n$, the *Chamfer distance* is defined as $\operatorname{dist}_{\mathrm{CH}}(A, B) := \sum_{a \in A} \min_{b \in B} \operatorname{dist}(a, b)$. For $\alpha \geq 1$, we say that a value $\tilde{\mu}$ is an α approximation to $\operatorname{dist}_{\mathrm{CH}}(A, B)$ if $\tilde{\mu} \leq \operatorname{dist}_{\mathrm{CH}}(A, B) \leq \tilde{\mu} \cdot \alpha$.

We will refer to $\min_{b \in B} \operatorname{dist}(a, b)$ as $\operatorname{dist}_{\mathrm{CH}}(a, B)$. We will use the following dynamic nearest-neighbor data structure as a subroutine in our algorithm.

Definition 2.2 (Dynamic nearest-neighbor data structure)**.** Given $\alpha > 0$ and a dynamic point set $B \subset \mathbb{R}^d$, $|B| \leq n$, a $(1+\alpha)$-approximate dynamic nearest neighbor data structure with update time and query time $\tau(\alpha)$ with respect to the ℓ_1 norm is a data structure which can be maintained in $\tau(\alpha)$ update time as B undergoes insertions and deletions, and when queried for point $a \in \mathbb{R}^d$, it returns a value $\tilde{\mu}_a$ such that $\tilde{\mu}_a \leq \min_{b \in B} \|a - b\|_1 \leq (1 + \alpha) \cdot \tilde{\mu}_a$ with $(1 - 1/\operatorname{poly}(n))$ probability in $\tau(\alpha)$ time.

Throughout this paper, we assume that the input points are contained in $[U]^d$ for some $U = \operatorname{poly}(n)$, which is a power of 2. Furthermore, we assume that during all updates to the input sequence, the *aspect-ratio* of the input points $\max_{a \in A, b \in B} \|a - b\|_1 / \min_{a \in A, b \in B} \|a - b\|_1$ is upper bounded by $\phi = \operatorname{poly}(n) = 2^{\mathcal{L}}$, for some integer $\mathcal{L} \geq 0$.

Lemma 2.3 (Dynamic Weighted Sampler)**.** *There exists a dynamic algorithm that maintains a weighted set of elements $A = \{a_1, \ldots, a_n\}$ with corresponding weights $W = \{w(a_1), \ldots, w(a_n)\}$ undergoing insertions and deletions. Upon query, the algorithm returns an element of A such that $a_i \in A$ is returned with probability $w(a_i)/\sum_{j \in [n]} w(a_j)$. Both updates and queries are supported in $O(\log n)$ worst-case update and query time.*

Data structures similar to that of Lemma 2.3 have appeared before in literature, but for sake of completeness we include an implementation in Appendix B.4.

2.1 Dynamic Quad-Tree

Our algorithm relies on the dynamic quad-tree data structure [dBHTT07]. The quad-tree is constructed as follows. We first choose a random vector described by $z \in [0, U]^d$. We then shift all input points with the vector described by z, hence after the shift they will be contained in $[0, 2 \cdot U]^d$. With a slight overload of notation, we will refer to the shifted points as A and B.

Consider a series of $O(\log n)$ grids drawn on the input space, where the i-th grid has side length $U \cdot 2^{1-i}$. The quad-tree is then a rooted tree T, where each node $v \in T$ is associated with some cell $\mathcal{C}_v$ of these grids with side length $L(v)$. The root r of T corresponds to the smallest cell of among all the grids which contains all input points. Any node v of the tree T such that $\mathcal{C}_v$ contains more then one point of the input has child nodes in T corresponding to its non-empty sub-cells on the next level of the grid decomposition with side length $L(v)/2$. The leaves of T corresponds to the largest cells of the decomposition containing a single point of the input. As the aspect ratio of the input is assumed to be $\phi = \operatorname{poly}(n)$, the tree consists of $O(\log \phi) = \mathcal{L}$ layers. As each input point may appear once in any of the cells of all these layers, T has at most $O(n \cdot \log n)$ nodes.

[dBHTT07] has shown how to maintain this representation of the input in $O(d \cdot \log n)$ worst-case update time such that all nodes of the tree store the number of input points in their respective cell and the leaves explicitly store the single input point stored in their cell. For our application, we further impose that every cell is aware of the number of input points in its cell from A and B separately.

3 Dynamic Algorithm

We start by describing an algorithm that achieves slightly worse guarantees than described in Theorem 1.1. Namely, this algorithm will maintain the estimate of the Chamfer distance between two sets with a constant probability. Formally, we will first prove the following result.

Theorem 3.1. *Let A, B be two sets in space $\mathbb{R}^d$, with $|A|, |B| \leq n$, $\alpha > 0$ and $\epsilon \in (0, 1)$ parameters, and let $\tau(\alpha)$ be the update and query times of a $(1 + \alpha)$-approximate dynamic nearest-neighbor datastructure. There exists a dynamic data structure which can be maintained in $O(d \cdot \log n + \log^2 n + \tau(\Theta(\alpha))$ worst-case update time as A and B undergoes point insertions and deletions and can be queried to return a $(1 + \alpha + \epsilon)$-approximation to $\operatorname{dist}_{\mathrm{CH}}(A, B)$ between A and B w.r.t. l_1-norm in $O(\log^2 n \cdot \epsilon^{-2} \max\{\alpha^2, 1\} \cdot (d \log^2 n + \tau(\Theta(\alpha))))$ time with $3/4$ probability.*

In Section 3.4, we show how to boost the the above result to obtain our main result, i.e. Theorem 1.1, which on query returns a $(1 + \alpha + \epsilon)$-approximation to $\operatorname{dist}_{\mathrm{CH}}(A, B)$ with probability $(1 - 1/\operatorname{poly}(n))$. In the following sections, we describe our dynamic algorithm on a high level. For sake of completeness, we include pseudo-codes of our algorithm in Appendix C.

3.1 Algorithm description

We say that a point $a \in A \cup B$ belongs to a node v of level i of quad-tree T if $a \in \mathcal{C}_v$. We say that point $a \in A$ is matched at v if v is the lowest level level node of T such that both a and any point of b belongs to it (where we assume that the root has the highest level as it corresponds to the largest cell containing all the input points). Note that every point of $A \cup B$ may belong to $\mathcal{L} = O(\log n)$ nodes of T, but all points of A are matched at exactly one node of T. For each $a \in A$ denote this unique node by v_a.

3.1.1 Handling an Update

We augment the dynamic tree structure of Section 2.1 with the following information being stored at each node $v \in T$: number of points $\gamma_A(v)$ from A belonging to v, number of points $\gamma_B(v)$ from B belonging to v, and the number of matched points $\gamma(v)$ from A at v. Note that $\gamma_B(v)$ and $\gamma_A(v)$ can be maintained using the algorithm of [dBHTT07].

We also maintain a dynamic sampler NODE-SAMPLER(v) corresponding to each node v of T, and a global sampler TREE-SAMPLER(T) for the whole T. The sampler NODE-SAMPLER(v) is for the set $\{u \in T : u \text{ is a child of } v \wedge \mathcal{C}_u \cap B = \emptyset\}$ w.r.t. weights $\gamma_A(u)$. The sampler TREE-SAMPLER(T) is for the set of all nodes v of T with $\gamma(v) > 0$ w.r.t. weights $w_T(v) = L(v) \cdot \gamma(v)$.

For sake of simplicity of the presentation, assume that at all times $B \neq \emptyset$. We will now describe how the algorithm updates γ values for all $v \in T$ after each update. Assume point x is deleted from or inserted into $A \cup B$. Through iterating along the path starting from the leaf of T containing x and ending at the root r, the algorithm finds the path of nodes $v_1, \ldots, v_k$ of T whose cell does not contain a point of $B \setminus x$ (ordered from the leaf). Let v' be the ancestor of v_k in T. We distinguish between four cases.

1. **Insertion of x into A:** set $\gamma(v') = \gamma(v') + 1$.
2. **Deletion of x from A:** set $\gamma(v') = \gamma(v') - 1$.
3. **Insertion of x into B:** set $\gamma(v') := \gamma(v') - \gamma_A(v_k)$, $\gamma(v_1) = \gamma_A(v_1)$ and $\gamma(v_i) := \gamma_A(v_i) - \gamma_A(v_{i-1})$ for $k \geq i > 1$.
4. **Deletion of x from B:** set $\gamma(v') := \gamma(v') + \sum_{i \in [k]} \gamma(v_i)$ and $\gamma(v_i) = 0$ for $i \in [k]$.

The correctness of our algorithm in maintaining $\gamma(v), v \in T$ is proven by Claim 3.3.

3.1.2 Answering Queries

The queries are answered using importance sampling based on a sampling process for points in A.

Sampling points in A. To sample a single point from A, we first sample a node v from T using $\textsc{Tree-Sampler}(T)$. Then, we sample a child u of v using $\textsc{Node-Sampler}(v)$. After this, we recursively call $\textsc{Node-Sampler}(u)$, until we reach a leaf of T, when we finally return the unique point $a \in A$ contained in it.

Claim 3.5 shows that this sampling process returns point $a \in A$ with probability $L(v_a)/\sum_{v \in T} \gamma(v) \cdot L(v)$, where we recall that v_a stands for the unique cell a is matched in. Claim 3.4 shows that $L(v_a) \sim \mathrm{dist}_{\mathrm{CH}}(a, B)$ within $\mathrm{poly}(\log n)$ factors.

Estimating the Chamfer distance. Using our sampler, estimating $\mathrm{dist}_{\mathrm{CH}}(A, B)$ turns into a standard application of importance sampling. Function $\mathbf{NN}(a, B, \alpha)$ refers to any procedure for finding a $(1+\alpha)$-approximation to the nearest neighbor of a in set B.

