Dimension Reduction with Certainty

Rasmus Pagh
IT University of Copenhagen

ECML-PKDD
September 21, 2016

Slides: goo.gl/hZoRWo

European Research Council
Established by the European Commission

SCALABLE SIMILARITY SEARCH

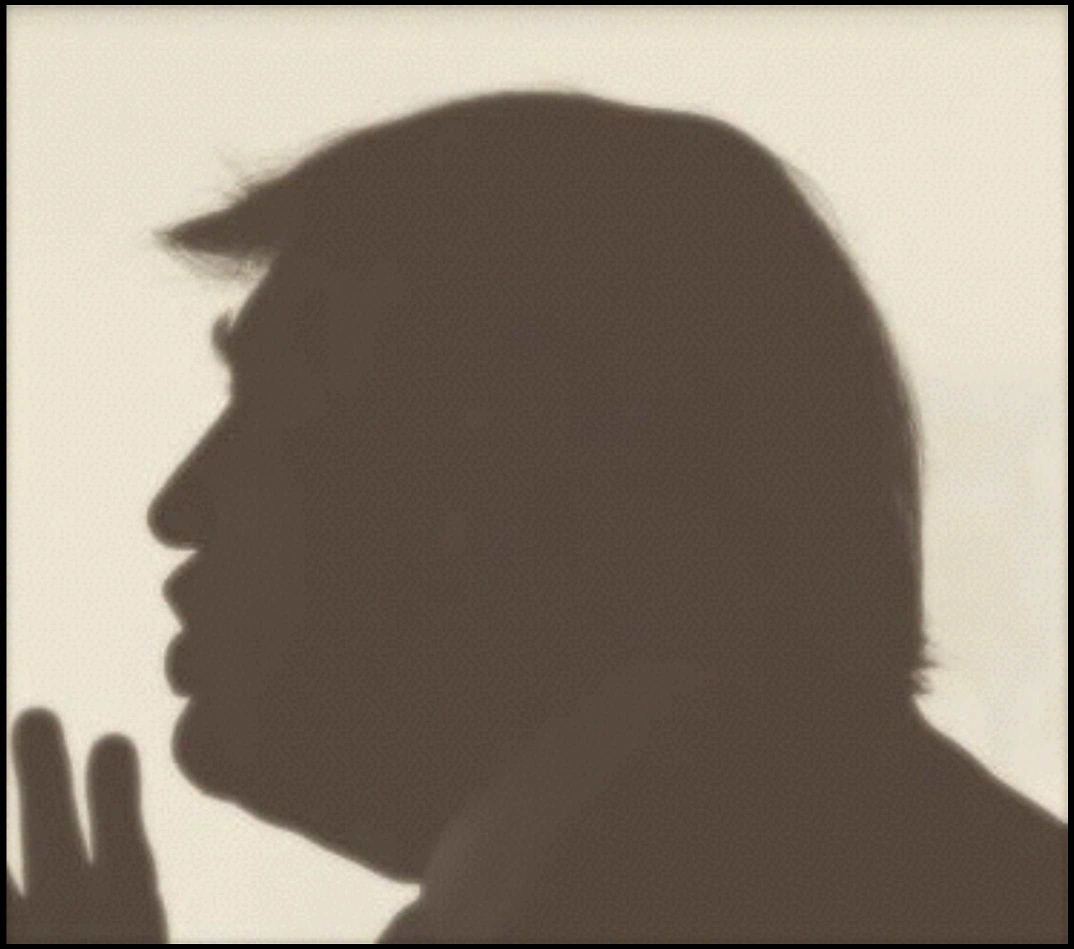


Photo by Charlie Neibergall - AP

What can we say about high-dimensional objects from a low-dimensional representation?

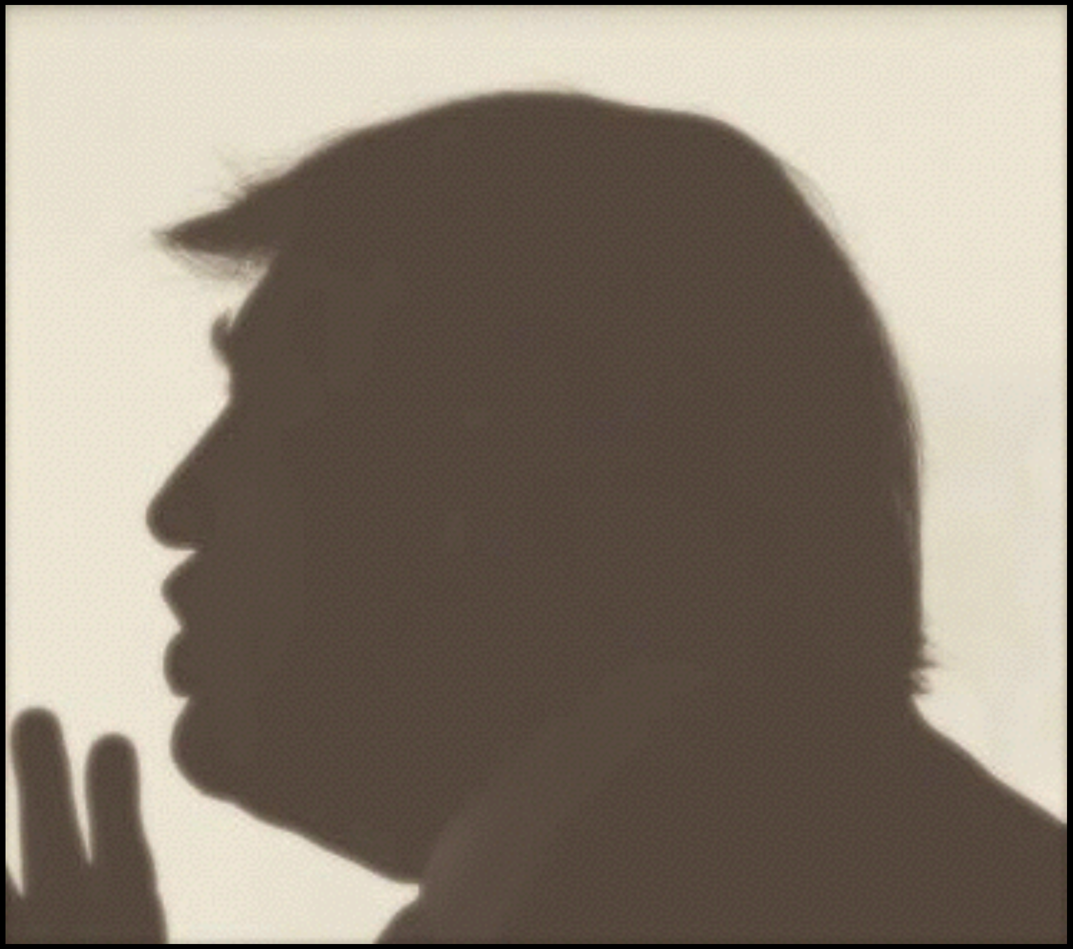


Photo by Charlie Neibergall - AP

What can we say *with certainty* *about high-dimensional objects from a low-dimensional representation?*

Outline

Part I:

Tools for randomized dimension reduction - greatest hits

Part II:

Transparency and interpretability

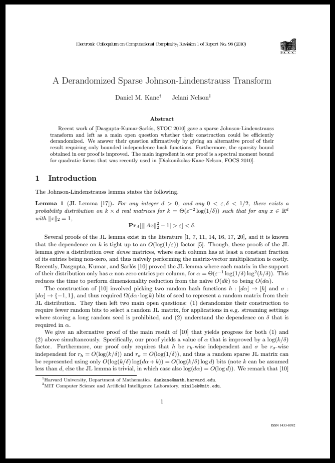
Part III:

Dimension reduction with certainty?

Dimension reduction

Technique for mapping objects from a *large space* into a *small space,* while preserving essential relations.

x



Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 98 (2010)

A Derandomized Sparse Johnson-Lindenstrauss Transform

Daniel M. Kane[†] Jelani Nelson[‡]

Abstract

Recent work of [Dasgupta-Kumar-Sarlós, STOC 2010] gave a sparse Johnson-Lindenstrauss transform and left as a main open question whether their construction could be efficiently derandomized. We answer their question affirmatively by giving an alternative proof of their result requiring only bounded independence hash functions. Furthermore, the sparsity bound obtained in our proof is improved. The main ingredient in our proof is a spectral moment bound for quadratic forms that was recently used in [Diakonikolas-Kane-Nelson, FOCS 2010].

1 Introduction

The Johnson-Lindenstrauss lemma states the following.

Lemma 1 (JL Lemma [17]). *For any integer $d > 0$, and any $0 < \varepsilon, \delta < 1/2$, there exists a probability distribution on $k \times d$ real matrices for $k = \Theta(\varepsilon^{-2}\log(1/\delta))$ such that for any $x \in \mathbb{R}^d$ with $\|x\|_2 = 1$,*

$$\mathbf{Pr}_A[|\|Ax\|_2^2 - 1| > \varepsilon] < \delta.$$

Several proofs of the JL lemma exist in the literature [1, 7, 11, 14, 16, 17, 20], and it is known that the dependence on k is tight up to an $O(\log(1/\varepsilon))$ factor [5]. Though, these proofs of the JL lemma give a distribution over *dense* matrices, where each column has at least a constant fraction of its entries being non-zero, and thus naïvely performing the matrix-vector multiplication is costly. Recently, Dasgupta, Kumar, and Sarlós [10] proved the JL lemma where each matrix in the support of their distribution only has α non-zero entries per column, for $\alpha = \Theta(\varepsilon^{-1}\log(1/\delta)\log^2(k/\delta))$. This reduces the time to perform dimensionality reduction from the naïve $O(dk)$ to being $O(d\alpha)$.

The construction of [10] involved picking two random hash functions $h : [d\alpha] \to [k]$ and $\sigma : [d\alpha] \to \{-1, 1\}$, and thus required $\Omega(d\alpha \cdot \log k)$ bits of seed to represent a random matrix from their JL distribution. They then left two main open questions: (1) derandomize their construction to require fewer random bits to select a random JL matrix, for applications in e.g. streaming settings where storing a long random seed is prohibited, and (2) understand the dependence on δ that is required in α.

We give an alternative proof of the main result of [10] that yields progress for both (1) and (2) above simultaneously. Specifically, our proof yields a value of α that is improved by a $\log(k/\delta)$ factor. Furthermore, our proof only requires that h be r_h-wise independent and σ be r_σ-wise independent for $r_h = O(\log(k/\delta))$ and $r_\sigma = O(\log(1/\delta))$, and thus a random sparse JL matrix can be represented using only $O(\log(k/\delta)\log(d\alpha + k)) = O(\log(k/\delta)\log d)$ bits (note k can be assumed less than d, else the JL lemma is trivial, in which case also $\log(d\alpha) = O(\log d)$). We remark that [10]

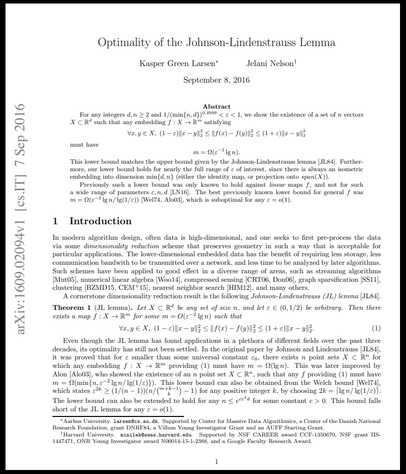
[1] Harvard University, Department of Mathematics. dankane@math.harvard.edu.

[2] MIT Computer Science and Artificial Intelligence Laboratory. minilek@mit.edu.

ISSN 1433-8092

$\hat{x}$

→ **0010111011101010101001010**



Optimality of the Johnson-Lindenstrauss Lemma

Kasper Green Larsen* Jelani Nelson[†]

September 8, 2016

Abstract

For any integers $d, n \geq 2$ and $1/(\min\{n, d\})^{0.4999} < \varepsilon < 1$, we show the existence of a set of n vectors $X \subset \mathbb{R}^d$ such that any embedding $f : X \to \mathbb{R}^m$ satisfying

$$\forall x, y \in X,\ (1-\varepsilon)\|x-y\|_2^2 \leq \|f(x) - f(y)\|_2^2 \leq (1+\varepsilon)\|x-y\|_2^2$$

must have

$$m = \Omega(\varepsilon^{-2}\lg n).$$

This lower bound matches the upper bound given by the Johnson-Lindenstrauss lemma [JL84]. Furthermore, our lower bound holds for nearly the full range of ε of interest, since there is always an isometric embedding into dimension $\min\{d, n\}$ (either the identity map, or projection onto *span*(X)).

Previously such a lower bound was only known to hold against *linear* maps f, and not for such a wide range of parameters ε, n, d [LN16]. The best previously known lower bound for general f was $m = \Omega(\varepsilon^{-2}\lg n/\lg(1/\varepsilon))$ [Wel74, Alo03], which is suboptimal for any $\varepsilon = o(1)$.

1 Introduction

In modern algorithm design, often data is high-dimensional, and one seeks to first pre-process the data via some *dimensionality reduction* scheme that preserves geometry in such a way that is acceptable for particular applications. The lower-dimensional embedded data has the benefit of requiring less storage, less communication bandwith to be transmitted over a network, and less time to be analyzed by later algorithms. Such schemes have been applied to good effect in a diverse range of areas, such as streaming algorithms [Mut05], numerical linear algebra [Woo14], compressed sensing [CRT06, Don06], graph sparsification [SS11], clustering [BZMD15, CEM+15], nearest neighbor search [HIM12], and many others.

A cornerstone dimensionality reduction result is the following *Johnson-Lindenstrauss (JL) lemma* [JL84].

Theorem 1 (JL lemma). *Let $X \subset \mathbb{R}^d$ be any set of size n, and let $\varepsilon \in (0, 1/2)$ be arbitrary. Then there exists a map $f : X \to \mathbb{R}^m$ for some $m = O(\varepsilon^{-2}\lg n)$ such that*

$$\forall x, y \in X,\ (1-\varepsilon)\|x-y\|_2^2 \leq \|f(x) - f(y)\|_2^2 \leq (1+\varepsilon)\|x-y\|_2^2. \quad (1)$$

Even though the JL lemma has found applications in a plethora of different fields over the past three decades, its optimality has still not been settled. In the original paper by Johnson and Lindenstrauss [JL84], it was proved that for ε smaller than some universal constant ε_0, there exists n point sets $X \subset \mathbb{R}^n$ for which any embedding $f : X \to \mathbb{R}^m$ providing (1) must have $m = \Omega(\lg n)$. This was later improved by Alon [Alo03], who showed the existence of an n point set $X \subset \mathbb{R}^n$, such that any f providing (1) must have $m = \Omega(\min\{n, \varepsilon^{-2}\lg n/\lg(1/\varepsilon)\})$. This lower bound can also be obtained from the Welch bound [Wel74], which states $\varepsilon^{2k} \geq (1/(n-1))(n/\binom{m+k-1}{k} - 1)$ for any positive integer k, by choosing $2k = \lceil \lg n/\lg(1/\varepsilon) \rceil$. The lower bound can also be extended to hold for any $n \leq e^{c\varepsilon^2 d}$ for some constant $c > 0$. This bound falls short of the JL lemma for any $\varepsilon = o(1)$.

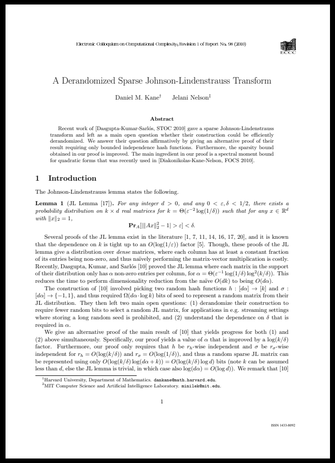
*Aarhus University. larsen@cs.au.dk. Supported by Center for Massive Data Algorithmics, a Center of the Danish National Research Foundation, grant DNRF84, a Villum Young Investigator Grant and an AUFF Starting Grant.

[†] Harvard University. minilek@seas.harvard.edu. Supported by NSF CAREER award CCF-1350670, NSF grant IIS-1447471, ONR Young Investigator award N00014-15-1-2388, and a Google Faculty Research Award.

→ **0110111010101010101101010**

Oblivious dimension reduction

Technique for mapping objects from a *large space* into a *small space,* while preserving essential relations, that is *data-independent* and does not need to be trained.



Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 98 (2010)

ECCC

A Derandomized Sparse Johnson-Lindenstrauss Transform

Daniel M. Kane[†] Jelani Nelson[‡]

Abstract

Recent work of [Dasgupta-Kumar-Sarlós, STOC 2010] gave a sparse Johnson-Lindenstrauss transform and left as a main open question whether their construction could be efficiently derandomized. We answer their question affirmatively by giving an alternative proof of their result requiring only bounded independence hash functions. Furthermore, the sparsity bound obtained in our proof is improved. The main ingredient in our proof is a spectral moment bound for quadratic forms that was recently used in [Diakonikolas-Kane-Nelson, FOCS 2010].

1 Introduction

The Johnson-Lindenstrauss lemma states the following.

Lemma 1 (JL Lemma [17]). *For any integer $d > 0$, and any $0 < \varepsilon, \delta < 1/2$, there exists a probability distribution on $k \times d$ real matrices for $k = \Theta(\varepsilon^{-2} \log(1/\delta))$ such that for any $x \in \mathbb{R}^d$ with $\|x\|_2 = 1$,*

$$\mathbf{Pr}_A[|\|Ax\|_2^2 - 1| > \varepsilon] < \delta.$$

Several proofs of the JL lemma exist in the literature [1, 7, 11, 14, 16, 17, 20], and it is known that the dependence on k is tight up to an $O(\log(1/\varepsilon))$ factor [5]. Though, these proofs of the JL lemma give a distribution over *dense* matrices, where each column has at least a constant fraction of its entries being non-zero, and thus naïvely performing the matrix-vector multiplication is costly. Recently, Dasgupta, Kumar, and Sarlós [10] proved the JL lemma where each matrix in the support of their distribution only has α non-zero entries per column, for $\alpha = \Theta(\varepsilon^{-1} \log(1/\delta) \log^2(k/\delta))$. This reduces the time to perform dimensionality reduction from the naïve $O(dk)$ to being $O(d\alpha)$.

The construction of [10] involved picking two random hash functions $h : [d\alpha] \to [k]$ and $\sigma : [d\alpha] \to \{-1, 1\}$, and thus required $\Omega(d\alpha \cdot \log k)$ bits of seed to represent a random matrix from their JL distribution. They then left two main open questions: (1) derandomize their construction to require fewer random bits to select a random JL matrix, for applications in e.g. streaming settings where storing a long random seed is prohibited, and (2) understand the dependence on δ that is required in α.

We give an alternative proof of the main result of [10] that yields progress for both (1) and (2) above simultaneously. Specifically, our proof yields a value of α that is improved by a $\log(k/\delta)$ factor. Furthermore, our proof only requires that h be r_h-wise independent and σ be r_σ-wise independent for $r_h = O(\log(k/\delta))$ and $r_\sigma = O(\log(1/\delta))$, and thus a random sparse JL matrix can be represented using only $O(\log(k/\delta) \log(d\alpha + k)) = O(\log(k/\delta) \log d)$ bits (note k can be assumed less than d, else the JL lemma is trivial, in which case also $\log(d\alpha) = O(\log d)$). We remark that [10]

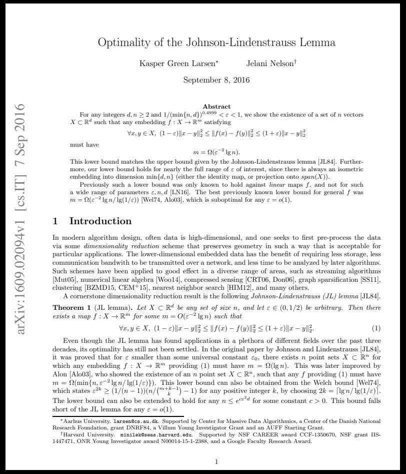
[1] Harvard University, Department of Mathematics. dankane@math.harvard.edu.

