Dimensionality Reduction for Categorical Data

Debajyoti Bera, Rameshwar Pratap, and Bhisham Dev Verma

Abstract—Categorical attributes are those that can take a discrete set of values, e.g., colours. This work is about compressing vectors over categorical attributes to low-dimension discrete vectors. The current hash-based methods compressing vectors over categorical attributes to low-dimension discrete vectors do not provide any guarantee on the Hamming distances between the compressed representations. Here we present `FSketch` to create sketches for sparse categorical data and an estimator to estimate the pairwise Hamming distances among the uncompressed data only from their sketches. We claim that these sketches can be used in the usual data mining tasks in place of the original data without compromising the quality of the task. For that, we ensure that the sketches also are categorical, sparse, and the Hamming distance estimates are reasonably precise. Both the sketch construction and the Hamming distance estimation algorithms require just a single-pass; furthermore, changes to a data point can be incorporated into its sketch in an efficient manner. The compressibility depends upon how sparse the data is and is independent of the original dimension – making our algorithm attractive for many real-life scenarios. Our claims are backed by rigorous theoretical analysis of the properties of `FSketch` and supplemented by extensive comparative evaluations with related algorithms on some real-world datasets. We show that `FSketch` is significantly faster, and the accuracy obtained by using its sketches are among the top for the standard unsupervised tasks of RMSE, clustering and similarity search.

Index Terms—Dimensionality Reduction, Sketching, Feature Hashing, Clustering, Classification, Similarity Search.

1 Introduction

Of the many types of digital data that are getting recorded every second, most can be ordered – they belong to the ordinal type (e.g., age, citation count, *etc.*), and a good proportion can be represented as strings but cannot be ordered — they belong to the nominal type (e.g., hair colour, country, publication venue, *etc.*). The latter datatype is also known as *categorical* which is our focus in this work. Categorical attributes are commonly present in survey responses, and have been used earlier to model problems in bioinformatics [1], [2], market-basket transactions [3], [4], [5], web-traffic [6], images [7], and recommendation systems [8]. The first challenge practitioners encounter with such data is how to process them using standard tools most of which are designed for numeric data, that too often are real-valued.

Two important operations are often performed before running statistical data analysis tools and machine learning algorithms on such datasets. The first is encoding the data points using numbers, and the second is dimensionality reduction; many approaches combine the two, with the final objective being numeric vectors of fewer dimensions. To the best of our knowledge, the approaches usually followed are ad-hoc adaptations of those employed for vectors in the real space, and suffer from computational inefficiency and/or unproven heuristics [9]. The motivation of this work is to provide a solution that is efficient in practice and has proven theoretical guarantees.

For the first operation, we use the standard method of label encoding in this paper. In this a feature with c categories is represented by an integer from $\{0, 1, 2, \dots c\}$ where 0 indicates a missing category and $i \in \{1, 2, \dots, c\}$ indicates the i-th category. Hence, an n-dimensional data point, where each feature can take at most c values, can be represented by a vector from $\{0, 1, 2 \dots c\}^n$ — we call such a vector as a categorical vector. Another approach is one-hot encoding (OHE) which is more popular since it avoids the implicit ordering among the feature values imposed by label-encoding. One-hot encoding of a feature with c possible values is a c-dimensional binary vector in which the i-th bit is set to 1 to represent the i-th feature value. Naturally, one-hot encoding of an n-dimensional vector will be nc dimensional — which can be very large if c is large (e.g., for features representing countries, *etc.*). Not only label encoding avoids this problem, but is essential for the crucial second step – that of dimensionality reduction.

Dimensionality reduction is important when data points lie in a high-dimensional space, e.g., when encoded using one-hot encoding or when described using tens of thousands of categorical attributes. High-dimensional data vectors not only increase storage and processing cost, but they suffer from the "curse of dimensionality" that points to the decrease in performance after the dimension of the data points crosses a peak. Hence it is suggested that the high-dimensional categorical vectors be compressed to smaller vectors, essentially retaining the information only from the useful features. Baraniuk et al. [10] characterised a good dimensionality reduction *in the Euclidean space* as a compression algorithm that satisfies the following two conditions for any two vectors x and y.

1) Information preserving: For any two distinct vectors x and y, $R(x) \neq R(y)$.

- *D. Bera is with the Department of Computer Science and Engineering, Indraprastha Institute of Information Technology (IIIT-Delhi), New Delhi, India, 110020.*
  E-mail: see http://www.michaelshell.org/contact.html
- *R. Pratap and B. D. Verma are with the Indian Institute of Technology, Mandi, Himachal Pradesh, India.*
- *Emails: dbera@iiitd.ac.in, rameshwar@iitmandi.ac.in and d18039@students.iitmandi.ac.in.*

Manuscript accepted for publication by IEEE Transactions on Knowledge and Data Engineering.

2) ϵ-Stability: (Euclidean) distances between all the points are approximately preserved (with ϵ inaccuracy).

We call these two conditions the *"well-designed"* conditions. To obtain their mathematically precise versions, we need to narrow down upon a distance measure for categorical vectors. A natural measure for categorical vectors is an extension of the binary Hamming distance. For two n-dimensional categorical data points x and y, the Hamming distance between them is defined as the number of features with different attributes in x and y, i.e.,

$$HD(x,y) = \Sigma_{i=1}^{n} \text{dist}(x[i], y[i]), \text{ where}$$
$$\text{dist}(x[i], y[i]) = \begin{cases} 1, & \text{if } x[i] \neq y[i], \\ 0, & \text{otherwise.} \end{cases}$$

Problem statement: The specific problem that we address is how to design a dimensionality reduction algorithm that can compress high-dimensional sparse label-encoded categorical vectors to low-dimensional categorical vectors so that (a) compressions of distinct vectors are distinct, and (b) the Hamming distance between two uncompressed vectors can be efficiently approximated from their compressed forms. These conditions, in turn, guarantee both information-preserving and stability. Furthermore, we would like to take advantage of the sparse nature of many real-world datasets. The most important requirement is the compressed vectors should be categorical as well, specifically not over real numbers and preferably not binary; this is to allow the statistical tests and machine learning tools for categorical datasets, e.g. k-mode, to run on the compressed datasets.

1.1 Challenges in the existing approaches

Dimensionality reduction is a well-studied problem [11] (also see Table 8 in Appendix) but Hamming space does not allow the usual approaches applicable in the Euclidean spaces. Methods that work for continuous-valued data or even ordinal data (such as integers) do not perform satisfactorily for unordered categorical data. Among those that specifically consume categorical data, techniques *via* feature selection have been well studied. For example, in the case of labelled data χ^2 [12] and Mutual Information [13] based methods select features based on their correlation with the label. This limits their applicability to only the classification tasks. Further, Kendall rank correlation coefficient [14] "learns" the important features based on the correlation among them. Learning approaches tend to be computationally heavy and do not work reliably with small training samples. So what about task-agnostic approaches that do not involve learning? PCA-based methods, e.g., MCA is popular among the practitioners of biology [11]; however, we consider them merely a better-than-nothing approach since PCA is fundamentally designed for continuous data.

A quick search among internet forums, tutorials and Q&A websites revealed that the more favourable approach to perform machine learning tasks on categorical datasets is to convert categorical feature vectors to binary vectors using one-hot encoding [15, see DictVectorizer] — a widely-viewed tutorial on Kaggle calls it "The Standard Approach for Categorical Data" [16]. The biggest problem with OHE

$$\begin{array}{rl} & \left.\begin{array}{l} \texttt{u} = 220 \longrightarrow 10 \cdot 10 \cdot 00 \\ \texttt{v} = 202 \longrightarrow 10 \cdot 00 \cdot 10 \end{array}\right\} \text{Hamming=2} \\ \text{Hamming=1} & \left\{\begin{array}{l} \texttt{w} = 201 \longrightarrow 10 \cdot 00 \cdot 01 \end{array}\right\} \text{Hamming=2} \end{array}$$

Fig. 1. An example showing that the Hamming distances of one-hot encoded sparse vectors are not functionally related to the distances between their unencoded forms. If a feature, say country, is missing, libraries differ in their handling of its one-hot encoding. In this paper, we follow the common practice of using the c-dimensional all-zero vector as its encoding. This retains sparsity since the number of non-missing attributes in the original vector equals the number of non-zero bits in the encoded vector.

is that it is impractical for large n or large c followed by a technical annoyance that some OHE implementations do not preserve the Hamming distances for sparse vectors (see illustration in Figure 1). Hence, this encoding is used in conjunction with problem-specific feature selection or followed by dimensionality reduction from binary to binary vectors [17], [18], [19]. The latter is a viable heuristic that we wanted to improve upon by allowing non-binary compressed vectors (see Appendix A for a quick analysis of OHE followed by a state-of-the-art binary compression).

Another popular alternative, especially when $n \times c$ is large, is *feature hashing* [20] that is now part of most libraries, e.g., `scikit-learn` [15, see FeatureHasher]. Feature hashing and other forms of hash-based approaches, also known as sketching algorithms, both encode and compress categorical feature vectors into integer vectors (sometimes signed) of a lower dimension, and furthermore, provide theoretical guarantees like stability, in some metric space. The currently known results for feature hashing apply only to the Euclidean space, however, Euclidean distance and Hamming distance are not monotonic for categorical vectors. It is neither known nor straightforward to ascertain whether feature hashing and its derivatives can be extended to the Hamming space which lacks the continuity that is crucial to their theoretical bounds. Other hash-based approaches either come with no guarantees and are used merely because of their compressibility or come with stability-like guarantees in a different space, e.g., cosine similarity by Simhash [21]. Our solution is a hashing approach that we prove to be stable in the Hamming space.

1.2 Overview of results

The commonly followed practices in dealing with categorical vectors, especially those with high dimensions and not involving supervised learning or training data, appear to be either feature hashing or one-hot encoding followed by dimensionality reduction of binary vectors [22, Chapter 5]. We provide a contender to these in the form of the `FSketch` *sketching algorithm* to construct lower-dimensional categorical vectors from high-dimensional ones.

The lower-dimensional vectors, *sketches*, produced by `FSketch` (we shall call these vectors as `FSketch` too) have the desired theoretical guarantees and perform well on real-world datasets vis-à-vis related algorithms. Now we summarise the important features of `FSketch`; in the summarisation, p is a constant that is typically chosen to be a prime number between 5-50.

Lightweight and unsupervised: First and foremost, `FSketch` is an unsupervised process, and in fact, quite

lightweight making a single pass over an input vector and taking $O(poly(\log p))$ steps per non-missing feature. The FSketch-es retain the sparsity of the input vectors and their size and dimension do not depend at all on c. To make our sketches applicable out-of-the-box for modern applications where data keeps changing, we present an extremely lightweight algorithm to incorporate any change in a feature vector into its sketch in $O(poly(\log p))$-steps per modified feature. It should be noted that FSketch supports change of an attribute, deletion of an attribute and insertion of a previously missing attribute unlike some state-of-the-art sketches; for example, BinSketch [17] does not support deletion of an attribute.

Estimator for Hamming distance: We want to advocate the use of FSketch-es for data analytic tasks like clustering, *etc.* that use Hamming distance for the (dis)similarity metric. We present an *estimator that can approximate the Hamming distance* between two points by making a single pass over their sketches. The estimator follows a tight concentration bound and has the ability to estimate the Hamming distance from very low-dimensional sketches. In the theoretical bounds, the dimensions could go as low as 4σ or even $\sqrt{\sigma}$ (and independent of the dimension of the data) where σ indicates the sparsity (maximum number of non-zero attributes) of the input vectors; however, we later show that a much smaller dimension suffices in practice. Our sketch generation and the Hamming distance estimation algorithms combined meet the two conditions of "well-designed" dimensionality reduction.

Theorem 1. *Let x and y be distinct categorical vectors, and $\phi(x)$ and $\phi(y)$ be their d-dimensional compressions.*

1) *$\phi(x)$ and $\phi(y)$ are distinct with probability $\approx HD(x,y)/d$.*
2) *Let $HD'(x,y)$ denote the approximation to the Hamming distance between x and y computed from $\phi(x)$ and $\phi(y)$. If d is set to 4σ, then with probability at least $1-\delta$ (for any δ of choice),*

$$\left|HD(x,y) - HD'(x,y)\right| = O\left(\sqrt{\sigma \ln \tfrac{2}{\delta}}\right).$$

The proof of (1) follows from Lemma 3 and the proof of (2) follows from Lemma 8 for which we used McDiarmid's inequality. The theorem allows us to use compressed forms of the vectors in place of their original forms for data analytic and statistical tools that depend largely on their pairwise Hamming distances.

Practical performance: All of the above claims are proved rigorously but one may wonder how do they perform in practice. For this, we design an elaborate array of experiments on real-life datasets involving many common approaches for categorical vectors. The experiments demonstrate these facts.

- Some of the baselines do not output categorical vectors (see Section 4). Our FSketch algorithm is super-fast among those that do and offer comparable accuracy.
- When used for typical data analytic tasks like clustering, similarity search, etc. low-dimension FSketch-es bring immense speedup *vis-a-vis* using the original (uncompressed) vectors, yet achieving very high accuracy. The NYTimes dataset saw 140x speedup upon compression to 0.1%.
- Even though highly compressed, the results of clustering, *etc.* on FSketch-es are close to what could be obtained from the uncompressed vectors and are comparable with the best alternatives. For example, we were able to compress the Brain cell dataset of dimensionality 1306127 to 1000 dimensions in a few seconds, yet retaining the ability to correctly approximating the pairwise Hamming distances from the compressed vectors. This is despite many other baselines giving either an out-of-memory error, not stopping even after running for a sufficiently long time, or producing significantly worse estimates of pairwise Hamming distances.
- The parameter p can be used to fine-tune the quality of results and the storage of the sketches.

We claim that FSketch is the best method today to compress categorical datasets for data analytic tasks that require pairwise Hamming distances with respect to both theoretical guarantee and practical performance.

1.3 Organisation of the paper

The rest of the paper is organised as follows. We discuss several related works in Section 2. In Section 3, we present our algorithm FSketch and derive its theoretical bounds. In Section 4, we empirically compare the performance of FSketch on several end tasks with state-of-the-art algorithms. We conclude our presentation in Section 5. The proofs of the theoretical claims and the results of additional experiments are included in Appendix.

2 Related work

Dimensionality reduction: Dimensionality reduction has been studied in-depth for real-valued vectors, and to some extent, also for discrete vectors. We categorise them into these broad categories — (a) random projection, (b) spectral projection, (c) locality sensitive hashing (LSH), (d) other hashing approaches, and (e) learning-based algorithms. All of them compress high-dimensional input vectors to low-dimensional ones that explicitly or implicitly preserve some measure of similarity between the input vectors.

The seminal result by Johnson and Lindenstrauss [23] is probably the most well known random projection-based algorithm for dimensionality reduction. This algorithm compresses real-valued vectors to low-dimensional real-valued vectors such that the Euclidean distances between the pairs of vectors are approximately preserved, but in such a manner that the compressed dimension does not depend upon the original dimension. The algorithm involves projecting a data matrix onto a random matrix whose each entry is sampled from a Gaussian distribution. This result has seen lots of enhancements, particularly with respect to generating the random matrix without affecting the accuracy [24], [25], [26]. However, it is not clear whether any of those ideas can be made to work for categorical data and that too, for approximating Hamming distances.

