Approximate Nearest Neighbor under Edit Distance via Product Metrics

Piotr Indyk*

Abstract

We present a data structure for the approximate nearest neighbor problem under edit metric (which is defined as the minimum number of insertions, deletions and character substitutions needed to transform one string into another). For any $l \geq 1$ and a set of n strings of length d, the data structure reports a 3^l-approximate Nearest Neighbor for any given query string q in $O(d)$ time. The space requirement of this data structure is roughly $O(n^{d^{1/(l+1)}})$, i.e., strongly subexponential. To our knowledge, this is the first data structure for this problem with both $o(n)$ query time and storage subexponential in d.

1 Introduction

The *Nearest Neighbor Search* (NN) problem is: Given a set P of n points in a metric space X, preprocess P so as to efficiently answer queries for finding the point in P closest to a query point q. NN and its approximate versions are among the most extensively studied problems in the fields of Computational Geometry and Algorithms, resulting in discovery of many efficient algorithms. In particular, for the case when the metric space X is a low-dimensional Euclidean space l_2^d, it is known how to construct data structure for exact [Cla88, Mei93] or approximate [AMN$^+$94, Kle97, IM98, HP01] NN with query time $(d + \log n)^{O(1)}$. Unfortunately, those data structures require space exponential in d. More recently, several data structures using space (quasi)-polynomial in n and d, and query time sublinear in n, have been discovered for approximate NN under l_1 and l_2 [KOR98, IM98, Ind00] and l_∞ [Ind98] norms.

While many metrics of interest for nearest neighbor search are norms, quite a few of them are not. A prominent example of the latter is the *Levenshtein* (or *edit*) metric, which, for two strings s and s', is defined as the minimum number of insertions, deletions and substitutions that transform s into s'. Edit distance is a natural measure of similarity between sequences of characters, and is widely used e.g., in computational biology and text processing. Unfortunately, the only nearest neighbor algorithms for edit metric known so far were the "trivial" ones, i.e., using linear scan (which suffers from $\Omega(n)$ query time) or using exhaustive storage (which suffers from memory requirements exponential in the length of strings).

A potential approach for devising an approximate NN data structure for edit metric is to *embed* it into a normed space, and use NN algorithms for normed spaces. In other words, we would like to map every sequence into a vector such that the distance between the vectors approximates the distance between the sequences. This approach has been successfully applied for some not-normed metric spaces. In particular, it was shown that a variant of the edit distance (called *block edit distance*) can be embedded into Hamming space or l_1 norm with distortion roughly $O(\log d)$ [CPSV00, MS00, CM02], where d is the maximum string length. The block edit distance measures the minimum number of *blocks* (not just character) operations needed to transform one string into another. Although it is plausible that the standard edit metric can be embedded into, say, l_1 norm with low distortion, so far researchers were unable to provide such an embedding. A (somewhat weak) lower bound of $3/2$ for distortion of any such embedding has been recently given in [ADG$^+$03].

In this paper we present the first non-trivial approximate algorithm for NN under edit metric. For any $l \geq 1$, our algorithm solves the 3^l-approximate NN using $O(d)$ query time and space roughly $n^{d^{1/(l+1)}}$ space. This, e.g., allows us to obtain a constant factor approximation using space that is exponential only in d^α, for arbitrarily small $\alpha > 0$. Alternatively, we can use $l = \alpha \log_3 d$, and obtain a d^α-approximate algorithm using polynomial space.

The result is obtained by representing edit metric as a product of several "smaller" metrics, using a method first proposed in [Ind02]. A product of metrics $M_1, \ldots M_k$ with distance functions $D_1, \ldots D_k$ and a function $f : \Re^k \to \Re$ is a metric over $M_1 \times \ldots \times M_k$ with distance function D such that $D((p_1, \ldots, p_k), (q_1, \ldots, q_k)) = f[D_1(p_1, q_1) \ldots D_k(p_k, q_k)]$. The most typical functions f are "max" and "sum"; in general, our result holds for any b-monotone function (see Preliminaries).