[2] MIT Computer Science and Artificial Intelligence Laboratory. minilek@mit.edu.

1

ISSN 1433-8092

→ 0010111011101010101001010



arXiv:1609.02094v1 [cs.IT] 7 Sep 2016

Optimality of the Johnson-Lindenstrauss Lemma

Kasper Green Larsen[*] Jelani Nelson[†]

September 8, 2016

Abstract

For any integers $d, n \geq 2$ and $1/(\min\{n, d\})^{0.4999} < \varepsilon < 1$, we show the existence of a set of n vectors $X \subset \mathbb{R}^d$ such that any embedding $f : X \to \mathbb{R}^m$ satisfying

$$\forall x, y \in X,\ (1 - \varepsilon)\|x - y\|_2^2 \leq \|f(x) - f(y)\|_2^2 \leq (1 + \varepsilon)\|x - y\|_2^2$$

must have

$$m = \Omega(\varepsilon^{-2} \lg n).$$

This lower bound matches the upper bound given by the Johnson-Lindenstrauss lemma [JL84]. Furthermore, our lower bound holds for nearly the full range of ε of interest, since there is always an isometric embedding into dimension $\min\{d, n\}$ (either the identity map, or projection onto *span*(X)).

Previously such a lower bound was only known to hold against *linear* maps f, and not for such a wide range of parameters ε, n, d [LN16]. The best previously known lower bound for general f was $m = \Omega(\varepsilon^{-2} \lg n / \lg(1/\varepsilon))$ [Wel74, Alo03], which is suboptimal for any $\varepsilon = o(1)$.

1 Introduction

In modern algorithm design, often data is high-dimensional, and one seeks to first pre-process the data via some *dimensionality reduction* scheme that preserves geometry in such a way that is acceptable for particular applications. The lower-dimensional embedded data has the benefit of requiring less storage, less communication bandwith to be transmitted over a network, and less time to be analyzed by later algorithms. Such schemes have been applied to good effect in a diverse range of areas, such as streaming algorithms [Mut05], numerical linear algebra [Woo14], compressed sensing [CRT06, Don06], graph sparsification [SS11], clustering [BZMD15, CEM+15], nearest neighbor search [HIM12], and many others.

A cornerstone dimensionality reduction result is the following *Johnson-Lindenstrauss (JL) lemma* [JL84].

Theorem 1 (JL lemma). *Let $X \subset \mathbb{R}^d$ be any set of size n, and let $\varepsilon \in (0, 1/2)$ be arbitrary. Then there exists a map $f : X \to \mathbb{R}^m$ for some $m = O(\varepsilon^{-2} \lg n)$ such that*

$$\forall x, y \in X,\ (1 - \varepsilon)\|x - y\|_2^2 \leq \|f(x) - f(y)\|_2^2 \leq (1 + \varepsilon)\|x - y\|_2^2. \quad (1)$$

Even though the JL lemma has found applications in a plethora of different fields over the past three decades, its optimality has still not been settled. In the original paper by Johnson and Lindenstrauss [JL84], it was proved that for ε smaller than some universal constant ε_0, there exists n point sets $X \subset \mathbb{R}^n$ for which any embedding $f : X \to \mathbb{R}^m$ providing (1) must have $m = \Omega(\lg n)$. This was later improved by Alon [Alo03], who showed the existence of an n point set $X \subset \mathbb{R}^n$, such that any f providing (1) must have $m = \Omega(\min\{n, \varepsilon^{-2} \lg n / \lg(1/\varepsilon)\})$. This lower bound can also be obtained from the Welch bound [Wel74], which states $\varepsilon^{2k} \geq (1/(n-1))(n/\binom{m+k-1}{k} - 1)$ for any positive integer k, by choosing $2k = \lceil \lg n / \lg(1/\varepsilon) \rceil$. The lower bound can also be extended to hold for any $n \leq e^{c\varepsilon^2 d}$ for some constant $c > 0$. This bound falls short of the JL lemma for any $\varepsilon = o(1)$.

[*] Aarhus University. larsen@cs.au.dk. Supported by Center for Massive Data Algorithmics, a Center of the Danish National Research Foundation, grant DNRF84, a Villum Young Investigator Grant and an AUFF Starting Grant.

[†] Harvard University. minilek@seas.harvard.edu. Supported by NSF CAREER award CCF-1350670, NSF grant IIS-1447471, ONR Young Investigator award N00014-15-1-2388, and a Google Faculty Research Award.

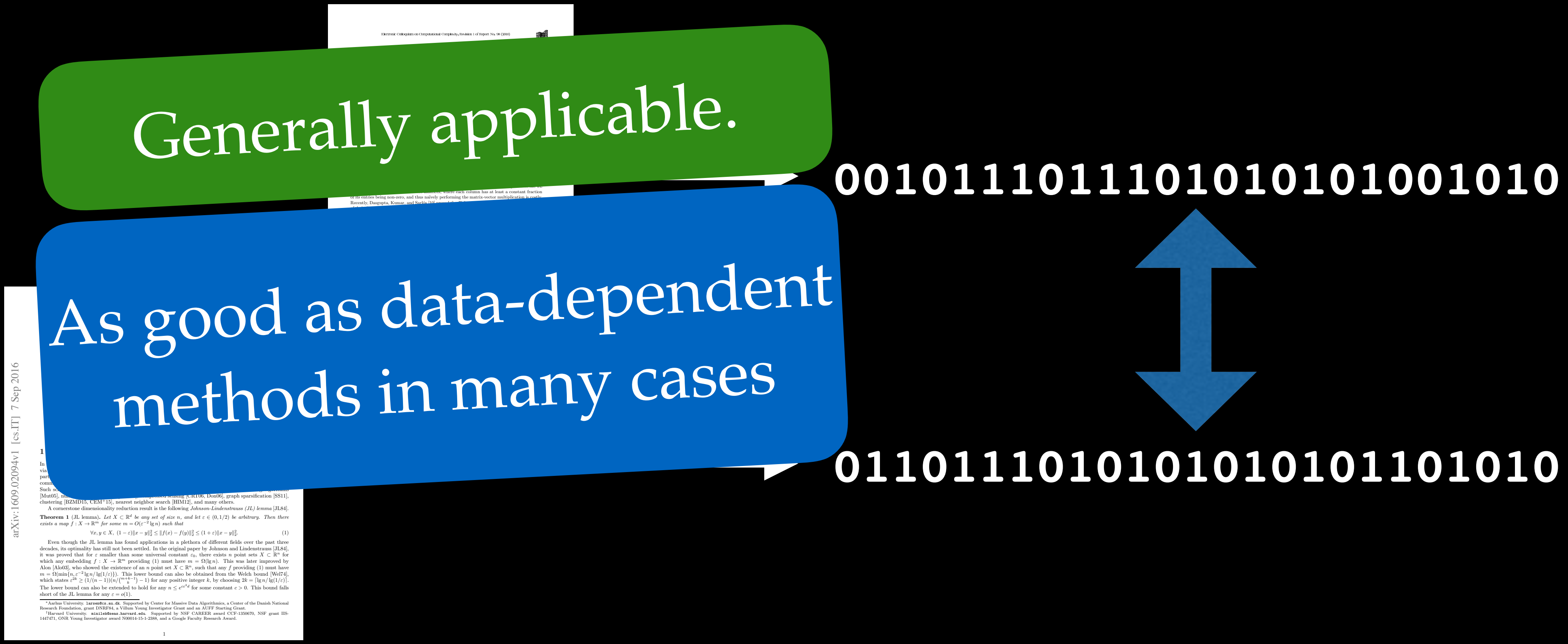
1

→ 0110111010101010101101010

↕

Oblivious dimension reduction

Technique for mapping objects from a *large space* into a *small space,* while preserving essential relations, that is *data-independent* and does not need to be trained.



Generally applicable.

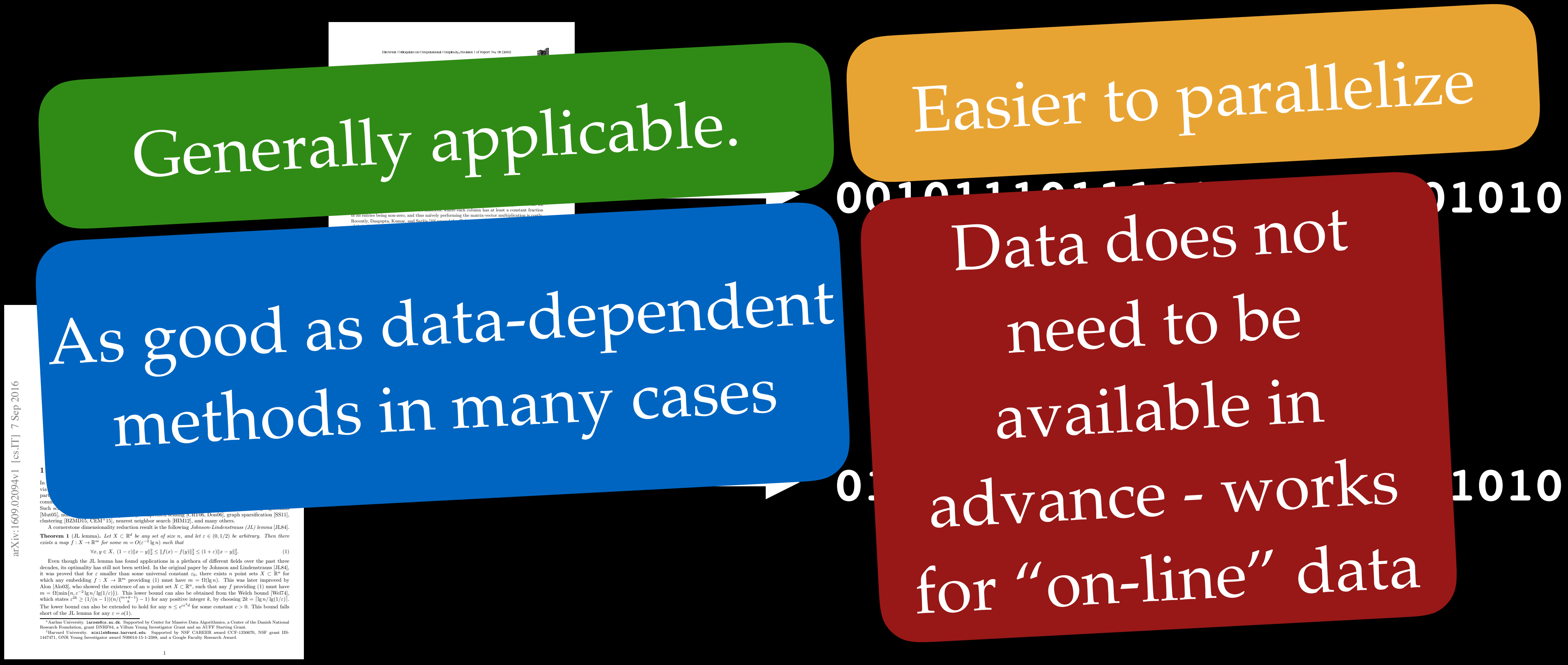
As good as data-dependent methods in many cases

0010111011101010101001010

0110111010101010101101010

Oblivious dimension reduction

Technique for mapping objects from a *large space* into a *small space,* while preserving essential relations, that is *data-independent* and does not need to be trained.



Generally applicable.

Easier to parallelize

As good as data-dependent methods in many cases

Data does not need to be available in advance - works for "on-line" data

Next: Three tools

- 🛠 Random projection
- 🛠 Random feature mapping
- 🛠 1-bit minwise hashing

Random projections

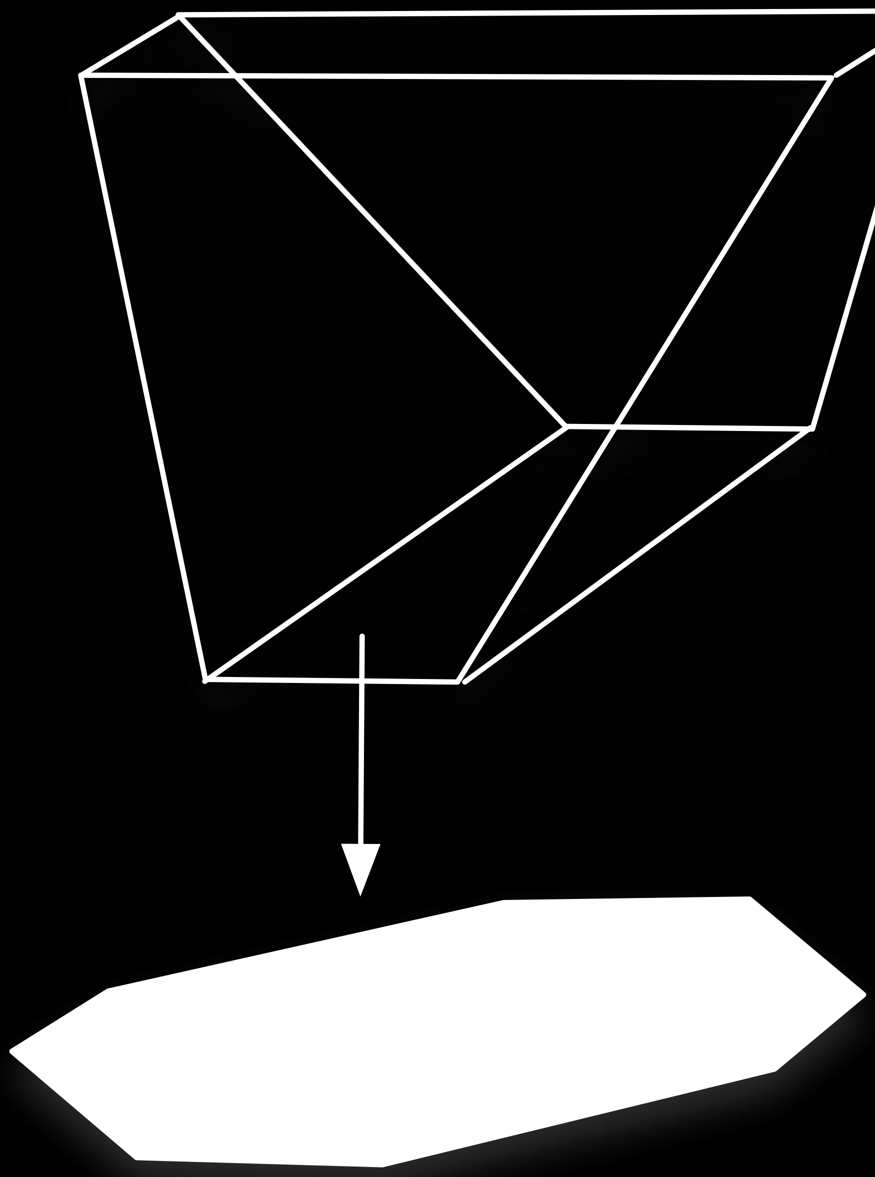


Figure courtesy of Suresh Venkatasubramanian

Photo by Giovanni Dall'Orto

[Johnson & Lindenstrauss ‘84]

Johnson-Lindenstrauss Transformation

- To preserve *n* Euclidean distances?

$$||\hat{x}_i - \hat{x}_j||_2 = (1 \pm \varepsilon)||x_i - x_j||_2$$

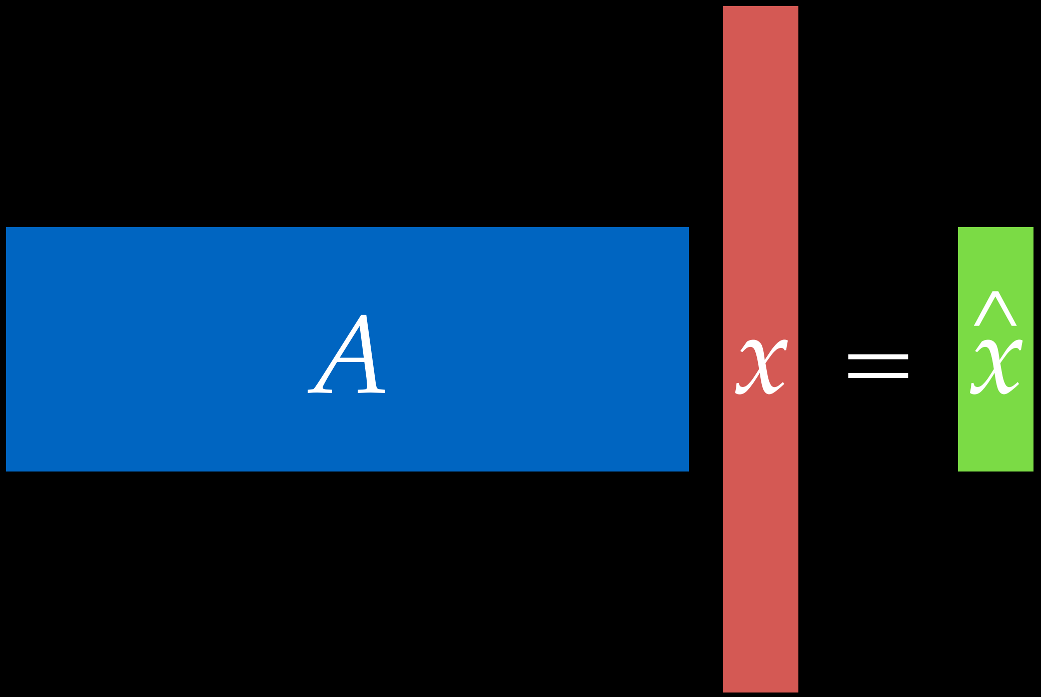
[Johnson & Lindenstrauss ‘84]

Johnson-Lindenstrauss Transformation

- To preserve n Euclidean distances?

$$||\hat{x}_i - \hat{x}_j||_2 = (1 \pm \varepsilon)||x_i - x_j||_2$$

- Use a random* linear mapping!



$$A \, x = \hat{x}$$

$m = O(\log(n)/\,\varepsilon^2)$ dimensions

[Johnson & Lindenstrauss ‘84]

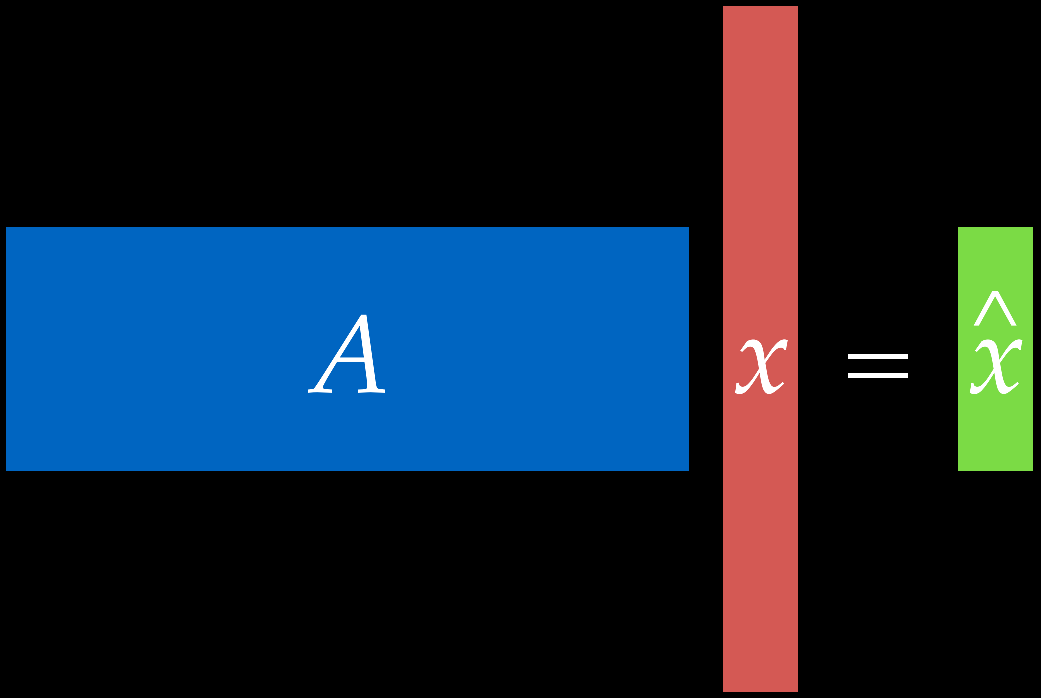
Johnson-Lindenstrauss Transformation

- To preserve *n* Euclidean distances? Dot products?

$$||\hat{x}_i - \hat{x}_j||_2 = (1 \pm \varepsilon)||x_i - x_j||_2$$

Yes, but error depends on vector lengths

- Use a random* linear mapping!