Principal component analysis (PCA) is a spectral projection-based technique for reducing the dimensionality of high dimensional datasets by creating new uncorrelated variables that successively maximise variance. There are extensions of PCA that employ kernel methods that try to

capture non-linear relationships [27]. Multiple Correspondence Analysis (MCA) [28] does the analogous job for the categorical datasets. However, these methods perform dimensionality reduction by creating un-correlated features in a low-dimensional space whereas our aim is to preserve the pairwise Hamming distances in a low-dimensional space.

Another line of dimensionality reduction techniques builds upon the "Locality Sensitive Hashing (LSH)" algorithms. LSH algorithms have been proposed for different data types and similarity measures, e.g., real-valued vectors and the Euclidean distance [29], real-valued vectors and the cosine similarity [21], binary vectors and the Jaccard similarity [30], binary vectors and the Hamming distance [31]. However, generally speaking, the objective of an LSH is to group items so that similar items are grouped together and dissimilar items are not; unlike `FSketch` they do not provide explicit estimators of any similarity metric.

There are quite a few learning-based dimensionality reduction algorithms available such as Latent Semantic Analysis (LSA) [32], Latent Dirichlet Allocation (LDA) [33], Non-negative Matrix Factorisation (NNMF) [34], Generalized feature embedding learning (GEL) [35] all of which strive to learn a low-dimensional representation of a dataset while preserving some inherent properties of the full-dimensional dataset. They are rather slow due to the optimization step involved during learning. T-distributed Stochastic Neighbour Embedding `(t-SNE)` [36] is a faster non-linear dimensionality reduction technique that is widely used for the visualisation of high-dimensional datasets. However, the low-dimensional representation obtained from `t-SNE` is not recommended for use for other end tasks such as clustering, classification, anomaly detection as it does not necessarily preserve densities or pairwise distances. An autoencoder [37] is another learning-based non-linear dimension reduction algorithm. It basically consists of two parts: An *encoder* which aims to learn a low-dimensional representation of the input and a *decoder* which tries to reconstruct the original input from the output of the encoder. However, these approaches involve optimising a learning objective function and are usually slow and CPU-intensive.

The other hashing approaches randomly assign each feature (dimension) to one of several bins, and then compute a summary value for each bin by aggregating all the feature values assigned to it. A list of such summaries can be viewed as a low-dimensional sketch of the input. Such techniques have been designed for real-valued vectors approximating inner product (e.g., feature hashing [20]), binary vectors allowing estimation of several similarity measures such as Hamming distance, Inner product, Cosine, and Jaccard similarity (e.g., BinSketch [17]), etc. This work is similar to these approaches but for categorical vectors and only aiming to estimate the Hamming distances.

Another approach in this direction could be to encode categorical vectors to binary and then apply dimensionality reduction for binary vectors; unfortunately, the popular encodings, e.g. OHE, do not preserve Hamming distance for vectors with missing features. Nevertheless, it is possible to encode using OHE and then reduce its dimension. However, our theoretical analysis led to a worse accuracy compared to that of `FSketch` (see Appendix A for the analysis) and this approach turned out to be one of the worst performers in our experiments (see Section 4).

While our motivation was to design an end-task agnostic dimensionality reduction algorithm, there exist several that are designed for specific tasks, e.g., for clustering [38], for regression and discriminant analysis of labelled data [39], and for estimating covariance matrix [40]. Deep learning has gained mainstream importance and several researchers have proposed a dimensionality reduction "layer" inside a neural network [41]; this layer is intricately interwoven with the other layers and cannot be separated out as a standalone technique that outputs compressed vectors.

Feature selection is a limited form of dimensionality reduction whose task is to identify a set of good features, and maybe learn their relative importance too. Banerjee and Pal [42] recently proposed an unsupervised technique that identifies redundant features and selects those with bounded correlation, but only for real-valued vectors. For our experiments we chose the Kendall-Tau rank correlation approach that is applicable to discrete-valued vectors.

Sketching algorithm: The use of "sketches" for computing Hamming distance has been explicitly studied in the streaming algorithm framework. The first well-known solution was proposed by Cormode et al. [43] where they showed how to estimate a Hamming distance with high accuracy and low error. There have been several improvements to this result, in particular, by Kane et al. [44] where a sketch with the optimal size was proposed. However, we neither found any implementation nor an empirical evaluation of those approaches (the algorithms themselves appear fairly involved). Further, their objective was to minimise the space usage in the asymptotic sense in a streaming setting, whereas, our objective is to design a solution that can be readily used for data analysis. This motivated us to compress categorical vectors onto low-dimensional categorical vectors, unlike the real-valued vectors that the theoretical results proposed. A downside of our solution is that it heavily relies on the sparsity of a dataset unlike the sketches output by the streaming algorithms.

TABLE 1
Notations

categorical data vectors	x, y
their Hamming distance	h
compressed categorical vectors (sketches)	$\phi(x), \phi(y)$
j-th bit of a sketch $\phi(x)$	$\phi_j(x)$
observed Hamming distance between sketches	f
expected Hamming distance between sketches	f^*
estimated Hamming distance between data vectors	$\hat{h}$

3 Category sketching and Hamming distance estimation

Our technical objective is to design an effective algorithm to compress high-dimensional vectors over $\{0, 1, \ldots, c\}$ to integer vectors of a low dimension, *aka.* sketches; c can even be set to an upper bound on the largest number of categories among all the features. The number of attributes in the input vectors is denoted n and the dimension of the compressed vector is denoted d. We will later show how to choose d depending on the sparsity of a dataset that we denote σ.

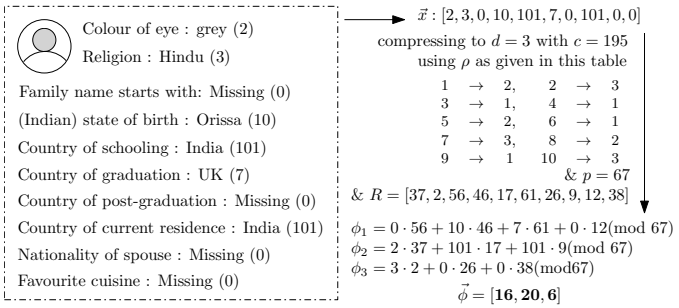


Fig. 2. An example illustrating how to compress a data point with categorical features using `FSketch` to a 3-dimensional integer vector. The data point has 10 feature values, each of which is a categorical variable (the corresponding label encoded values are present inside the brackets). c is chosen as 195 since the fifth, sixth, seventh, and eighth features have 195 categories which is the largest. ρ, p and R are internal variables of `FSketch`.

The commonly used notations in this section are listed in Table 1.

Algorithm 1 Constructing d-dimensional `FSketch` of n-dimensional vector x

procedure INITIALIZE
  Choose random mapping $\rho : \{1, \dots n\} \to \{1, \dots d\}$
  Choose some prime p
  Choose n random numbers $R = r_1, \dots, r_n$ with each $r_i \in \{0, \dots p-1\}$
end procedure
procedure CREATESKETCH($x \in \{0, 1, \dots c\}^n$)
  Create empty sketch $\phi(x) = 0^d$
  for $i = 1 \dots n$ **do**
    $j = \rho(i)$
    $\phi_j(x) = (\phi_j(x) + x_i \cdot r_i) \mod p$
  end for
  return $\phi(x)$
end procedure

3.1 `FSketch` construction

Our primary tool for sketching categorical data is a randomised sketching algorithm named `FSketch` that is described in Algorithm 1; see Figure 2 for an example.

Let $x \in \{0, 1, \dots c\}^n$ denote the input vector, and the i-th feature or co-ordinate of x is denoted by x_i. The sketch of input vector x will be denoted $\phi(x) \in \{0, 1, \dots p-1\}^d$ whose coordinates will be denoted $\phi_1(x), \phi_2(x), \dots, \phi_d(x)$. Note that the initialisation step of `FSketch` needs to run only once for a dataset. We are going to use the following characterisation of the sketches in the rest of this section; a careful reader may observe the similarity to Freivald's algorithm for verifying matrix multiplication [45].

Observation 2. *It is obvious from Algorithm 1 that the sketches created by* `FSketch` *satisfy* $\phi_j(x) = (\sum_{i \in \rho^{-1}(j)} x_i \cdot r_i) \mod p$.

3.2 Hamming distance estimation

Here we explain how the Hamming distance between x and y denoted $HD(x, y)$, percolates to their sketches as well. The objective is derive an estimator for $HD(x, y)$ from the Hamming distance between $\phi(x)$ and $\phi(y)$.

The sparsity of a set of vectors denoted σ, is the maximum number of non-zero coordinates in them. For the theoretical analysis, we assume that we know the sparsity of the dataset, or at least an upper bound of the same. Note that, for a pair of sparse vectors $x, y \in \{0, 1, \dots, c\}^n$, the Hamming distance between them can vary from 0 (when they are same) to 2σ (when they are completely different).

We first prove case (a) of Theorem 1 which states that sketches of different vectors are rarely the same.

Lemma 3. *Let h denote $HD(x, y)$ for two input vectors x, y to* `FSketch`*. Then*

$$\Pr_{\rho,R}[\phi_j(x) \neq \phi_j(y)] = (1 - \tfrac{1}{p})(1 - (1 - \tfrac{1}{d})^h).$$

Proof. Fix a mapping ρ and then define $F_j(x)$ as the vector $[x_{i_1}, x_{i_2}, \dots \ : \ i_k \in \{1, \dots n\}]$ of values of x that are mapped to j in $\phi(x)$ in the increasing order of their coordinates, i.e., $\rho(i_k) = j$ and $i_1 < \dots i_k < i_{k+1}$. Since ρ is fixed, $F_j(y)$ is also a vector of the same length. The key observation is that if $F_j(x) = F_j(y)$ then $\phi_j(x) = \phi_j(y)$ but the converse is not always true. Therefore we separately analyse both the conditions (a) $F_j(x) \neq F_j(y)$ and (b) $F_j(x) = F_j(y)$.

It is given that x and y differ at h coordinates. Therefore, $F_j(x) \neq F_j(y)$ iff any of those coordinates are mapped to j by ρ. Thus,

$$\Pr_{\rho}[F_j(x) = F_j(y)] = (1 - \tfrac{1}{d})^h. \tag{1}$$

Next we analyse the chance of $\phi_j(x) = \phi_j(y)$ when $F_j(x) \neq F_j(y)$. Note that $\phi_j(x) = (x_{i_1} \cdot r_{i_1} + x_{i_2} \cdot r_{i_2} + \dots) \mod p$ (and a similar expression exists for y), where r_is are randomly chosen during initialisation (they are fixed for x and y). Using a similar analysis as that in the Freivald's algorithm [46, Ch 1(Verifying matrix multiplication)],

$$\Pr_{\rho,R}[\phi_j(x) = \phi_j(y) \mid F_j(x) \neq F_j(y)] = \tfrac{1}{p}. \tag{2}$$

Due to Equations 1, 2, we have

$$\begin{aligned}
&\Pr_{\rho,R}[\phi_j(x) \neq \phi_j(y)] \\
&= \Pr_{\rho,R}[\phi_j(x) \neq \phi_j(y) \mid F_j(x) \neq F_j(y)] \cdot \Pr_{\rho,R}[F_j(x) \neq F_j(y)] \\
&+ \Pr_{\rho,R}[\phi_j(x) \neq \phi_j(y) \mid F_j(x) = F_j(y)] \cdot \Pr_{\rho,R}[F_j(x) = F_j(y)] \\
&= (1 - \tfrac{1}{p})(1 - (1 - \tfrac{1}{d})^h).
\end{aligned}$$

□

The right-hand side of the expression in the statement of the lemma can be approximated as $(1 - \frac{1}{p})\frac{h}{d}$ which is stated as case (a) of Theorem 1. The lemma also allows us to relate the Hamming distance of the sketches to the Hamming distance of the vectors which is our main tool to define an estimator.

Lemma 4. *Let h denote $HD(x, y)$ for two input vectors x, y to* `FSketch`*, f denote $HD(\phi(x), \phi(y))$ and f^* denote $\mathbb{E}[HD(\phi(x), \phi(y))]$. Then*

$$f^* = \mathbb{E}[f] = d\left(1 - \tfrac{1}{p}\right)\left(1 - \left(1 - \tfrac{1}{d}\right)^h\right).$$

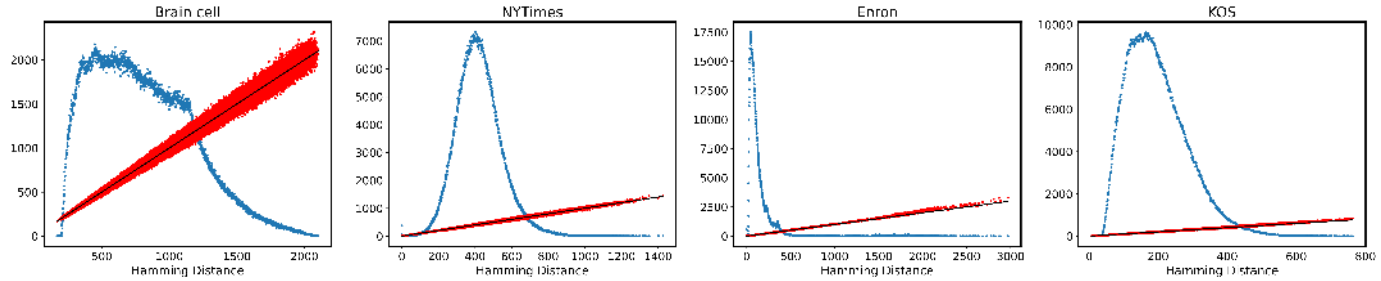


Fig. 3. The distributions of Hamming distances for some of the datasets used in our experiments are shown in blue — the Y-axis shows the frequency of each distance. The black points represent the actual Hamming distances and the red points are the estimates, i.e., a red-point plotted against a Hamming distance d (on the X-axis) shows the estimated Hamming distance between two points with actual Hamming distance d. Observe that the Hamming distances follow a long-tailed distribution and that most distances are fairly low — moreover, our estimates are more accurate for those high frequent Hamming distances.

The lemma is easily proved using Lemma 3 by applying the linearity of expectation on the number of coordinates j such that $\phi_j(x) \neq \phi_j(y)$. We are now ready to define an estimator for the Hamming distance.

Using $D = \left(1 - \frac{1}{d}\right)$ and $P = \left(1 - \frac{1}{p}\right)$, we can write

$$f^* = dP(1 - D^h) \text{ and } h = \ln\left(1 - \frac{f^*}{dP}\right) / \ln D. \quad (3)$$

Our proposal to estimate h is to obtain a tight approximation of f^* and then use the above expression.

Definition 5 (Estimator of Hamming distance)**.** *Given sketches $\phi(x)$ and $\phi(y)$ of data points x and y, suppose f represents $HD(\phi(x), \phi(y))$. We define the estimator of $HD(x, y)$ as $\hat{h} = \ln\left(1 - \frac{f}{dP}\right) / \ln D$ if $f < dP$ and 2σ otherwise.*

Observe that $\hat{h}$ is set to 2σ if $f \geq dP$. However, we shall show in the next section that this occurs very rarely.

3.3 Analysis of Estimator

$\hat{h}$ is pretty reliable when the actual Hamming distance is 0; in that case $\phi(x) = \phi(y)$ and thus, $f = 0$ and so is $\hat{h}$. However, in general, $\hat{h}$ could be different from h. The main result of this section is that their difference can be upper bounded when we set the dimension of `FSketch` to $d = 4\sigma$.