We prove the following general result about NN

*Laboratory for Computer Science, MIT, Cambridge, MA 01239.
This work was supported in part by NSF CAREER grant CCR-0133849.

for product metrics. Assume that each M_i is equipped with a c-approximate NN data structure with query time $Q(n)$ and using space $S(n)$. Then we can construct a data structure for $O(c)$-approximate NN, with query time $kQ(n)$ and space roughly $kS(n) + n^k$. Given the above theorem, we show that NN for edit metric can be reduced to NN over a product of several edit metrics over shorter sequences. Then, the NN data structure for each individual metric can be constructed either recursively, or using the exhaustive storage approach.

Although the main focus of this paper is on algorithms for edit metric, we mention that NN data structures for product metrics are also of significant interest by itself. Product metrics are of interest in scenarios where one searches for an object satisfying a set of metric constraints, e.g., "Find a person with similar signature and similar face photo". We can answer such a query by using NN data structure for a product of individual metrics.

Because of this reason, we further study approximate NN algorithms for product metrics. In particular, we obtain the following result for the case when f is a sum. As before, assume that each metric M_i is equipped with an approximate NN data structure with query time $Q(n)$ and space roughly $S(n)$. Then we construct an $O(\log n \cdot \log k \cdot \log\log n)$-approximate data structure that solves the *decision version* (see Preliminaries) of NN for the sum-product of M_i's, with query time $Q(n)(k+\log n)^{O(1)}$ and space $n^{O(1)} + O(kS(n))$. Note that, unlike in the previous case, the space bound does not depend exponentially on k. However, the $\Omega(\log n)$ approximation factor makes it impossible to use this data structure in the context of edit metric.

Related results. The approach of solving NN in not-normed metrics via product metrics was introduced in [Ind02]. In that paper, the author gave a $O(c \log\log n)$-approximate algorithm for the case when the function f computes the maximum of its arguments. The space bound of that data structure lacked the n^k term present here; other parameters were similar. The data structure was then used to provide a subexponential space approximate NN data structure for the *Frechet* metric.

Unfortunately, the approach of [Ind02] relied significantly on properties of the maximum function, since it essentially generalized the earlier algorithm of [Ind98] for l_∞^k norms. On the other hand, in order to obtain an algorithm for the edit metric, one needs to design an algorithm for the case when f computes the *sum* of its arguments. Thus, one cannot use the data structure of [Ind02] to obtain the result presented in this paper.

We mention that by using our product metric data structure, we can construct an approximation algorithm for NN under Frechet metric, with space and query time as in [Ind02], that has only *constant* approximation factor.

Our techniques. Our first data structure (for any "monotone" function f) is quite simple. Its basic idea is to partition the product metric M into roughly n^k regions, within which an approximate nearest neighbor can be fixed and pre-computed during the preprocessing. In addition, we can determine the region that a given query belongs to, by only invoking the nearest neighbor oracles for the individual metrics M_i.

Our second algorithm can be viewed as an abstract version of the Locality-Sensitive Hashing algorithm of [IM98] . It allows us to reduce the problem of approximate NN in sum-product metrics to approximate NN in max-product metrics, by embedding the former into the latter. Then we can use the algorithm of [Ind02] to solve the problem in the latter space.

2 Preliminaries

Let $M_i = (X_i, D_i)$ be metrics, $i = 1 \dots k$, and let $f : \Re^k \to \Re$. The *f-product* $M = (X, D)$ of $M_1 \dots M_k$ is defined by setting $X = X_1 \times \dots \times X_k$, and, for any $p, q \in X$, setting

$$D(p, q) = f(D_1(p_1, q_1), \dots, D_k(p_k, q_k)).$$

Note that if for any $a, b, c \in \Re^k$, $a + b \geq c$ implies $f(a) + f(b) \geq f(c)$, then M is a metric.

A function f is called *b-monotone*, if for any $0 \leq x, x' \in \Re^k$ and $c \in \Re$, if $x \leq x' \leq cx$ then $f(x) \leq f(x') \leq bcf(x)$. Note that the functions $f(x_1 \dots x_k) = |x|_p$, for any $0 < p \leq \infty$, are 1-monotone.