Specifically, we estimate $\mathrm{dist}_{\mathrm{CH}}(A, B)$ through taking $m = 240 \cdot \mathcal{L} \cdot \log n \max\{\alpha^2, 1\} \cdot \epsilon^{-2} = O(\log^2 n \cdot \max\{\alpha^2, 1\} \cdot \epsilon^{-2})$ samples S from A through the after-mentioned sampling procedure. For each $a \in S$, we then query the nearest neighbor data structure to generate a $(1 + \alpha/4)$-approximation to $\mathrm{dist}_{\mathrm{CH}}(a, B)$. Refer to these values as $\mathbf{NN}(a, B, \alpha/4)$ for $a \in S$.

We assign a weight of $\mathbf{NN}(a, B, \alpha/4) \cdot \sum_{v \in T} \gamma(v) \cdot L(v)/L(v_a)$ to each $a \in S$. This implies that the weight of each sample is a $(1 + \alpha/4)$-approximation of $\mathrm{dist}_{\mathrm{CH}}(A, B)$ in expectation. We then finally return the average of these weights (shifted by $1/(1 + \epsilon/2)$ to fit our definition of approximation).

3.2 Correctness Analysis

This section is devoted to an overview of the proof of Lemma 3.2, which establishes the correctness of our algorithm.

Lemma 3.2. *On query, the algorithm returns a $(1 + \epsilon + \alpha)$-approximation to $\mathrm{dist}_{\mathrm{CH}}(A, B)$ with $7/8$ probability.*

We start with examining the variables $\gamma(v)$, corresponding to the number of points from A matched at node v.

Claim 3.3. *The value of $\gamma(v)$ for all $v \in T$ is maintained correctly after an update.*

Proof. When an update occurs to $a \in A$, the only γ value changes is $\gamma(v_a)$. The algorithm greedily finds v_a and updates $\gamma(v_a)$ accordingly.

When an update occurs to $b \in B$, the algorithm finds all nodes of T that contain only b from B, $v_1, \ldots, v_k$ (ordered from the leaf upwards) and v' the node with the smallest cell containing b and an other point of B. Observe that only $v_1, \ldots, v_k$ and v' may have its γ value updated.

In the case of a deletion, all points of A matched in $v_1, \ldots, v_k$ should be matched in v' after an update, and the algorithm updates γ values accordingly. In case of an insertions, every point of A in $\mathcal{C}_{v'}$ should be matched in the smallest cell among $\mathcal{C}_{v_1}, \ldots, \mathcal{C}_{v_k}$ they are contained in (if there is such a cell). The algorithm similarly updates γ values accordingly. ☐

The following claim allows us to estimate $\mathrm{dist}_{\mathrm{CH}}(a, B)$ with value $L(v_a)$. The proof is deferred to Appendix B.2. Our proof is similar to a lemma of [BIJ+23], however our lower bound on $L(v_a)$ with respect to $\mathrm{dist}_{\mathrm{CH}}(a, B)$ is slightly weaker due to the limitations of the dynamic model.

Claim 3.4. *The following statements hold with respect to the random shift of the quad-tree (see Section 2.1).*

(i) With $1 - 1/\mathrm{poly}(n)$ probability $L(v_a) \geq \frac{\mathrm{dist}_{\mathrm{CH}}(a,B)}{\log n \cdot 3}$ holds for all $a \in A$.
(ii) $\mathbb{E}[L(v_a)] \leq 2 \cdot \mathcal{L} \cdot \mathrm{dist}_{\mathrm{CH}}(a, B))$ holds for all $a \in A$, where $\mathcal{L}$ denotes the height of T.

From Claim 3.4 we have that $L(v_a) \sim \mathrm{dist}_{\mathrm{CH}}(a, B)$. Hence, the following claim shows that the sampling process of the algorithm samples $a \in A$ with probability roughly proportional to its contribution to $\mathrm{dist}_{\mathrm{CH}}(A, B)$. Its proof is deferred to Appendix B.3.

Claim 3.5. *The algorithm samples $a \in A$ with probability $L(v_a)/\sum_{v \in T} L(v) \cdot \gamma(v)$.*

Table 1: Specifications of datasets and experiment parameters.

dataset	dimension d	$\|A\|$	$\|B\|$	window size	sample size
Text Embedding	300	~1.9k	~1.2k	100	150
ShapeNet	3	~2k	~2k	100	150
Fashion-MNIST	784	60k	10k	500	200
SIFT	128	1000k	10k	500	300

The remaining proofs follow the standard analysis for importance sampling. Our main goal is to bound the variance of a single random variable defined by the weight assigned to each of the m samples of A the algorithm draws on query. The proof is deferred to Appendix B.5

Claim 3.6. *Let X stand for the random variable defined by $NN(a, B, \alpha/4) \cdot \sum_{v \in V} \gamma(v) \cdot L(v)/L(v_a)$ when the algorithm samples point $a \in A$. Then, with high $(1 - 1/\operatorname{poly}(n))$ probability* $\operatorname{Var}[X] \leq CH(A, B)^2 \cdot \log n \cdot \mathcal{L} \cdot 12\alpha^2$.

We are now ready to finally prove Lemma 3.2, which is a standard application of importance sampling. Given random variable X with expectation $\mathbb{E}[X]$ and variance $\mathbb{E}[X]^2 \cdot \phi$, the average of $O(\phi/\epsilon^2)$ i.i.d. samples of X is a $(1+\epsilon)$-approximation to $\mathbb{E}[X]$ with $> 1/2$ probability by Chebyshev's inequality.

In our case, $\mathbb{E}[X]$ is $(1+\alpha/4)$-approximate to $\operatorname{dist}_{\mathrm{CH}}(A, B)$ and $\phi = O(\alpha^2 \epsilon^{-2} \log^2 n)$ by Claim 3.6. Hence, we conclude the main lemma of this section Lemma 3.2. We defer the proof to Appendix B.6

3.3 Running time

Lemma 3.7. *The algorithm of Section 3.1 has $O(d \cdot \log n + \log^2 n + \tau(\Theta(\alpha))$ and $O(\log^2 n \cdot \epsilon^{-2} \max\{1, \alpha^2\}(d \log^2 n + \tau(\Theta(\alpha))))$ worst-case update and query times, respectively.*

Proof. Note that updating the quad tree itself (and the corresponding γ_A, γ_B values) takes $O(d \cdot \log n)$ time using algorithms from literature [dBHTT07]. Observe that insertions and deletions both to A and B require the algorithm to identify the set of vertices of T the affected point falls in and simply to adjust the γ value on a subset of these nodes. Hence, this requires at most $O(d \cdot \mathcal{L}) = O(d \cdot \log n)$ time.

The algorithm also needs to update its internal global TREE-SAMPLER(T) and NODE-SAMPLERS for all $v \in T$. Observe that the total number of updates these samplers undergo is proportional to the number of nodes of T which change their $\gamma_v(A)$ or γ_v values, that is $O(\log n)$. By Lemma 2.3 this takes $O(\log^2 n)$ time as each sampler contains at most $|A| \leq n$ elements. In addition, the algorithm needs to maintain its dynamic nearest-neighbor datastructure.

On query the algorithm needs to sample $m = O(\log^2 n \cdot \max\{1, \alpha^2\} \cdot \epsilon^{-2})$ samples, for each of which it queries the nearest neighbor datastructure. Each query requires first a call to TREE-SAMPLER(T), then a walk from the sampled node to a leaf at each step of which a NODE-SAMPLER is queried. Hence, this takes $O(d \log^2 n \cdot m) = O(d \log^4 n \cdot \max\{1, \alpha^2\} \cdot \epsilon^{-2})$ time. □

3.4 Boosting for a High-Probability Guarantee After Every Update

We show to derive Theorem 1.1 from Theorem 3.1. To this end, while maintaining the internal data structures of Theorem 3.1, on query output the median result of $O(\log n)$ queries to the baseline algorithm. This will result in the same update time, but an $O(\log n)$ blowup in query time.

In terms of correctness, consider the $\log n$ outputs the algorithm produces. Each of them is an $(1 + \alpha + \epsilon)$-approximation to $\operatorname{dist}_{\mathrm{CH}}(A, B)$ with $3/4 > 1/2$ probability. By a standard application of Chernoff bound with high $(1 - 1/\operatorname{poly}(n))$ probability more than half of these query results will be $(1 + \alpha + \epsilon)$-approximation, hence so is their median value.