The results of this subsection rely on the following lemma that proves that an observed value of f is concentrated around its expected value f^*.

Lemma 6. *Let α denote a desired additive accuracy. Then, for any x, y with sparsity σ,*

$$\Pr\left[|f - f^*| \geq \alpha\right] \leq 2\exp\left(-\frac{\alpha^2}{4\sigma}\right).$$

The proof of the lemma employs martingales and McDiarmid's inequality and is available in Appendix B. The lemma allows us to upper bound the probability of $f \geq dP$.

Lemma 7. $\Pr[f \geq dP] \leq 2\exp(-P^2\sigma)$.

The right-hand side is a very small number, e.g., it is of the order of 10^{-278} for $p = 5$ and $\sigma = 1000$. The proof is a straightforward application of Lemma 6 and is explained in Appendix B. Now we are ready to show that the estimator $\hat{h}$, which uses f instead of f^* (refer to Equation 3) is almost equal to the actual Hamming distance.

Lemma 8. *Choose $d = 4\sigma$ as the dimension of `FSketch` and choose a prime p and an error parameter $\delta \in (0, 1)$ (ensure that $1 - \frac{1}{p} \geq \frac{4}{\sqrt{\sigma}}\sqrt{\ln\frac{2}{\delta}}$ — see the proof for discussion). Then the estimator defined in Definition 5 is close to the Hamming distance between x and y with high probability, i.e.,*

$$\Pr\left[|\hat{h} - h| \geq \frac{32}{1-1/p}\sqrt{\sigma \ln\frac{2}{\delta}}\right] \leq \delta.$$

If the data vectors are not too dissimilar which is somewhat evident from Figure 3, then a better compression is possible which is stated in the next lemma. The proofs of both these lemmas are fairly algebraic and use standard inequalities; they are included in Appendix B.

Lemma 9. *Suppose we know that $h \leq \sqrt{\sigma}$ and choose $d = 16\sqrt{\sigma \ln\frac{2}{\delta}}$ as the dimension for `FSketch`. Then (a) also $f < dP$ with high probability and moreover we get a better estimator. That is, (b) $\Pr\left[|\hat{h} - h|\right] \geq \frac{8}{1-1/p}\sqrt{\sigma \ln\frac{2}{\delta}}\right] \leq \delta$.*

The last two results prove case (b) of Theorem 1 which states that the estimated Hamming distances are almost always fairly close to the actual Hamming distances. We want to emphasise that the above claims on d and accuracy are only theoretical bounds obtained by worst-case analysis. We show in our empirical evaluations that an even smaller d leads to better accuracy in practice for real-life instances.

There is a way to improve the accuracy even further by generating multiple `FSketch` using several independently generated internal variables and combining the estimates obtained from each. We observed that the median of the estimates can serve as a good statistic, both theoretically and empirically. We discuss this in detail in Appendix H.

3.4 Complexity analysis

The results in the previous section show that the accuracy of the estimator $\hat{h}$ can be tightened, or a smaller probability of error can be achieved, by choosing large values of p which has a downside of a larger storage requirement. In this section, we discuss these dependencies and other factors that affect the complexity of our proposal.

The USP of `FSketch` is its efficiency. There are two major operations with respect to `FSketch`— construction of sketches and estimation of Hamming distance from two sketches. Their time and space requirements are given in the following table and explained in detail in Appendix C.

TABLE 2
Space savings offered by FSketch on an example scenario with 2^{20} data points, each of 2^{10} dimensions but having only 2^7 non-zero entries where non-zero entry belongs to one of 2^3 categories. FSketch dimension is 2^9 (as prescribed theoretically) and its parameter p is close to 2^5. (*) The data required to construct the sketches is no longer required after the construction.

Uncompressed		Compressed	
Naive	Sparse vector format	FSketch construction (*)	Storage of sketches
$2^{20} \times 2^{10} \times 3$	$2^{20} \times 2^7 \times (\log 2^3 + \log 2^{10})$	$2^{10} \times (\log 2^9 + \log 2^5) + 5$	$2^{20} \times \log(2^5)$

Construction		Estimation	
time per sketch	$O(n)$	time per pair	$O(d \log p)$
space per sketch	$O(d \log p)$		

We are aware of efficient representations of sparse data vectors, but for the sake of simplicity we assume full-size arrays to store vectors in this table; similarly, we assume simple dictionaries for storing the internal variables ρ, R and p. While it may be possible to reduce the number of random bits by employing k-wise independent bits and mappings, we left it out of the scope of this work.

Both the operations are quite fast compared to the matrix-based and learning-based methods. There is very little space overhead too; we explain the space requirement with the help of an example in Table 2 — one should keep in mind that a sparse representation of a vector has to store the non-zero entries as well as their positions in it.

Apart from the efficiency in both time and space measures, FSketch provides additional benefits. Recall that each entry of an FSketch is an integral value from 0 to $p-1$. Even though 0 does not necessarily indicate a missing feature in a compressed vector, we show below that 0 has a predominant presence in the sketches. The sketches can therefore be treated as sparse vectors that further facilitates their efficient storage.

Lemma 10. *If $d = 4\sigma$ (as required by Lemma 8), then the expected number of non-zero entries of $\phi(x)$ is upper bounded by $\frac{d}{4}$. Further, at least 50% of $\phi(x)$ will be zero with probability at least $\frac{1}{2}$.*

The lemma can be proved using a balls-and-bins type analysis (see Appendix D for the entire proof).

3.5 Sketch updating

Imagine a situation where the categories of attributes can change dynamically, and they can both "increase", "decrease" or even "vanish". We present Algorithm 2 to incorporate such changes *without recomputing the sketch afresh.* The algorithm simply uses the formula for a sketch entry as given in Observation 2.

Most hashing-based sketching and dimensionality reduction algorithms that we have encountered either require complete regeneration of $\phi(x)$ when some attributes of x change or are able to handle addition of previously missing attributes but not their removal.

Algorithm 2 Update sketch $\sigma(x)$ of x after i-th attribute of x changes from v to v'

input: data vector x and its existing sketch $\phi(x) = \langle \phi_1(x), \phi_2(x), \ldots \phi_d(x) \rangle$
input: change $x_i : v \mapsto v'$ ▷ v' can be any value in $\{0, 1, \ldots, c\}$
parameters: $\rho, R = [r_1 \ldots r_n], p$ (same as that was used for generating the sketch)
1: $j = \rho(i)$
2: update $\phi_(x) = \big(\phi_j(x) + (v' - v) \cdot r_i\big) \mod p$
3: **return** updated $\phi(x)$

4 EXPERIMENTS

We performed our experiments on a machine having Intel(R) Xeon(R) CPU E5-2650 v3 @ 2.30GHz, 94 GB RAM, and running a Ubuntu 64-bits OS.

We first study the effect of the internal parameters of our proposed solution on its performance. We start with the effect of the prime number p; then we compare FSketch with the appropriate baselines for several unsupervised data-analytic tasks (see Table 5) and objectively establish these advantages of FSketch over the others.

(a) Significant speed-up in the dimensionality reduction time,
(b) considerable savings in the time for the end-tasks (e.g., clustering) which now runs on the low-dimensional sketches,
(c) but with comparable accuracy of the end-tasks (e.g., clustering).

Several baselines threw *out-of-memory* errors or did not stop on certain datasets. We discuss the errors separately in Section F in Appendix.

4.1 Dataset description

The efficacy of our solution is best described for high-dimensional datasets. Publicly available categorical datasets being mostly low-dimensional, we treated several integer-valued freely available real-world datasets as categorical. Our empirical evaluation was done on the following seven such datasets with dimensions between 5000 and 1.3 million, and sparsity from 0.07% to 30%.

- Gisette Data Set [47], [48]: This dataset consists of integer feature vectors corresponding to images of handwritten digits and was constructed from the MNIST data. Each image, of 28×28 pixels, has been pre-processed (to retain the pixels necessary to disambiguate the digit 4 from 9) and then projected onto a higher-dimensional feature space represented to construct a 5000-dimension integer vector.
- BoW (Bag-of-words) [47], [49]: We consider the following five corpus – NIPS full papers, KOS blog entries, Enron Emails, NYTimes news articles, and tagged web pages from the social bookmarking site delicious.com. These datasets are "BoW"(Bag-of-words) representations of the corresponding text cor-

TABLE 3
Datasets

Datasets	Categories	Dimension	Sparsity	No. of points
Gisette [47], [48]	999	5000	1480	13500
Enron Emails [47]	150	28102	2021	39861
DeliciousMIL [47], [49]	58	8519	200	12234
NYTimes articles [47]	114	102660	871	10000
NIPS full papers [47]	132	12419	914	1500
KOS blog entries [47]	42	6906	457	3430
Million Brain Cells from E18 Mice [50]	2036	1306127	1051	2000

TABLE 4
13 baselines

1.	SSD	Sketching via Stable Distribution [51]
2.	OHE	One Hot Encoding+BinSketch [17]
3.	FH	Feature Hashing [20]
4.	SH	Signed-random projection/SimHash [21]
5.	KT	Kendall rank correlation coefficient [14]
6.	LSA	Latent Semantic Analysis [32]
7.	LDA	Latent Dirichlet Allocation [33]
8.	MCA	Multiple Correspondence Analysis [28]
9.	NNMF	Non-neg. Matrix Factorization [34]
10.	PCA	Vanilla Principal component analysis
11.	VAE	Variational autoencoder [52]
12.	CATPCA	Categorical PCA [53]
13.	HCA	Hierarchical Cluster Analysis [53]

pora. In all these datasets, the attribute takes integer values which we consider as categories.

- `1.3 Million Brain Cell Dataset` [50]: This dataset contains the result of a single cell RNA-sequencing (`scRNA-seq`) of 1.3 million cells captured and sequenced from an `E18.5` mouse brain [1]. Each gene represents a data point and for every gene, the dataset stores the read-count of that gene corresponding to each cell – these read-counts form our features.

We chose the last dataset due to its very high dimension and the earlier ones due to their popularity in dimensionality-reduction experiments. We consider all the data points for KOS, Enron, Gisette, DeliciousMIL, a $10,000$ sized sample for NYTimes, and a 2000 sized samples for BrainCell. We summarise the dimensionality, the number of categories, and the sparsity of these datasets in the Table 3.

4.2 Baselines

Recall that `FSketch` (hence `Median-FSketch`) *compresses categorical vectors to shorter categorical vectors in an unsupervised manner that "preserves" Hamming distances*.

Our first baseline is based on one-hot-encoding (OHE) which is one of the most common methods to convert categorical data to a numeric vector and can approximate pairwise Hamming distance (refer to Appendix A). Since OHE actually increases the dimension to very high levels (e.g., the dimension of the binary vectors obtained by encoding the NYTimes dataset is $11,703,240$), the best way to use it is by further compressing the one-hot encoded vectors. For empirical evaluation we applied BinSketch [17] which is the state-of-the-art binary-to-binary dimensionality reduction technique that preserves Hamming distance. We refer to the entire process of OHE followed by BinSketch simply by OHE in the rest of this section.

To the best of our knowledge, there is no sketching algorithm other than OHE that compresses high-dimensional categorical vectors to low-dimensional categorical (or integer) vectors that preserves the original pairwise Hamming distances. Hence, we chose as baseline state-of-the-art and popularly employed algorithms that either preserve Hamming distance or output discrete-valued sketches (preserving some other similarity measure). We list them in Table 4 and tabulate their characteristic in Table 5. Their implementation details are discussed in Appendix E.1.

We include Kendall rank correlation coefficient (KT) [14] – a feature selection algorithm which generates discrete valued sketches. Note that if we apply Feature Hashing (FH), SimHash (SH), and KT naively on categorical datasets, we get discrete valued sketches on which Hamming distance can be computed.We also include a few other well known dimensionality reduction methods such as Principal component analysis (PCA), Non-negative Matrix Factorisation (NNMF) [34], Latent Dirichlet Allocation (LDA) [33], Latent Semantic Analysis (LSA) [32], Variational Autoencoder (VAE) [52], Categorical PCA (CATPCA) [53], Hierarchical Cluster Analysis (HCA) [53] all of which output real-valued sketches.

4.3 Choice of p

We discussed in Section 3 that a larger value of p (a prime number) leads to a tighter estimation of Hamming distance but degrades sketch sparsity, which negatively affects performance at multiple fronts, and moreover, demands more space to store a sketch. We conducted an experiment to study this trade-off, where we ran our proposal with different values of p, and computed the corresponding RMSE values. The RMSE is defined as the square-root of the average error, among all pairs of data points, between their actual Hamming distances and the corresponding estimate obtained via `FSketch`. Note that a lower RMSE indicates that the sketch correctly estimates the underlying pairwise Hamming distance. We also note the corresponding *space overhead* which is defined as the ratio of the space used by uncompressed vector and its sketch obtained from `FSketch`. We consider storing a data point in a typical sparse vector format – a list of non-zero entries and their positions (see Table 2). We summarise our results in Figures 4 and 5, respectively. We observe that a large value of p leads to a lower RMSE (in Figure 4), however simultaneously it leads to a smaller space compression (Figure 5). As a heuristic, we decided to set p as the next prime after c as shown in this table.

Brain cell	2039	NYTimes	127	Enron	151
KOS	43	Delicious	59	Gisette	1009
NIPS	137				

That said, the experiments reveal that, at least for the datasets in the above experiments, setting p to be at least $c/4$ may be practically sufficient, since there does not appear to be much advantage in using a larger p.

4.4 Variance of `FSketch`

In Section 3.3 we explained that the bias of our estimator is upper bounded with a high likelihood. However, there re-

1. https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.3.0/1M_neurons

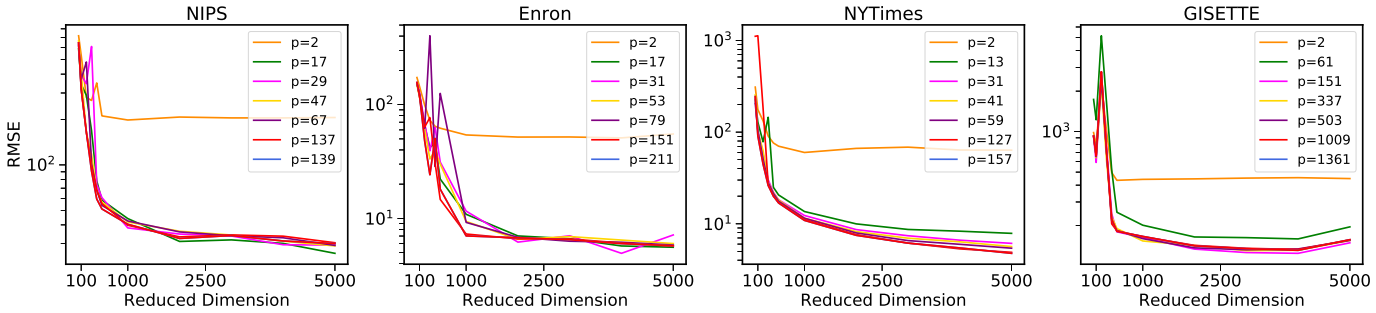


Fig. 4. Comparison of RMSE measure obtained from `FSketch` algorithm on various choices of p. Values of c for NIPS, Enron, NYTimes, and GISETTE are 132, 150, 114, and 999, respectively.