We define $D_E(s, s')$ to be the edit distance between strings $s, s' \in \Sigma^*$, i.e., the minimum number of insertions and deletions and substitutions of symbols needed to transform s into s'. In addition, we set $M_E^d = (\Sigma^{\leq d}, D_E)$.

The c-approximate Nearest Neighbor problem (c-NN), under a metric $M = (X, D)$, is defined as follows: given $P \subset X$, $|P| = n$, build a data structure which given any $q \in X$ finds $p' \in P$ such that $D(q, p') \leq c \min_{p \in P} D(q, p)$. The decision version of the above problem is defined as: given $P \subset X$, $|P| = n$, and $r > 0$, build a data structure which, given any $q \in X$, if there is any $p \in P$ that $D(q, p) \leq r$, finds $p' \in P$ such that $D(q, p') \leq cr$. Note that r is fixed before the data structure is constructed. If the data structures are are randomized, we require that the above specifications holds for a fixed q with constant probability. As shown in [IM98, HP01], the decision and optimization versions of approximate NN are equivalent, up to polylogarithmic factors in space and query time bounds.

For any metric $M = (X, D)$ and $a > 0$, we define $aM = (X, aD)$ to be a metric such that $(aD)(p, q) = aD(p, q)$ for all $p, q \in X$.

3 Nearest neighbor under f-product metrics

Consider metric spaces M_i, $i = 1 \dots k$. Assume that all distances in metrics M_i are integers from $\{1 \dots \Delta\}$. In addition, let f be a b-monotone function $f : \Re^k \to \Re$. Let $M = (X, D)$ be the f-product of $M_1 \dots M_k$. Let $P \subset X$, $|P| = n$. Define $P_i = \{p_i : p \in P\}$, and let $m = \max_i |P_i|$.

THEOREM 3.1. *Assume that each metric M_i is equipped with a c-NN data structure with query time $Q(n)$ and using space $S(n)$. Then, for any $\epsilon \geq 0$, there is a data structure for $b(1 + 2c)(1 + O(\epsilon))$-NN, with query time $kQ(n)$ using space $kS(n) + (m \log_{1+\epsilon} \Delta)^k$ (we denote $\log_1 \Delta = \Delta$).*

Proof: For simplicity, we focus on the case $\epsilon = 0$. Generalization to $\epsilon > 0$ is straightforward.

The data structure is constructed as follows. First, we build a c-NN data structure for each M_i. Then, we prepare a lookup table $A : P_1 \times \{1 \dots \Delta\} \times \dots \times P_k \times \{1 \dots \Delta\} \to P$. We describe how to construct A later.

Given a query point $q = (q_1, \dots, q_k)$, the data structure proceeds as follows. First, for each i, it queries the c-NN data structure for M_i with an argument q_i. Let p_i be the point returned by the i-th data structure. Then the data structure returns a point in A indexed by $(p_1, D_1(q_1, p_1), \dots, p_k, D_k(q_k, p_k))$.

The query time and space bound follows directly from the above description. It remains how to construct A so that the algorithm returns an approximate nearest neighbor.

Let q and $p_1 \dots p_k$ be as above. Consider any point $p' = (p'_1 \dots p'_k) \in P$. We define $\overline{D}(q, p')$ to be an approximation of the distance $D(q, p')$. Specifically, we set

$$\overline{D}_i(q_i, p'_i) = \max[D_i(q_i, p_i)/c, D_i(p_i, p'_i) - D_i(q_i, p_i)]$$

and $\overline{D}(q, p') = f(\overline{D}_1(q_1, p'_1), \dots, \overline{D}_k(q_k, p'_k))$. Observe that $\overline{D}(q, p')$ can be computed using exclusively the information presented to the lookup table, without the full knowledge of q.