4 Experiments

We implement our dynamic algorithm and validate its performance on various datasets. Observe that handling update of A is straightforward (by simply querying an NN oracle on B), whereas

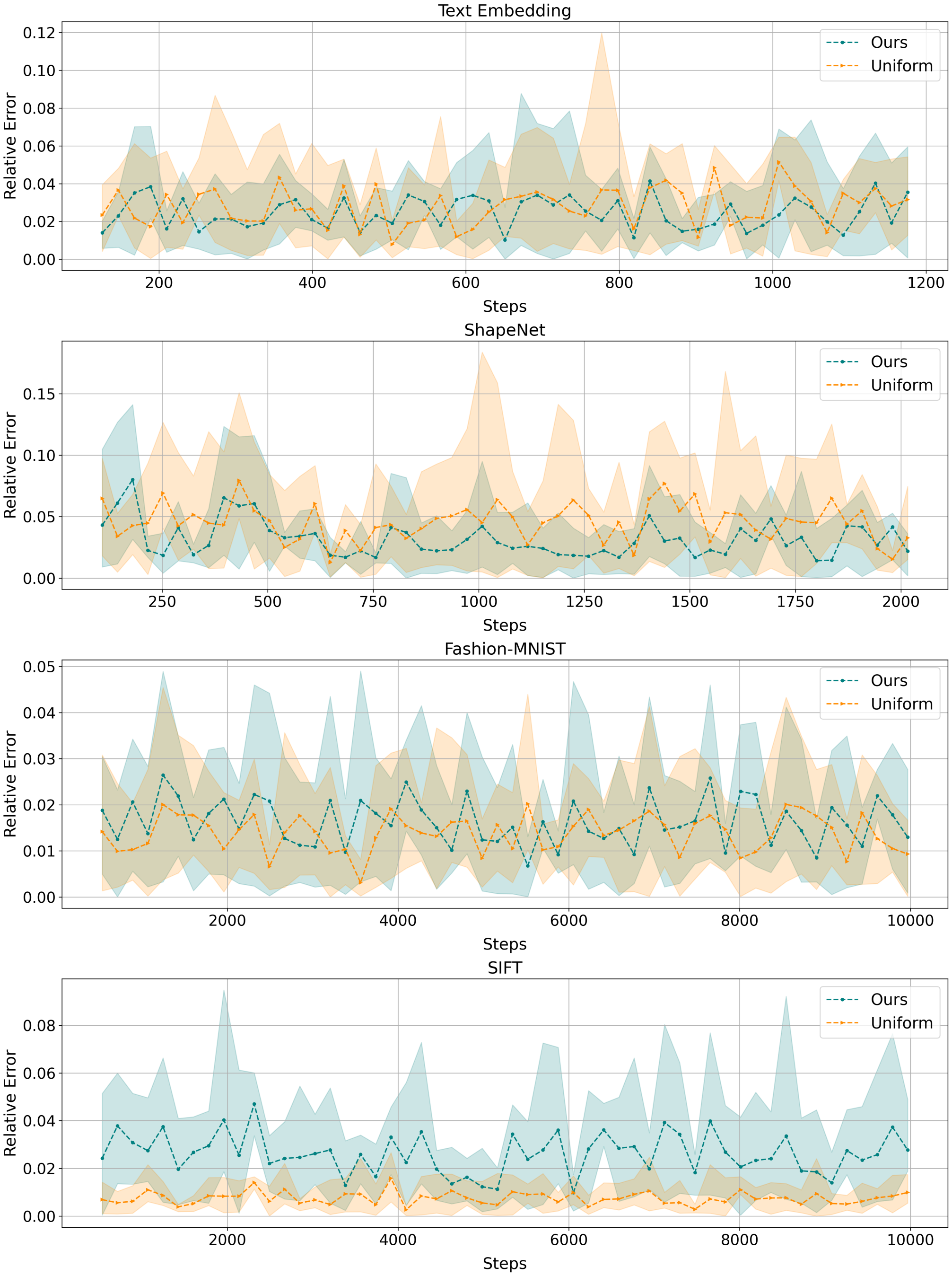

Figure 1: Relative error curves for datasets *without* outliers. These experiments are independently run for 5 times, and we report the average (the dot), max and min value (the shaded area) after every $\frac{w}{5}$ to $\frac{w}{3}$ updates (depending on the dataset) where w is the window size.

allowing updates on B is more challenging. Hence, to simplify the exposition (but still keeping the key challenge), we focus on the default setting where B is dynamic and A is a static set. For completeness, we also include the case that both A and B are dynamic in Section E, and the results are similar as in the default setting.

Baselines. Our first baseline is a naive exact algorithm, which we call "Benchmark", where for each update of B, it re-computes $\mathrm{dist}(a, B)$ in $O(d|B|)$ time for each a whose current $\mathrm{dist}(a, B)$

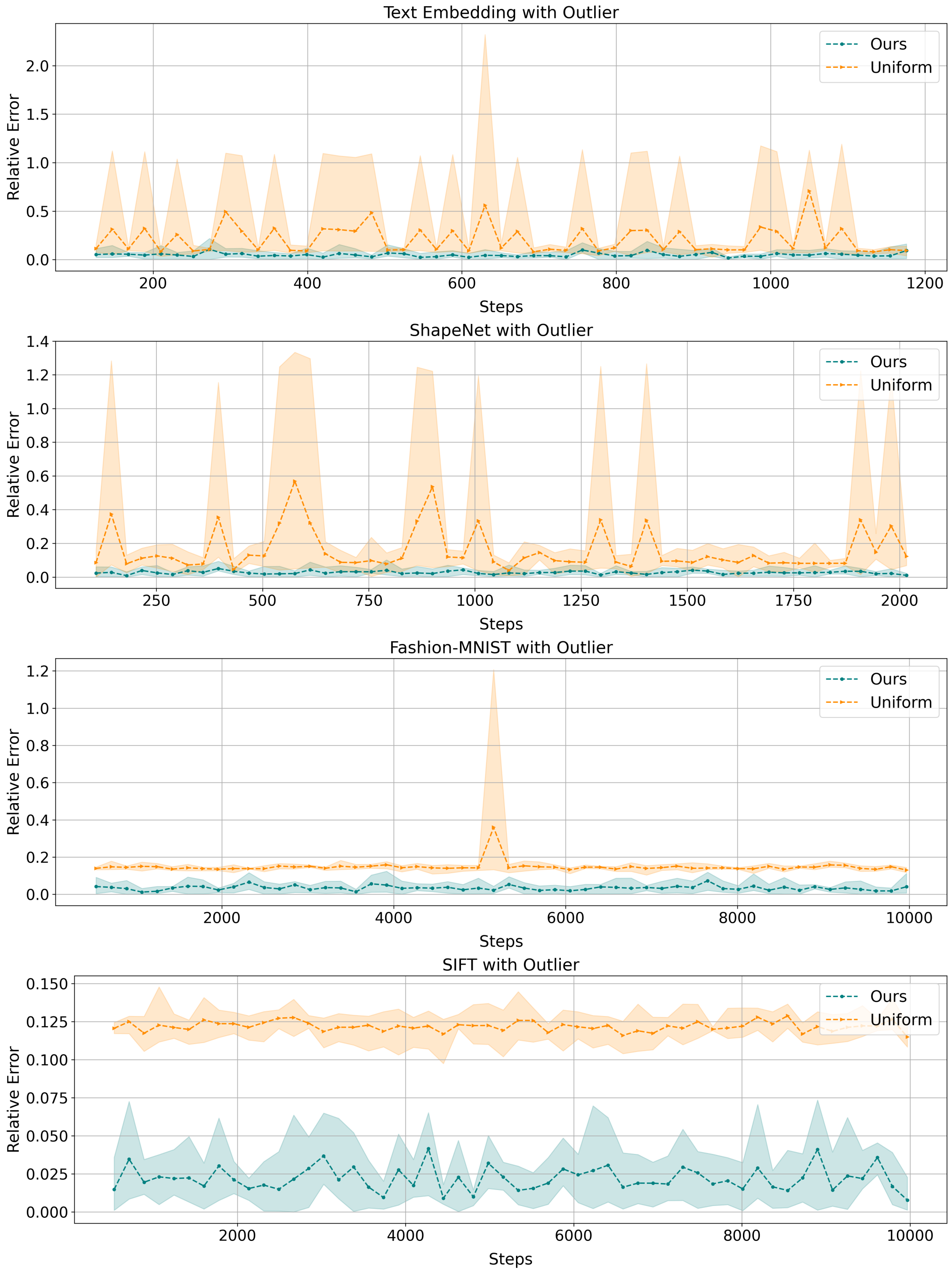

Figure 2: Relative error curves for datasets *with* outliers, with the same setup as in Figure 1.

value may be affected by the update. This serves as a benchmark for the approximation ratio. The second baseline, called “Uniform”, replaces our important sampling with a uniform sampling (while keeping the other steps the same).

Datasets and Experiment Setup. We employ four real datasets covering both high and low dimensions in the experiment: Text Embedding [KSKW15], ShapeNet [CFG$^+$15], Fashion-MNIST [XRV17], and SIFT [JDS11]. Each dataset consists of a larger set which we use as A, and a smaller set which we use as B. The ShapeNet dataset consists of 3D point clouds and is widely used to measure the similarity between different shapes. The Fashion-MNIST and SIFT datasets were

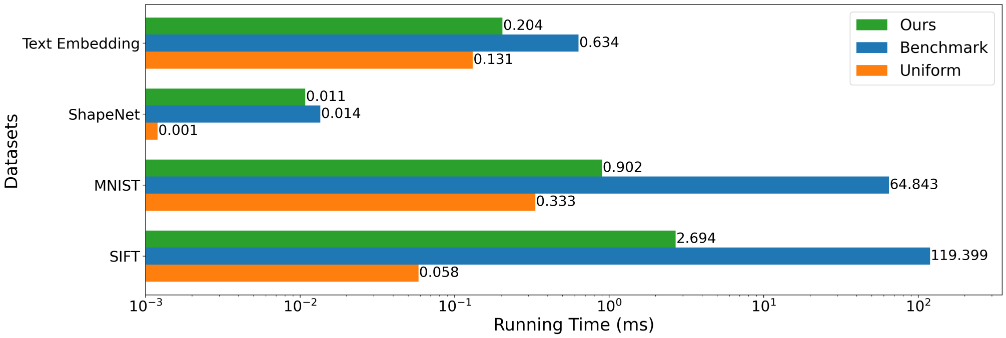

Figure 3: Average running time per window update for all algorithms on datasets *without* outliers.

also known to be used as benchmarks for approximate nearest neighbor search. Similar choice of using ANN benchmark datasets was also made in a previous work, to evaluate the performance of static algorithm for Chamfer distance [BIJ+23]. Furthermore, to evaluate the robustness of the algorithms, we inject an outlier point into each dataset: we compute the geometric mean $c := \frac{1}{|A|}\sum_{a\in A} a$, pick an arbitrary $a^* \in A$, and generate an outlier as $\tilde{a} = 0.1 \cdot |A| \cdot (a^* - c) + c$. Intuitively, this "moves" a^* along the direction of $a^* - c$ by a large distance. Finally, since these datasets do not contain information of dynamic update, we employ a sliding window (on B) to simulate the insertions and deletions. The detailed specification of the dataset and the experiment parameters can be found in Table 1.