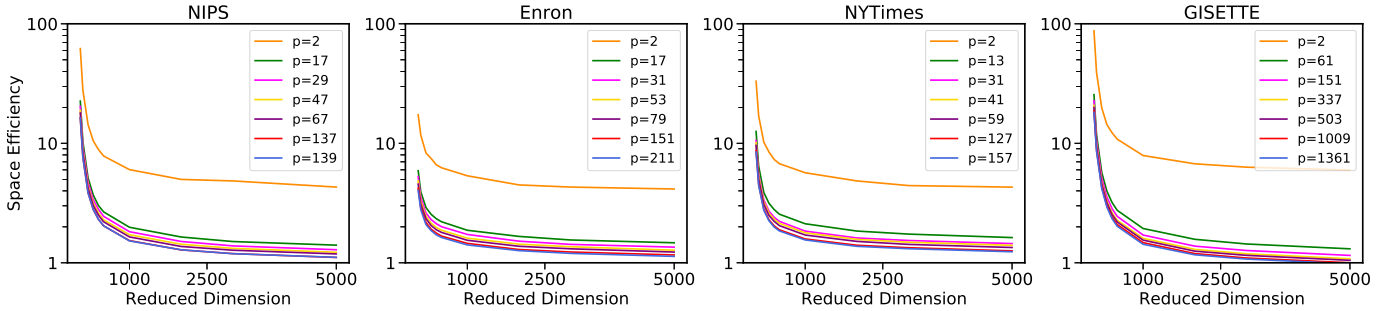


Fig. 5. Space overhead of uncompressed vectors stored as a list of non-zero entries and their positions. Y-axis represents the ratio of the space used by uncompressed vector to that obtained from `FSketch`.

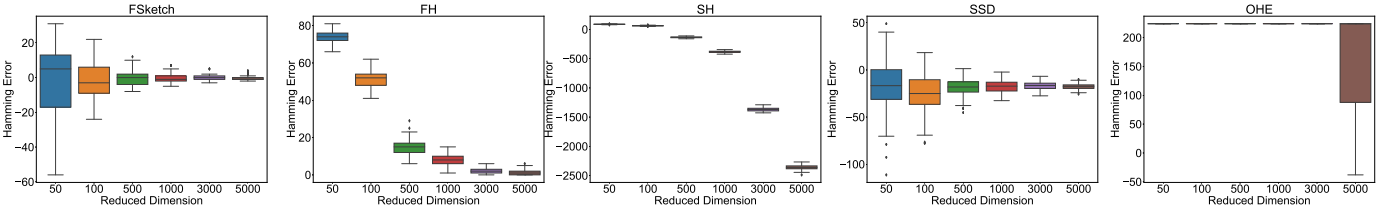


Fig. 6. Comparison of avg. error in estimating Hamming distance of a pair of points from the Enron dataset.

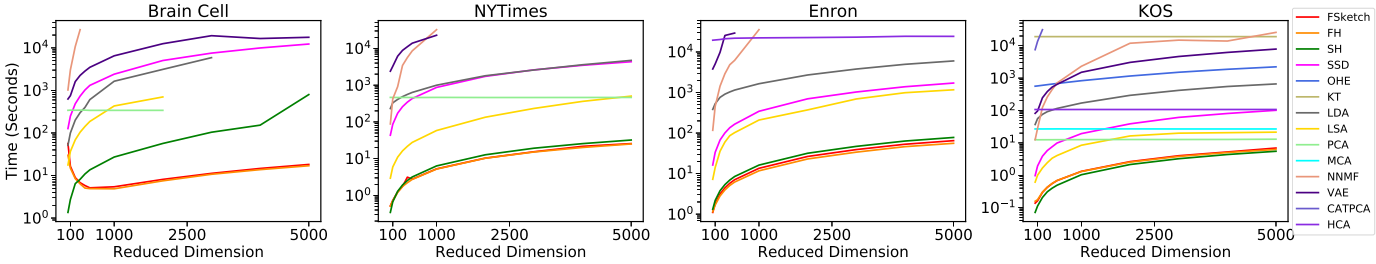


Fig. 7. Comparison among the baselines on the dimensionality reduction time. See Appendix G for results on the other datasets which show a similar trend and Section F for the errors encountered by some baselines.

TABLE 5
Summarisation of the baselines.

Characteristics	FSketch	FH	SH	SSD	OHE	KT	NNMF	MCA	LDA	LSA	PCA	VAE	CATPCA	HCA
Output discrete sketch	✓	✓	✓	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗	✓
Output real-valued sketch	✗	✗	✗	✓	✗	✗	✓	✓	✓	✓	✓	✓	✓	✗
Approximating distance measure	Hamming	Dot product	Cosine	Hamming	Hamming	NA	NA	NA	NA	NA	NA	NA	NA	NA
Require labelled data	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Dependency on the size of sample *	✗	✗	✗	✗	✗	✗	✗	✓	✗	✓	✓	✗	✓	✗
End tasks comparison	All	All	All	All	All	All	Clustering, Similarity Search	Clustering, Similarity Search	Clustering, Similarity Search	Clustering, Similarity Search	Clustering, Similarity Search	Clustering, Similarity Search	Clustering, Similarity Search	Clustering, Similarity Search

* The size of the maximum possible reduced dimension is the minimum of the number of data points and the dimension.

TABLE 6
Speedup of FSketch *w.r.t.* baselines on the reduced dimension 1000. OOM indicates "out-of-memory error" and DNS indicates "did not stop" after a sufficiently long time.

Dataset	OHE	KT	NNMF	MCA	LDA	LSA	PCA	VAE	SSD	SH	FH	CATPCA	HCA
NYTimes	OOM	OOM	6149×	OOM	189×	11.5×	88.14×	4340×	164.9×	1.2×	0.99×	DNS	DNS
Enron	OOM	DNS	2624×	OOM	122×	15.5×	OOM	DNS	25.5×	1.25×	0.87×	DNS	1268.2×
KOS	629×	14455×	1754×	20.41×	128×	6.40×	9.5×	1145×	14.62×	0.79×	0.98×	DNS	81.24×
DeliciousMIL	1332×	14036×	1753×	40.39×	136×	6.6×	18.1×	1557×	29.2×	0.61×	0.90×	DNS	117.6×
Gisette	399×	1347×	459×	5.7×	269×	5.4×	4.2×	285×	8.1×	0.69×	0.98×	DNS	16.78×
NIPS	378×	15863×	1599×	26.6×	302×	6.4×	3.17×	451×	29.9×	0.47×	1.20×	DNS	58.49×
Brain Cell	OOM	OOM	DNS	OOM	322×	79.38×	62.7×	1198×	443×	5×	0.89×	DNS	DNS

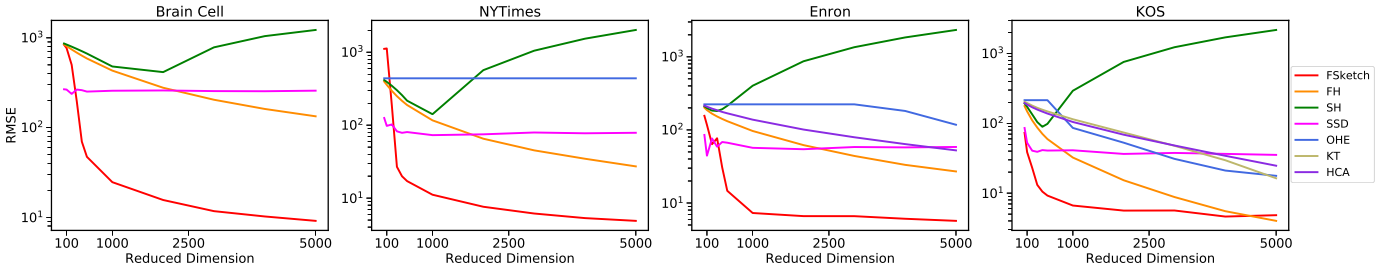


Fig. 8. Comparison on RMSE among baselines. A lower value is an indication of better performance. See Appendix G for results on the other datasets which show a similar trend.

mains the question of its variance. To decide the worthiness of our method, we compared the variance of the estimates of the Hamming distance obtained from FSketch and from the other randomised sketching algorithms with integer-valued sketches (KT was not included as it is a deterministic algorithm, and hence, has zero variance).

Figure 6 shows the Hamming error (estimation error) for a randomly chosen pair of points from the Enron dataset, averaged over 100 iterations. We make two observations.

First is that the estimate using FSketch is closer to the actual Hamming distance even at a smaller reduced dimension; in fact, as the reduced dimension is increased, the variance becomes smaller and the Hamming error converges to zero. Secondly, FSketch causes a smaller error compared to the other baselines. On the other hand, feature hashing highly underestimates the actual Hamming distance, but has low variance, and tends to have negligible Hamming error with an increase of the reduced dimension. The behaviour of SimHash is counter-intuitive as on lower reduced dimensions it closely estimates the actual Hamming distances, but on larger dimensions it starts to highly underestimate the actual Hamming distances. This creates an ambiguity on the choice of a dimension for generating a low-dimensional sketch of a dataset.Similar to FSketch, the sketches produced by SSD, though real-valued, allow estimation of pairwise Hamming distances. However the estimation error increases with the reduced dimension. Lastly, OHE seems to be highly underestimating pairwise Hamming distances.

4.5 Speedup in dimensionality reduction

We compress the datasets to several dimensions using FSketch and the baselines and report their running times in Figure 7. We notice that FSketch has a comparable speed *w.r.t.* Feature hashing and SimHash, and is significantly faster than the other baselines. However, both feature

TABLE 7
Speedup from running tasks on 1000-dimensional sketches instead of the full dimensional dataset. We got a DNS error while running clustering on the uncompressed BrainCell dataset.

Task	Brain cell	NYTimes	Enron	NIPS	KOS	Gisette	DeliciousMIL
Clustering	NA	$139.64\times$	$21.15\times$	$10.6\times$	$3.93\times$	$4.35\times$	$5.84\times$
Similarity Search	$1231.6\times$	$118.12\times$	$48.15\times$	$15.1\times$	$10.56\times$	$8.34\times$	$17.76\times$

hashing and SimHash are not able to accurately estimate the Hamming distance between data points and hence perform poorly on RMSE measure (Subsection 4.6) and the other tasks. Many baselines such as OHE, KT, NNMF, MCA, CATPCA, HCA give *"out-of-memory"* (OOM) error, and also didn't stop (DNS) even after running for a sufficiently long time ($\sim$ 10 hrs) on high dimensional datasets such as Brain Cell and NYTimes. On other moderate dimensional datasets such as Enron and KOS, our speedup *w.r.t.* these baselines are of the order of a few thousand. We report the numerical speedups that we observed in Table 6.

4.6 Performance on root-mean-squared-error (RMSE)

How good are the sketches for estimating Hamming distances between the uncompressed points in practice? To answer this, we compare `FSketch` with integer-valued sketching algorithms, namely, feature hashing, SimHash, Kendall correlation coefficient and OHE+BinSketch. Note that feature hashing and SimHash are known to approximate inner product and cosine similarity, respectively. However, we consider them in our comparison nonetheless as they output discrete sketches and Hamming distance can be computed on their sketch. We also include SSD for comparison which outputs real-valued sketches and estimates original pairwise Hamming distance. For each of the methods we compute its RMSE as the square-root of the average error, among all pairs of data points, between their actual Hamming distances and their corresponding estimates (for `FSketch` the estimate was obtained using Definition 5). Figure 8 compares these values of RMSE for different dimensions; note that a lower RMSE is an indication of better performance. It is immediately clear that the RMSE of `FSketch` is the lowest among all; furthermore, it falls to zero rapidly with increase in reduced dimension. This demonstrates that our proposal `FSketch` estimates the underlying pairwise Hamming distance better than the others.

4.7 Performance on clustering

We compare the performance of `FSketch` with baselines on the task of clustering and similarity search, and present the results for the first task in this section. The objective of the clustering experiment was to test if the data points in the reduced dimension maintain the original clustering structure. If they do, then it will be immensely helpful for those techniques that use a clustering, e.g., spam filtering. We used the *purity index* to measure the quality of k-mode and k-means clusters on the reduced datasets obtained through the compression algorithms; the ground truth was obtained using k-mode on the uncompressed data (for more details refer to Appendix E.2).

We summarise our findings on quality in Figure 9. The compressed versions of the NIPS, Enron, and KOS datasets that were obtained from `FSketch` yielded the best purity index as compared to those obtained from the other baselines; for the other datasets the compressed versions from `FSketch` are among the top. Even though it appears that KT offers comparable performance on the KOS, DeliciousMIL, and Gisette datasets *w.r.t.* `FSketch`, the downside of using KT is that its compression time is much higher than that of `FSketch` (see Table 6) on those datasets, and moreover it gives OOM/DNS error on the remaining datasets. Performance of FH also remains in the top few. However, its performance degrades on the NIPS dataset.

We tabulate the speedup of clustering of `FSketch`-compressed data over uncompressed data in Table 7 where we observe significant speedup in the clustering time, e.g., $139\times$ when run on a 1000 dimensional `FSketch`.

Recall that the dimensionality reduction time of our proposal is among the fastest among all the baselines which further reduces the total time to perform clustering by speeding up the dimensionality reduction phase. Thus the overall observation is that `FSketch` appears to be the most suitable method for clustering among the current alternatives, especially, for high-dimensional datasets on which clustering would take a long time.

4.8 Performance on similarity search

We take up another unsupervised task – that of similarity search. The objective here is to show that after dimensionality reduction the similarities of points with respect to some query points are maintained. To do so, we randomly split the dataset in two parts 5% and 95% – the smaller partition is referred to as the *query partition* and each point of this partition is called a query vector; we call the larger partition as *training partition*. For each query vector, we find top-k similar points in the training partition. We then perform dimensionality reduction using all the methods (for various values of reduced dimensions). Next, we process the compressed dataset where, for each query point, we compute the top-k similar points in the corresponding low-dimensional version of the training points, by maintaining the same split. For each query point, we compute the accuracy of baselines by taking the Jaccard ratio between the set of top-k similar points obtained in full dimensional data with the top-k similar points obtained in reduced dimensional dataset. We repeat this for all the points in the querying partition, compute the average, and report this as accuracy.

We summarise our findings in Figure 10. Note that PCA, MCA and LSA can reduce the data dimension up to the minimum of the number of data points and the original data dimension. Therefore their reduced dimension is at most 2000 for Brain cell dataset.

The top few methods appear to be feature hashing (FH), Kendall-Tau (KT), HCA along with `FSketch`. However, KT give *OOM* and *DNS* on the Brain cell, NYTimes and Enron datasets, and HCA give DNS error on BrainCell and NYTimes datasets. Ffurther, their dimensionality reduction time are much worse than `FSketch` (see Table 6).