$m = O(\log(n)/\varepsilon^2)$ dimensions

[Johnson & Lindenstrauss ‘84]

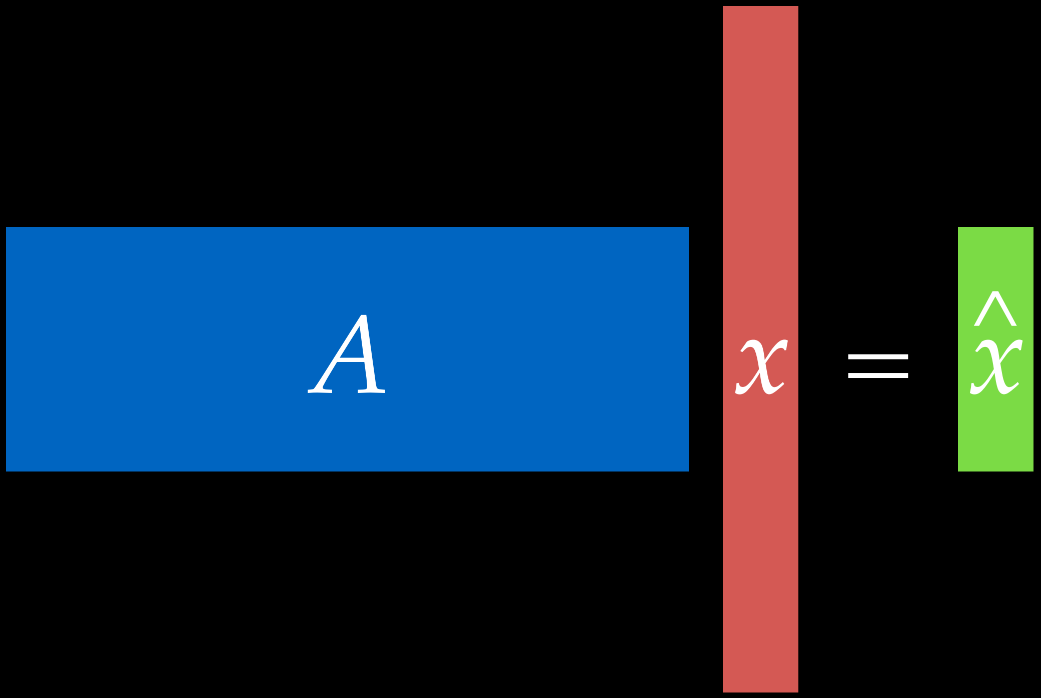
Johnson-Lindenstrauss Transformation

- To preserve n Euclidean distances? Dot products?

$$||\hat{x}_i - \hat{x}_j||_2 = (1 \pm \varepsilon)||x_i - x_j||_2$$

Yes, but error depends on vector lengths

- Use a random* linear mapping!



$m = O(\log(n)/\,\varepsilon^2)$ dimensions

Still too many dimensions!

[Johnson & Lindenstrauss '84]

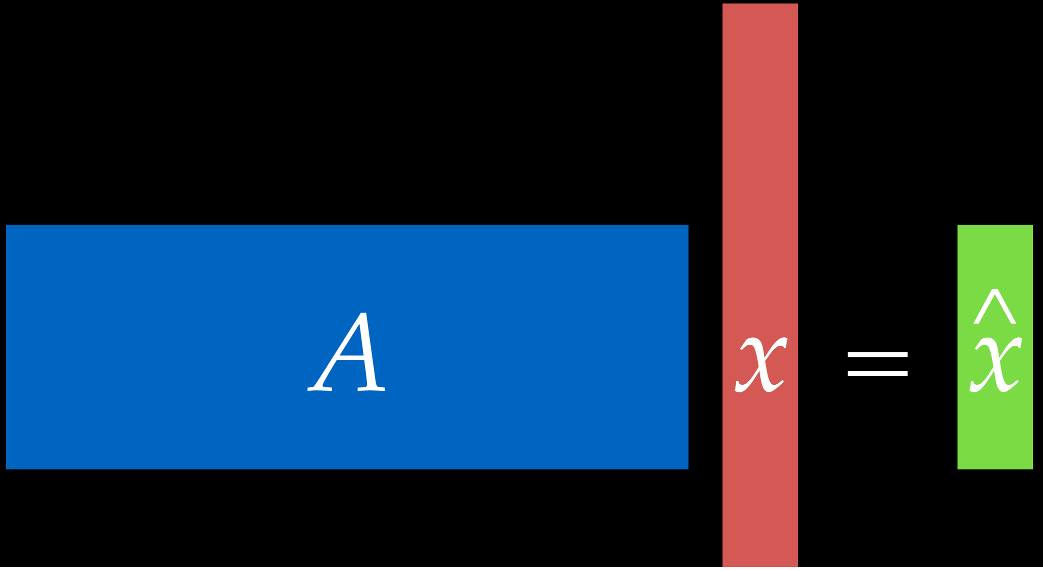
Johnson-Lindenstrauss Transformation

- To preserve *n* Euclidean distances? Dot products?

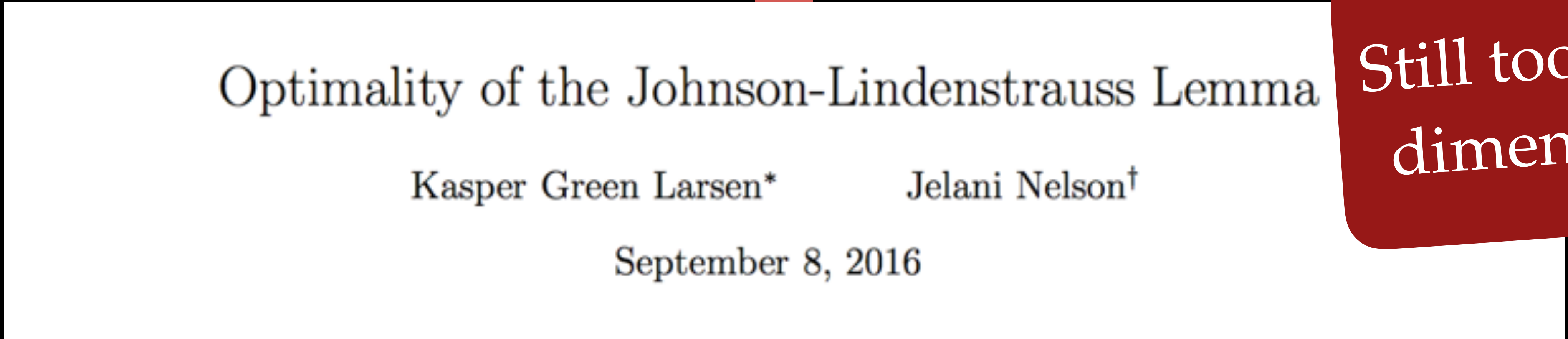
$$||\hat{x}_i - \hat{x}_j||_2 = (1 \pm \varepsilon)||x_i - x_j||_2$$

- Use a random* linear mapping!

Yes, but error depends on vector lengths



$m = O(\log(n)/\varepsilon^2)$ dimensions


Optimality of the Johnson-Lindenstrauss Lemma

Kasper Green Larsen* Jelani Nelson†

September 8, 2016

Still too many dimensions!

Oblivious subspace embeddings

- Do better if data has nice structure?
  For example constrained to d-dim. subspace.

Oblivious subspace embeddings

- Do better if data has nice structure?
  For example constrained to *d*-dim. subspace.
  - Principal component analysis (PCA) works, but mapping is *data-dependent*.

Oblivious subspace embeddings

- Do better if data has nice structure?
  For example constrained to *d*-dim. subspace.
  - Principal component analysis (PCA) works, but mapping is *data-dependent*.
  - A suitable random linear map works with $m = O(d/\varepsilon^2)$ dimensions! [Sarlós ‘06]

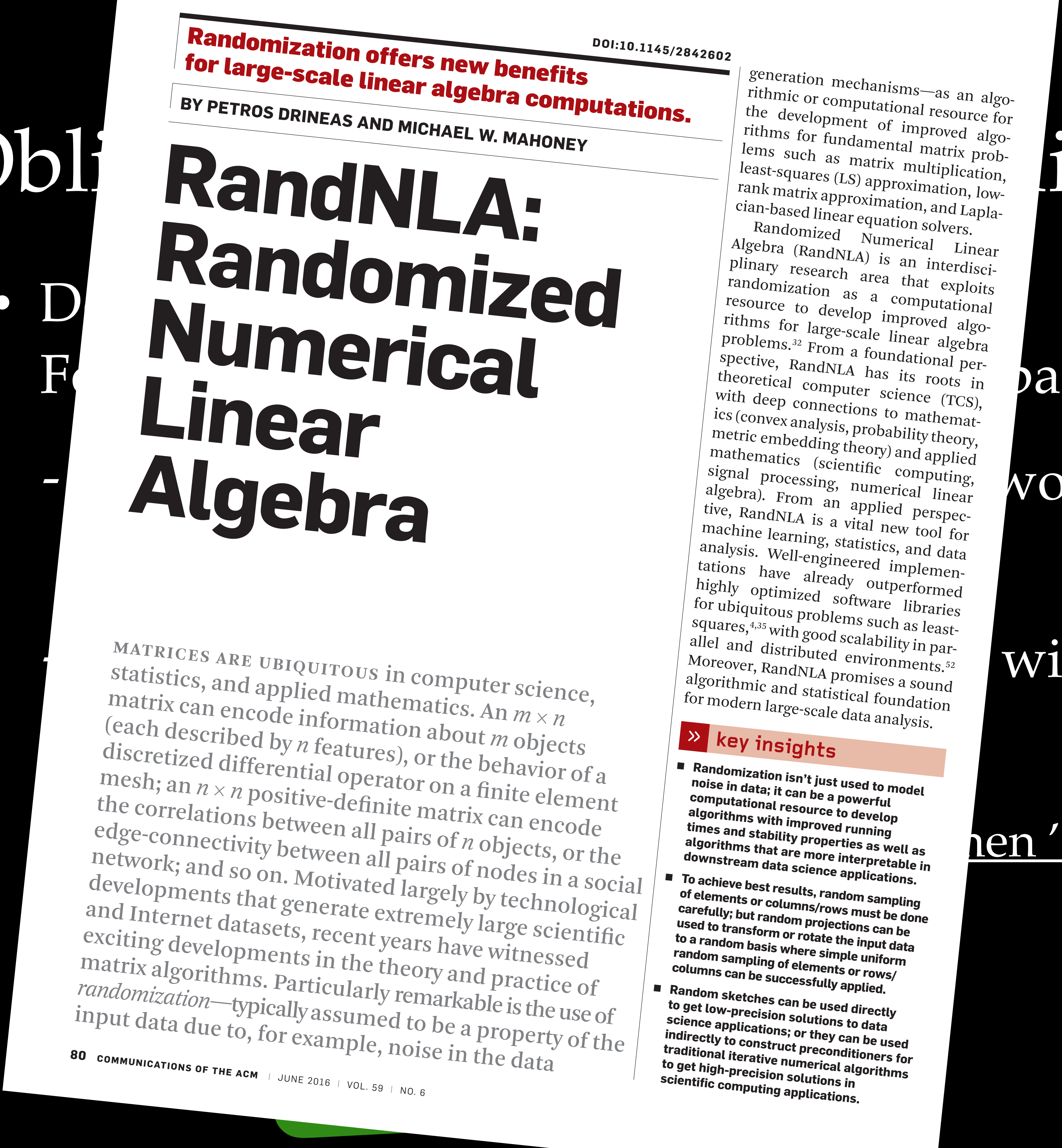
Oblivious subspace embeddings

- Do better if data has nice structure?
  For example constrained to d-dim. subspace.
  - Principal component analysis (PCA) works, but mapping is *data-dependent*.
  - A suitable random linear map works with $m = O(d / \varepsilon^2)$ dimensions! [Sarlós '06]
  - Sparse matrices almost as good [Cohen '16].

Key tool in randomized numerical linear algebra (RandNLA) methods

Obli ings

- D
  F pace.
  - works,
  with
  hen ’16].



DOI:10.1145/2842602

Randomization offers new benefits for large-scale linear algebra computations.

BY PETROS DRINEAS AND MICHAEL W. MAHONEY

RandNLA: Randomized Numerical Linear Algebra

MATRICES ARE UBIQUITOUS in computer science, statistics, and applied mathematics. An $m \times n$ matrix can encode information about m objects (each described by n features), or the behavior of a discretized differential operator on a finite element mesh; an $n \times n$ positive-definite matrix can encode the correlations between all pairs of n objects, or the edge-connectivity between all pairs of nodes in a social network; and so on. Motivated largely by technological developments that generate extremely large scientific and Internet datasets, recent years have witnessed exciting developments in the theory and practice of matrix algorithms. Particularly remarkable is the use of *randomization*—typically assumed to be a property of the input data due to, for example, noise in the data generation mechanisms—as an algorithmic or computational resource for the development of improved algorithms for fundamental matrix problems such as matrix multiplication, least-squares (LS) approximation, low-rank matrix approximation, and Laplacian-based linear equation solvers.

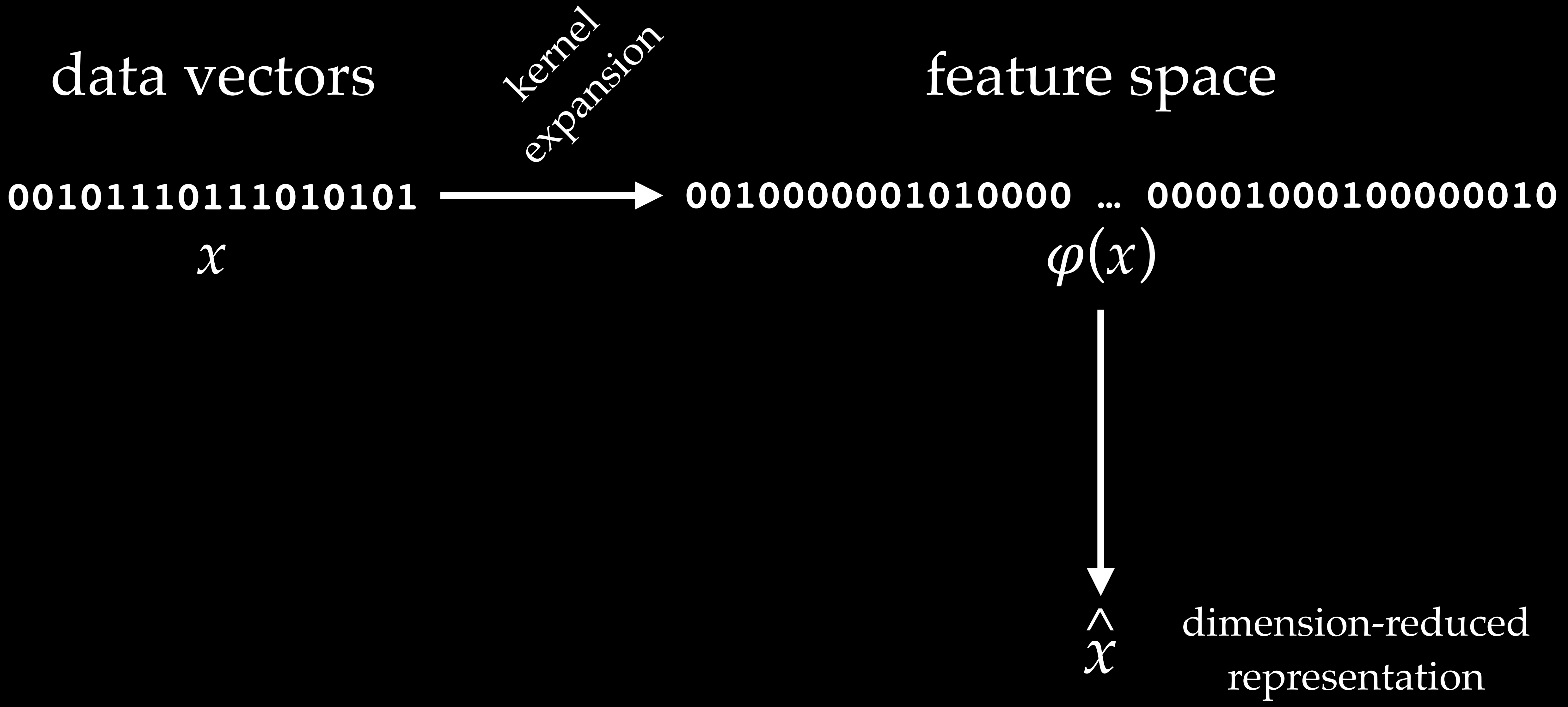
Randomized Numerical Linear Algebra (RandNLA) is an interdisciplinary research area that exploits randomization as a computational resource to develop improved algorithms for large-scale linear algebra problems.[32] From a foundational perspective, RandNLA has its roots in theoretical computer science (TCS), with deep connections to mathematics (convex analysis, probability theory, metric embedding theory) and applied mathematics (scientific computing, signal processing, numerical linear algebra). From an applied perspective, RandNLA is a vital new tool for machine learning, statistics, and data analysis. Well-engineered implementations have already outperformed highly optimized software libraries for ubiquitous problems such as least-squares,[4,35] with good scalability in parallel and distributed environments.[52] Moreover, RandNLA promises a sound algorithmic and statistical foundation for modern large-scale data analysis.