Implementation Details. Recall that our algorithm (and the Uniform baseline) consists of a sampling step and then a second step to build the estimator that makes use of a nearest neighbor query structure. Since the window size for each dataset is relatively small, we choose to use the *exact* nearest neighbor algorithm, which does not introduce additional errors and allows for a more accurate evaluation. For the 3D ShapeNet dataset, we implement nearest neighbor queries with KD-trees, which can efficiently perform exact search in low-dimensional spaces. All algorithms are implemented in C++ and compiled with Apple Clang version 15.0.0 at -O3 optimization level. All the experiments are run on a MacBook Air 15.3 with an Apple M3 chip (8 cores, 2.22 GHz), 16GB RAM, and macOS 14.4.1 (23E224).

Experiment Results. Our main experiment evaluates both the error of the estimated Chamfer distance and the running time of the algorithms, over the sliding windows. We depict the relative error curve in Figure 1 and Figure 2.

Here, the relative error for an estimate $\widehat{E}$ over the accurate Chamfer distance E is defined as $\frac{|E-\widehat{E}|}{E}$. Overall, our algorithm achieves less than 10% error using only hundreds of samples. Compared with Uniform baseline, our algorithm achieves comparable error and variance for datasets without outliers, and shows clear advantage when the dataset has the outlier. This showcases the robustness of our algorithm. We observe that our algorithm performs slightly worse than Uniform in the SIFT dataset (without outliers), but this is because the distance $\{\mathrm{dist}(a, B) : a \in A\}$ is very uniform (see Figure 4 in Section D), hence uniform sampling is already the "optimal" sampling strategy.

We report the average running time per sliding window update in Figure 3. Our algorithm is magnitudes more efficient than the Benchmark on larger datasets. It incurs a slightly higher time cost than Uniform which is expected since Uniform does not need to maintain any additional structure to generate a sample. The results for datasets with outliers are similar to that in Figure 3 and can be found in Figure 5 (Section D).

5 Acknowledgments

The authors would like to thank the anonymous reviewers for their insightful comments. This research was funded in part by a national key R&D program of China No. 2021YFA1000900, and the Austrian Science Fund (FWF) 10.55776/ESP6088024.

References

[AM19] Kubilay Atasu and Thomas Mittelholzer. Linear-complexity data-parallel earth mover's distance approximations. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *36th International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pages 364–373. PMLR, 2019. URL: http://proceedings.mlr.press/v97/atasu19a.html.

[AMN$^+$98] Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, and Angela Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM*, 45(6):891–923, 1998. doi:10.1145/293347.293348.

[AR15] Alexandr Andoni and Ilya P. Razenshteyn. Optimal data-dependent hashing for approximate near neighbors. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *47th Annual ACM Symposium on Theory of Computing (STOC)*, pages 793–801. ACM, 2015. doi:10.1145/2746539.2746553.

[AS03] Vassilis Athitsos and Stan Sclaroff. Estimating 3d hand pose from a cluttered image. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2003)*, pages 432–442. IEEE Computer Society, 2003. doi:10.1109/CVPR.2003.1211500.

[AvdBM25] Emile Anand, Jan van den Brand, and Rose McCarty. The structural complexity of matrix-vector multiplication. *CoRR*, abs/2502.21240, 2025. URL: https://doi.org/10.48550/arXiv.2502.21240, arXiv:2502.21240, doi:10.48550/ARXIV.2502.21240.

[BCLP23] Sayan Bhattacharya, Martín Costa, Silvio Lattanzi, and Nikos Parotsidis. Fully dynamic k-clustering in $\tilde{o}(k)$ update time. In *36th Conference on Neural Information Processing Systems (NeurIPS)*, 2023. URL: http://papers.nips.cc/paper_files/paper/2023/hash/3b7ba46201bf15e5c3935272afae50db-Abstract-Conference.html.

[BEF$^+$23] MohammadHossein Bateni, Hossein Esfandiari, Hendrik Fichtenberger, Monika Henzinger, Rajesh Jayaram, Vahab Mirrokni, and Andreas Wiese. Optimal fully dynamic k-center clustering for adaptive and oblivious adversaries. In Nikhil Bansal and Viswanath Nagarajan, editors, *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2677–2727. SIAM, 2023. URL: https://doi.org/10.1137/1.9781611977554.ch101, doi:10.1137/1.9781611977554.CH101.

[BGJ$^+$24] Sayan Bhattacharya, Gramoz Goranci, Shaofeng H.-C. Jiang, Yi Qian, and Yubo Zhang. Dynamic facility location in high dimensional euclidean spaces. In *41st International Conference on Machine Learning (ICML)*. OpenReview.net, 2024. URL: https://openreview.net/forum?id=rucbIsWoEV.

[BHMS23] Leyla Biabani, Annika Hennes, Morteza Monemizadeh, and Melanie Schmidt. Faster query times for fully dynamic k-center clustering with outliers. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *36th Conference on Neural Information Processing Systems (NeurIPS)*, 2023. URL: http://papers.nips.cc/paper_files/paper/2023/hash/1d8e261c241aa72f9b4a02af7f52587e-Abstract-Conference.html.

[BIJ$^+$23] Ainesh Bakshi, Piotr Indyk, Rajesh Jayaram, Sandeep Silwal, and Erik Waingarten. Near-linear time algorithm for the chamfer distance. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *36th Conference on Neural Information Processing Systems (NeurIPS)*, 2023. URL: http://papers.nips.cc/paper_files/paper/2023/hash/d2fe3a5711a6d488da9e9a78b84ee24c-Abstract-Conference.html.

[CFG$^+$15] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *CoRR*, abs/1512.03012, 2015. URL: http://arxiv.org/abs/1512.03012, arXiv:1512.03012.

[CFG$^+$24] Emilio Cruciani, Sebastian Forster, Gramoz Goranci, Yasamin Nazari, and Antonis Skarlatos. Dynamic algorithms for k-center on graphs. In David P. Woodruff, editor, *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3441–3462. SIAM, 2024. doi:10.1137/1.9781611977912.123.

[CGS18] T.-H. Hubert Chan, Arnaud Guerquin, and Mauro Sozio. Fully dynamic k-center clustering. In Pierre-Antoine Champin, Fabien Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis, editors, *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018*, pages 579–587. ACM, 2018. doi:10.1145/3178876.3186124.

[CHP$^+$19a] Vincent Cohen-Addad, Niklas Hjuler, Nikos Parotsidis, David Saulpic, and Chris Schwiegelshohn. Fully dynamic consistent facility location. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *32nd Conference on Neural Information Processing Systems (NeurIPS)*, pages 3250–3260, 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/fface8385abbf94b4593a0ed53a0c70f-Abstract.html.

[CHP$^+$19b] Vincent Cohen-Addad, Niklas Hjuler, Nikos Parotsidis, David Saulpic, and Chris Schwiegelshohn. Fully dynamic consistent facility location. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *32nd Conference on Neural Information Processing Systems (NeurIPS)*, pages 3250–3260, 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/fface8385abbf94b4593a0ed53a0c70f-Abstract.html.

[CLMP24] Vincent Cohen-Addad, Silvio Lattanzi, Andreas Maggiori, and Nikos Parotsidis. Dynamic correlation clustering in sublinear update time. In *41st International Conference on Machine Learning (ICML)*. OpenReview.net, 2024. URL: https://openreview.net/forum?id=3YG55Lbcnr.

[CLSW24] T.-H. Hubert Chan, Silvio Lattanzi, Mauro Sozio, and Bo Wang. Fully dynamic k-center clustering with outliers. *Algorithmica*, 86(1):171–193, 2024. URL: https://doi.org/10.1007/s00453-023-01159-3, doi:10.1007/S00453-023-01159-3.

[dBHTT07] Mark de Berg, Herman J. Haverkort, Shripad Thite, and Laura Toma. I/o-efficient map overlay and point location in low-density subdivisions. In Takeshi Tokuyama, editor, *Algorithms and Computation, 18th International Symposium, ISAAC 2007*, volume 4835 of *Lecture Notes in Computer Science*, pages 500–511. Springer, 2007. doi:10.1007/978-3-540-77120-3_44.

[FI25] Ying Feng and Piotr Indyk. Even faster algorithm for the chamfer distance. In Keren Censor-Hillel, Fabrizio Grandoni, Joël Ouaknine, and Gabriele Puppis, editors, *52nd International Colloquium on Automata, Languages and Programming (ICALP)*, volume 334 of *LIPIcs*, pages 76:1–76:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. URL: https://doi.org/10.4230/LIPIcs.ICALP.2025.76, doi:10.4230/LIPICS.ICALP.2025.76.

[FSG17] Haoqiang Fan, Hao Su, and Leonidas J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pages 2463–2471. IEEE Computer Society, 2017. doi:10.1109/CVPR.2017.264.

[GHL18] Gramoz Goranci, Monika Henzinger, and Dariusz Leniowski. A tree structure for dynamic facility location. In Yossi Azar, Hannah Bast, and Grzegorz Herman, editors, *26th European Symposium on Algorithms (ESA)*, volume 112 of *LIPIcs*, pages 39:1–39:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. URL: https://doi.org/10.4230/LIPIcs.ESA.2018.39, doi:10.4230/LIPICS.ESA.2018.39.

[GHL$^+$21] Gramoz Goranci, Monika Henzinger, Dariusz Leniowski, Christian Schulz, and Alexander Svozil. Fully dynamic k-center clustering in low dimensional metrics. In Martin Farach-Colton and Sabine Storandt, editors, *Proceedings of the Symposium on Algorithm Engineering and Experiments, ALENEX 2021*, pages 143–153. SIAM, 2021. doi:10.1137/1.9781611976472.11.