`FSketch` outperforms FH on the BrainCell and the Enron datasets; however, on the remaining datasets, both

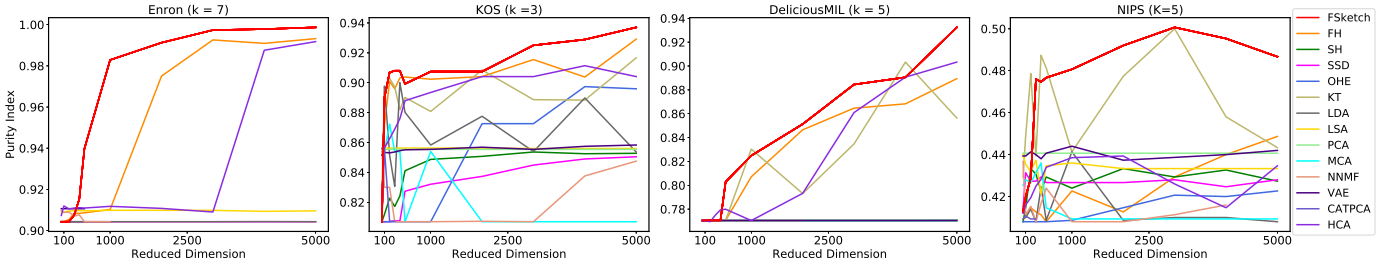


Fig. 9. Comparing the quality of clusters on the compressed datasets. See Appendix G for results on the other datasets which show a similar trend.

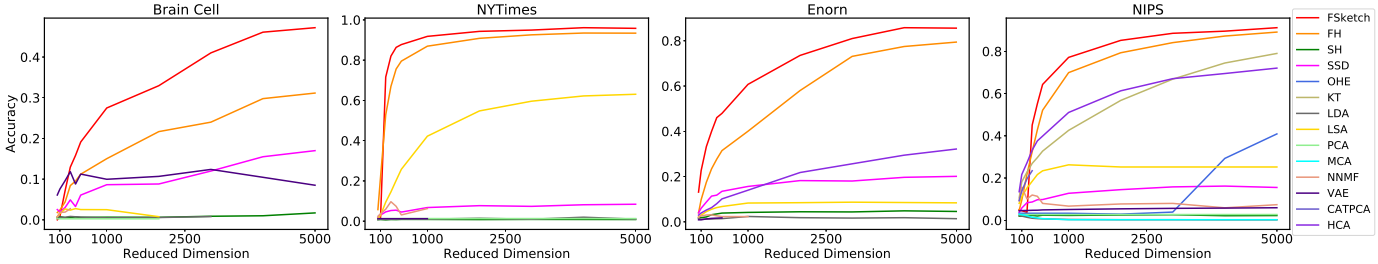


Fig. 10. Comparing the performance of the similarity search task (estimating top-k similar points with $k = 100$) achieved on the reduced dimensional data obtained from various baselines. See Appendix G for results on the other datasets which show a similar trend.

of them appear neck to neck for similarity search despite the fact that there is no known theoretical understanding of FH for Hamming distance — in fact, it was included in the baselines as a heuristic because it offers discrete-valued sketches on which Hamming distance can be calculated. Here want to point out that FH was not a consistent top-performer for clustering and similarity search.

The two other methods that are designed for Hamming distance, namely SSD and OHE, perform significantly worse than FSketch; in fact, the accuracy of OHE lies almost to the bottom on all the four datasets.

We also summarise the speedup of `FSketch`-compressed data over uncompressed data, on similarity search task, in Table 7. We observe a significant speedup – *e.g.* 1231.6× speedup on the BrainCell dataset when run on a 1000 dimensional `FSketch`.

To summarise, `FSketch` is one of the best approaches towards similarity search for high-dimensional datasets and the best if we also require theoretical guarantees or applicability towards other data analytic tasks.

5 Conclusion

In this paper, we proposed a sketching algorithm named `FSketch` for sparse categorical data such that the Hamming distances estimated from the sketches closely approximate the original pairwise Hamming distances. The low-dimensional data obtained by `FSketch` are discrete-valued, and therefore, enjoy the flexibility of running the data analytics tasks suitable for categorical data. The sketches allow tasks like clustering, similarity search to run which might not be possible on a high-dimensional dataset.

Our method does not require learning from the dataset and instead, exploits randomization to bring forth large speedup and high-quality output for standard data analytic tasks. We empirically validated the performance of our algorithm on several metric and end tasks such as RMSE, clustering, similarity search, and observed comparable performance while simultaneously getting significant speed up in dimensionality reduction and end-task with respect to several baselines. A common practice to analyse high-dimensional datasets is to partition them into smaller datasets. Given the simplicity, efficiency, and effectiveness of our proposal, we hope that `FSketch` will allow such analysis to be done on the full datasets and on general-purpose hardware.

References

[1] J. Moody, D. T. (eds, M. Kaufmann, M. O. Noordewier, G. G. Towell, and J. W. Shavlik, "Training knowledge-based neural networks to recognize genes in dna sequences," 1991.

[2] T. Rognvaldsson, L. You, and D. Garwicz, "State of the art prediction of hiv-1 protease cleavage sites," *Bioinformatics (Oxford, England)*, vol. 31, 2014.

[3] W. Hämäläinen and M. Nykänen, "Efficient discovery of statistically significant association rules," in *2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 203–212.

[4] J. Lavergne, R. Benton, and V. V. Raghavan, "Min-max itemset trees for dense and categorical datasets," in *Foundations of Intelligent Systems*, L. Chen, A. Felfernig, J. Liu, and Z. W. Raś, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 51–60.

[5] R. Agrawal, T. Imielinski, and A. Swami, "Mining association rules between sets of items in large databases," in *SIGMOD '93: Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, 1993, pp. 207–216.

[6] I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White, "Visualization of navigation patterns on a web site using model-based clustering," in *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000, pp. 280–284.

[7] L. Kurgan, K. Cios, R. Tadeusiewicz, M. Ogiela, and L. Goodenday, "Knowledge discovery approach to automated cardiac spect diagnosis," *Artificial intelligence in medicine*, vol. 23, pp. 149–69, 2001.

[8] S. Sidana, C. Laclau, and M.-R. Amini, "Learning to recommend diverse items over implicit feedback on pandor," 2018, pp. 427–431.

[9] J. T. Hancock and T. M. Khoshgoftaar, "Survey on categorical data for neural networks," *Journal of Big Data*, vol. 7, pp. 1–41, 2020.

[10] R. G. Baraniuk, V. Cevher, and M. B. Wakin, "Low-dimensional models for dimensionality reduction and signal recovery: A geometric perspective," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 959–971, 2010.

[11] L. H. Nguyen and S. Holmes, "Ten quick tips for effective dimensionality reduction," *PLOS Computational Biology*, vol. 15, no. 6, pp. 1–19, 2019.

[12] H. Liu and R. Setiono, "Chi2: feature selection and discretization of numeric attributes," in *Seventh International Conference on Tools with Artificial Intelligence, ICTAI '95, Herndon, VA, USA, November 5-8, 1995*, 1995, pp. 388–391. [Online]. Available: https://doi.org/10.1109/TAI.1995.479783

[13] H. Peng, F. Long, and C. H. Q. Ding, "Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1226–1238, 2005. [Online]. Available: https://doi.org/10.1109/TPAMI.2005.159

[14] M. G. Kendall, "A new measure of rank correlation," *Biometrika*, vol. 30, no. 1/2, pp. 81–93, 1938.

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

[16] D. Becker, "Using categorical data with one hot encoding," 2018. [Online]. Available: https://www.kaggle.com/dansbecker/using-categorical-data-with-one-hot-encoding

[17] R. Pratap, D. Bera, and K. Revanuru, "Efficient sketching algorithm for sparse binary data," in *2019 IEEE International Conference on Data Mining, ICDM 2019, Beijing, China, November 8-11, 2019*, 2019, pp. 508–517. [Online]. Available: https://doi.org/10.1109/ICDM.2019.00061

[18] M. Mitzenmacher, R. Pagh, and N. Pham, "Efficient estimation for high similarities using odd sketches," in *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014*, 2014, pp. 109–118. [Online]. Available: http://doi.acm.org/10.1145/2566486.2568017

[19] R. Pratap, I. Sohony, and R. Kulkarni, "Efficient compression technique for sparse sets," in *Advances in Knowledge Discovery and Data Mining - 22nd Pacific-Asia Conference, PAKDD 2018, Melbourne, VIC, Australia, June 3-6, 2018, Proceedings, Part III*, 2018, pp. 164–176. [Online]. Available: https://doi.org/10.1007/978-3-319-93040-4_14

[20] K. Q. Weinberger, A. Dasgupta, J. Langford, A. J. Smola, and J. Attenberg, "Feature hashing for large scale multitask learning," in *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, 2009, pp. 1113–1120. [Online]. Available: http://doi.acm.org/10.1145/1553374.1553516

[21] M. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, 2002, pp. 380–388. [Online]. Available: http://doi.acm.org/10.1145/509907.509965

[22] A. Zheng and A. Casari, *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*. O'Reilly Media, 2018. [Online]. Available: https://books.google.co.in/books?id=sthSDwAAQBAJ

[23] W. B. Johnson and J. Lindenstrauss, "Extensions of lipschitz mappings into a hilbert space," *Conference in modern analysis and probability (New Haven, Conn., 1982), Amer. Math. Soc., Providence, R.I.*, pp. 189–206, 1983. [Online]. Available: http://dx.doi.org/10.1016/S0022-0000(03)00025-4

[24] D. Achlioptas, "Database-friendly random projections: Johnson-lindenstrauss with binary coins," *J. Comput. Syst. Sci.*, vol. 66, no. 4, pp. 671–687, 2003. [Online]. Available: http://dx.doi.org/10.1016/S0022-0000(03)00025-4

[25] P. Li, T. Hastie, and K. W. Church, "Very sparse random projections," in *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, 2006, pp. 287–296. [Online]. Available: https://doi.org/10.1145/1150402.1150436

[26] D. M. Kane and J. Nelson, "Sparser johnson-lindenstrauss transforms," *J. ACM*, vol. 61, no. 1, pp. 4:1–4:23, 2014. [Online]. Available: https://doi.org/10.1145/2559902

[27] B. Schölkopf, A. J. Smola, and K. Müller, "Kernel principal component analysis," in *Artificial Neural Networks - ICANN '97, 7th International Conference, Lausanne, Switzerland, October 8-10, 1997, Proceedings*, 1997, pp. 583–588.

[28] J. Blasius and M. Greenacre, "Multiple correspondence analysis and related methods," *Multiple Correspondence Analysis and Related Methods*, 2006.

[29] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, 1998, pp. 604–613.

[30] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher, "Min-wise independent permutations (extended abstract)," in *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, 1998, pp. 327–336. [Online]. Available: http://doi.acm.org/10.1145/276698.276781

[31] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *VLDB'99, Proceedings of 25th International Conference on Very Large Data Bases, September 7-10, 1999, Edinburgh, Scotland, UK*, 1999, pp. 518–529. [Online]. Available: http://www.vldb.org/conf/1999/P49.pdf

[32] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE*, vol. 41, no. 6, pp. 391–407, 1990.

[33] D. M. Blei, A. Y. Ng, M. I. Jordan, and J. Lafferty, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, p. 2003, 2003.

[34] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization." in *NIPS*, T. K. Leen, T. G. Dietterich, and V. Tresp, Eds., 2000, pp. 556–562.

[35] E. Golinko and X. Zhu, "Generalized feature embedding for supervised, unsupervised, and online learning tasks," *Information Systems Frontiers*, vol. 21, no. 1, pp. 125–142, 2019.

[36] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[37] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[38] X. Li, M. Chen, and Q. Wang, "Discrimination-aware projected matrix factorization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 4, pp. 809–814, 2020.

[39] X. Zhang, Q. Mai, and H. Zou, "The maximum separation subspace in sufficient dimension reduction with categorical response," *Journal of Machine Learning Research*, vol. 21, no. 29, pp. 1–36, 2020. [Online]. Available: http://jmlr.org/papers/v21/17-788.html

[40] X. Chen, H. Yang, S. Zhao, M. R. Lyu, and I. King, "Effective data-aware covariance estimator from compressed data," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 31, no. 7, pp. 2441–2454, 2020. [Online]. Available: https://doi.org/10.1109/TNNLS.2019.2929106

[41] Q. Wang, Z. Qin, F. Nie, and X. Li, "C2dnda: A deep framework for nonlinear dimensionality reduction," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 2, pp. 1684–1694, 2021.

[42] M. Banerjee and N. R. Pal, "Unsupervised feature selection with controlled redundancy (ufescor)," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 12, pp. 3390–3403, 2015.

[43] G. Cormode, M. Datar, P. Indyk, and S. Muthukrishnan, "Comparing data streams using hamming norms (how to zero in)," *IEEE Trans. Knowl. Data Eng.*, vol. 15, no. 3, pp. 529–540, 2003.

[44] D. M. Kane, J. Nelson, and D. P. Woodruff, "An optimal algorithm for the distinct elements problem," in *Proceedings of the Twenty-Ninth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2010, June 6-11, 2010, Indianapolis, Indiana, USA*, 2010, pp. 41–52.

[45] R. Freivalds, "Probabilistic machines can use less running time," in *IFIP Congress*, 1977.

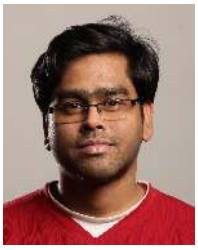
[46] M. Mitzenmacher and E. Upfal, *Probability and computing - randomized algorithms and probabilistic analysis*, 2005.

[47] M. Lichman, "UCI machine learning repository," 2013.

[48] I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror, "Result analysis of the nips 2003 feature selection challenge," in *Advances in Neural Information Processing Systems 17*, 2005, pp. 545–552.

[49] H. Soleimani and D. J. Miller, "Semi-supervised multi-label topic models for document classification and sentence labeling," in *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*, 2016, pp. 105–114. [Online]. Available: https://doi.org/10.1145/2983323.2983752

[50] X. Genomics, "1.3 million brain cells from e18 mice," *CC BY*, vol. 4, 2017.
[51] G. Cormode, M. Datar, P. Indyk, and S. Muthukrishnan, "Comparing data streams using hamming norms (how to zero in)," *IEEE Trans. Knowl. Data Eng.*, vol. 15, no. 3, pp. 529–540, 2003.
[52] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.
[53] Z. Sulc and H. Rezanková, "Dimensionality reduction of categorical data: Comparison of hca and catpca approaches," 2015.
[54] C. McDiarmid, *On the method of bounded differences*, ser. London Mathematical Society Lecture Note Series. Cambridge University Press, 1989, p. 148–188.
[55] Z. Huang, "Extensions to the k-means algorithm for clustering large data sets with categorical values," *Data Mining and Knowledge Discovery*, vol. 2, no. 3, pp. 283–304, 1998. [Online]. Available: https://doi.org/10.1023/A:1009769707641
[56] G. Cormode and S. Muthukrishnan, "An improved data stream summary: the count-min sketch and its applications," *J. Algorithms*, vol. 55, no. 1, pp. 58–75, 2005. [Online]. Available: https://doi.org/10.1016/j.jalgor.2003.12.001
[57] M. Charikar, K. Chen, and M. Farach-Colton, "Finding frequent items in data streams," *Theoretical Computer Science*, vol. 312, no. 1, pp. 3 – 15, 2004, automata, Languages and Programming.
[58] P. Indyk, "Stable distributions, pseudorandom generators, embeddings, and data stream computation," *J. ACM*, vol. 53, no. 3, pp. 307–323, 2006. [Online]. Available: http://doi.acm.org/10.1145/1147954.1147955
[59] R. Fisher, "The statistical utilization of multiple measurements," *Annals of Eugenics*, vol. 8, pp. 376–386, 1938.
[60] G. E. Hinton and R. S. Zemel, "Autoencoders, minimum description length and helmholtz free energy," in *Proceedings of the 6th International Conference on Neural Information Processing Systems*, ser. NIPS'93. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993, p. 3–10.
[61] S. Mika, B. Schölkopf, A. Smola, K.-R. Müller, M. Scholz, and G. Rätsch, "Kernel pca and de-noising in feature spaces," in *Advances in Neural Information Processing Systems*, M. Kearns, S. Solla, and D. Cohn, Eds., vol. 11. MIT Press, 1999. [Online]. Available: https://proceedings.neurips.cc/paper/1998/file/226d1f15ecd35f784d2a20c3ecf56d7f-Paper.pdf
[62] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, p. 2319, 2000.
[63] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological Cybernetics*, vol. 43, no. 1, pp. 59–69, Jan. 1982. [Online]. Available: http://dx.doi.org/10.1007/BF00337288



Debajyoti Bera received his B.Tech. in Computer Science and Engineering in 2002 at Indian Institute of Technology (IIT), Kanpur, India and his Ph.D. degree in Computer Science from Boston University, Massachusetts, USA in 2010. Since 2010 he is an assistant professor at Indraprastha Institute of Information Technology, (IIIT-Delhi), New Delhi, India. His research interests include quantum computing, randomized algorithms, and engineering algorithms for networks, data mining, and information security.