» key insights

- **Randomization isn’t just used to model noise in data; it can be a powerful computational resource to develop algorithms with improved running times and stability properties as well as algorithms that are more interpretable in downstream data science applications.**
- **To achieve best results, random sampling of elements or columns/rows must be done carefully; but random projections can be used to transform or rotate the input data to a random basis where simple uniform random sampling of elements or rows/columns can be successfully applied.**
- **Random sketches can be used directly to get low-precision solutions to data science applications; or they can be used indirectly to construct preconditioners for traditional iterative numerical algorithms to get high-precision solutions in scientific computing applications.**

[Rahimi & Recht '07]

Random feature mappings



data vectors

kernel expansion

feature space

0010111011101010 1

x

0010000001010000 … 00001000100000010

$\varphi(x)$

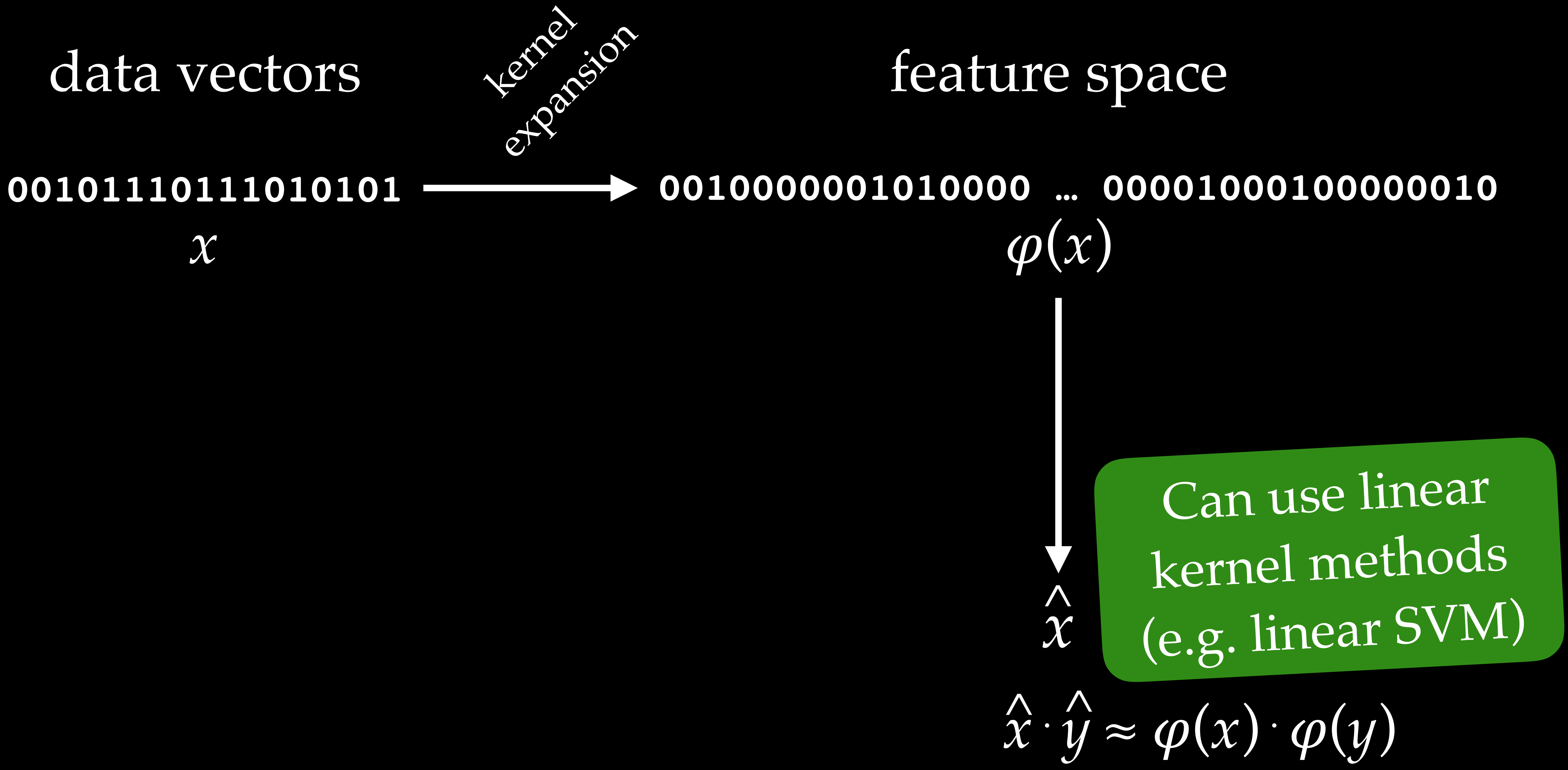
$\hat{x}$

dimension-reduced representation

$$\hat{x} \cdot \hat{y} \approx \varphi(x) \cdot \varphi(y)$$

[Rahimi & Recht '07]

Random feature mappings



data vectors

kernel expansion

feature space

`0010111011101010` → `0010000001010000 … 0000100010000010`

x → $\varphi(x)$

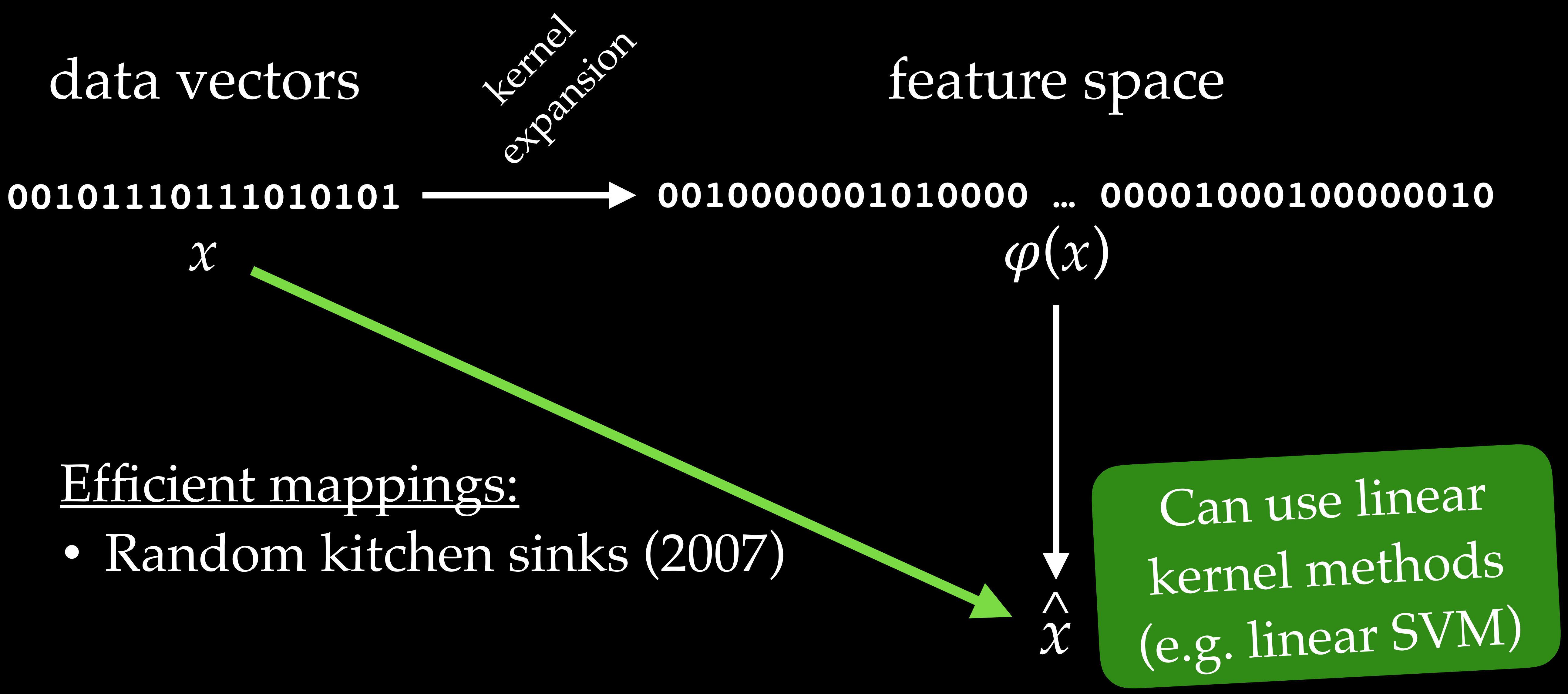
$\varphi(x)$ → $\hat{x}$

Can use linear kernel methods (e.g. linear SVM)

$$\hat{x} \cdot \hat{y} \approx \varphi(x) \cdot \varphi(y)$$

[Rahimi & Recht ‘07]

Random feature mappings



Can use linear kernel methods (e.g. linear SVM)

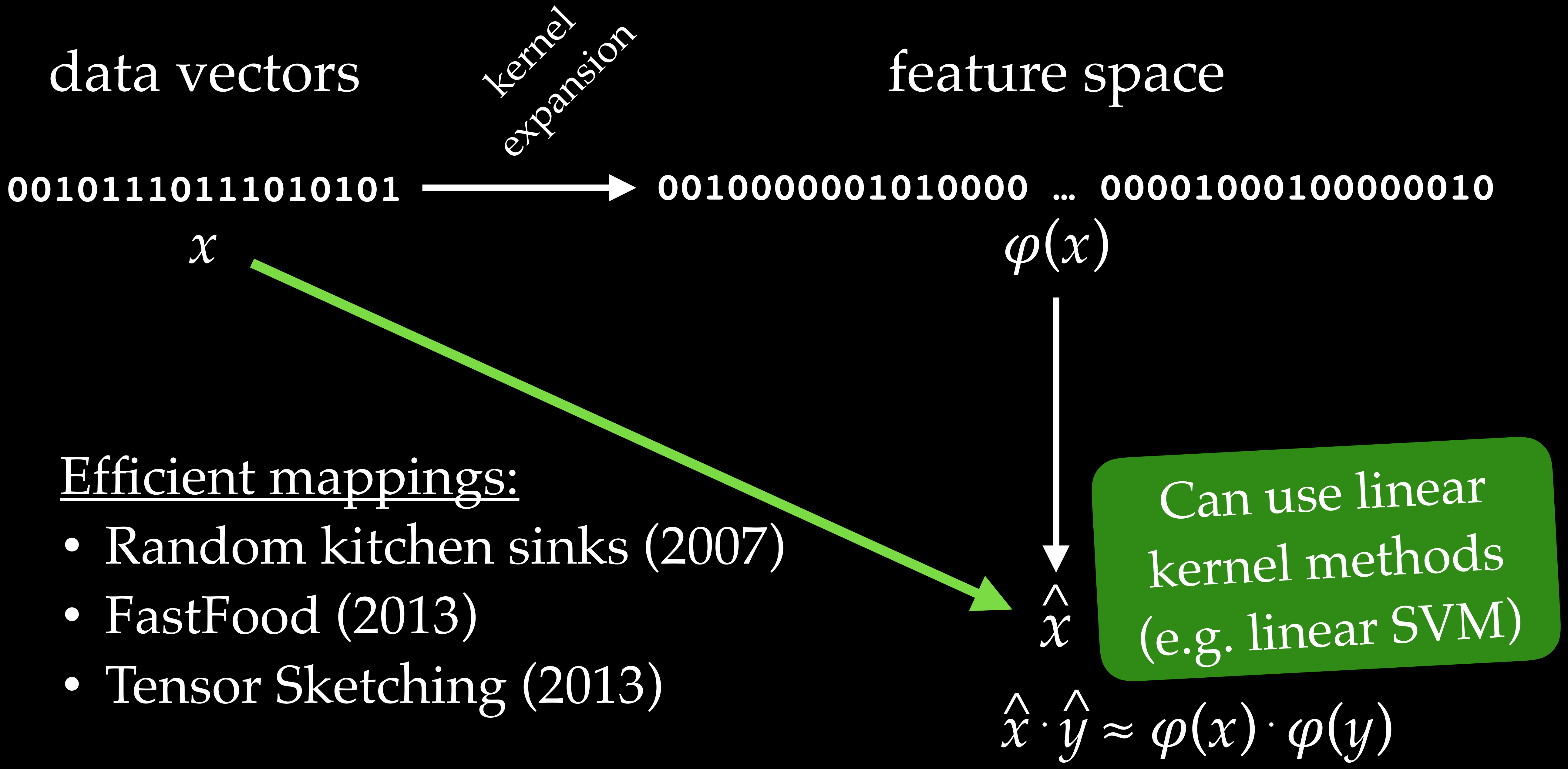
Efficient mappings:

- Random kitchen sinks (2007)

$$\hat{x} \cdot \hat{y} \approx \varphi(x) \cdot \varphi(y)$$

[Rahimi & Recht ‘07]

Random feature mappings



data vectors

feature space

`0010111011101010 1` $\xrightarrow{\text{kernel expansion}}$ `0010000001010000 … 0000100010000001 0`

x

$\varphi(x)$

$\hat{x}$

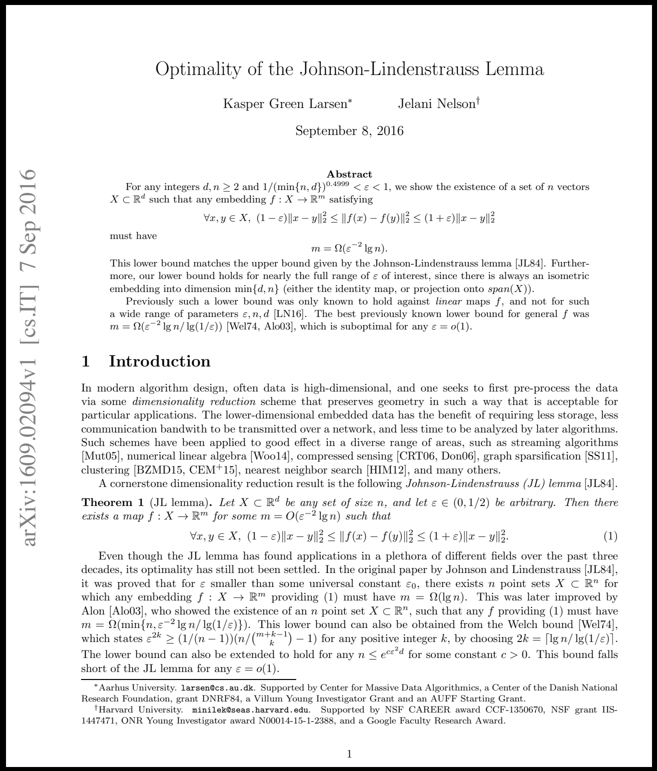
Can use linear kernel methods (e.g. linear SVM)

$$\hat{x} \cdot \hat{y} \approx \varphi(x) \cdot \varphi(y)$$

Efficient mappings:

- Random kitchen sinks (2007)
- FastFood (2013)
- Tensor Sketching (2013)

Sparse vectors



arXiv:1609.02094v1 [cs.IT] 7 Sep 2016

Optimality of the Johnson-Lindenstrauss Lemma

Kasper Green Larsen* Jelani Nelson†

September 8, 2016

Abstract

For any integers $d, n \geq 2$ and $1/(\min\{n, d\})^{0.4999} < \varepsilon < 1$, we show the existence of a set of n vectors $X \subset \mathbb{R}^d$ such that any embedding $f : X \to \mathbb{R}^m$ satisfying

$$\forall x, y \in X,\ (1-\varepsilon)\|x-y\|_2^2 \leq \|f(x)-f(y)\|_2^2 \leq (1+\varepsilon)\|x-y\|_2^2$$

must have

$$m = \Omega(\varepsilon^{-2} \lg n).$$

This lower bound matches the upper bound given by the Johnson-Lindenstrauss lemma [JL84]. Furthermore, our lower bound holds for nearly the full range of ε of interest, since there is always an isometric embedding into dimension $\min\{d, n\}$ (either the identity map, or projection onto $span(X)$).

Previously such a lower bound was only known to hold against *linear* maps f, and not for such a wide range of parameters ε, n, d [LN16]. The best previously known lower bound for general f was $m = \Omega(\varepsilon^{-2} \lg n / \lg(1/\varepsilon))$ [Wel74, Alo03], which is suboptimal for any $\varepsilon = o(1)$.

1 Introduction

In modern algorithm design, often data is high-dimensional, and one seeks to first pre-process the data via some *dimensionality reduction* scheme that preserves geometry in such a way that is acceptable for particular applications. The lower-dimensional embedded data has the benefit of requiring less storage, less communication bandwith to be transmitted over a network, and less time to be analyzed by later algorithms. Such schemes have been applied to good effect in a diverse range of areas, such as streaming algorithms [Mut05], numerical linear algebra [Woo14], compressed sensing [CRT06, Don06], graph sparsification [SS11], clustering [BZMD15, CEM+15], nearest neighbor search [HIM12], and many others.

A cornerstone dimensionality reduction result is the following *Johnson-Lindenstrauss (JL) lemma* [JL84].

Theorem 1 (JL lemma)**.** *Let $X \subset \mathbb{R}^d$ be any set of size n, and let $\varepsilon \in (0, 1/2)$ be arbitrary. Then there exists a map $f : X \to \mathbb{R}^m$ for some $m = O(\varepsilon^{-2} \lg n)$ such that*

$$\forall x, y \in X,\ (1-\varepsilon)\|x-y\|_2^2 \leq \|f(x)-f(y)\|_2^2 \leq (1+\varepsilon)\|x-y\|_2^2. \tag{1}$$

Even though the JL lemma has found applications in a plethora of different fields over the past three decades, its optimality has still not been settled. In the original paper by Johnson and Lindenstrauss [JL84], it was proved that for ε smaller than some universal constant ε_0, there exists n point sets $X \subset \mathbb{R}^n$ for which any embedding $f : X \to \mathbb{R}^m$ providing (1) must have $m = \Omega(\lg n)$. This was later improved by Alon [Alo03], who showed the existence of an n point set $X \subset \mathbb{R}^n$, such that any f providing (1) must have $m = \Omega(\min\{n, \varepsilon^{-2} \lg n / \lg(1/\varepsilon)\})$. This lower bound can also be obtained from the Welch bound [Wel74], which states $\varepsilon^{2k} \geq (1/(n-1))(n/\binom{m+k-1}{k} - 1)$ for any positive integer k, by choosing $2k = \lceil \lg n / \lg(1/\varepsilon) \rceil$. The lower bound can also be extended to hold for any $n \leq e^{c\varepsilon^2 d}$ for some constant $c > 0$. This bound falls short of the JL lemma for any $\varepsilon = o(1)$.

[*] Aarhus University. larsen@cs.au.dk. Supported by Center for Massive Data Algorithmics, a Center of the Danish National Research Foundation, grant DNRF84, a Villum Young Investigator Grant and an AUFF Starting Grant.

[†] Harvard University. minilek@seas.harvard.edu. Supported by NSF CAREER award CCF-1350670, NSF grant IIS-1447471, ONR Young Investigator award N00014-15-1-2388, and a Google Faculty Research Award.

1

Term vector (or TF/IDF)

→ **0010000001010000 … 0000100010000010**

$> 10^5$ dimensions

Sparse vectors

Optimality of the Johnson-Lindenstrauss Lemma

Kasper Green Larsen[*] Jelani Nelson[†]

September 8, 2016

Abstract

For any integers $d, n \geq 2$ and $1/(\min\{n, d\})^{0.4999} < \varepsilon < 1$, we show the existence of a set of n vectors $X \subset \mathbb{R}^d$ such that any embedding $f : X \to \mathbb{R}^m$ satisfying

$$\forall x, y \in X,\ (1-\varepsilon)\|x-y\|_2^2 \leq \|f(x)-f(y)\|_2^2 \leq (1+\varepsilon)\|x-y\|_2^2$$

must have

$$m = \Omega(\varepsilon^{-2} \lg n).$$

This lower bound matches the upper bound given by the Johnson-Lindenstrauss lemma [JL84]. Furthermore, our lower bound holds for nearly the full range of ε of interest, since there is always an isometric embedding into dimension $\min\{d, n\}$ (either the identity map, or projection onto $span(X)$).

Previously such a lower bound was only known to hold against *linear* maps f, and not for such a wide range of parameters ε, n, d [LN16]. The best previously known lower bound for general f was $m = \Omega(\varepsilon^{-2} \lg n / \lg(1/\varepsilon))$ [Wel74, Alo03], which is suboptimal for any $\varepsilon = o(1)$.