[GKP$^+$25] Gramoz Goranci, Peter Kiss, Neel Patel, Martin P. Seybold, Eva Szilagyi, and Da Wei Zheng. Fully dynamic Euclidean bi-chromatic matching in sublinear update time. 267:20162–20186, 13–19 Jul 2025. URL: https://proceedings.mlr.press/v267/goranci25a.html.

[HB91] Gabor T. Herman and Carolyn A. Bucholtz. Shape-based interpolation using a chamfer distance. In Alan C. F. Colchester and David J. Hawkes, editors, *Information Processing in Medical Imaging, 12th International Conference, IPMI'91*, volume 511 of *Lecture Notes in Computer Science*, pages 314–325. Springer, 1991. URL: https://doi.org/10.1007/BFb0033762, doi:10.1007/BFB0033762.

[HHT+23] Shreyas Hampali, Tomas Hodan, Luan Tran, Lingni Ma, Cem Keskin, and Vincent Lepetit. In-hand 3d object scanning from an RGB sequence. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 17079–17088. IEEE, 2023. doi:10.1109/CVPR52729.2023.01638.

[HK20] Monika Henzinger and Sagar Kale. Fully-dynamic coresets. 173:57:1–57:21, 2020. URL: https://doi.org/10.4230/LIPIcs.ESA.2020.57, doi:10.4230/LIPICS.ESA.2020.57.

[HSS+24] Linus Härenstam-Nielsen, Lu Sang, Abhishek Saroha, Nikita Araslanov, and Daniel Cremers. Diffcd: A symmetric differentiable chamfer distance for neural implicit surface fitting. In Ales Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Computer Vision - ECCV 2024 - 18th European Conference, Proceedings, Part LXXIII*, volume 15131 of *Lecture Notes in Computer Science*, pages 432–447. Springer, 2024. doi:10.1007/978-3-031-73464-9_26.

[JDS11] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(1):117–128, 2011. doi:10.1109/TPAMI.2010.57.

[JSQJ18] Li Jiang, Shaoshuai Shi, Xiaojuan Qi, and Jiaya Jia. GAL: geometric adversarial loss for single-view 3d-object reconstruction. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VIII*, volume 11212 of *Lecture Notes in Computer Science*, pages 820–834. Springer, 2018. doi:10.1007/978-3-030-01237-3_49.

[KSKW15] Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. From word embeddings to document distances. In Francis R. Bach and David M. Blei, editors, *32nd International Conference on Machine Learning (ICML)*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 957–966. JMLR.org, 2015. URL: http://proceedings.mlr.press/v37/kusnerb15.html.

[LHG+24] Jakub Lacki, Bernhard Haeupler, Christoph Grunau, Rajesh Jayaram, and Václav Rozhon. Fully dynamic consistent k-center clustering. In David P. Woodruff, editor, *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3463–3484. SIAM, 2024. doi:10.1137/1.9781611977912.124.

[LJG05] Wei Liang, Yunde Jia, and Cheng Ge. Visual hand tracking using nonparametric sequential belief propagation. In De-Shuang Huang, Xiao-Ping (Steven) Zhang, and Guang-Bin Huang, editors, *Advances in Intelligent Computing, International Conference on Intelligent Computing, ICIC 2005, Proceedings, Part I*, volume 3644 of *Lecture Notes in Computer Science*, pages 679–687. Springer, 2005. doi:10.1007/11538059_71.

[LLZ+24] Fangzhou Lin, Haotian Liu, Haoying Zhou, Songlin Hou, Kazunori D. Yamada, Gregory S. Fischer, Yanhua Li, Haichong K. Zhang, and Ziming Zhang. Loss distillation via gradient matching for point cloud completion with weighted chamfer distance. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2024*, pages 511–518. IEEE, 2024. doi:10.1109/IROS58592.2024.10801828.

[LYH+23] Fangzhou Lin, Yun Yue, Songlin Hou, Xuechu Yu, Yajun Xu, Kazunori D. Yamada, and Ziming Zhang. Hyperbolic chamfer distance for point cloud completion. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 14549–14560. IEEE, 2023. doi:10.1109/ICCV51070.2023.01342.

[Mat13] Jirı Matoušek. Lecture notes on metric embeddings. Technical report, Technical report, ETH Zürich, 2013.

[MDJT14] M Mathews, J Deepa, Tonu James, and Shari Thomas. Segmentation of head from ultrasound fetal image using chamfer matching and hough transform based approaches. *Int. J. Eng. Res. Technol*, 3(5):1065–1068, 2014.

[RLT+20] Martin Rünz, Kejie Li, Meng Tang, Lingni Ma, Chen Kong, Tanner Schmidt, Ian D. Reid, Lourdes Agapito, Julian Straub, Steven Lovegrove, and Richard A. Newcombe. Frodo: From detections to 3d objects. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 14708–14717. Computer Vision Foundation / IEEE, 2020. URL: https://openaccess.thecvf.com/content_CVPR_2020/html/Runz_FroDO_From_Detections_to_3D_Objects_CVPR_2020_paper.html, doi:10.1109/CVPR42600.2020.01473.

[TMPF22] Michal J. Tyszkiewicz, Kevis-Kokitsi Maninis, Stefan Popov, and Vittorio Ferrari. Raytran: 3d pose estimation and shape reconstruction of multiple objects from videos with ray-traced transformers. In Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part X*, volume 13670 of *Lecture Notes in Computer Science*, pages 211–228. Springer, 2022. doi:10.1007/978-3-031-20080-9_13.

[VHK94] Marcel Van Herk and Hanne M Kooy. Automatic three-dimensional correlation of ct-ct, ct-mri, and ct-spect using chamfer matching. *Medical physics*, 21(7):1163–1178, 1994.

[WCL+19] Ziyu Wan, Dongdong Chen, Yan Li, Xingguang Yan, Junge Zhang, Yizhou Yu, and Jing Liao. Transductive zero-shot learning with visual structure constraint. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *32nd Conference on Neural Information Processing Systems (NeurIPS)*, pages 9972–9982, 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/5ca359ab1e9e3b9c478459944a2d9ca5-Abstract.html.

[Wil18] Virginia Vassilevska Williams. On some fine-grained questions in algorithms and complexity. In *Proceedings of the international congress of mathematicians: Rio de janeiro 2018*, pages 3447–3487. World Scientific, 2018. doi:10.1142/9789813272880_0188.

[WPZ+21] Tong Wu, Liang Pan, Junzhe Zhang, Tai Wang, Ziwei Liu, and Dahua Lin. Density-aware chamfer distance as a comprehensive metric for point cloud completion. volume abs/2111.12702, 2021. URL: https://arxiv.org/abs/2111.12702, arXiv:2111.12702.

[XRV17] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. URL: http://arxiv.org/abs/1708.07747, arXiv:1708.07747.

A Our Algorithms for the ℓ_2 Norm

[Mat13] has shown the following useful lemma (which we state in the dynamic setting):

Lemma A.1. *There is a dynamic algorithm which for* n *points* $A \subset \mathbb{R}^d$ *undergoing insertions and deletions in* $O(d/\epsilon^2)$ *worst-case update time maintains* $A \subset \mathbb{R}^{O(d/\epsilon^2)}$ *such that* $\|A'_i - A'_j\|_1 \le \|A_i - A_j\|_2 \le \|A'_i - A'_j\|_1 \cdot (1+\epsilon)$ *for all* $i, j \in [n]$ *with high* $(1 - 1/\operatorname{poly}(n))$ *probability.*

That is, we may embed the input points into the $O(d/\epsilon^2)$ space such that ℓ_1 distances between the embedded points roughly correspond to ℓ_2 distances between the input points. Substituting the embedded point set into Theorem 1.1, and using the state of the art dynamic nearest neighbor datastructures of [AMN$^+$98, AR15], we obtain the following algorithms for the ℓ_2 norm.

- For $d = O(1)$ and $0 < \epsilon < 1$, there exists an algorithm which, with high probability, maintains a $(1+\epsilon)$-approximation to the Chamfer distance between $A, B \subset \mathbb{R}^d$, $|A|, |B| \le n$ with respect to the ℓ_2 norm, as A and B undergo point insertions and deletions, in $\tilde{O}(\epsilon^{-O(d/\epsilon^2)})$ worst-case update time.
- There exists a dynamic algorithm which maintains an $O(\epsilon^{-1})$-approximation for $0 < \epsilon < 1$ to the Chamfer distance between $A, B \subset \mathbb{R}^d$, $|A|, |B| \le n$ with respect to the ℓ_2 norm, as A and B undergo point insertions and deletions, in $\tilde{O}(d \cdot n^{\epsilon^2} \cdot \epsilon^{-6})$ worst-case update time.

B Deferred Proofs

B.1 Linear Recourse for Maintaining Assignments

Lemma B.1. *Any dynamic algorithm that maintains an* α*-approximate assignment between* A *and* B *must have at least* $\Omega(n)$ *recourse, and thus also* $\Omega(n)$ *update time.*

Proof. Consider an instance where points of A are in close proximity to each other, and B contains two points, one at distance 1 from A and the other at distance $\Omega(\alpha)$. Any α-approximate assignment of this instance must assign the majority of points in A to the closer point in B. If the closer point of B is removed from the input, then any assignment must assign all points of A to the single remaining point in B, leading to the claimed recourse. In fact, for any constant $\delta > 0$, [BIJ$^+$23] show that even in the *static* setting, reporting a $(1+\epsilon)$-approximate mapping still requires $\Omega(n^{2-\delta})$ time under the hitting set conjecture [Wil18]. ☐

B.2 Proof of Claim 3.4

We will first prove some useful properties of the underlying quad-tree datastructure, which are based on the random shift of the points introduced at initialization. Our proof is analogous to a similar separation lemma of [BIJ$^+$23], and we include it here for sake of completeness.