Rameshwar Pratap has earned Ph.D in Theoretical Computer Science in 2014 from Chennai Mathematical Institute (CMI). Earlier, he completed Masters in Computer Application (MCA) from Jawaharlal Nehru University and BSc in Mathematics, Physics, and Computer Science from University of Allahabad. Post Ph.D he has worked TCS Innovation Labs (New Delhi, India), and Wipro AI-Research (Bangalore, India). Since 2019 he is working as an assistant professor at School of Computing and Electrical Engineering (SCEE), IIT Mandi. His research interests include algorithms for dimensionality reduction, robust sampling, and algorithmic fairness.

Bhisham Dev Verma is pursuing Ph.D from IIT Mandi. He has done his Masters in Applied Mathematics from IIT Mandi and BSc in Mathematics, Physics, and Chemistry from Himachal Pradesh University. His research interest includes data mining, algorithms for dimension reduction, optimization and machine learning.

APPENDIX A
ANALYSIS OF ONE-HOT ENCODING + BINARY COMPRESSION

Let x and y be two n-dimensional categorical vectors with sparsity at most σ; c will denote the maximum number of values any attribute can take. Let x' and y' be the one-hot encodings of x and y, respectively. Further, let x'' and y'' denote the compression of x' and y', respectively, using BinSketch [17] which is the state-of-the-art dimensionality reduction for binary vector using Hamming distance.

Observe that the sparsity of x' is same as that of x and a similar claim holds for y' and y. However, $HD(x',y')$ does not hold a monotonic relationship with $HD(x,y)$. It is easy to show that $HD(x,y) \leq HD(x',y') \leq 2HD(x,y)$. Therefore,

$$|HD(x,y) - HD(x',y')| \leq HD(x,y) \leq 2\sigma. \tag{4}$$

We need the following lemma that was used to analyse BinSketch [17, Lemma 12,Appendix A].

Lemma 11. *Suppose we compress two n'-dimensional binary vectors x' and y' with sparsity at most σ to g-dimensional binary sketches, denotes x'' and y'' respectively, by following an algorithm proposed in the BinSketch work. If g is set to $\sigma\sqrt{\frac{\sigma}{2}\ln\frac{6}{\delta}}$ for any $\delta \in (0,1)$, then the following holds with probability at least $1-\delta$.*

$$|HD(x',y') - HD(x'',y'')| \leq 6\sqrt{\tfrac{\sigma}{2}\ln\tfrac{6}{\delta}}.$$

Combining the above inequality with that in Equation 4 gives us

$$|HD(x,y) - HD(x'',y'')| \leq 2\sigma + 6\sqrt{\tfrac{\sigma}{2}\ln\tfrac{6}{\delta}} \leq 2\sigma\sqrt{\ln\tfrac{2}{\delta}}$$

if we set the reduced dimension to $\sigma\sqrt{\frac{\sigma}{2}\ln\frac{6}{\delta}}$.

This bound is worse compared to that of `FSketch` where we able to prove an accuracy of $\Theta(\sqrt{\sigma\ln\frac{2}{\delta}})$ using reduced dimension value of 4σ (see Lemma 8).

APPENDIX B
PROOFS FROM SECTION 3.3

Lemma 6. *Let α denote a desired additive accuracy. Then, for any x, y with sparsity σ,*

$$\Pr\Big[|f - f^*| \geq \alpha\Big] \leq 2\exp(-\tfrac{\alpha^2}{4\sigma}).$$

Proof. Fix any R and x, y; the rest of the proof applies to any R, and therefore, holds for a random R as well. Define a vector $z \in \{0, \pm1, \ldots, \pm c\}^n$ in which $z_i = (x_i - y_i)$; the number of non-zero entries of z are at most 2σ since the number of non-zero entries of x and y are at most σ. Let J_0 be the set of coordinates from $\{1,\ldots,n\}$ at which z is 0, and let J_1 be the set of the rest of the coordinates; from above, $J_1 \leq 2\sigma$.

Define the event E_j as "$[\phi_j(x) \neq \phi_j(y)]$". Note that f can be written as a sum of indicator random variables, $\sum_j I(E_j)$, and we would like to prove that f is almost always close to $f^* = \mathbb{E}[f]$.

Observe that $\phi_j(x) = \phi_j(y)$ iff $\sum_{i\in\rho^{-1}(j)} z_i \cdot r_i = 0 \mod p$ iff $\sum_{i\in\rho^{-1}(j)\cap J_1} z_i \cdot r_i = 0 \mod p$. In other words, $\rho(i)$ could be set to anything for $i \in J_0$ without any effect on the event E_j; hence, we will assume that the mapping ρ is defined as a random mapping only for $i \in J_1$, and further for the ease of analysis, we will denote them as $\rho(i_1), \rho(i_2), \ldots, \rho(i_{2\sigma})$ (if $|J_1| < 2\sigma$ then move a few coordinates from J_0 to J_1 without any loss of correctness).

To prove the concentration bound we will employ martingales. Consider the sequence of these random variables $\rho' = \rho(i_1), \rho(i_2), \ldots, \rho(i_{2\sigma})$ – these are independent. Define a function $g(\rho')$ of these random variables as a sum of indicator random variables as stated below (note that R and $\rho(i)$, for $i \in J_0$, are fixed at this point)

$$\begin{aligned} &g(\rho(i_1), \rho(i_2), \ldots \rho(i_{2\sigma})) \\ &= \sum_j I\left(\sum_{i\in\rho^{-1}(j)\cap J_1} z_i \cdot r_i \neq 0 \mod p\right) \\ &= \sum_j I(E_j) = f \end{aligned}$$

Now consider an arbitrary $t \in \{1,\ldots,2\sigma\}$ and let $q = \rho(i_t)$; observe that z_{i_t} influences only E_q. Choose an arbitrary value $q' \in \{1,\ldots,d\}$ that is different from q. Observe that, if ρ is modified only by setting $\rho(i_t) = q'$ then we claim that "bounded difference holds".

Proposition 12. $|\ g(\rho(i_1),\ldots,\rho(i_{t-1}),q,\ldots,\rho(i_{2\sigma})) - g(\rho(i_1),\ldots,\rho(i_{t-1}),q',\ldots,\rho(i_{2\sigma}))\ | \leq 2.$

The proposition holds since the only effects of the change of $\rho(i_t)$ from q to q' are seen in E_q and $E_{q'}$ (earlier E_q depended upon z_{i_t} that now changes to $E_{q'}$ being depended upon z_{i_t}). Since $g()$ obeys bounded difference, therefore, we can apply McDiarmid's inequality [46, Ch 17], [54].

Theorem 13 (McDiarmid's inequality)**.** *Consider independent random variables $X_1,\ldots,X_m \in \mathcal{X}$, and a mapping $f : \mathcal{X}^m \to \mathbb{R}$ which for all i and for all $x_1,\ldots x_m, x_i'$ satisfies the property: $|f(x_1,\ldots,x_i,\ldots,x_m) - f(x_1,\ldots,x_i',\ldots,x_m)| \leq c_i$, where $x_1,\ldots x_m, x_i'$ are possible values for the input variables of the function f. Then,*

$$\Pr\Big[|\mathbb{E}[f(X_1,\ldots,X_m) - f(X_1,\ldots,X_m)]| \geq \varepsilon\Big] \leq 2\exp\left(\frac{-2\varepsilon^2}{\sum_{i=1}^m c_i^2}\right).$$

This inequality implies that, for every x, y, R,

$$\Pr_\rho\Big[|\mathbb{E}[f] - f| \geq \alpha\Big] \leq 2\exp\left(-\frac{2\alpha^2}{(2\sigma)2^2}\right) = \exp\left(-\frac{\alpha^2}{4\sigma}\right).$$

Hence, the lemma is proved. □

Lemma 7. $\Pr[f \geq dP] \leq 2\exp(-P^2\sigma)$.

Proof. Since $f^* = dP(1-D^h) = dP - dPD^h$, if $f \geq dP$ then $|f - f^*| \geq dPD^h$.

$$\begin{aligned} \Pr[f \geq dP] &\leq \Pr[|f - f^*| \geq dPD^h] \\ &\leq 2\exp(-\tfrac{d^2P^2D^{2h}}{4\sigma}) && \text{(using Lemma 6)} \\ &= 2\exp(-\tfrac{P^2}{4\sigma}d^2(1-\tfrac{1}{d})^{2h}) \\ &\leq 2\exp(-\tfrac{P^2}{4\sigma}(d-h)^2) && (\because\ (1-\tfrac{1}{d})^h \geq 1-\tfrac{h}{d}) \\ &\leq 2\exp(-P^2\sigma) && (\because\ \tfrac{(d-h)^2}{4\sigma} \geq \sigma) \end{aligned}$$

Here we have used the fact that $h \leq 2\sigma$ which, along with the setting $d = 4\sigma$, implies that $(d-h) \geq 2\sigma$. □

Lemma 8. *Choose $d = 4\sigma$ as the dimension of* `FSketch` *and choose a prime p and an error parameter $\delta \in (0,1)$ (ensure that $1 - \frac{1}{p} \geq \frac{4}{\sqrt{\sigma}}\sqrt{\ln \frac{2}{\delta}}$ — see the proof for discussion). Then the estimator defined in Definition 5 is close to the Hamming distance between x and y with high probability, i.e.,*

$$\Pr\left[|\hat{h} - h| \geq \tfrac{32}{1-1/p}\sqrt{\sigma \ln \tfrac{2}{\delta}}\right] \leq \delta.$$

Proof. Denote $|\hat{h} - h|$ by Δh and let $\alpha = \sqrt{d \ln \frac{2}{\delta}}$. We will prove that $\Delta h < \frac{32}{P}\sqrt{\sigma \ln \frac{2}{\delta}}$ for the case $|f - f^*| \leq \alpha$ which, by Lemma 6, happens with probability at least $(1 - 2\exp(-\frac{\alpha^2}{4\sigma})) = 1 - \delta$.

First we make a few technical observations all of which are based on standard inequalities of binomial series and logarithmic functions. It will be helpful to remember that $D = 1 - 1/d \in (0,1)$.

Observation 14. *For reasonable values of σ, and reasonable values of δ, almost all primes satisfy the bound $P \geq \frac{4}{\sqrt{\sigma}}\sqrt{\ln \frac{2}{\delta}}$. We will assume this inequality to hold without loss of generality* [2].

For example, $p = 2$ is sufficient for $\sigma \approx 1000$ and $\delta \approx 0.001$ (remember that $P = 1 - \frac{1}{p}$). Furthermore, observe that P is an increasing function of p, and the right hand side is a decreasing function of σ, increasing with decreasing delta but at an extremely slow logarithmic rate.

Observation 15. *$\frac{dP}{\alpha} > 4$ can be assumed without loss of generality. This holds since the left hand side is $\frac{dP}{\sqrt{d}\sqrt{ln(2/\delta)}} = \frac{P\sqrt{d}}{\sqrt{\ln(2/\delta)}} \geq \frac{4\sqrt{d}}{\sqrt{\sigma}}$ (by Observation 14) which is at least 4.*

Observation 16. *Based on the above assumptions, $f < dP$.*

Proof of Observation. We will prove that $\sqrt{d \ln \frac{2}{\delta}} < dPD^h$. Since $|f - f^*| \leq \sqrt{d \ln \frac{2}{\delta}}$ and $f^* = dP(1 - D^h)$, it follows that $f \leq f^* + \sqrt{d \ln \frac{2}{\delta}} < dP$.

$$\begin{aligned}
\sqrt{d}PD^h = \frac{dPD^h}{\sqrt{d}} &\geq \frac{P}{\sqrt{d}} d(1 - \tfrac{1}{d})^h \geq \frac{P}{\sqrt{d}} d(1 - \tfrac{h}{d}) \\
&= \frac{P}{\sqrt{d}}(d - h) \geq \frac{P}{\sqrt{d}}\frac{d}{2} \\
&\qquad (\because h \leq 2\sigma,\ d - h \geq 2\sigma = \tfrac{d}{2}) \\
&= P\sqrt{\sigma} \geq 4\sqrt{\ln \tfrac{2}{\delta}} \qquad \text{(Observation 14)}
\end{aligned}$$

which proves the claim stated at the beginning of the proof. □

Based on this observation, $\hat{h}$ is calculated as $\ln\left(1 - \frac{f}{dP}\right) / \ln D$ (see Definition 5). Thus, we get $D^{\hat{h}} = 1 - \frac{f}{dP}$. Further, from Equation 3 we get $D^h = 1 - \frac{f^*}{dP}$.

Observation 17. *$D^h \geq D^{2\sigma} \geq \frac{9}{16}$. This is since $h \leq 2\sigma$ and $D^\sigma = (1 - \frac{1}{d})^\sigma \geq 1 - \frac{\sigma}{d} = \frac{3}{4}$.*

2. If the reader is wondering why we are not proving this fact, it may be observed that this relationship does not hold for small values of σ, e.g., $\sigma = 16, \delta = 0.01$.

Observation 18. *$D^{\hat{h}} > \frac{5}{16}$.*

This is not so straight forward as Observation 17 since $\hat{h}$ is calculated using a formula and is not guaranteed, ab initio, to be upper bounded by 2σ.

Proof of Observation. We will prove that $\frac{f}{dP} < \frac{11}{16}$ which will imply that $D^{\hat{h}} = 1 - \frac{f}{dP} > \frac{5}{16}$.

For the proof of the lemma we have considered the case that $f \leq f^* + \alpha$. Therefore, $\frac{f}{dP} \leq \frac{f^*}{dP} + \frac{\alpha}{dP}$. Substituting the value of $f^* = dP(1 - D^h)$ from Equation 3 and using Observation 17 we get the bound $\frac{f}{dP} \leq \frac{7}{16} + \frac{\alpha}{dP}$. We can further simplify the bound using Observation 15:

$\frac{f}{dP} \leq \frac{7}{16} + \frac{\alpha}{dP} \leq \frac{7}{16} + \frac{1}{4} < \frac{11}{16}$, validating the observation.

□

Now we get into the main proof which proceeds by considering two possible cases.

(Case $\hat{h} \geq h$, i.e., $\Delta h = \hat{h} - h$:) We start with the identity $D^h - D^{\hat{h}} = \frac{f - f^*}{dP}$.