1 Introduction

In modern algorithm design, often data is high-dimensional, and one seeks to first pre-process the data via some *dimensionality reduction* scheme that preserves geometry in such a way that is acceptable for particular applications. The lower-dimensional embedded data has the benefit of requiring less storage, less communication bandwith to be transmitted over a network, and less time to be analyzed by later algorithms. Such schemes have been applied to good effect in a diverse range of areas, such as streaming algorithms [Mut05], numerical linear algebra [Woo14], compressed sensing [CRT06, Don06], graph sparsification [SS11], clustering [BZMD15, CEM+15], nearest neighbor search [HIM12], and many others.

A cornerstone dimensionality reduction result is the following *Johnson-Lindenstrauss (JL) lemma* [JL84].

Theorem 1 (JL lemma)**.** *Let $X \subset \mathbb{R}^d$ be any set of size n, and let $\varepsilon \in (0, 1/2)$ be arbitrary. Then there exists a map $f : X \to \mathbb{R}^m$ for some $m = O(\varepsilon^{-2} \lg n)$ such that*

$$\forall x, y \in X,\ (1-\varepsilon)\|x-y\|_2^2 \leq \|f(x)-f(y)\|_2^2 \leq (1+\varepsilon)\|x-y\|_2^2. \quad (1)$$

Even though the JL lemma has found applications in a plethora of different fields over the past three decades, its optimality has still not been settled. In the original paper by Johnson and Lindenstrauss [JL84], it was proved that for ε smaller than some universal constant ε_0, there exists n point sets $X \subset \mathbb{R}^n$ for which any embedding $f : X \to \mathbb{R}^m$ providing (1) must have $m = \Omega(\lg n)$. This was later improved by Alon [Alo03], who showed the existence of an n point set $X \subset \mathbb{R}^n$, such that any f providing (1) must have $m = \Omega(\min\{n, \varepsilon^{-2} \lg n / \lg(1/\varepsilon)\})$. This lower bound can also be obtained from the Welch bound [Wel74], which states $\varepsilon^{2k} \geq (1/(n-1))(n/\binom{m+k-1}{k} - 1)$ for any positive integer k, by choosing $2k = \lceil \lg n / \lg(1/\varepsilon) \rceil$. The lower bound can also be extended to hold for any $n \leq e^{c\varepsilon^2 d}$ for some constant $c > 0$. This bound falls short of the JL lemma for any $\varepsilon = o(1)$.

[*] Aarhus University. `larsen@cs.au.dk`. Supported by Center for Massive Data Algorithmics, a Center of the Danish National Research Foundation, grant DNRF84, a Villum Young Investigator Grant and an AUFF Starting Grant.

[†] Harvard University. `minilek@seas.harvard.edu`. Supported by NSF CAREER award CCF-1350670, NSF grant IIS-1447471, ONR Young Investigator award N00014-15-1-2388, and a Google Faculty Research Award.

1

Term vector (or TF/IDF)

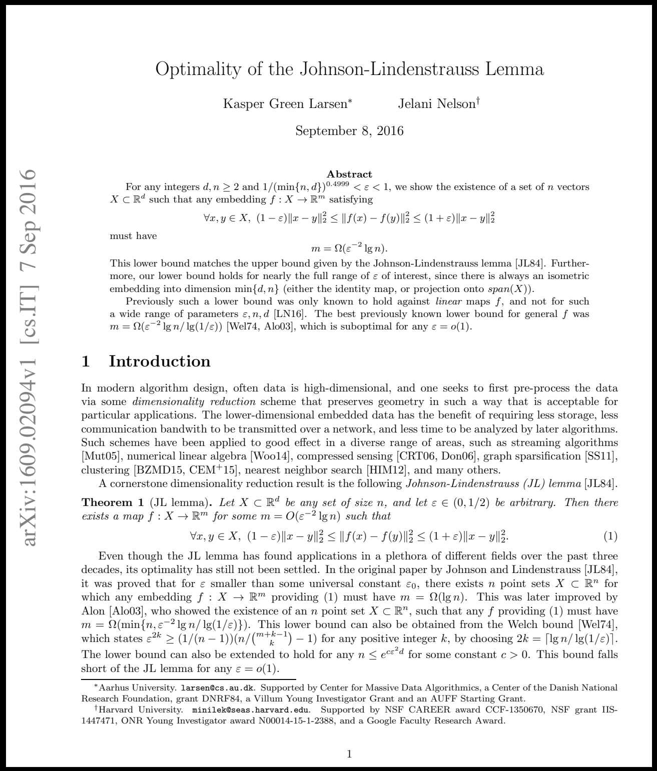
0010000001010000 … 0000100010000010

$> 10^5$ dimensions

0010111011101010101001010

dimension-reduced representation

Sparse vectors



Optimality of the Johnson-Lindenstrauss Lemma

Kasper Green Larsen[*] Jelani Nelson[†]

September 8, 2016

Abstract

For any integers $d, n \geq 2$ and $1/(\min\{n, d\})^{0.4999} < \varepsilon < 1$, we show the existence of a set of n vectors $X \subset \mathbb{R}^d$ such that any embedding $f : X \to \mathbb{R}^m$ satisfying

$$\forall x, y \in X,\ (1-\varepsilon)\|x-y\|_2^2 \leq \|f(x)-f(y)\|_2^2 \leq (1+\varepsilon)\|x-y\|_2^2$$

must have

$$m = \Omega(\varepsilon^{-2} \lg n).$$

This lower bound matches the upper bound given by the Johnson-Lindenstrauss lemma [JL84]. Furthermore, our lower bound holds for nearly the full range of ε of interest, since there is always an isometric embedding into dimension $\min\{d, n\}$ (either the identity map, or projection onto $span(X)$).

Previously such a lower bound was only known to hold against *linear* maps f, and not for such a wide range of parameters ε, n, d [LN16]. The best previously known lower bound for general f was $m = \Omega(\varepsilon^{-2} \lg n / \lg(1/\varepsilon))$ [Wel74, Alo03], which is suboptimal for any $\varepsilon = o(1)$.

1 Introduction

In modern algorithm design, often data is high-dimensional, and one seeks to first pre-process the data via some *dimensionality reduction* scheme that preserves geometry in such a way that is acceptable for particular applications. The lower-dimensional embedded data has the benefit of requiring less storage, less communication bandwith to be transmitted over a network, and less time to be analyzed by later algorithms. Such schemes have been applied to good effect in a diverse range of areas, such as streaming algorithms [Mut05], numerical linear algebra [Woo14], compressed sensing [CRT06, Don06], graph sparsification [SS11], clustering [BZMD15, CEM$^+$15], nearest neighbor search [HIM12], and many others.

A cornerstone dimensionality reduction result is the following *Johnson-Lindenstrauss (JL) lemma* [JL84].

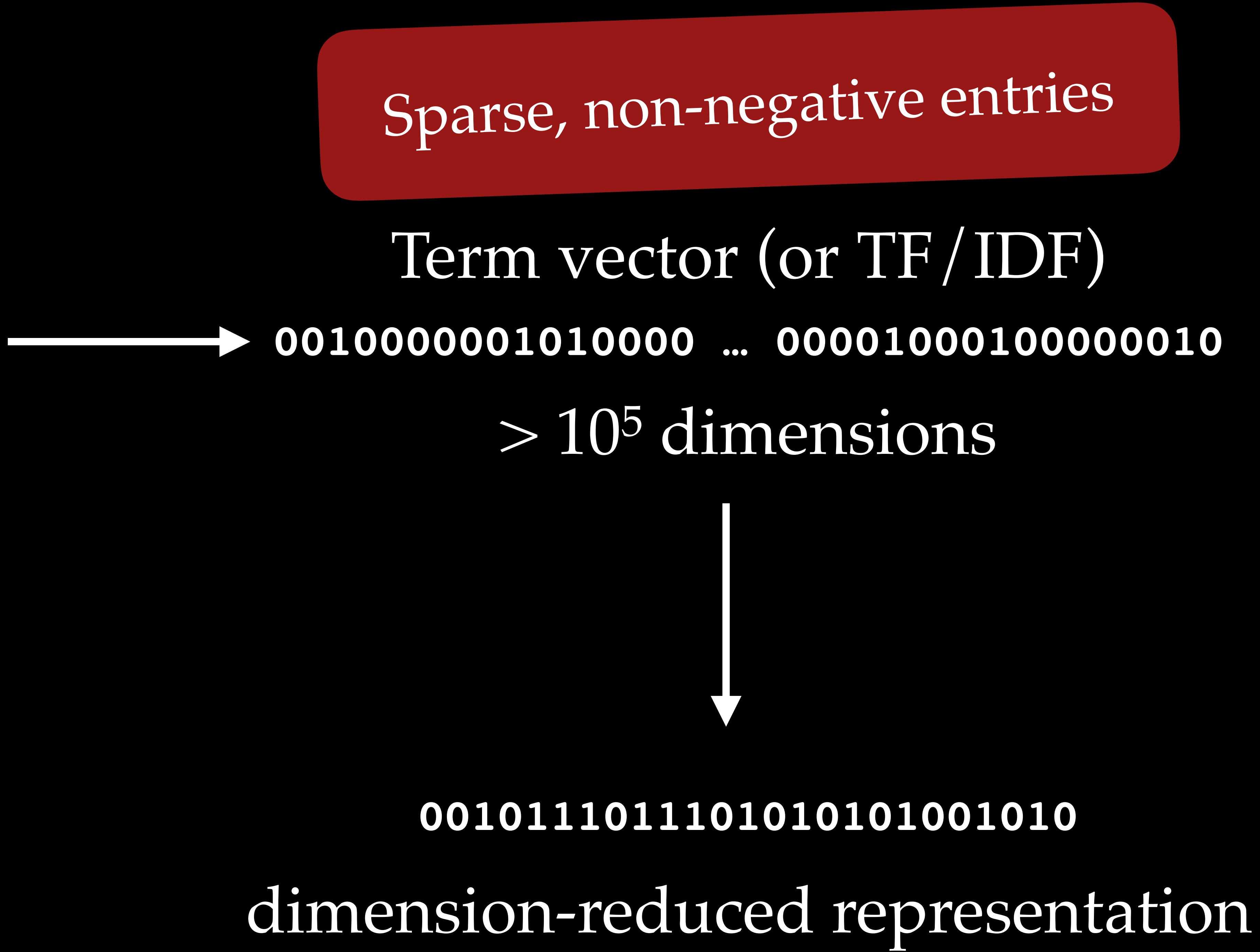
Theorem 1 (JL lemma). *Let $X \subset \mathbb{R}^d$ be any set of size n, and let $\varepsilon \in (0, 1/2)$ be arbitrary. Then there exists a map $f : X \to \mathbb{R}^m$ for some $m = O(\varepsilon^{-2} \lg n)$ such that*

$$\forall x, y \in X,\ (1-\varepsilon)\|x-y\|_2^2 \leq \|f(x)-f(y)\|_2^2 \leq (1+\varepsilon)\|x-y\|_2^2. \quad (1)$$

Even though the JL lemma has found applications in a plethora of different fields over the past three decades, its optimality has still not been settled. In the original paper by Johnson and Lindenstrauss [JL84], it was proved that for ε smaller than some universal constant ε_0, there exists n point sets $X \subset \mathbb{R}^n$ for which any embedding $f : X \to \mathbb{R}^m$ providing (1) must have $m = \Omega(\lg n)$. This was later improved by Alon [Alo03], who showed the existence of an n point set $X \subset \mathbb{R}^n$, such that any f providing (1) must have $m = \Omega(\min\{n, \varepsilon^{-2} \lg n / \lg(1/\varepsilon)\})$. This lower bound can also be obtained from the Welch bound [Wel74], which states $\varepsilon^{2k} \geq (1/(n-1))(n/\binom{m+k-1}{k} - 1)$ for any positive integer k, by choosing $2k = \lceil \lg n / \lg(1/\varepsilon) \rceil$. The lower bound can also be extended to hold for any $n \leq e^{c\varepsilon^2 d}$ for some constant $c > 0$. This bound falls short of the JL lemma for any $\varepsilon = o(1)$.

[*] Aarhus University. larsen@cs.au.dk. Supported by Center for Massive Data Algorithmics, a Center of the Danish National Research Foundation, grant DNRF84, a Villum Young Investigator Grant and an AUFF Starting Grant.

[†] Harvard University. minilek@seas.harvard.edu. Supported by NSF CAREER award CCF-1350670, NSF grant IIS-1447471, ONR Young Investigator award N00014-15-1-2388, and a Google Faculty Research Award.



Sparse, non-negative entries

Term vector (or TF/IDF)

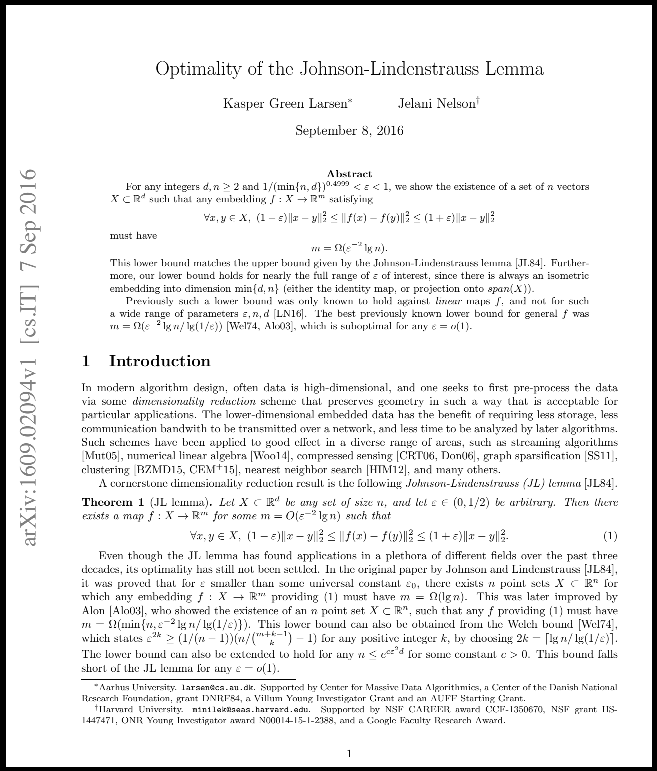
0010000001010000 … 0000100010000010

$> 10^5$ dimensions

0010111011101010101001010

dimension-reduced representation

Sparse vectors



arXiv:1609.02094v1 [cs.IT] 7 Sep 2016

Optimality of the Johnson-Lindenstrauss Lemma

Kasper Green Larsen[*] Jelani Nelson[†]

September 8, 2016

Abstract

For any integers $d, n \geq 2$ and $1/(\min\{n,d\})^{0.4999} < \varepsilon < 1$, we show the existence of a set of n vectors $X \subset \mathbb{R}^d$ such that any embedding $f : X \to \mathbb{R}^m$ satisfying

$$\forall x, y \in X,\ (1-\varepsilon)\|x-y\|_2^2 \leq \|f(x)-f(y)\|_2^2 \leq (1+\varepsilon)\|x-y\|_2^2$$

must have

$$m = \Omega(\varepsilon^{-2} \lg n).$$

This lower bound matches the upper bound given by the Johnson-Lindenstrauss lemma [JL84]. Furthermore, our lower bound holds for nearly the full range of ε of interest, since there is always an isometric embedding into dimension $\min\{d, n\}$ (either the identity map, or projection onto $span(X)$).

Previously such a lower bound was only known to hold against *linear* maps f, and not for such a wide range of parameters ε, n, d [LN16]. The best previously known lower bound for general f was $m = \Omega(\varepsilon^{-2} \lg n / \lg(1/\varepsilon))$ [Wel74, Alo03], which is suboptimal for any $\varepsilon = o(1)$.

1 Introduction

In modern algorithm design, often data is high-dimensional, and one seeks to first pre-process the data via some *dimensionality reduction* scheme that preserves geometry in such a way that is acceptable for particular applications. The lower-dimensional embedded data has the benefit of requiring less storage, less communication bandwith to be transmitted over a network, and less time to be analyzed by later algorithms. Such schemes have been applied to good effect in a diverse range of areas, such as streaming algorithms [Mut05], numerical linear algebra [Woo14], compressed sensing [CRT06, Don06], graph sparsification [SS11], clustering [BZMD15, CEM+15], nearest neighbor search [HIM12], and many others.

A cornerstone dimensionality reduction result is the following *Johnson-Lindenstrauss (JL) lemma* [JL84].

Theorem 1 (JL lemma). *Let $X \subset \mathbb{R}^d$ be any set of size n, and let $\varepsilon \in (0, 1/2)$ be arbitrary. Then there exists a map $f : X \to \mathbb{R}^m$ for some $m = O(\varepsilon^{-2} \lg n)$ such that*

$$\forall x, y \in X,\ (1-\varepsilon)\|x-y\|_2^2 \leq \|f(x)-f(y)\|_2^2 \leq (1+\varepsilon)\|x-y\|_2^2. \quad (1)$$

Even though the JL lemma has found applications in a plethora of different fields over the past three decades, its optimality has still not been settled. In the original paper by Johnson and Lindenstrauss [JL84], it was proved that for ε smaller than some universal constant ε_0, there exists n point sets $X \subset \mathbb{R}^n$ for which any embedding $f : X \to \mathbb{R}^m$ providing (1) must have $m = \Omega(\lg n)$. This was later improved by Alon [Alo03], who showed the existence of an n point set $X \subset \mathbb{R}^n$, such that any f providing (1) must have $m = \Omega(\min\{n, \varepsilon^{-2} \lg n / \lg(1/\varepsilon)\})$. This lower bound can also be obtained from the Welch bound [Wel74], which states $\varepsilon^{2k} \geq (1/(n-1))(n/\binom{m+k-1}{k} - 1)$ for any positive integer k, by choosing $2k = \lceil \lg n / \lg(1/\varepsilon) \rceil$. The lower bound can also be extended to hold for any $n \leq e^{c\varepsilon^2 d}$ for some constant $c > 0$. This bound falls short of the JL lemma for any $\varepsilon = o(1)$.

[*] Aarhus University. larsen@cs.au.dk. Supported by Center for Massive Data Algorithmics, a Center of the Danish National Research Foundation, grant DNRF84, a Villum Young Investigator Grant and an AUFF Starting Grant.

[†] Harvard University. minilek@seas.harvard.edu. Supported by NSF CAREER award CCF-1350670, NSF grant IIS-1447471, ONR Young Investigator award N00014-15-1-2388, and a Google Faculty Research Award.