Namely, let $x, y \in A \cup B$ and v be some node of T. Then, with respect to the random shift defined by $z \in [0, U]^d$, we have:

1. $\Pr[y \notin \mathcal{C}_v | x \in \mathcal{C}_v] \le \frac{\|x-y\|_1}{L(v)}$
2. $\Pr[y \in \mathcal{C}_v | x \in \mathcal{C}_v] \le \exp(-\frac{\|x-y\|_1}{L(v)})$

The random shift introduced at the initialization of our algorithm can be described as follows: the algorithms draws a uniform random point z with coordinates $z_1, \ldots z_d$ in $[U]^d$. Then the algorithm defines sets $A := \{a + z | a \in A\}$ and $B = \{b + z | b \in B\}$.

Fix some cell $\mathcal{C}_v = [v_1, v_1 + L(v)] \times \cdots \times [v_d, v_d + L(v)]$ (we disregard the case where a shifted point falls exactly on the grid, as it occurs with 0 probability). If point $x \in A \cup B$ with coordinates $x_1, \ldots, x_d$ falls in cell $\mathcal{C}_v$, we know that $z_i \in [x_i - v_i, x_i - v_i + L(v)]$ for all $i \in [d]$. If we condition on this event, then we know that z_i is uniformly distributed on this interval.

Fix some $y \in A \cup B$ with coordinates $y_1, \ldots, y_d$. Conditioning on the event that $x \in \mathcal{C}_v$, the event $y \notin \mathcal{C}_v$ is for all $i \in [d]$ if $z_i \in [x_i - v_i, x_i - v_i + L(v)], \notin [y_i - v_i, y_i - v_i + L(v)]$ that is $|x_i - y_i| > L(v)$ or $z_i \in [x_i - v_i, y_i - v_i]$ or $z_i \in [y_i - v_i + L(v), x_i - v_i + L(v)]$ (depending on which of x_i, y_i is larger) which happens with probability at most $|x_i - y_i|/L(v)$. Hence, by union bounding over these events:

$$\Pr[y \notin \mathcal{C}_v | x \in \mathcal{C}_v] \leq \sum_{i=1}^{d} \frac{|x_i - y_i|}{L(v)} = \frac{\|x - y\|_1}{L(v)}$$

Now observe that the above argument holds for all coordinates of z independently. This implies:

$$\Pr[y \in \mathcal{C}_v | x \in \mathcal{C}_v] \leq \prod_{i=1}^{d} (1 - \frac{|x_i - y_i|}{L(v)}) \leq \exp(-\frac{\|x - y\|_1}{L(v)})$$

We are now ready to prove the claim. Fix some $a \in A$. If $L(v_a) < \mathrm{dist}_{\mathrm{CH}}(a, B) \cdot 1/(4 \cdot \log n)$, that implies that for some $U/2^{i_0} \leq \mathrm{dist}_{\mathrm{CH}}(a, B)/(3 \cdot \log n)$ there exists a $b \in B$ such that both a and b belong to the same cell in the tree T with side length $U/2^{i_0}$. Let $\mathcal{C}_a^{i_0}, \ldots, \mathcal{C}_a^{k}$ be the cells of T with side length at most $U/2^{i_0}$ that a belongs to. Recall that this means $k \leq \mathcal{L}$. By property (ii), we know that for all $b \in B$:

$$\begin{aligned}
\sum_{i \in \{i_0, \ldots, k\}} \Pr[b \in \mathcal{C}_a^i | a \in \mathcal{C}_a^i] &\leq \sum_{\{i_0, \ldots, k\}} \exp\left(\frac{\|a - b\|_1}{2^i}\right) \\
&\leq \mathcal{L} \cdot \exp(\frac{\|a - b\|_1}{2^i}) \\
&\leq \mathcal{L} \cdot \exp\left(\frac{\mathrm{dist}_{\mathrm{CH}}(a, B)}{2^i}\right) \\
&\leq \mathcal{L} \cdot \exp(-3 \cdot \log n) \\
&\leq 1/n^3
\end{aligned}$$

By union bounding over all $b \in B$ and $a \in A$, and observing that these inequalities hold regardless which cells a falls on in specific levels of the tree, we get that the first item holds with high $1 - \mathrm{poly}(n)$ probability for all $a \in A$.

To prove point (ii), fix some $a \in A$ and $b = \mathrm{argmin}_{b \in B} \|a - b\|_1$. Let $\mathcal{C}_a^{i_0}, \ldots, \mathcal{C}_a^{0}$ be the cells of T with size lengths at least $\mathrm{dist}_{\mathrm{CH}}(a, B)$ containing a where $\mathcal{C}_a^i$ has side length $U/2^i$. If $L(v_a) \geq U/2^{i-1}$ then $b \notin \mathcal{C}_a^i$. Hence, by property (i) we have that $\Pr[L(v_a) \geq U/2^{i-1}] \leq \Pr[b \notin \mathcal{C}_a^i | a \in \mathcal{C}_a^i] \leq 2^i \cdot \|a - b\|_1/U$. Note that $L(v_a) \leq L(\mathcal{C}_a^0)$ by definition. Hence, summing over the $\mathcal{L}$ levels of the tree we get:

$$\begin{aligned}
\mathbb{E}[L(v_a)] &\leq \sum_{\{i_0, \ldots, 0\}} \Pr[L(v_a) \geq \frac{U}{2^{i-1}}] \cdot L(\mathcal{C}_a^i) \\
&\leq \sum_{\{i_0, \ldots, 0\}} \Pr[b \notin \mathcal{C}_a^i | a \in \mathcal{C}_i^i] \cdot \frac{U}{2^{i-1}} \\
&\leq \sum_{\{i_0, \ldots, 0\}} \frac{2^i \cdot \|a - b\|_1}{U} \cdot \frac{U}{2^{i-1}} \\
&\leq \sum_{\{i_0, \ldots, 0\}} 2 \cdot \|a - b\|_1 \\
&\leq 2 \cdot \mathcal{L} \cdot \mathrm{dist}_{\mathrm{CH}}(a, B)
\end{aligned}$$

Note that the second inequality holds as it does not matter which cells a falls on different levels of the tree for the set of proceeding inequalities.

B.3 Proof of Claim 3.5

Fix a point $a \in A$, and let X be the event when a is sampled in Algorithm 2. Let $v_a, v_2, ..., v_l$ be the path on tree starting at v_a ending at the leaf v_l containing a (hence $\gamma_A(v_l) = 1$). First, denote by Y the event when TREE-SAMPLER(T) returns node v_a, and by X_i the event that NODE-SAMPLER(v)E in line 4 of Algorithm 3 choose the node v_i on level i of T, for $1 < i \leq l$. By definition, we have that $\Pr[Y] = \frac{L(v_a)\cdot\gamma(v_a)}{\sum_{v\in T} L(v)\cdot\gamma(v)}$. Further,

$$\Pr[X_2|Y] = \frac{w(v_2)}{\sum_{u:u\in\mathbb{C}(v_1)} w(u)} = \frac{\gamma_A(v_2)}{\gamma(v_a)},$$

by definition of $w(\cdot)$. Similarly, for $3 \leq i \leq l$

$$\Pr[X_i|X_{i-1}] = \frac{w(v_i)}{\sum_{u:u\in\mathbb{C}(v_{i-1})} w(u)} = \frac{\gamma_A(v_i)}{\gamma_A(v_{i-1})}$$

Finally, we have that

$$\begin{aligned}
\Pr[X] &= \Pr[Y \cap X_2 \cap ... \cap X_l] \\
&= \Pr[Y] \cdot \frac{\gamma_A(v_2)}{\gamma(v)} \cdot \prod_{i=3}^{l} \frac{\gamma_A(v_i)}{\gamma_A(v_{i-1})} \\
&= \Pr[Y] \cdot \frac{\gamma_A(v_l)}{\gamma(v_a)} \\
&= \frac{L(v_a)}{\sum_{v\in T} L(v)\cdot\gamma(v)}.
\end{aligned}$$

B.4 Proof of Lemma 2.3

The data structure maintains a balanced binary tree of the elements of A, where each leaf corresponds to some element. In addition, each node of the binary search tree keeps track of the sum of the weights of the elements corresponding to the leaves in its sub-tree.

On query, the data-structure draws a random real value x in $[0, \sum_{i=1}^{n} w_i]$. Starting from the root of the search tree, it completes a walk to a leaf. When deciding between child nodes v and u (from left to right) with leaf weight sums w_v, w_u in a step of the walk, it chooses v if $x \leq w_v$. Once reaching a leaf, it returns the element assigned to it.

Observe that if the ordering of the leafs from left to right is $a_1, \ldots a_n$, then this process chooses element a_i if $x > \sum_{j<i} w_i$ and $x \leq \sum_{j\leq i} w_i$, that is with probability $w_i / \sum_{j\in[n]} w_j$. As the tree will have depth at most $O(\log n)$, the query takes $O(\log n)$ time. The maintenance of a balanced binary tree of n elements in $O(\log n)$ is folklore.