Observe that $D_i(q_i, p'_i) \geq D_i(q_i, p_i)/c$ and $D_i(q_i, p'_i) \geq D_i(p_i, p'_i) - D_i(q_i, p_i)$; this implies $\overline{D}_i(q_i, p'_i) \leq D_i(q_i, p'_i)$. In addition, we have

$$\begin{aligned} D_i(q_i, p'_i) &\leq D_i(p_i, p'_i) + D_i(q_i, p_i) \\ &= [D_i(p_i, p'_i) - D_i(q_i, p_i)] + 2D_i(q_i, p_i) \\ &\leq \overline{D}_i(q_i, p'_i) + 2D_i(q_i, p_i) \\ &\leq (1 + 2c)\overline{D}_i(q_i, p'_i) \end{aligned}$$

Thus, $\overline{D}_i(q_i, p'_i) \leq D_i(q_i, p'_i) \leq (1 + 2c)\overline{D}_i(q_i, p'_i)$, which implies $\overline{D}(q, p') \leq D(q, p') \leq b(1 + 2c)\overline{D}(q, p')$. Therefore, if we set $A[p_1, D_1(q_1, p_1), \dots, p_k, D_1(q_k, p_k)]$ to a point p' which minimizes $\overline{D}(q, p)$ over $p \in P$, the algorithm will report a $b(2c + 1)$-approximate nearest neighbor.

□

4 Edit distance

THEOREM 4.1. *For any integer $l \geq 1$, there is a data structure solving 3^l-NN under the edit metric M_E^d, using $(d|\Sigma|n)^{O(d^{1/(l+1)})}$ space and $O(d)$ query time.*

In the following we will show a data structure for 3-NN under the edit distance, which uses $(d|\Sigma|)^{\sqrt{d \log n}} \cdot (nd^3)^{\sqrt{d/\log n}}$ space and has $O(d)$ query time. The generalization to the bounds stated are obtained by recursive application of the same idea.

Consider two strings $q, p \in \Sigma^d$. Partition q into k substrings $q_1 \dots q_k$ of length d/k each (assuming k divides d).

CLAIM 1. *The edit distance D_E satisfies the following equality*

$$D_E(q, p) = \min_{p_1 \dots p_k = p} \sum_{i=1}^{k} D_E(q_i, p_i)$$

Set $k = \sqrt{d/\log n}$. We create an auxiliary set of points P' containing all k-tuples $(p_1, \dots, p_k)$ such that $p_1 \dots p_k \in P$. By Claim 1, the (approximate) nearest neighbor of point q in the set P with respect to D_E can be computed by finding the (approximate) nearest neighbor a query point $q' = (q_1, \dots, q_k)$ in P' with respect to the distance $\sum_i D_E(\cdot, \cdot)$ (informally, we can combine the "min" from the nearest neighbor definition with the "min" of Claim 1 into one "min"). The latter distance function is a sum-product of k copies of M_E^d. The key property of this transformation, however, is that each of q_i's in $(q_1, \dots, q_k)$ has length d/k. Therefore, the 1-NN problem under the component metrics can be trivially solved with $O(d/k)$ query time and using $|\Sigma|^{d/k}$ space, via exhaustive storage of answers to all queries. At the same time, the sum function is 1-monotone. Thus, using the results from the previous section, we can build a data structure solving the 3-NN problem under the $\sum_i D_E(\cdot, \cdot)$ metric. Since $\max_i |P'_i| = m \leq d^2 n$ and $\Delta = d$, the space bound follows.

5 Sum-product

In this section we show how to perform an embedding of a sum-product M of metrics $M_1, \dots, M_k$ into a max-product of those metrics; the max-product includes several (scaled) copies of each M_i. The embedding is somewhat weak, but is nevertheless sufficient to reduce the decision version of the problem of computing approximate nearest neighbor under sum-product of metrics to the same problem over max-product of those metrics. The latter problem can be solved using the algorithm of [Ind02].

The embedding is based on the following approach. Assume k is a power of 2. We start by choosing $I_i \subset \{1 \dots k\}$, $i = 1 \dots \log k$. Each I_i is chosen by picking $k/2^i$ elements independently and uniformly at random from $\{1 \dots k\}$, without replacement. In addition, we define $I_0 = \{1 \dots k\}$.