1

Sparse, non-negative entries

Term vector (or TF / IDF)

$\rightarrow$ **0010000001010000 … 0000100010000010**

$> 10^5$ dimensions

$\downarrow$

- Min-wise hashing (1997)
- *b*-bit min-wise hashing (2010)

001011101110101010101001010

dimension-reduced representation

[Li & König '10]

1-bit minwise hashing

- Min-max kernel: $k(x, y) = \frac{\sum_i \min(x_i, y_i)}{\sum_i \max(x_i, y_i)}$

[Li & König '10]

1-bit minwise hashing

- Min-max kernel: $k(x, y) = \frac{\sum_i \min(x_i, y_i)}{\sum_i \max(x_i, y_i)}$

  Now: Binary vectors/Jaccard similarity

[Li & König '10]

1-bit minwise hashing

- Min-max kernel: $k(x,y) = \frac{\sum_i \min(x_i, y_i)}{\sum_i \max(x_i, y_i)}$

  Now: Binary vectors/Jaccard similarity

- Random hash functions h_i: $\mathbf{N} \longrightarrow [0;1]$
  - Min-hash: $z_i(x) = \arg\min_{x_j \neq 0} h_i(j)$

[Li & König '10]

1-bit minwise hashing

- Min-max kernel: $k(x,y) = \frac{\sum_i \min(x_i, y_i)}{\sum_i \max(x_i, y_i)}$

  Now: Binary vectors / Jaccard similarity

- Random hash functions h_i: $\mathbf{N} \longrightarrow [0;1]$
  - Min-hash: $z_i(x) = \arg\min_{x_j \neq 0} h_i(j)$
  - 1-bit min-hash: $b_i(z_i(x))$, for random b_i: $\mathbf{N} \longrightarrow \{0,1\}$

[Li & König '10]

1-bit minwise hashing

- Min-max kernel: $k(x,y) = \dfrac{\sum_i \min(x_i, y_i)}{\sum_i \max(x_i, y_i)}$

  Now: Binary vectors / Jaccard similarity

- Random hash functions h_i: $\mathbf{N} \longrightarrow [0;1]$
  - Min-hash: $z_i(x) = \arg\min\limits_{x_j \neq 0} h_i(j)$
  - 1-bit min-hash: $b_i(z_i(x))$, for random b_i: $\mathbf{N} \longrightarrow \{0,1\}$
- Binary dimension-reduced representation:

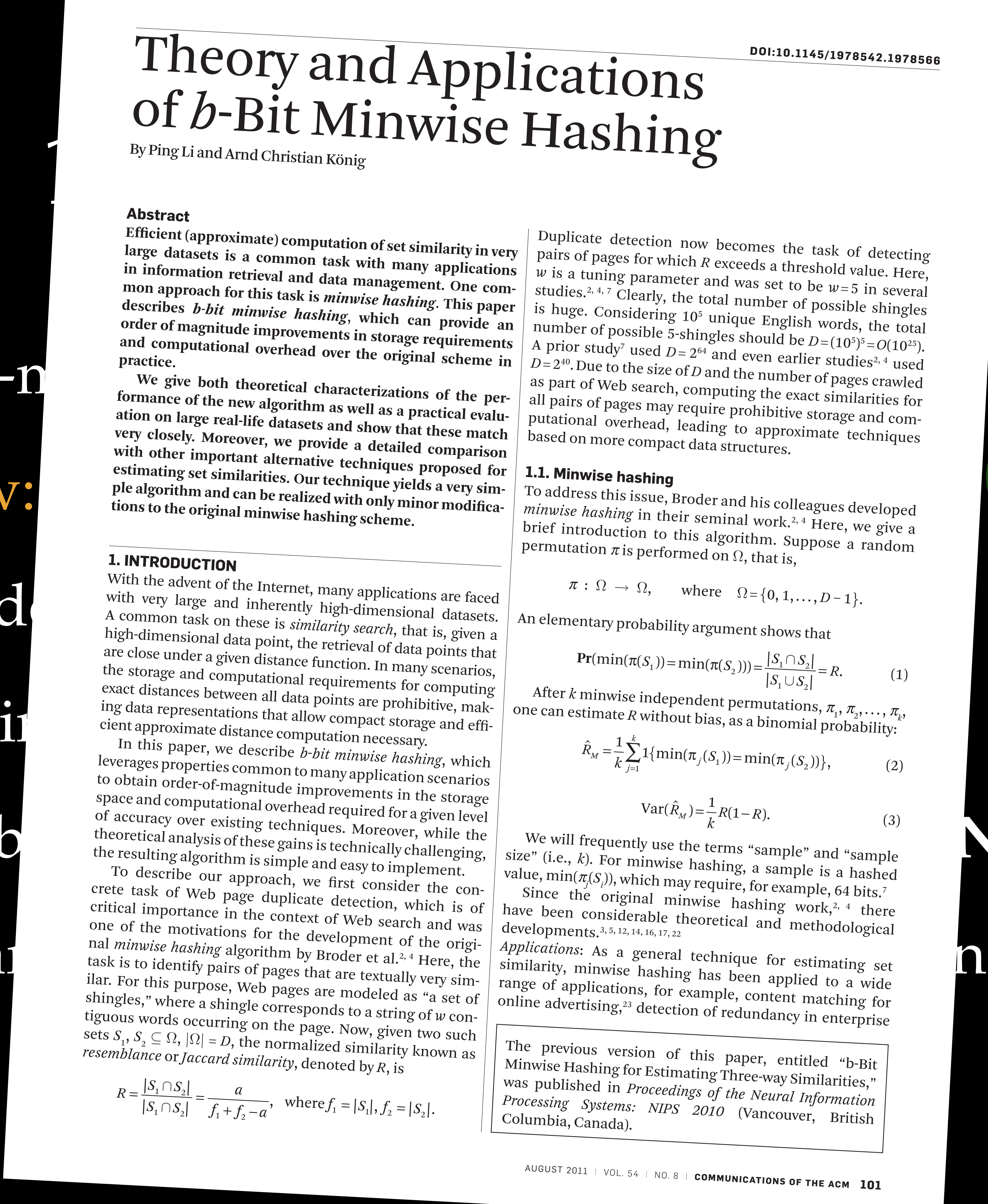
$$b_1(z_1(x)) \ldots b_m(z_m(x))$$

i & König ‘10]

- Min-n

Now:

- Rand
  - Mi
  - 1-b N → {0,1}
- Binar n:



DOI:10.1145/1978542.1978566

Theory and Applications of *b*-Bit Minwise Hashing

By Ping Li and Arnd Christian König

Abstract

Efficient (approximate) computation of set similarity in very large datasets is a common task with many applications in information retrieval and data management. One common approach for this task is *minwise hashing*. This paper describes *b-bit minwise hashing*, which can provide an order of magnitude improvements in storage requirements and computational overhead over the original scheme in practice.

We give both theoretical characterizations of the performance of the new algorithm as well as a practical evaluation on large real-life datasets and show that these match very closely. Moreover, we provide a detailed comparison with other important alternative techniques proposed for estimating set similarities. Our technique yields a very simple algorithm and can be realized with only minor modifications to the original minwise hashing scheme.

1. INTRODUCTION

With the advent of the Internet, many applications are faced with very large and inherently high-dimensional datasets. A common task on these is *similarity search*, that is, given a high-dimensional data point, the retrieval of data points that are close under a given distance function. In many scenarios, the storage and computational requirements for computing exact distances between all data points are prohibitive, making data representations that allow compact storage and efficient approximate distance computation necessary.

In this paper, we describe *b-bit minwise hashing*, which leverages properties common to many application scenarios to obtain order-of-magnitude improvements in the storage space and computational overhead required for a given level of accuracy over existing techniques. Moreover, while the theoretical analysis of these gains is technically challenging, the resulting algorithm is simple and easy to implement.

To describe our approach, we first consider the concrete task of Web page duplicate detection, which is of critical importance in the context of Web search and was one of the motivations for the development of the original *minwise hashing* algorithm by Broder et al.[2, 4] Here, the task is to identify pairs of pages that are textually very similar. For this purpose, Web pages are modeled as "a set of shingles," where a shingle corresponds to a string of w contiguous words occurring on the page. Now, given two such sets $S_1, S_2 \subseteq \Omega$, $|\Omega| = D$, the normalized similarity known as *resemblance* or *Jaccard similarity*, denoted by R, is

$$R = \frac{|S_1 \cap S_2|}{|S_1 \cap S_2|} = \frac{a}{f_1 + f_2 - a}, \quad \text{where } f_1 = |S_1|, f_2 = |S_2|.$$

Duplicate detection now becomes the task of detecting pairs of pages for which R exceeds a threshold value. Here, w is a tuning parameter and was set to be $w = 5$ in several studies.[2, 4, 7] Clearly, the total number of possible shingles is huge. Considering 10^5 unique English words, the total number of possible 5-shingles should be $D = (10^5)^5 = O(10^{25})$. A prior study[7] used $D = 2^{64}$ and even earlier studies[2, 4] used $D = 2^{40}$. Due to the size of D and the number of pages crawled as part of Web search, computing the exact similarities for all pairs of pages may require prohibitive storage and computational overhead, leading to approximate techniques based on more compact data structures.

1.1. Minwise hashing

To address this issue, Broder and his colleagues developed *minwise hashing* in their seminal work.[2, 4] Here, we give a brief introduction to this algorithm. Suppose a random permutation π is performed on Ω, that is,

$$\pi : \Omega \rightarrow \Omega, \qquad \text{where} \quad \Omega = \{0, 1, \ldots, D - 1\}.$$

An elementary probability argument shows that

$$\mathbf{Pr}(\min(\pi(S_1)) = \min(\pi(S_2))) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} = R. \qquad (1)$$

After k minwise independent permutations, $\pi_1, \pi_2, \ldots, \pi_k$, one can estimate R without bias, as a binomial probability:

$$\hat{R}_M = \frac{1}{k}\sum_{j=1}^{k} 1\{\min(\pi_j(S_1)) = \min(\pi_j(S_2))\}, \qquad (2)$$

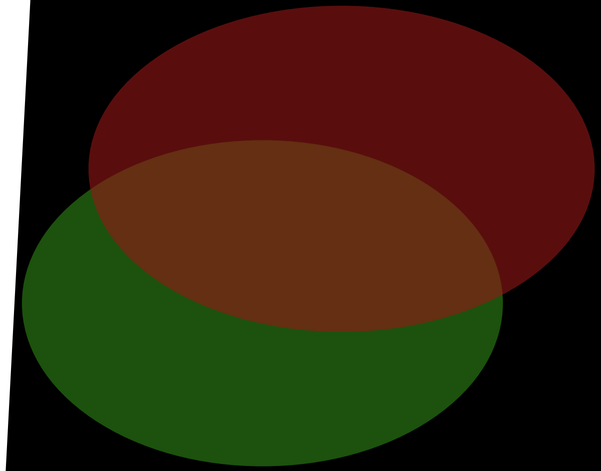
$$\mathrm{Var}(\hat{R}_M) = \frac{1}{k} R(1 - R). \qquad (3)$$

We will frequently use the terms "sample" and "sample size" (i.e., k). For minwise hashing, a sample is a hashed value, $\min(\pi_j(S_i))$, which may require, for example, 64 bits.[7]

Since the original minwise hashing work,[2, 4] there have been considerable theoretical and methodological developments.[3, 5, 12, 14, 16, 17, 22]

Applications: As a general technique for estimating set similarity, minwise hashing has been applied to a wide range of applications, for example, content matching for online advertising,[23] detection of redundancy in enterprise

The previous version of this paper, entitled "b-Bit Minwise Hashing for Estimating Three-way Similarities," was published in *Proceedings of the Neural Information Processing Systems: NIPS 2010* (Vancouver, British Columbia, Canada).



Part II:

Transparency and interpretability

EU regulations on algorithmic decision-making and a "right to explanation"

Bryce Goodman BRYCE.GOODMAN@STX.OX.AC.UK
Oxford Internet Institute, Oxford

Seth Flaxman FLAXMAN@STATS.OX.AC.UK
Department of Statistics, Oxford

Abstract

We summarize the potential impact that the European Union's new General Data Protection Regulation will have on the routine use of machine learning algorithms. Slated to take effect as law across the EU in 2018, it will restrict automated individual decision-making (that is, algorithms that make decisions based on user-level predictors) which "significantly affect" users. The law will also create a "right to explanation," whereby a user can ask for an explanation of an algorithmic decision that was made about them. We argue that while this law will pose large challenges for industry, it highlights opportunities for machine learning researchers to take the lead in designing algorithms and evaluation frameworks which avoid discrimination.

1. Introduction

On 14 April 2016, for the first time in over two decades, the European Parliament adopted a set of comprehensive regulations for the collection, storage and use of personal information, the General Data Protection Reg-

However, while the bulk of language deals with how data is collected and stored, the regulation contains a short article entitled "Automated individual decision-making" (see figure 1) potentially prohibiting a wide swath of algorithms currently in use in, e.g. recommendation systems, credit and insurance risk assessments, computational advertising, and social networks. This raises important issues that are of particular concern to the machine learning community. In its current form, the GDPR's requirements could require a complete overhaul of standard and widely used algorithmic techniques. The GDPR's policy on the right of citizens to receive an explanation for algorithmic decisions highlights the pressing importance of human interpretability in algorithm design. If, as expected, the GDPR takes effect in its current form in mid-2018, there will be a pressing need for effective algorithms which can operate within this new legal framework.

Article 11. Automated individual decision making

1. Member States shall provide for a decision based solely on automated processing, including profiling, which produces an adverse legal effect concerning the

We summarize the potential impact that the European Union's new General Data Protection Regulation will have on the routine use of machine learning algorithms. Slated to take effect as law across the EU in 2018, it will restrict automated individual decision-making (that is, algorithms that make decisions based on user-level predictors) which "significantly affect" users. The law will also create a "right to explanation," whereby a user can ask for an explanation of an algorithmic decision that was made about them. We argue that while this law will pose large challenges for industry, it highlights opportunities for machine learning researchers to take the lead in

We summarize the potential impact that the European Union's new General Data Protection Regulation will have on the routine use of machine learning algorithms. Slated to take effect as law across the EU in 2018, it will restrict automated individual decision-making (that is, algorithms that make decisions based on user-level predictors) which "significantly affect" users. The law will also create a "right to explanation," whereby a user can ask for an explanation of an algorithmic decision that was made about them. We argue that while this law will pose large challenges for industry, it highlights opportunities for machine learning researchers to take the lead in

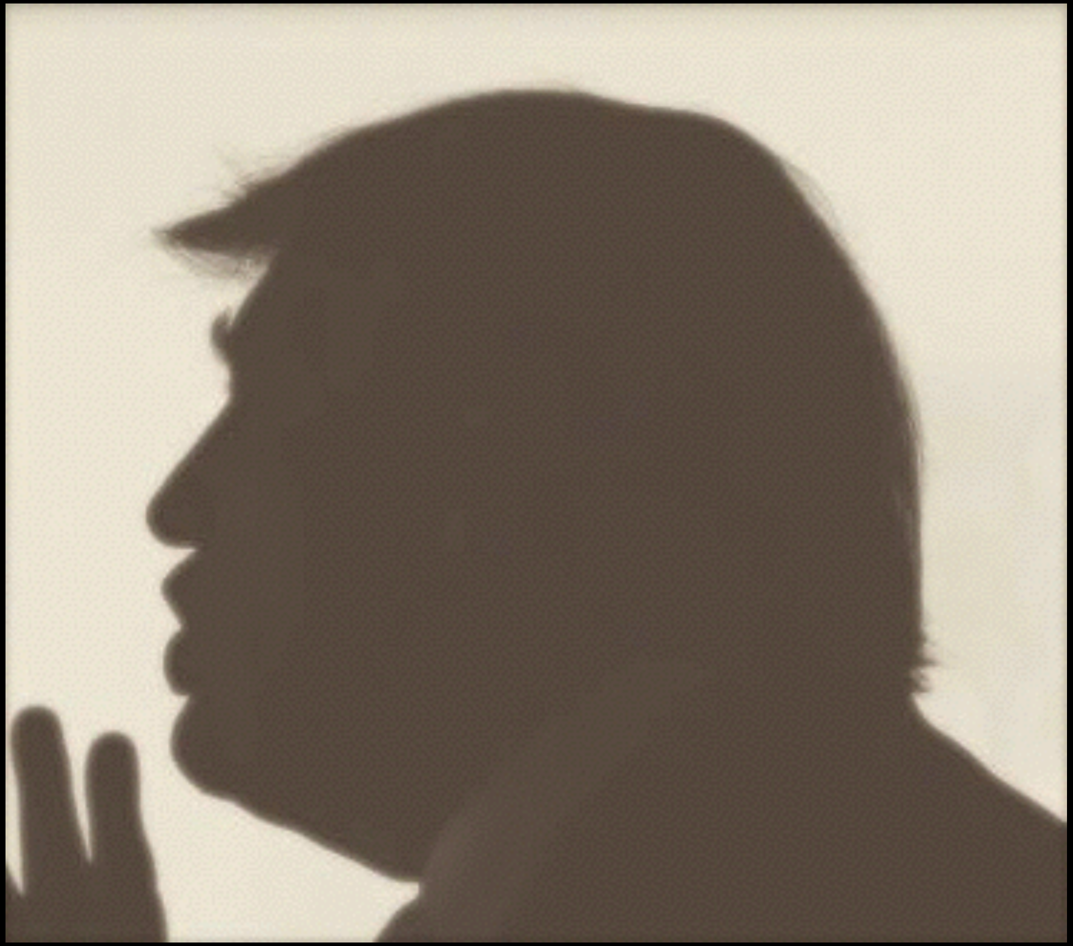


Photo by Charlie Neibergall - AP

Biff Tannen in Back to the Future

Issues for dimension reduction

- Dimension reduction creates features that are not easy to describe in terms of original data.

Issues for dimension reduction

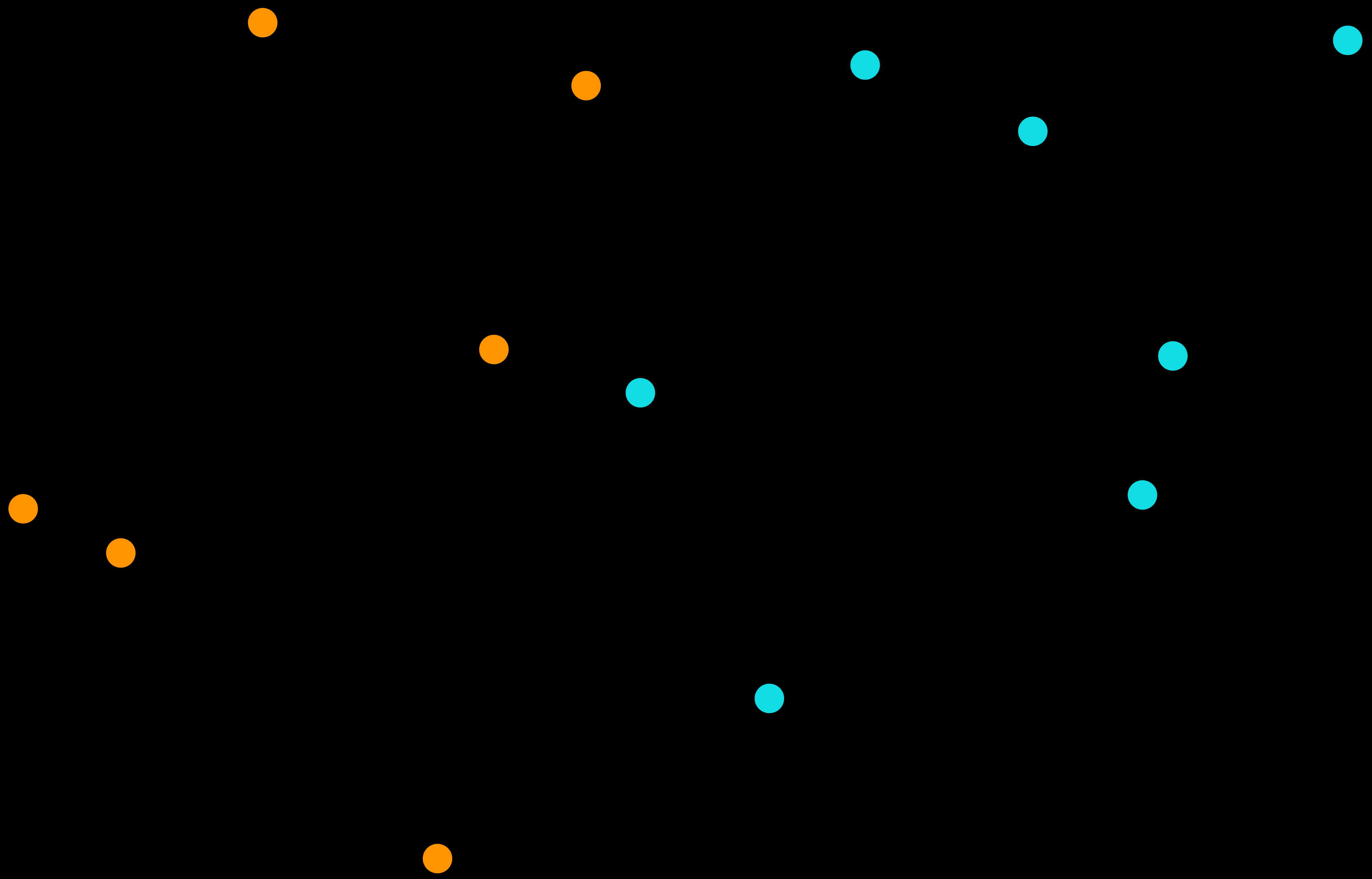
- Dimension reduction creates features that are not easy to describe in terms of original data.