B.5 Proof Claim 3.6

First, note that when we decide to query the distance of some $a \in A$ from B, we use a $(\alpha + \epsilon/4)$-approximate nearest neighbor datastructure, which might be randomized. Hence, instead of returning $\mathrm{dist}_{\mathrm{CH}}(a, B)$, we obtain some $Z_a \cdot \mathrm{dist}_{\mathrm{CH}}(a, B)$, where Z_a is a random variable taking real values in $[1, 1 + \alpha/4 + \epsilon/4]$. Note that this implies that $\mathbb{E}[Z_a^2] \leq (1 + \alpha/4 + \epsilon/4)^2 \leq 2\alpha^2$. Also, note that the randomization of $Z_a | a \in A$ is independent from the randomization of the rest of the algorithm (i.e. the randomness used for sampling elements of A and that of $\mathcal{H}$). Furthermore, the randomness we use to sample $a \in A$ is independent of the randomness of the quad-tree (that is of $L(v_a)$ and $\gamma(v)$ values).

$$
\begin{aligned}
\mathrm{Var}[X] &\le \mathbb{E}[X^2]\\
&= \sum_{a\in A} \mathbb{E}\left[\frac{L(v_a)}{\sum_{v\in T} L(v)\cdot\gamma(v)}\frac{(\sum_{v\in T}\gamma(v)\cdot L(v))^2}{L(v_a)^2}\right]\cdot \mathbb{E}[Z_a^2]\cdot \mathrm{dist}_{\mathrm{CH}}(a,B)^2\\
&= \sum_{a\in A} \mathbb{E}\left[\frac{\sum_{v\in T}\gamma(v)\cdot L(v)\cdot \mathrm{dist}_{\mathrm{CH}}(a,B)}{L(v_a)}\right]\cdot \mathbb{E}[Z_a^2]\cdot \mathrm{dist}_{\mathrm{CH}}(a,B)\\
&\le \mathbb{E}[\sum_{v\in T}\gamma(v)\cdot L(v)]\cdot \sum_{a\in A}\log n\cdot \mathrm{dist}_{\mathrm{CH}}(a,B)\cdot 6\alpha^2\\
&\le \log^2 n\cdot \mathcal{L}\cdot \mathrm{dist}_{\mathrm{CH}}(A,B)^2\cdot 12\alpha^2
\end{aligned}
$$

The second inequality holds due to Claim 3.4, part (i), with high $(1-1/\mathrm{poly}(n))$ probability. The last inequality follows from the same claim, part (ii).

B.6 Proof of Lemma 3.2

On query, the algorithm returns the value of random variable $\tilde{\mu}/(m\cdot(1+\epsilon/2))$. First, in order for this to be a $(1+\alpha+\epsilon)$-approximation to $\mathrm{dist}_{\mathrm{CH}}(A,B)$, by definition it must hold that

$$
\frac{\tilde{\mu}}{m\cdot(1+\epsilon/2)} \le \mathrm{dist}_{\mathrm{CH}}(A,B) \le \frac{\tilde{\mu}}{m\cdot(1+\epsilon/2)}\cdot(1+\alpha+\epsilon). \tag{1}
$$

Let X be a random variable described by Claim 3.6, that is a random variable taking value $\mathbf{NN}(a,B,\alpha/4)\cdot\sum_{v\in T}\gamma(v)\cdot L(v)/L(v_a)$ with probability $L(v_a)/\sum_{v\in T}\gamma(v)\cdot L(v)$. By the definition of the nearest neighbor oracle we have that $\mathrm{dist}_{\mathrm{CH}}(A,B)/(1+\alpha/4)\le\mathbb{E}[X]\le\mathrm{dist}_{\mathrm{CH}}(A,B)$. Since $\tilde{\mu}/\alpha$ is the average of m i.i.d. copies of X, then $\mathrm{Var}[\tilde{\mu}/m]\le\mathrm{Var}[X]/m$ and hence

$$
\mathrm{Var}[\tilde{\mu}/m] \le \mathrm{Var}[X]\cdot\frac{\epsilon^2\cdot\log^2 n}{240\cdot\alpha^2} \le \frac{\mathrm{dist}_{\mathrm{CH}}(A,B)^2\cdot\epsilon^2}{20\cdot\max\{\alpha^2,1\}} \tag{2}
$$

with high probability by Claim 3.6. Now, by a simple application of Chebyshev's inequality we have

$$
\Pr\left[\left|\frac{\tilde{\mu}}{m}-\mathbb{E}\left[\frac{\tilde{\mu}}{m}\right]\right| > 4\cdot\sqrt{\mathrm{Var}\left[\frac{\tilde{\mu}}{m}\right]}\right] \le \frac{1}{16} .
$$

Since by linearity of expectation it holds that $\mathrm{dist}_{\mathrm{CH}}(A,B)/(1+\alpha/4)\le\mathbb{E}[\frac{\tilde{\mu}}{m}]\le\mathrm{dist}_{\mathrm{CH}}(A,B)$, we have

$$
\Pr\left[\frac{\tilde{\mu}}{m} > \mathrm{dist}_{\mathrm{CH}}(A,B)+4\cdot\sqrt{\mathrm{Var}\left[\frac{\tilde{\mu}}{m}\right]}\right]+\Pr\left[\frac{\tilde{\mu}}{m} < \frac{\mathrm{dist}_{\mathrm{CH}}(A,B)}{1+\alpha/4}-4\cdot\sqrt{\mathrm{Var}\left[\frac{\tilde{\mu}}{m}\right]}\right] \le \frac{1}{16}.
$$

Plugging in 2, we obtain

$$
\Pr\left[\frac{\tilde{\mu}}{m} > \mathrm{dist}_{\mathrm{CH}}(A,B)+\frac{\epsilon\cdot\mathrm{dist}_{\mathrm{CH}}(A,B)}{\sqrt{5}\cdot\max\{\alpha,1\}}\right]+\Pr\left[\frac{\tilde{\mu}}{m} < \frac{\mathrm{dist}_{\mathrm{CH}}(A,B)}{1+\alpha/4}-\frac{\epsilon\cdot\mathrm{dist}_{\mathrm{CH}}(A,B)}{\sqrt{5}\cdot\max\{\alpha,1\}}\right] \le \frac{1}{16}.
$$

For all $\alpha>0$ and $0<\epsilon<1$ this implies that:

$$
\Pr\left[\frac{\tilde{\mu}}{m\cdot(1+\epsilon/2)} > \mathrm{dist}_{\mathrm{CH}}(A,B)\right]+\Pr\left[\frac{\tilde{\mu}}{m\cdot(1+\epsilon/2)} < \mathrm{dist}_{\mathrm{CH}}(A,B)\cdot(1+\alpha+\epsilon)\right] \le \frac{1}{16}.
$$

Therefore, the output of the query is a $(1+\alpha+\epsilon)$-approximation to $\mathrm{dist}_{\mathrm{CH}}(A,B)$ with $15/16>3/4$ probability.

C Pseudo-code of Our Algorithm

In this section, we give pseudo-codes for the dynamic algorithm described in Section 3.

The updates to $\gamma_A(v)$ and $\gamma_B(v)$ in the last line of each algorithm can be done using standard results from literature (see e.g. [dBHTT07]). The following pseudo-code shows how the algorithm updates γ values after point x is inserted into/deleted from $A \cup B$.

Algorithm 1 UPDATE POINT(x)

```
$v_1 \leftarrow$ leaf containing $x$
$\Pi \leftarrow$ path $(v_1, v_2, \ldots, r)$ from $v_1$ to root $r$ in $T$
$v_1, \ldots, v_k \leftarrow v \in \Pi | \mathcal{C}_v \cap (B \setminus x) = \emptyset$
$v' \leftarrow$ ancestor of $v_k$ in $T$
if Insertion into $A$ then
    $\gamma(v') \leftarrow \gamma(v') + 1$
end if
if Deletion from $A$ then
    $\gamma(v') \leftarrow \gamma(v') - 1$
end if
if Insertion into $B$ then
    $\gamma(v') \leftarrow \gamma(v') - \gamma_A(v_k)$
    $\gamma(v_1) \leftarrow \gamma_A(v_1)$
    $\gamma(v_i) \leftarrow \gamma_A(v_i) - \gamma_A(v_{i-1})$ for $k \geq i > 1$
end if
if Deletion from $B$ then
    $\gamma(v') \leftarrow \gamma(v') + \sum_{i \in [k]} \gamma(v_i)$
    $\gamma(v_i) \leftarrow 0$ for $i \in [k]$
end if
```

The algorithm also updates the global sampler NODE-SAMPLER(T) and affected TREE-SAMPLER(v)s such that after an update:

- TREE-SAMPLER(T) contains all nodes $v \in T$ with weight $w(v) = L(v) \cdot \gamma(v)$ (if non-negative),
- NODE-SAMPLER(v) for all $v \in T$ contains all children u of v with weight $\gamma_A(u)$ if $\mathcal{C}_u \cap B = \emptyset$.

Note that $\sum_{v \in T} \gamma(v) = |A| \leq n$ hence TREE-SAMPLER(T) contains at most n elements at all times. Similarly, for any $v \in T$ for the set of child nodes U of T it holds that $\sum_{u \in U} \gamma_A(u) \leq |A| = n$ hence all NODE-SAMPLER(v) also contain at most n elements (and hence updates and answers queries in $O(\log n)$ time by Lemma 2.3).

The following pseudo-codes describes how queries are handled by the algorithm of Section 3.1.2. For a single query, we invoke Algorithm 4.

Algorithm 2 SAMPLER(A) procedure

```
Sample vertex $v$ of $T$ using TREE-SAMPLER$(T)$
Return SAMPLER-FIND$(v)$
```

Algorithm 3 SAMPLER-FIND(v) procedure

```
if $v$ is a leaf of quad-tree $T$ then
    return $a$ contained in $v$
else
    Sample child $u$ of $v$ using NODE-SAMPLER$(v)$
    SAMPLER-FIND$(u)$
end if
```

Algorithm 4 QUERY-CHAMFER(A, B, ϵ, α) procedure

```
$\tilde{\mu} = 0$
$m = \frac{120\cdot\mathcal{L}\cdot\log n \max\{\alpha^2,1\}}{\epsilon^2}$        ▷ $m = O(\frac{\log^2 n\cdot\max\{\alpha^2,1\}}{\epsilon^2})$
for $i = 1$ to $m$ do
    $a \leftarrow$ SAMPLER$(A)$
    $x_a \leftarrow$ NN$(a, B, \alpha/4) \cdot \frac{\sum_{v\in T}\gamma(v)\cdot L(v)}{L(v_a)}$
    $\tilde{\mu} = \tilde{\mu} + x_a$
end for
Return $\frac{\tilde{\mu}}{m\cdot(1+\epsilon/2)}$
```

D Missing Figures in Section 4

In this section we give the missing Figures 4 and 5.