Notice that the RHS is bounded from the above by $\frac{\alpha}{dP}$ and the LHS can bounded from the below as

$$D^h - D^{\hat{h}} = D^h(1 - D^{\Delta h}) > \tfrac{9}{16}(1 - D^{\Delta h})$$

where we have used Observation 17. Combining these facts we get $\frac{\alpha}{dP} > \frac{9}{16}(1 - D^{\Delta h})$.

(Case $h \geq \hat{h}$, i.e., $\Delta h = h - \hat{h}$:) In a similar manner, we start with the identity $D^{\hat{h}} - D^h = \frac{f^* - f}{dP}$ in which the RHS we bound again from the above by $\frac{\alpha}{dP}$ and the LHS is treated similarly (but now using Observation 18).

$$D^{\hat{h}} - D^h = D^{\hat{h}}(1 - D^{\Delta h}) > \tfrac{5}{16}(1 - D^{\Delta h})$$

and then, $\frac{\alpha}{dP} > \frac{5}{16}(1 - D^{\Delta h})$.

So in both the cases we show that $\frac{\alpha}{dP} > \frac{5}{16}(1 - D^{\Delta h})$. Our desired bound on Δh can now be obtained.

$$\begin{aligned}
\Delta h \ln D &\geq \ln\left(1 - \tfrac{16}{5}\tfrac{\alpha}{dP}\right) \geq -\tfrac{16\alpha}{5dP} / (1 - \tfrac{16\alpha}{5dP}) = -\tfrac{16\alpha}{5dP - 16\alpha} \\
&\qquad \text{(using the inequality } \ln(1+x) \geq \tfrac{x}{1+x} \text{ for } x > -1) \\
\therefore \Delta h &\leq \frac{1}{\ln \frac{1}{D}} \frac{16\alpha}{5dP - 16\alpha} \leq \frac{16\alpha d}{5dP - 16\alpha} \\
&\qquad \text{(it is easy to show that } \ln \tfrac{1}{D} = \ln \tfrac{1}{1 - 1/d} \geq 1/d) \\
&= \frac{\frac{16}{5}d}{\frac{dP}{\alpha} - \frac{16}{5}} \\
&< \frac{\frac{16}{5}d}{\frac{dP}{5\alpha}} \text{ (using Observation 15, } \tfrac{dP}{\alpha} - \tfrac{16}{5} > \tfrac{dP}{5\alpha}) \\
&= \frac{16\alpha}{P} = \frac{16}{P}\sqrt{d \ln \tfrac{2}{\delta}} = \frac{32}{P}\sqrt{\sigma \ln \tfrac{2}{\delta}}
\end{aligned}$$

□

Lemma 9. *Suppose we know that $h \leq \sqrt{\sigma}$ and choose $d = 16\sqrt{\sigma \ln \frac{2}{\delta}}$ as the dimension for* `FSketch`*. Then (a) also $f < dP$ with high probability and moreover we get a better estimator. That is, (b) $\Pr\left[|\hat{h} - h|\right] \geq \frac{8}{1-1/p}\sqrt{\sigma \ln \frac{2}{\delta}}] \leq \delta$.*

Proof of (a) $f < dP$ with high probability. Following the steps of the proof of Lemma 7,

$$\begin{aligned}
\Pr[f \geq dP] &\leq 2\exp(-\tfrac{d^2P^2D^{2h}}{4\sigma}) \\
&\leq 2\exp(-P^2\tfrac{(d-h)^2}{4\sigma})
\end{aligned}$$

Let L denote $\sqrt{\ln \frac{2}{\delta}}$; note that $L > 1$. Now, $d = 16L\sqrt{\sigma}$ and $h \leq \sqrt{\sigma}$. So, $d - h \geq 15L\sqrt{\sigma} > 15\sqrt{\sigma}$ and, therefore, $\frac{(d-h)^2}{\sigma} > 225$. Using this bound in the equation above, we can upper bound the right-hand side as $2\exp(-225(1 - \frac{1}{p})^2/4)$ which is a decreasing function of p, the lowest (for $p = 2$) being $2\exp(-225/4 * 4) \approx 10^{-6}$. □

Proof of (b) a better estimator of h. The proof is almost exactly same as that of Lemma 8, with only a few differences. We set $\alpha = d/8$ where $d = 16\sqrt{\sigma \ln \frac{2}{\delta}}$. Incidentally, the value of α remains the same in terms of σ ($\alpha = \sqrt{4\sigma \ln \frac{2}{\delta}}$). Thus, the probability of error remains same as before;

$$2\exp\left(-\tfrac{d^2}{64 \cdot 4\sigma}\right) = \delta.$$

Observation 14 is true without any doubt. $\frac{dP}{\alpha} = 8P$ which is greater than 4 for any prime number; so Observation 15 is true in this scenario.

Observation 16 requires a new proof. Following the steps of the above proof of Observation 16, it suffices to prove that $dPD^h > \frac{d}{8}$.

$$\begin{aligned} PD^h &= P(1 - \tfrac{1}{d})^h \geq P(1 - \tfrac{h}{d}) \\ &= P(\tfrac{d-h}{d}) \geq P\tfrac{15L\sqrt{\sigma}}{16L\sqrt{\sigma}} = P\tfrac{15}{16} > \tfrac{1}{2}\tfrac{15}{16} > \tfrac{1}{8} \end{aligned}$$

Observation 17 is now tighter since $D^h \geq D^{\sqrt{\sigma}} = (1 - \frac{1}{d})^{\sqrt{\sigma}} \geq 1 - \frac{\sqrt{\sigma}}{d} = 1 - \frac{1}{16\sqrt{\ln 2/\delta}} \geq \frac{3}{4}$ for reasonable values of δ. Similarly Observation 18 is also tighter (it relies on only the above observations) since $\frac{f^*}{dP} = 1 - D^h \leq 1 - \frac{3}{4}$ and $\frac{\alpha}{dP} < \frac{1}{4}$; we get $D^{\hat{h}} > \frac{1}{2}$.

Following similar steps as above, for the case $\hat{h} \geq h$, we get $\frac{\alpha}{dP} > \frac{3}{4}(1 - D^{\Delta h})$ and for the case $\hat{h} < h$, we get $\frac{\alpha}{dP} > \frac{1}{2}(1 - D^{\Delta h})$ leading to the common condition that $\frac{\alpha}{dP} > \frac{1}{2}(1 - D^{\Delta h})$.

The final thing to calculate is the bound on Δh.

$$\begin{aligned} \Delta h \ln D &\geq \ln\left(1 - \tfrac{2\alpha}{dP}\right) \geq -\tfrac{2\alpha}{dP}/(1 - \tfrac{2\alpha}{dP}) = -\tfrac{2\alpha}{dP - 2\alpha} \\ &\quad \text{(using the inequality } \ln(1+x) \geq \tfrac{x}{1+x} \text{ for } x > -1) \\ \therefore \Delta h &\leq \frac{1}{\ln \frac{1}{D}} \frac{2\alpha}{dP - 2\alpha} \leq \frac{2\alpha d}{dP - 2\alpha} \\ &\quad \text{(it is easy to show that } \ln \tfrac{1}{D} = \ln \tfrac{1}{1 - 1/d} \geq 1/d) \\ &= \frac{2d}{\frac{dP}{\alpha} - 2} \\ &< \frac{2d}{\frac{dP}{2\alpha}} \text{ (using Observation 15, } \tfrac{dP}{\alpha} - 2 > \tfrac{dP}{2\alpha}) \\ &= \frac{4\alpha}{P} = \frac{4}{P}\sqrt{4\sigma \ln \tfrac{2}{\delta}} = \frac{8}{P}\sqrt{\sigma \ln \tfrac{2}{\delta}} \end{aligned}$$

□

APPENDIX C
COMPLEXITY ANALYSIS OF `FSketch`

There are two major operations with respect to `FSketch`—construction of sketches and estimation of Hamming distance from two sketches. We will discuss their time and space requirements. There are efficient representations of sparse data vectors, but for the sake of simplicity we assume full-size arrays to store vectors; similarly we assume simple dictionaries for storing the interval variables ρ, R by `FSketch`. While it may be possible to reduce the number of random bits by employing k-wise independent bits and mappings, we left it out of the scope of this work and for future exploration.

1) Construction: Sketches are constructed by the `FSketch` algorithm which does a linear pass over the input vector, maps every non-zero attribute to some entry of the sketch vector and then updates that corresponding entry. The time to process one data vector becomes $\Theta(n) + O(\sigma \cdot poly(\log p))$ which is $O(n)$ for constant p. The interval variables, ρ, R, p, require space $\Theta(n \log d)$, $\Theta(n \log p)$ and $\Theta(\log p)$, respectively, which is almost $O(n)$ if $\sigma \ll n$. Furthermore, ρ and R, that can consume bulk of this space, can be freed once the sketch construction phase is over. A sketch itself consumes $\Theta(d \log p)$ space.
2) Estimation: There is no additional space requirement for estimating the Hamming distance of a pair of points from their sketches. The estimator scans both the sketches and computes their Hamming distance; finally it computes an estimate by using Definition 5. The running time is $O(d \log p)$.

APPENDIX D
PROOFS FROM SECTION 3.4

Lemma 10. *If $d = 4\sigma$ (as required by Lemma 8), then the expected number of non-zero entries of $\phi(x)$ is upper bounded by $\frac{d}{4}$. Further, at least 50% of $\phi(x)$ will be zero with probability at least $\frac{1}{2}$.*

Proof. The lemma can be proved by treating it as a balls-and-bins problem. Imagine throwing σ balls (treat them as the non-zero attributes of x) into d bins (treat them as the sketch cells) independently and uniformly at random. If the jth-bin remains empty then $\phi_j(x)$ must be zero (the converse is not true). Therefore, the expected number of non-zero cells in the sketch is upper bounded by the expected number of empty bins, which can be easily shown to be $d[1 - (1 - \frac{1}{d})^\sigma]$. Using the stated value of d, this expression can further be upper bounded.

$$d[1 - (1 - \tfrac{1}{d})^\sigma] \leq d[1 - (1 - \tfrac{\sigma}{d})] = \tfrac{d}{4}$$

Furthermore, let NZ denote the number of non-zero entries in $\phi(x)$. We derived above $\mathbb{E}[NZ] \leq \frac{d}{4}$. Markov inequality can help in upper bounding the probability that $\phi(x)$ contains many non-zero entries.

$$\Pr[NZ \geq \tfrac{d}{2}] \leq \mathbb{E}[NZ]/\tfrac{d}{2} \leq \tfrac{1}{2}$$

□

APPENDIX E
REPRODUCIBILITY DETAILS

E.1 Baseline implementations

1) We implemented the feature hashing (FH) [20], SimHash (SH) [21], Sketching via Stable Distribution

(SSD) [51] and One Hot Encoding (OHE) [17] algorithms on own own; we have made these implementations publicly available [3].
2) For Kendall rank correlation coefficient [14] we used the implementation provided by `pandas data frame` [4].
3) For Latent Semantic Analysis (LSA) [32], Latent Dirichlet Allocation (LDA) [33], Non-negative Matrix Factorisation (NNMF) [34], and vanilla Principal component analysis (PCA), we used their implementations available in the `sklearn.decomposition` library [5].
4) For Multiple Correspondence Analysis (MCA) [28], we used a Python library [6].
5) For HCA [53], we performed hierarchical clustering [7] over the features in which we set the number of clusters to the value of reduced dimension. We then randomly sampled one feature from each of the clusters, and considered the data points restricted to the sampled features.
6) For CATPCA [53], we used an R package [8].

It should be noted that PCA, MCA and LSA cannot reduce the dimension beyond the number of data points.

E.2 Reproducibility details for clustering task

We first generated the ground truth clustering results on the datasets using k-mode [55] (we used a Python library [9]).

We then compressed the datasets using the baselines. Of them, feature hashing [20], SimHash [21], and Kendall rank correlation coefficient [14] generate integer/discrete valued sketches on which we can define a Hamming distance. Therefore we use the k-mode algorithm on compressed datasets. On the other hand, Latent Semantic Analysis (LSA) [32], Latent Dirichlet Allocation (LDA) [33], Non-negative Matrix Factorisation (NNMF) [34], Principal component analysis (PCA), and Multiple Correspondence Analysis (MCA) [28] generate real-valued sketches. For these we used the k-means algorithm (available in the `sklearn` library [10]) on the compressed datasets. We set `random_state = 42` for both k-mode and k-means.

We evaluated the clustering outputs using *purity index*. Let m be the number of data points and $\Omega = \{\omega_1, \omega_2, \ldots, \omega_k\}$ be a set of clusters obtained on the original data. Further, let $\mathcal{C} = \{c_1, c_2, \ldots, c_k\}$ be a set of clusters obtained on reduced dimensional data. Then the *purity index* of the clusters $\mathcal{C}$ is defined as

$$\textit{purity index}(\Omega, \mathcal{C}) = \frac{1}{m} \sum_{i=1}^{k} \max_{1 \le j \le k} |\omega_i \cap c_j|.$$

3. https://github.com/Anonymus135/F-Sketch
4. https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html
5. https://scikit-learn.org/stable/modules/classes.html#module-sklearn.decomposition
6. https://pypi.org/project/mca/
7. https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html
8. https://rdrr.io/rforge/Gifi/man/princals.html
9. https://pypi.org/project/kmodes/
10. https://scikit-learn.org/stable/modules/classes.html#module-sklearn.cluster

APPENDIX F
ERRORS DURING DIMENSIONALITY REDUCTION EXPERIMENTS

Several baselines give *out-of-memory* error or their running time is quite high on some datasets. This makes it infeasible to include them in empirical comparison on RMSE and other end tasks.

We list these errors here. OHE gives *out-of-memory* error for Brain cell dataset. HCA gives DNS errors on NYTimes and BrainCell datasets. CATPCA could only on KOS and DeliciousMIL datasets that too upto only 300 reduced dimension. Other than that it gives a DNS error. VAE gives DNS errors on Enron datasets. KT gives *out-of-memory* error for NYTimes and Brain cell and on Enron it didn't stop even after 10 hrs. MCA also gives *out-of-memory* error for NYTimes and Brain cell datasets. Further, the dimensionality reduction time for NNMF was quite high – on NYTimes it takes around 20 hrs to do the dimensionality reduction for 3000 dimension, and on the Brain cell dataset, NNMF didn't stop even after 10 hrs. These errors prevented us from performing dimensionality reduction for all dimension using some of the algorithms.

APPENDIX G
EXTENDED EXPERIMENTAL RESULTS

This section contains the remaining comparative plots for the RMSE (Figure 11), clustering (Figure 12), similarity search experiments (Figure 13) and the dimensionality reduction time (Figure 14).

APPENDIX H
`Median-FSketch`: COMBINING MULTIPLE `FSketch`

We proved in Lemma 8 that our estimate $\hat{h}$ is within an additive error of h. A standard approach to improve the accuracy in such situations is to obtain several independent estimates and then compute a suitable statistic of the estimates. We were faced with a choice of mean, median and minimum of the estimates of which we decided to choose median after extensive empirical evaluation (see Section H.3) and obtaining theoretical justification (explained in Section H.2). We first explain our algorithms in the next subsection.