Our mapping F will map a point $p \in M$ into $M' = (X', D')$, which is a max-product of metrics $M^0 \dots M^{\log k}$. Each M^i itself is a max-product of metrics $M_j, j \in I_i$, multiplied by 2^i. The coordinate $F^i_j(p)$ of $F(p)$ corresponding to i, j as above is defined to be p_j. The mapping $F(p)$ is obtained by concatenating $F^i_j(p)$ over all i, j.

The key properties of the mapping F follow from the following lemma.

LEMMA 5.1. *For any $p, q \in M$, $t, c > 1$ we have:*

1. *If $D(p,q) \le 1$, then*

$$\Pr[\forall i \max_{j \in I_i} D_j(p_j, q_j) \le 2^{-i}] \ge 1/k^2$$

2. *If $D(p,q) > t \cdot c \cdot \log k$, then*

$$\Pr[\exists i \max_{j \in I_i} D_j(p_j, q_j) > t2^{-i}] > 1 - e^{-c}$$

Proof: We start from part (1). Consider first $i \ge 1$. By Markov inequality, the fraction of j's such that $D_j(p_j, q_j) > 2^{-i} = k/2^i \cdot 1/k$ is at most $2^i/k$. Thus, the probability that I_i does not contain any of such j's is at least $(1 - 2^i/k)^{k/2^i} \ge 1/4$. The case of $i = 0$ holds trivially. Therefore, the probability that this holds for all $i = 0 \dots \log k$ is at least $1/4^{\log k} = 1/k^2$.

Now we consider part (2). Let s_i denote the number of j's such that $D_j(p_j, q_j) > tc2^{-i}$. We will show first that there exists i such that $s_i \ge 2^i$. Indeed, assume otherwise, i.e., $s_i < 2^i$ for all i. This would imply that $s_0 = 0$. In addition, we could upper bound $D_j(p_j, q_j)$ by

$$\sum_{i=1}^{\log k} (s_i - s_{i-1}) ct2^{-i+1} \le ct \sum_{i=1}^{\log k} 2^{-i} s_i < ct \log k$$

which yields a contradiction.

By the above argument, there exists i such that $s_i \ge 2^i$. Consider the following two cases:

Case 1: $c2^{-i} \ge 1$. In this case, $\max_{j \in I_0} D_j(p_j, q_j) > t$, and thus $i = 0$ satisfies the conditions of the lemma.

Case 2: $c2^{-i} < 1$. In this case, set $i' = i - \log c$. Note that $i' \ge 1$. From the above description we know that $I_{i'}$ contains $k/2^{i'} = ck/2^i$ elements. The probability that all of them are $\le t2^{-i'}$ is at most

$$(1 - (ck/2^i)/k)^{2^i} = (1 - c/2^i)^{2^i} \le e^{-c}$$

Thus i' satisfies the conditions of the lemma.

□

THEOREM 5.1. *The embedding $F : M \to M'$ satisfies the following conditions: for any $p, q \in M$ we have:*

1. *If $D(p,q) \le 1$, then*

$$\Pr[D'(F(p), F(q)) \le 1] \ge 1/k^2$$

2. *If $D(p,q) > t \cdot c \cdot \log k$, then*

$$\Pr[D'(F(p), F(q)) > t] > 1 - e^{-c}$$

COROLLARY 5.1. *Assume that each metric M_i is equipped with a b-NN data structure with query time $Q(n)$ and using space $S(n)$. Then there is a $O(b \log n \cdot \log k \cdot \log\log n)$-NN data structure for M, with query time $Q(n)(k + \log n)^{O(1)}$ and space $n^{O(1)} + O(kS(n))$.*

Proof: Follows by combining Theorem 5.1 and the algorithm of [Ind02]. Specifically, we set $c = \Theta(\log n)$ and $t = \Theta(b \log\log n)$. By scaling, we can assume that the threshold r in the decision version of approximate NN is equal to 1. Assume that there is a data point p which is within distance 1 from q. By performing the embedding as in Theorem 5.1 we are guaranteed that:

- The distance between q and p does not increase with probability $1/k^2$
- The distance between q and any p' such that $D(q, p') > t \cdot c \cdot \log k$ remains greater than t with overwhelming probability.