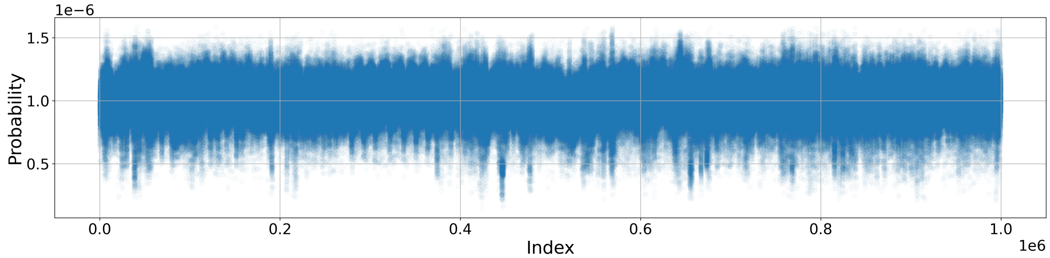

Figure 4: The value $\text{dist}(a, B)/\sum_{a'\in A}\text{dist}(a', B)$ over all points $a \in A$ for SIFT dataset, which are the "ideal" probabilities for importance sampling.

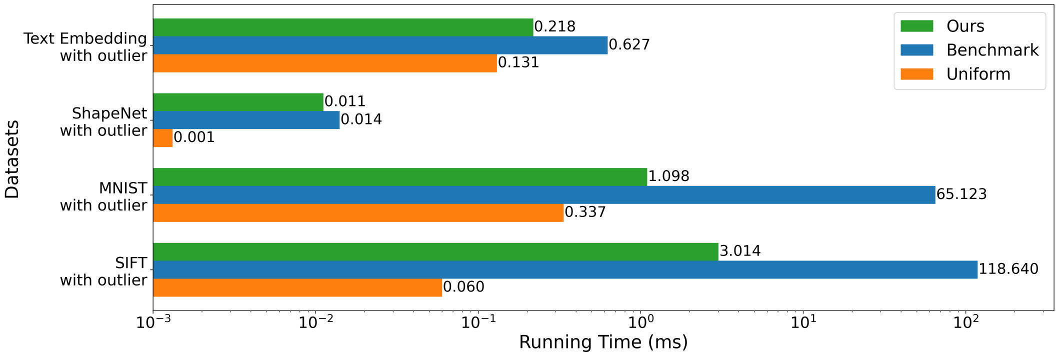

Figure 5: Average running time per window update for all algorithms on datasets *with* outliers.

E Additional Experiments

Table 2: Experiment parameters when both A and B are dynamic.

dataset	window size	sample size
Text Embedding	1.5k	100
ShapeNet	1.5k	100
Fashion-MNIST	3.5k	100
SIFT	50k	500

To evaluate the case when both A and B are dynamic, we simulate the dynamic updates via a sliding window, similar to the default setting where only B is dynamic. However, the main difference is

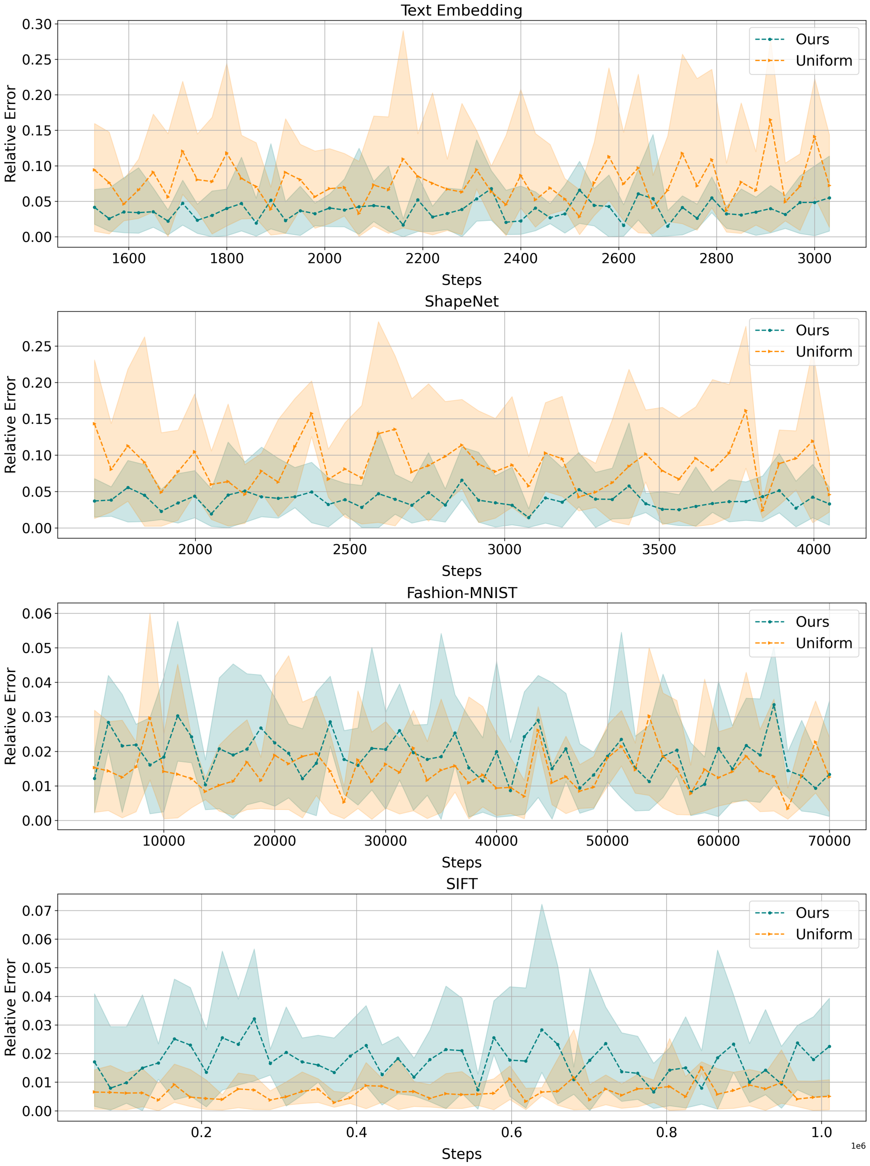

Figure 6: Relative error curves when both A and B are dynamic, with the same setup as in Figure 1.

that the window not only consists of insertions of points in B, but also those in A. Specifically, we insert alternatively between A and B in proportion to their dataset sizes (e.g., for MNIST dataset, $|A| : |B| = 6 : 1$, we perform six insertions from A followed by one from B). Experiment parameters for this case are summarized in Table 2. We depict the relative error curve in Figure 6 and report the average running time per sliding window update in Figure 7.

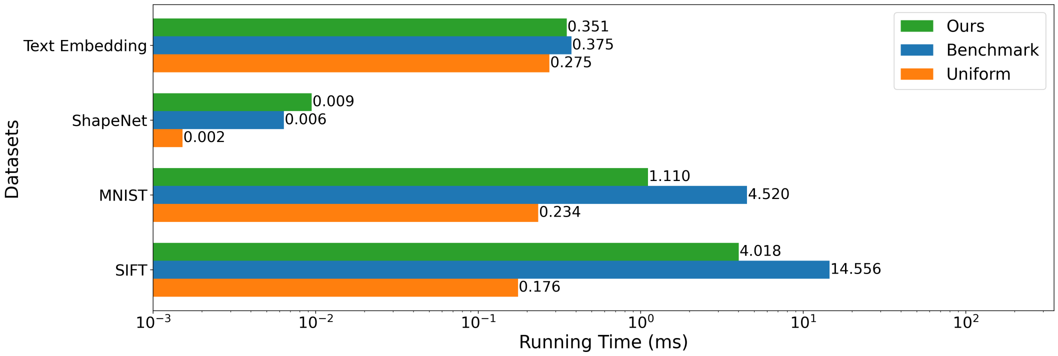

Figure 7: Average running time per window update for all algorithms when both A and B are dynamic.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes, the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope, encompassing both theoretical guarantees and practical results.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses theoretical run-time bounds for the proposed algorithm. Additionally, the paper provides a comprehensible comparison with the "Benchmark" and "Uniform" baseline algorithms. The results highlight the limitations of our technique on the SIFT dataset, where our algorithm performs slightly worse than the "Uniform" benchmark, likely due to the highly uniform nature of the SIFT data.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.

- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: All theoretical claims provide the full set of assumptions and are supported by detailed proofs, either presented in the main body of the paper or outlined through concise sketches, with full proofs deferred to the appendices.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Yes, the paper fully discloses all the information necessary to reproduce the main experimental results relevant to its claims and conclusions. The code is included as part of the supplementary material, and all datasets used in the experiments are publicly available online.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same

dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The supplementary material includes clear instructions and all necessary resources, including open access to the datasets and code.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

Justification: The paper specifies all relevant training and testing details. Additionally, it provides a comprehensive set of experimental plots that support and illustrate the theoretical guarantees of the algorithm.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The plots present clear relative bounds that effectively convey the variability and statistical significance of the experimental results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All experiments were conducted on the same machine, with its specifications clearly stated in the paper

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research adheres fully to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: The paper does not explicitly discuss societal impacts, but the nature of the work does not suggest any apparent negative societal implications.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We added references to the datasets we use. We do not use any external library in the code.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Yes, the code is provided as supplementary material and is well documented.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.