- Use of randomization causes issues of:
    - Trust. *"Is it a coincidence that my feature vector is similar to Donald Trump's, or was this arranged?"*
    - Fairness. *"Would I have gotten a loan if the random choices had been different?"*
    - …

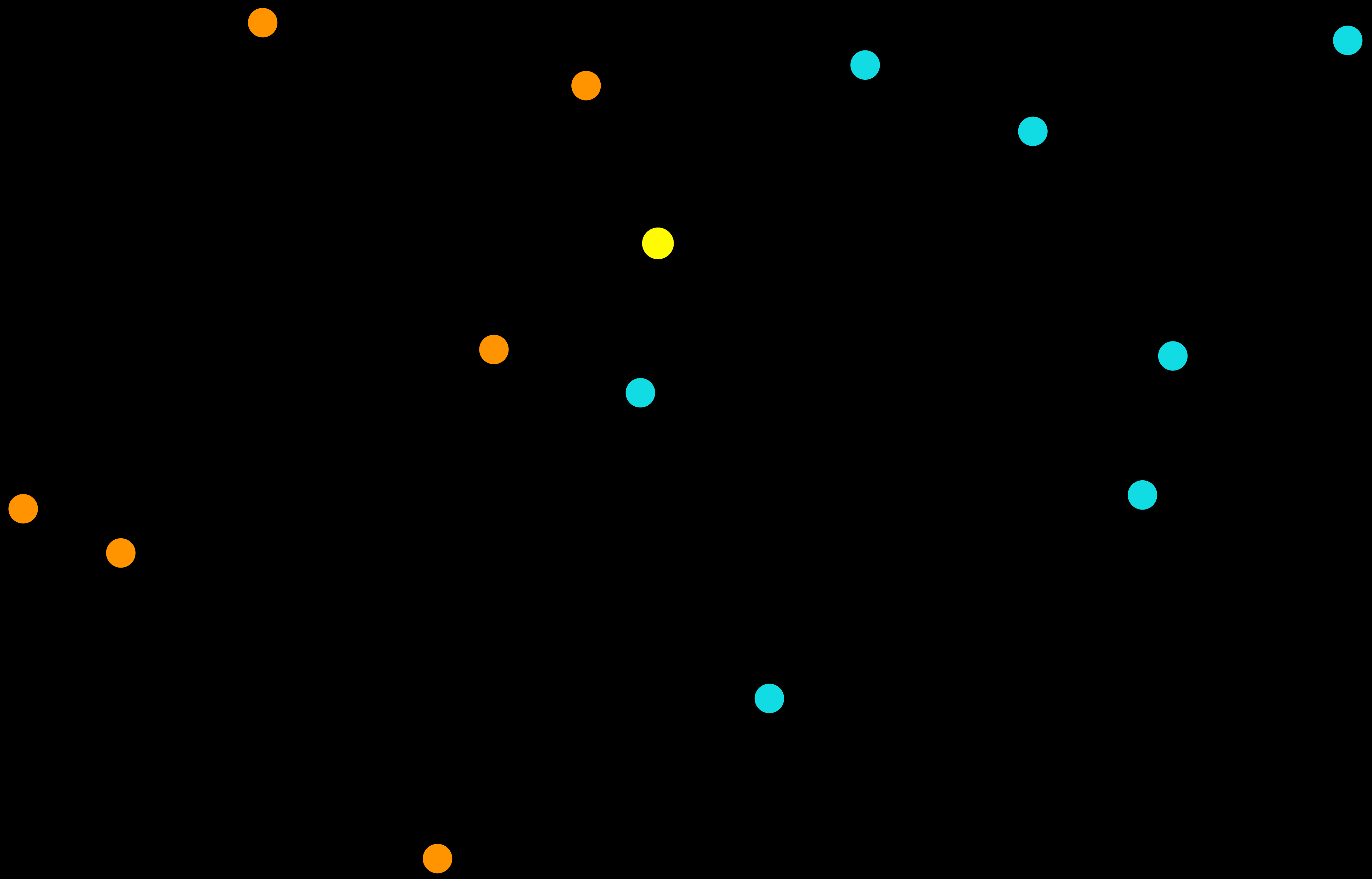
Part III:

Dimension reduction with certainty?

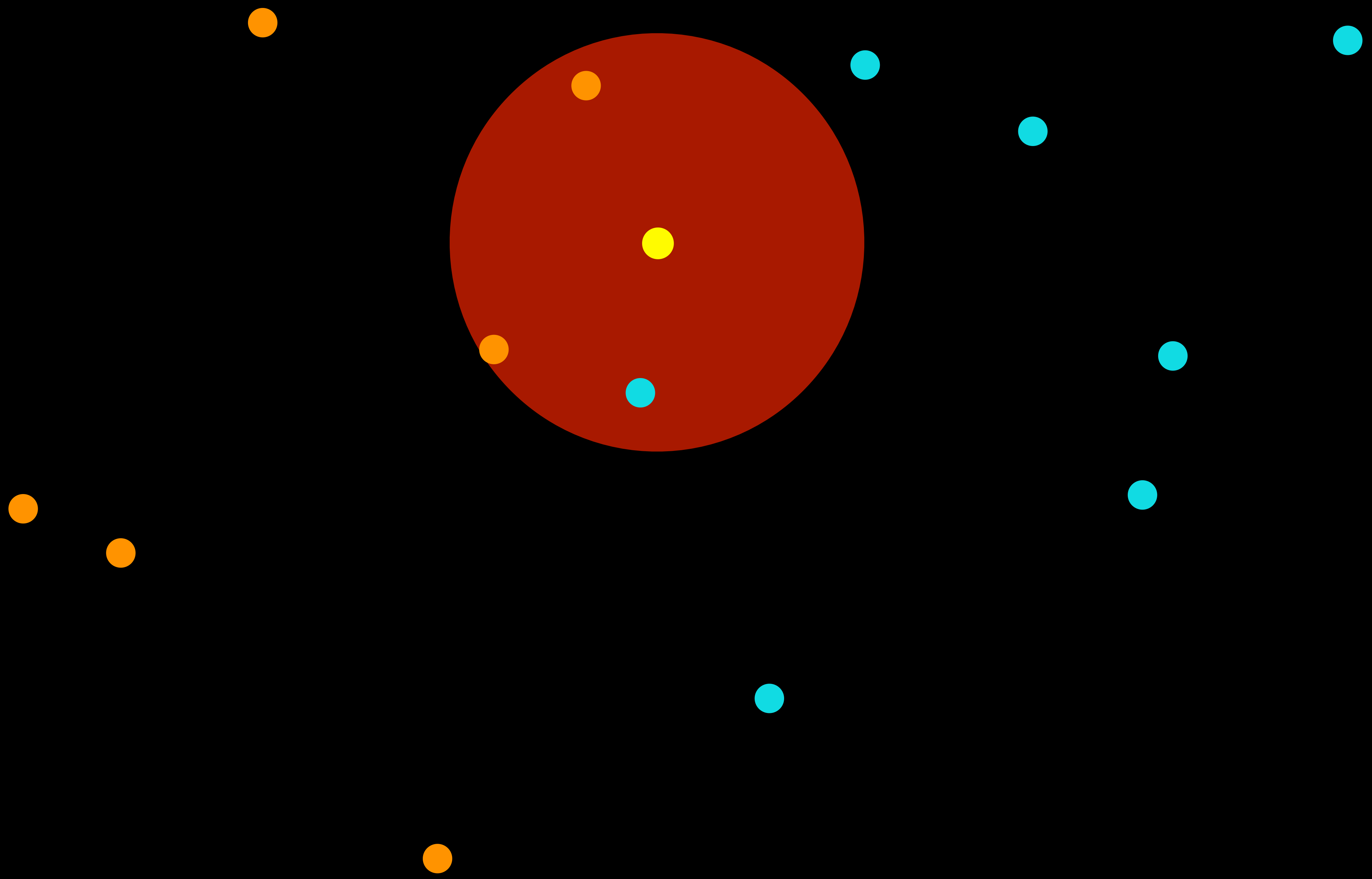
kNN classifier



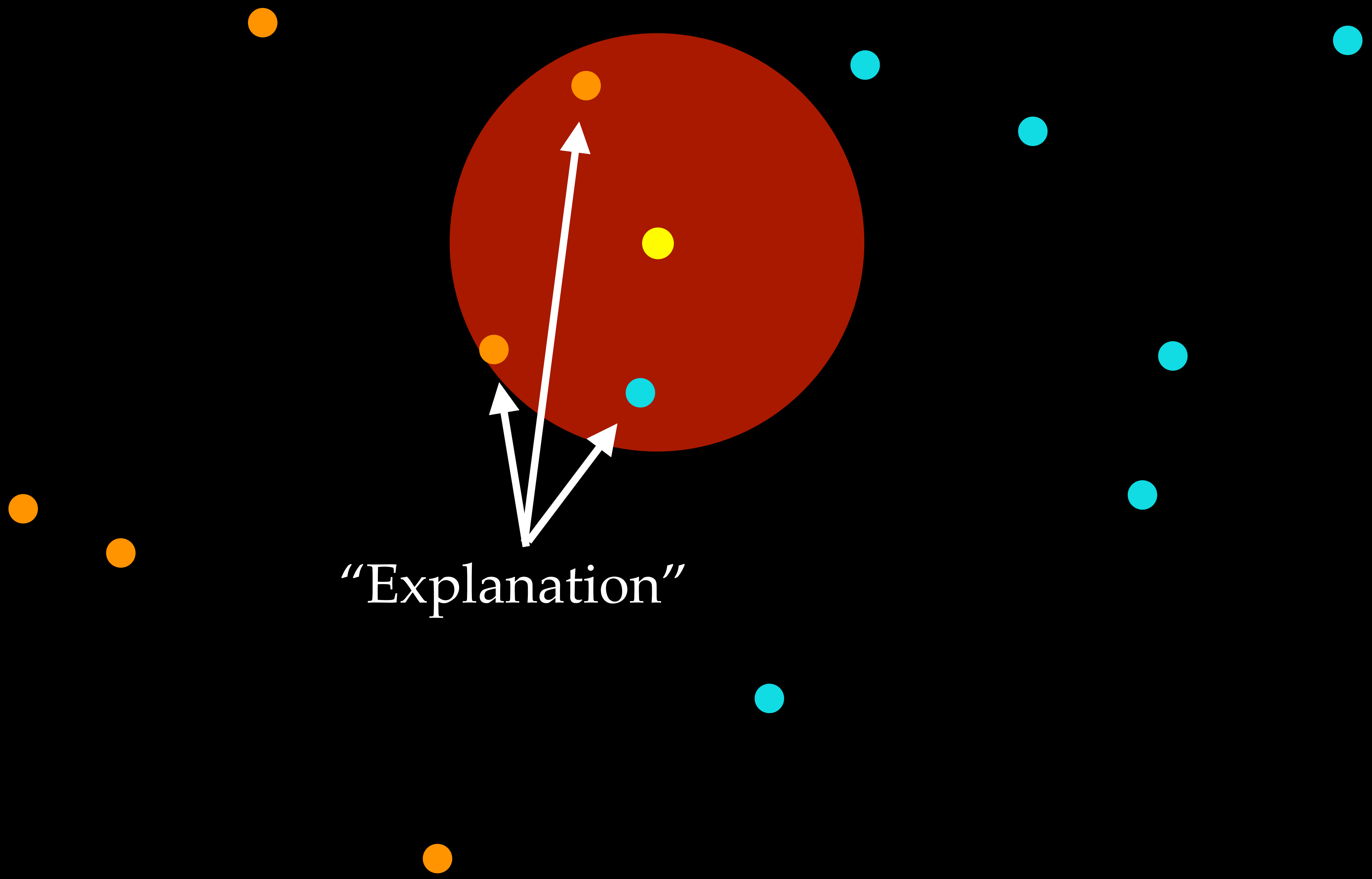
kNN classifier



kNN classifier

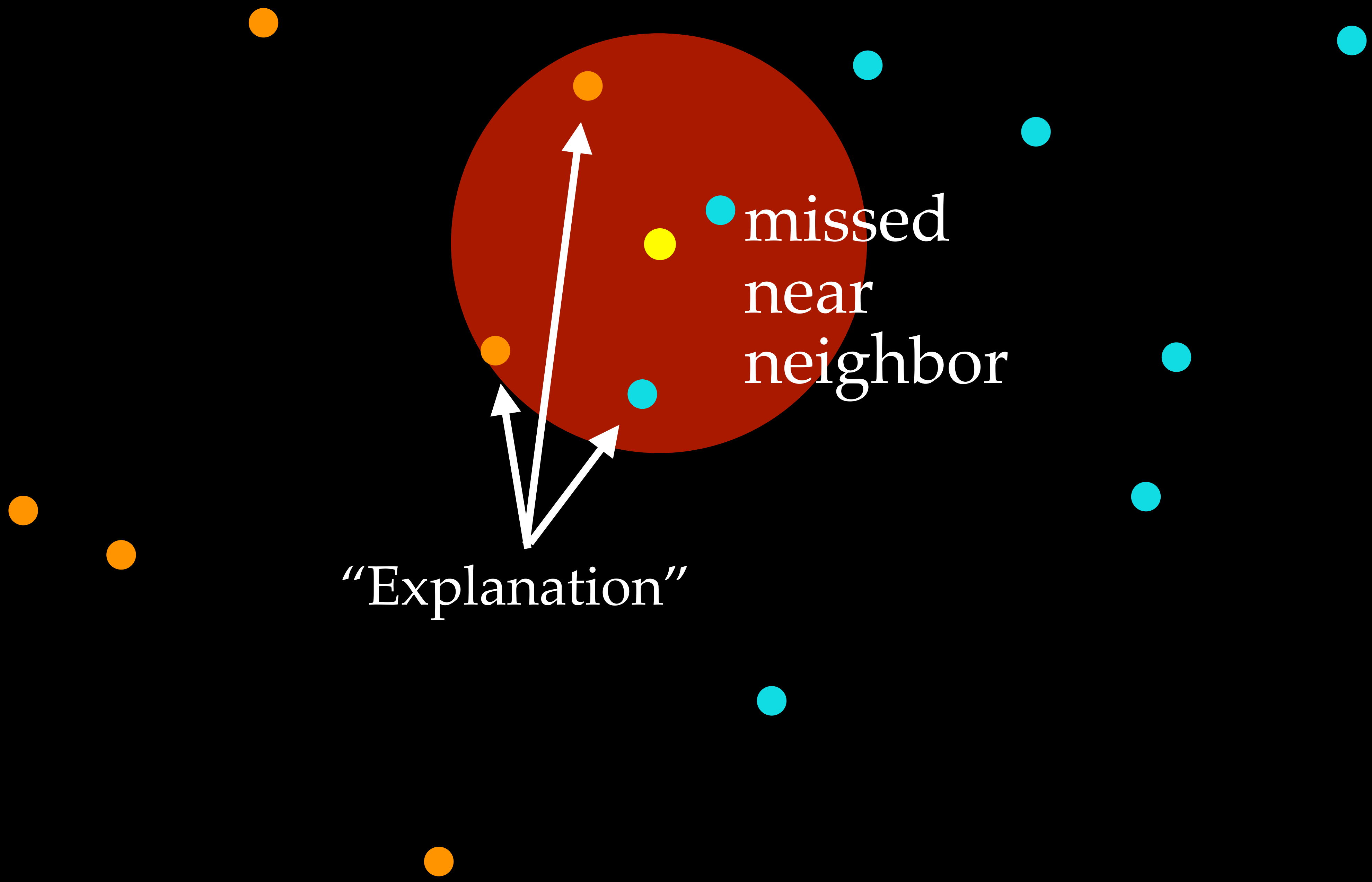


kNN classifier



“Explanation”

kNN classifier



missed near neighbor

“Explanation”

Getting *rid* of randomness?

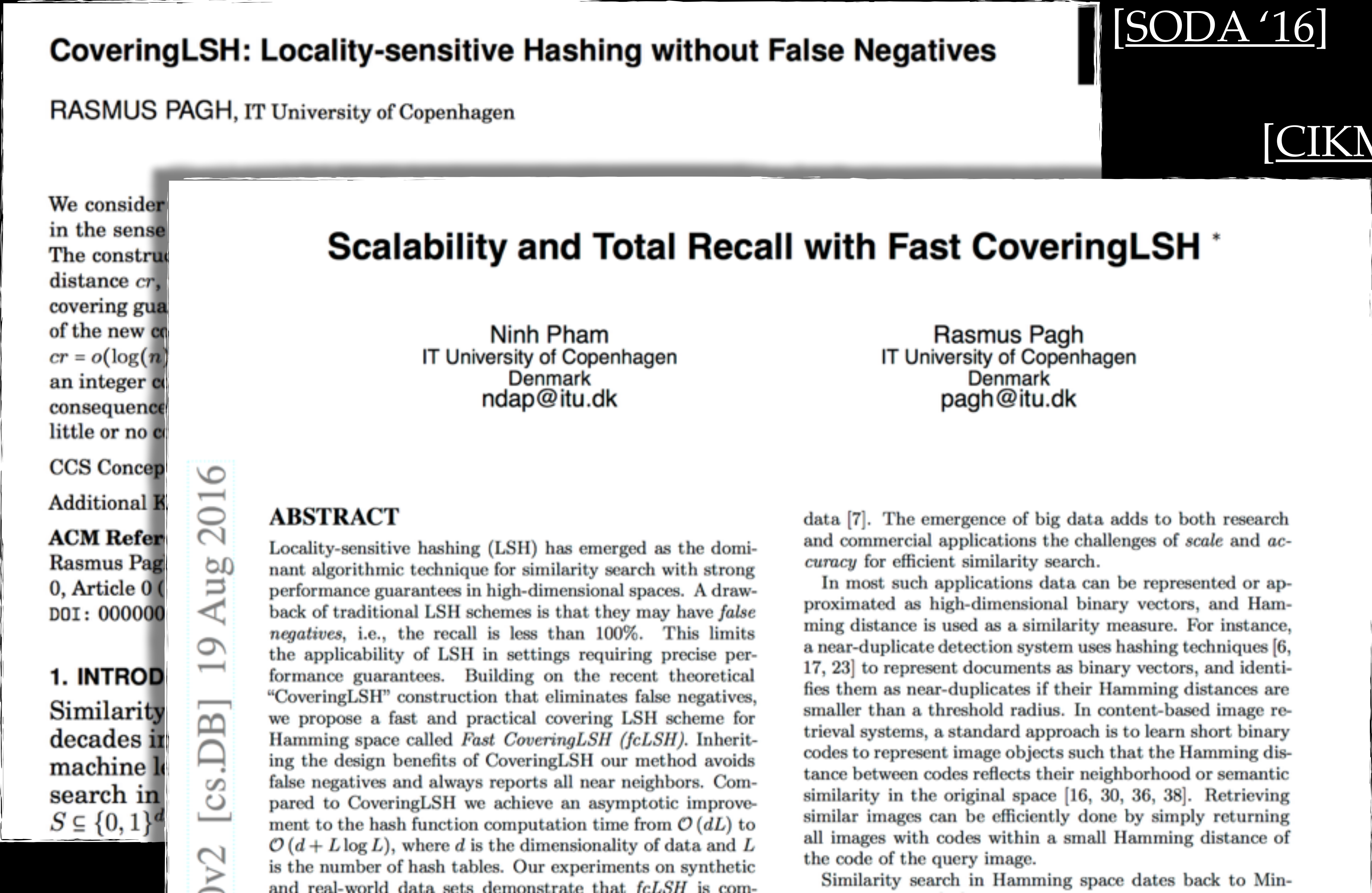
- Easy to argue that a *deterministic* dimension reduction cannot work without assumptions on data (“incompressibility”).