H.1 Algorithms for generating a sketch and estimaing Hamming distance

Let k, d be some suitably chosen integer parameters. An arity-k dimension-d `Median-FSketch` for a categorical data, say x, is an array of k sketches: $\Phi(x) = \langle \phi^1(x), \phi^2(x), \ldots \phi^k(x) \rangle$; the i-th entry of $\Phi(x)$ is a d-dimensional `FSketch`. See Figure 15 for an illustration. Note that the internal parameters ρ, R, p required to run `FSketch` to obtain the i-entry are same across all data points; the parameters corresponding to different i are, however, chosen independently (p can be the same).

Our algorithm for Hamming distance estimation is inspired from the Count-Median sketch [56] and Count sketch [57]. It estimates the Hamming distances between the pairs of "rows" from $\Phi(x)$ and $\Phi(y)$ and returns the median

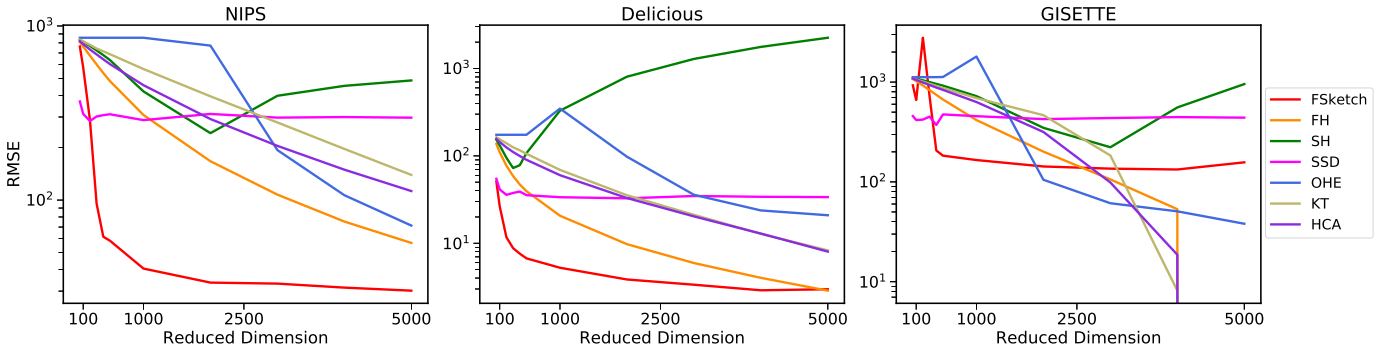


Fig. 11. Comparison of RMSE values. A lower value is an indication of better performance. The GISETTE dataset is of 5000 dimensions and hence, FSketch suffers from an increase in RMSE as the embedding dimension also reaches 5000.

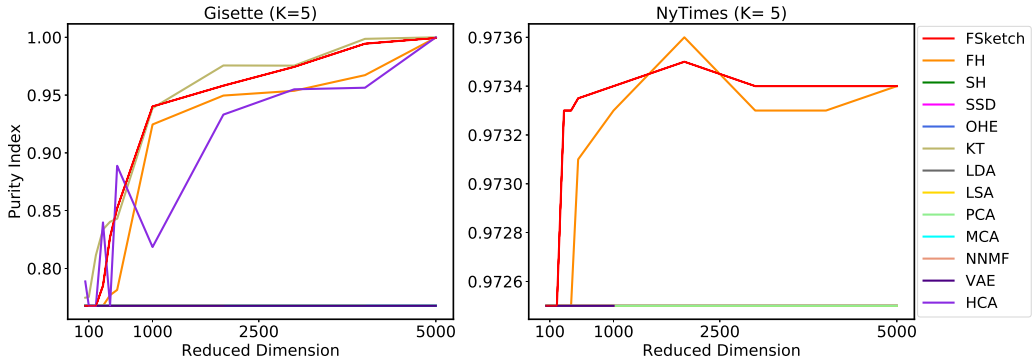


Fig. 12. Comparing the quality of clusters on the compressed datasets.

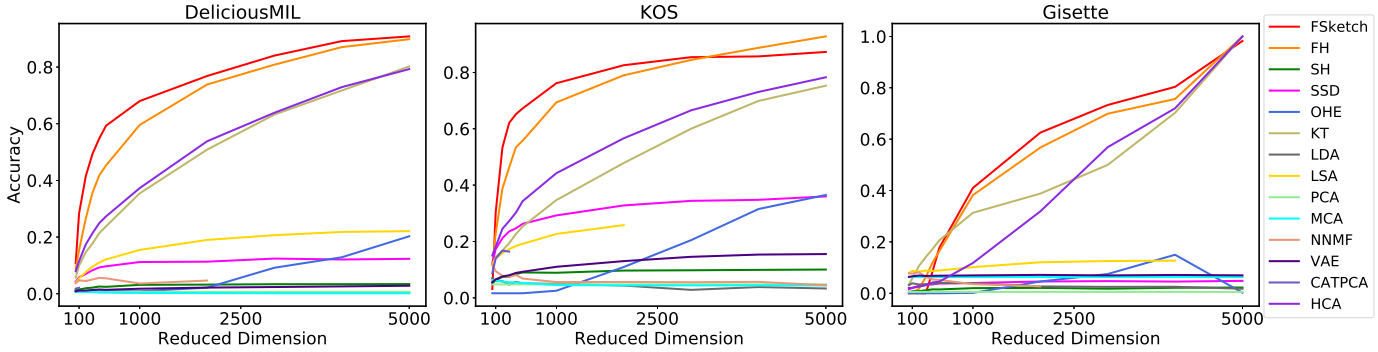


Fig. 13. Comparing the performance of the similarity search task (estimating top-k similar points with $k = 100$) achieved on the reduced dimensional data obtained from various baselines.

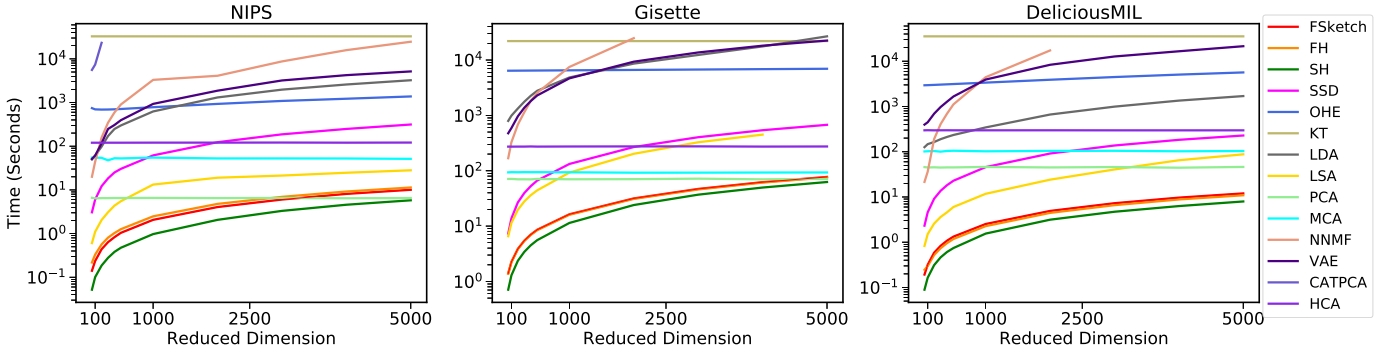


Fig. 14. Comparison of the dimensionality reduction times.

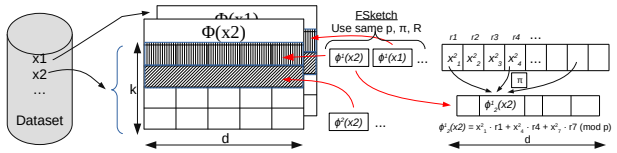


Fig. 15. `Median-FSketch` for categorical data — sketch of each data point is a 2-dimensional array whose each row is an `FSketch`. The i-th rows corresponding to all the data points use the same values of ρ, R.

of the estimated distances. This procedure is followed in Algorithm 3.

Algorithm 3 Estimate Hamming distance between x and y from their `Median-FSketch`

Input: $\Phi(x) = \langle \phi^1(x), \phi^2(x), \ldots \phi^k(x) \rangle$, $\Phi(y) = \langle \phi^1(y), \phi^2(y), \ldots \phi^k(y) \rangle$

1: **for** $i = 1 \ldots k$ **do**
2: Compute $f =$ Hamming distance between $\phi^i(x)$ and $\phi^i(y)$
3: If $f < dP$, $\widehat{h}^i = \ln\left(1 - \frac{f}{dP}\right) / \ln D$
4: Else $\widehat{h}^i = 2\sigma$
5: **end for**
6: **return** $\hat{h} = \min\{\widehat{h}^1, \widehat{h}^2, \ldots \widehat{h}^k\}$

H.2 Theoretical justification

We now give a proof that our `Median-FSketch` estimator offers a better approximation. Recall that σ indicates the maximum number of non-zero attributes in any data vector, and is often much small compared to the their dimension, n. Surprisingly, our results are independent of n.

Lemma 19. *Let h^m denote the median of the estimates of Hamming distances obtained from t independent `FSketch` vectors of dimension 4σ and let h denote the actual Hamming distance. Then,*

$$\Pr\left[|h^m - h| \geq 18\sqrt{\sigma}\right] \leq \delta$$

for any desired $\delta \in (0, 1)$ if we use $t \geq 48 \ln \frac{1}{\delta}$.

Proof. We start by using Lemma 8 with $p = 3$ and error (δ in the lemma statement) = $\frac{1}{4}$. Let $\hat{h}^i$ denote the k-th estimate. From the lemma we get that

$$\Pr\left[|\hat{h}^i - h| \geq 18\sqrt{\sigma}\right] \leq \tfrac{1}{4}$$

Define indicator random variables $W_1 \ldots W_t$ as $W_i = 1$ iff $|\hat{h}^i - h| \geq 18\sqrt{\sigma}$. We immediately have $\Pr[W_i] \leq \frac{1}{4}$. Notice that $W_i = 1$ can also be interpreted to indicate the event $h - 18\sqrt{\sigma} \leq \hat{h}^i \leq h + 18\sqrt{\sigma}$. Now, h^m is the median of $\{\hat{h}^1, \hat{h}^2, \ldots \hat{h}^t\}$, and so, h^m falls outside the range $[h - 18\sqrt{\sigma}, h + 18\sqrt{\sigma}]$ only if more than half of the estimates fall outside this range., i.e., if $\sum_{i=1}^{t} W_i > t/2$. Since $\mathbb{E}[\sum_i W_i] \leq t/4$, the probability of this event is easily bounded by $\exp\left(-(\frac{1}{2}^2 \frac{t}{4}/3)\right) = e^{-t/48} \leq \delta$ using Chernoff's bound. □

H.3 Choice of statistics in `Median-FSketch`

We conducted an experiment to decide whether to take median, mean or minimum of k `FSketch` estimates in the `Median-FSketch` algorithm. We randomly sampled a pair of points and estimated the Hamming distance from its low-dimensional representation obtained from `FSketch`. We repeated this 10 times over different random mappings and computed the median, mean, and minimum of those 10 different estimates. We further repeat this experiment 10 times and generate a box-plot of the readings which is presented in Figure 16. We observe that median has the lowest variance and also closely estimates the actual Hamming distance between the pair of points.

APPENDIX I
DIMENSIONALITY REDUCTION ALGORITHMS

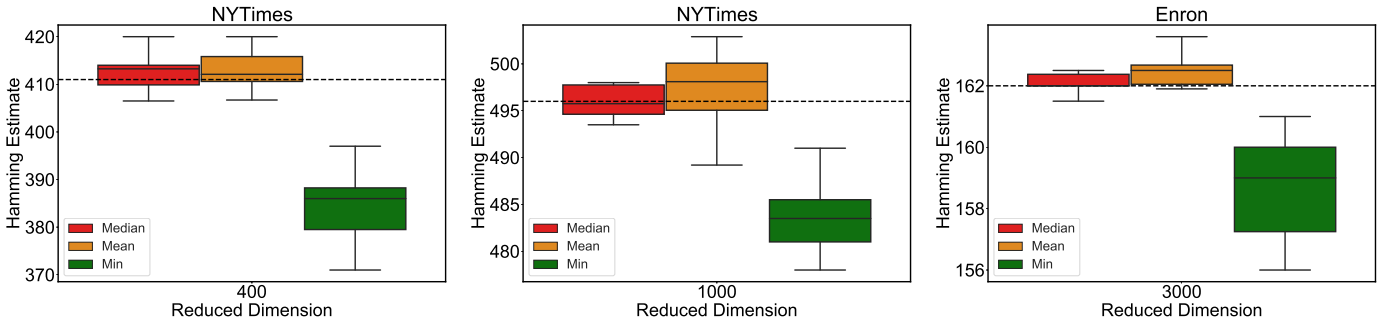


Fig. 16. Box plot for the median, mean, and minimum of the `FSketch`'s estimate obtain *via* it's from its 10 repetitions, then each experiment is repeated 10 times for computing the variance of these statistics. The black dotted line corresponds to the actual Hamming distance.

TABLE 8
A tabular summary of popular dimensionality reduction algorithms. Linear dimensionality reduction algorithms are those whose features in reduced dimension are linear combinations of the input features, and the others are known as non-linear algorithms. Supervised dimensionality reduction methods are those that require labelled datasets for dimensionality reduction.

S. No.	Data type of input vectors	Objective/ Properties	Data type of sketch vectors	Result	Supervised or Unsupervised	Type of dimensionality reduction
1	Real-valued vectors	Approximating pairwise euclidean distance, inner product	Real-valued vectors	JL-lemma [23]	Unsupervised	Linear
2	Real-valued vectors	Approximating pairwise euclidean distance, inner product	Real-valued vectors	Feature Hashing [20]	Unsupervised	Linear
3	Real-valued vectors	Approximating pairwise cosine or angular similarity	Binary vectors	SimHash [21]	Unsupervised	Non-Linear
4	Real-valued vectors	Approximating pairwise ℓ_p norm for $p \in (0, 2]$	Real-valued vectors	p-stable random projection (SSD) [58]	Unsupervised	Linear
5	Sets	Approximating pairwise Jaccard similarity	Integer valued vectors	MinHash [30]	Unsupervised	Non-linear
6	Sparse binary vectors	Approximating pairwise Hamming distance, Inner product, Jaccard and Cosine similarity	Binary vectors	BinSketch [17]	Unsupervised	Non-linear
7	Real-valued vectors	Minimize the variance in low dimension	Real-valued vectors	Principal Component Analysis (PCA)	Unsupervised	Linear
8	Real-valued vectors (labelled input)	Maximizes class separability in the reduced dimensional space	Real-valued vectors	Linear Discriminant Analysis [59]	Supervised	Linear
9	Real-valued vectors	Embedding high-dimensional data for visualization in a low-dimensional space of two or three dimensions	Real-valued vectors	t-SNE [36]	Unsupervised	Non-linear
10	Real-valued vectors	Minimize the reconstruction error	Real-valued vectors	Auto-encoder [60]	Unsupervised	Non-linear
11	Real-valued vectors	Extracting nonlinear structures in low-dimension via Kernel function	Real-valued vectors	Kernel-PCA [61]	Unsupervised	Non-linear
12	Real-valued vectors	Factorize input matrix into two small size non-negative matrices	Real-valued vectors	Non-negative matrix factorization (NNMF) [34]	Unsupervised	Linear
13	Real-valued vectors	Compute a *quasi-isometric* low-dimensional embedding	Real-valued vectors	Isomap [62]	Unsupervised	Non-linear
14	Real-valued vectors	Preserves the *topological structure* of the data	Real-valued vectors	Self-organizing map [63]	Unsupervised	Non-linear