Thus, with probability slightly smaller than $1/k^2$, a t-approximate nearest neighbor of $F(q)$ is a correct solution to the decision version of a $O(t \log n \cdot \log k)$-approximate NN for q. The probability can be amplified to a constant by building $O(k^2)$ data structures and using all of them. It suffices to solve a decision version of an t-approximate NN under M'. Since M' is a max-product metric, where each of components is equipped with a b-approximate NN data structure, we can solve the latter problem using the result of [Ind02] (as mentioned in the Introduction). □

6 Conclusions

In this paper we showed how to solve NN in edit metrics by designing a new data structure for product metrics. The approach undertaken in this paper is fairly general, and can be probably used for other metrics which do not seem amenable to constant distortion embeddings into normed spaces. However, the latter option would still be more attractive, due to much lower space requirements.

This work raises several interesting open problems on the complexity of the nearest neighbor problem under product metrics. One natural question is if it is possible to improve the approximation factor of 3, and maybe even solve the *exact* NN problem. However, there is evidence that the factor of 3 could be tight, even for the case when $f = \max$. Indeed, assume we could get a c-NN with space simply exponential in k, for $c < 3$. Then we could obtain a c-approximation algorithm for NN under l_∞^d norm that uses space only exponential in $O(\sqrt{d})$, by setting $k = \sqrt{d}$. By the result of [Ind98], this would imply a data structure for the partial match query over $\{0,1\}^d$, with space only exponential in $\sqrt{d}$. Such a result, although possible, appears to be somewhat unlikely (see [CIP02] for more background on the partial match problem).

References

[ADG$^+$03] A. Andoni, M. Deza, A. Gupta, P. Indyk, and S. Raskhodnikova. Lower bounds for embedding of edit distance into normed spaces. *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, 2003.

[AMN$^+$94] S. Arya, D.M. Mount, N.S. Netanyahu, R. Silverman, and A. Wu. An optimal algorithm for approximate nearest neighbor searching. *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 573–582, 1994.

[CIP02] Moses Charikar, Piotr Indyk, and Rina Panigrahy. New algorithms for subset query, partial match, orthogonal range searching and related problems. *International Colloquium on Automata,Languages, and Programming*, 2002.

[Cla88] K. Clarkson. A randomized algorithm for closest-point queries. *SIAM Journal on Computing*, 17:830–847, 1988.

[CM02] G. Cormode and S. Muthukrishnan. The string edit distance matching problem with moves. *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, 2002.

[CPSV00] G. Cormode, M. Paterson, C. Sahinalp, and U. Vishkin. Communication complexity of document exchange. *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, 2000.

[HP01] S. Har-Peled. A replacement for voronoi diagrams of near linear size. *Proceedings of the Symposium on Foundations of Computer Science*, 2001.

[IM98] P. Indyk and R. Motwani. Approximate nearest neighbor: towards removing the curse of dimensionality. *Proceedings of the Symposium on Theory of Computing*, 1998.

[Ind98] P. Indyk. On approximate nearest neighbors in l_∞ norm. *Journal of Computer and System Sciences, to appear. Preliminary version appeared in Proceedings of the Symposium on Foundations of Computer Science*, 1998.

[Ind00] P. Indyk. Dimensionality reduction techniques for proximity problems. *Proceedings of the Ninth ACM-SIAM Symposium on Discrete Algorithms*, 2000.

[Ind02] P. Indyk. Approximate nearest neighbor algorithms for frechet metric via product metrics. *Symposium on Computational Geometry*, 2002.

[Kle97] J. Kleinberg. Two algorithms for nearest-neighbor search in high dimensions. *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, 1997.

[KOR98] E. Kushilevitz, R. Ostrovsky, and Y. Rabani. Efficient search for approximate nearest neighbor in high dimensional spaces. *Proceedings of the Thirtieth ACM Symposium on Theory of Computing*, pages 614–623, 1998.

[Mei93] S. Meiser. Point location in arrangements of hyperplanes. *Information and Computation*, 106:286–303, 1993.

[MS00] S. Muthukrishnan and C. Sahinalp. Approximate sequence nearest neighbors. *Proceedings of the Symposium on Theory of Computing*, 2000.