Getting *rid* of randomness?

- Easy to argue that a *deterministic* dimension reduction cannot work without assumptions on data ("incompressibility").

- Second best option? Randomized algorithms whose output is guaranteed, but may fail to produce result within a given time/space usage. ("Las Vegas", a la quicksort.)

NN-search with certainty



[SODA '16]

CoveringLSH: Locality-sensitive Hashing without False Negatives

RASMUS PAGH, IT University of Copenhagen

We consider
in the sense
The constru
distance cr,
covering gua
of the new co
$cr = o(\log(n$
an integer co
consequence
little or no co

CCS Concep

Additional K

ACM Refer
Rasmus Pag
0, Article 0 (
DOI: 000000

1. INTROD

Similarity
decades in
machine l
search in
$S \subseteq \{0,1\}^d$

[CIKM '16]

Scalability and Total Recall with Fast CoveringLSH *

Ninh Pham
IT University of Copenhagen
Denmark
ndap@itu.dk

Rasmus Pagh
IT University of Copenhagen
Denmark
pagh@itu.dk

0v2 [cs.DB] 19 Aug 2016

ABSTRACT

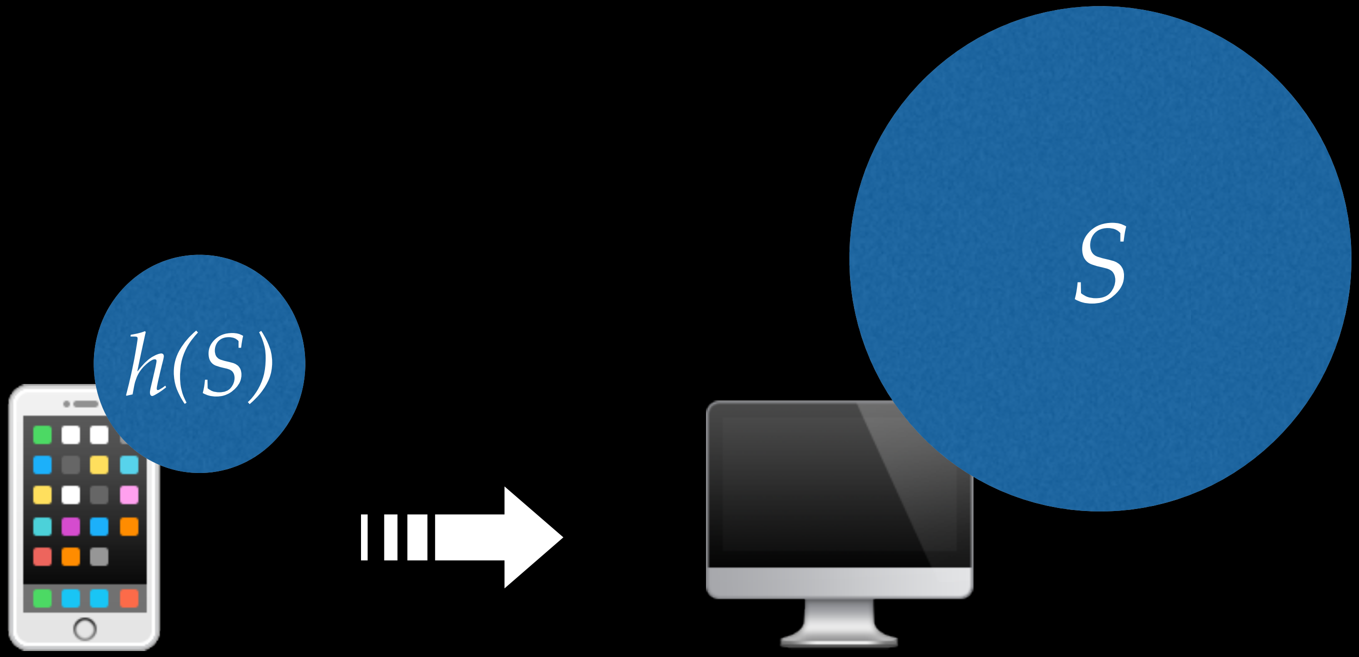
Locality-sensitive hashing (LSH) has emerged as the dominant algorithmic technique for similarity search with strong performance guarantees in high-dimensional spaces. A drawback of traditional LSH schemes is that they may have *false negatives*, i.e., the recall is less than 100%. This limits the applicability of LSH in settings requiring precise performance guarantees. Building on the recent theoretical "CoveringLSH" construction that eliminates false negatives, we propose a fast and practical covering LSH scheme for Hamming space called *Fast CoveringLSH (fcLSH)*. Inheriting the design benefits of CoveringLSH our method avoids false negatives and always reports all near neighbors. Compared to CoveringLSH we achieve an asymptotic improvement to the hash function computation time from $\mathcal{O}(dL)$ to $\mathcal{O}(d + L \log L)$, where d is the dimensionality of data and L is the number of hash tables. Our experiments on synthetic and real-world data sets demonstrate that *fcLSH* is com-

data [7]. The emergence of big data adds to both research and commercial applications the challenges of *scale* and *accuracy* for efficient similarity search.

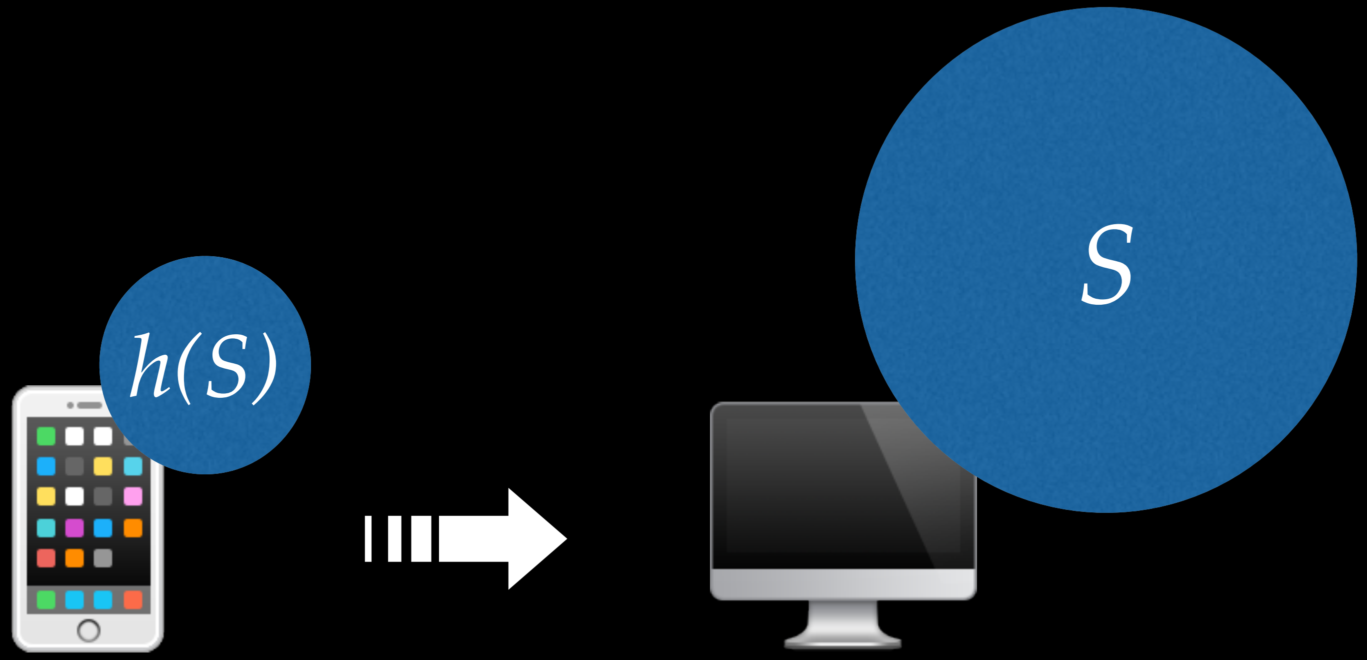
In most such applications data can be represented or approximated as high-dimensional binary vectors, and Hamming distance is used as a similarity measure. For instance, a near-duplicate detection system uses hashing techniques [6, 17, 23] to represent documents as binary vectors, and identifies them as near-duplicates if their Hamming distances are smaller than a threshold radius. In content-based image retrieval systems, a standard approach is to learn short binary codes to represent image objects such that the Hamming distance between codes reflects their neighborhood or semantic similarity in the original space [16, 30, 36, 38]. Retrieving similar images can be efficiently done by simply returning all images with codes within a small Hamming distance of the code of the query image.

Similarity search in Hamming space dates back to Min-

Bloom filters



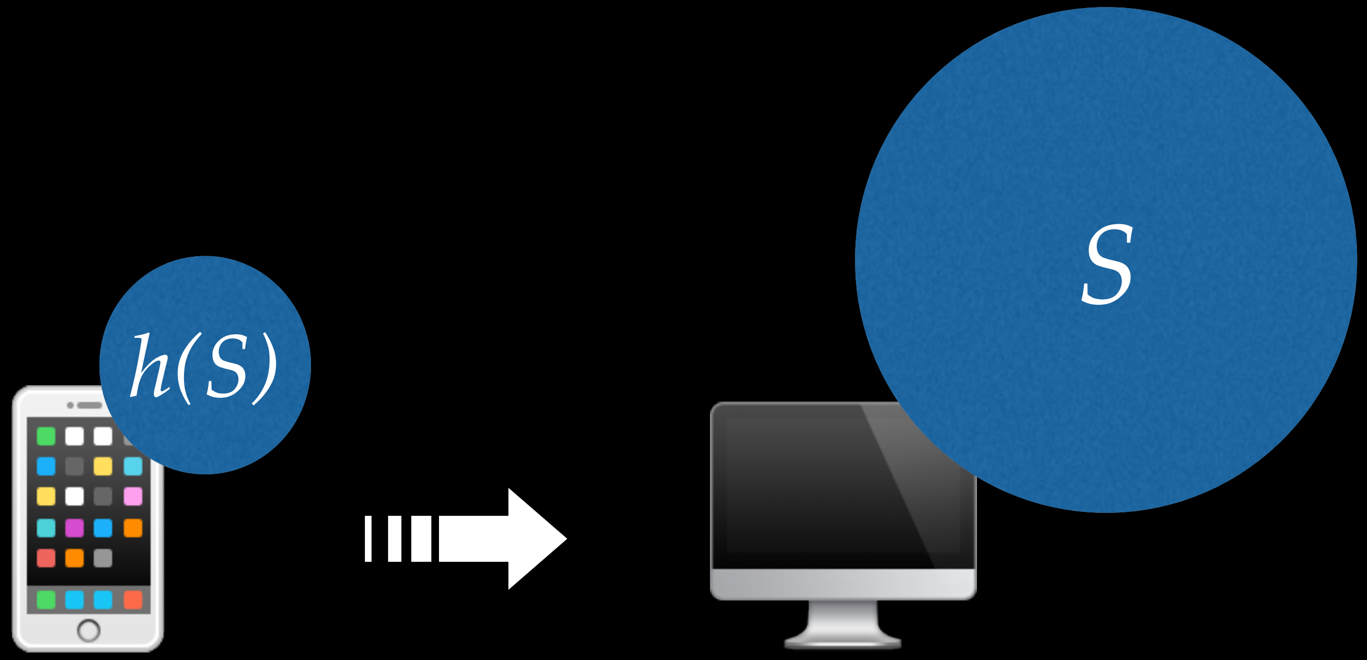
Bloom filters



- Use cases:
  - Detecting identical data in a remote server.
  - Make it possible to test for inclusion in S while revealing very little about S.

Bloom filters

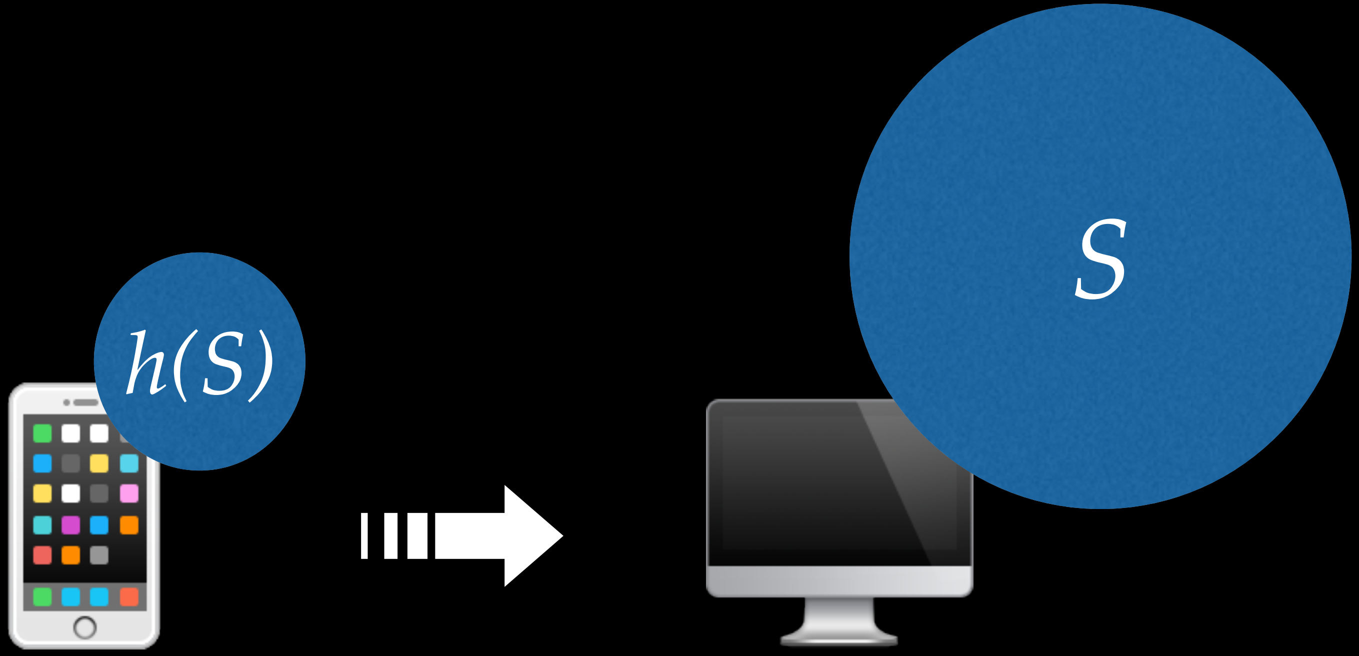
Allow ε fraction "false positives"



- Use cases:
  - Detecting identical data in a remote server.
  - Make it possible to test for inclusion in S while revealing very little about S.

Distance sensitive Bloom filters

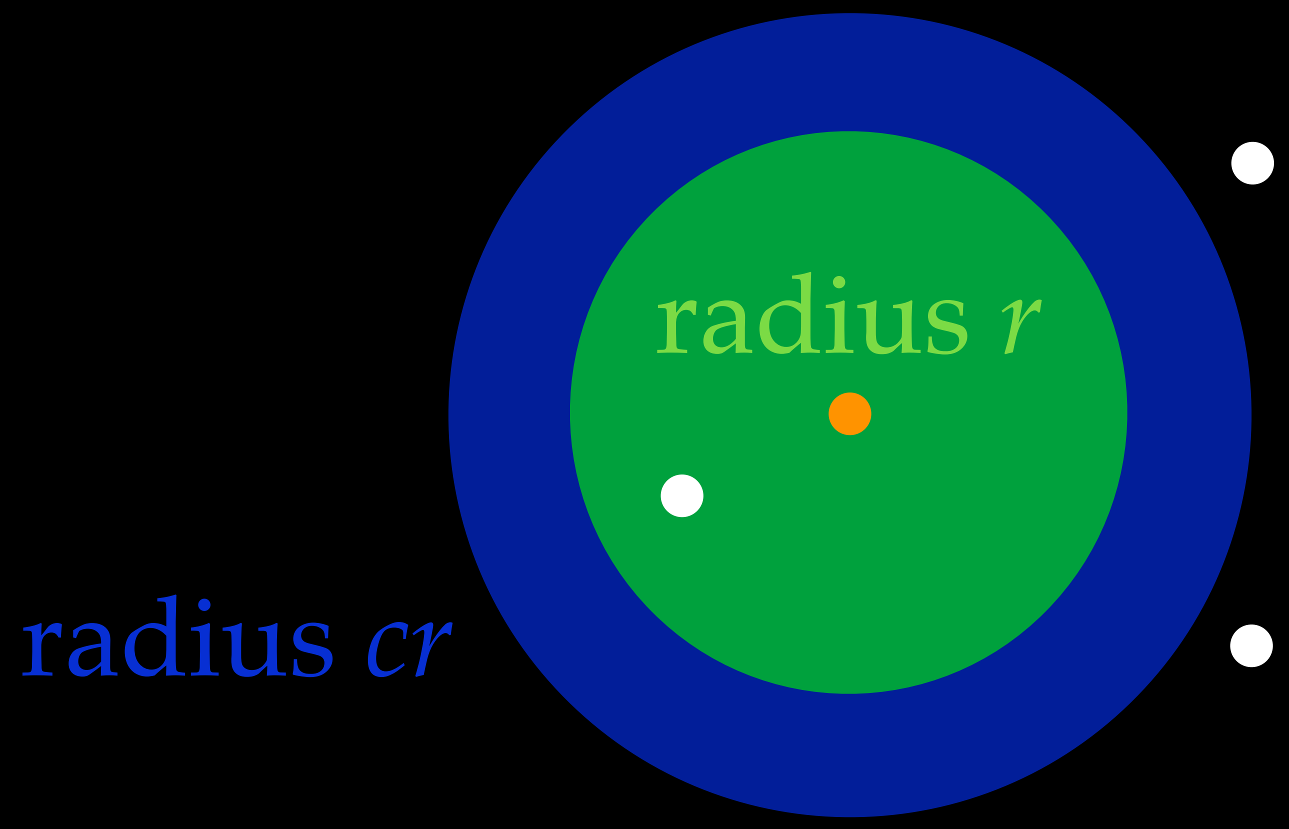
Allow ε fraction
"false positives"



- Use cases:
  - Detecting nearly identical vectors in a remote server.
  - Make it possible to test for proximity to a vector x in S while revealing very little about S.

[Kirsch & Mitzenmacher '06]

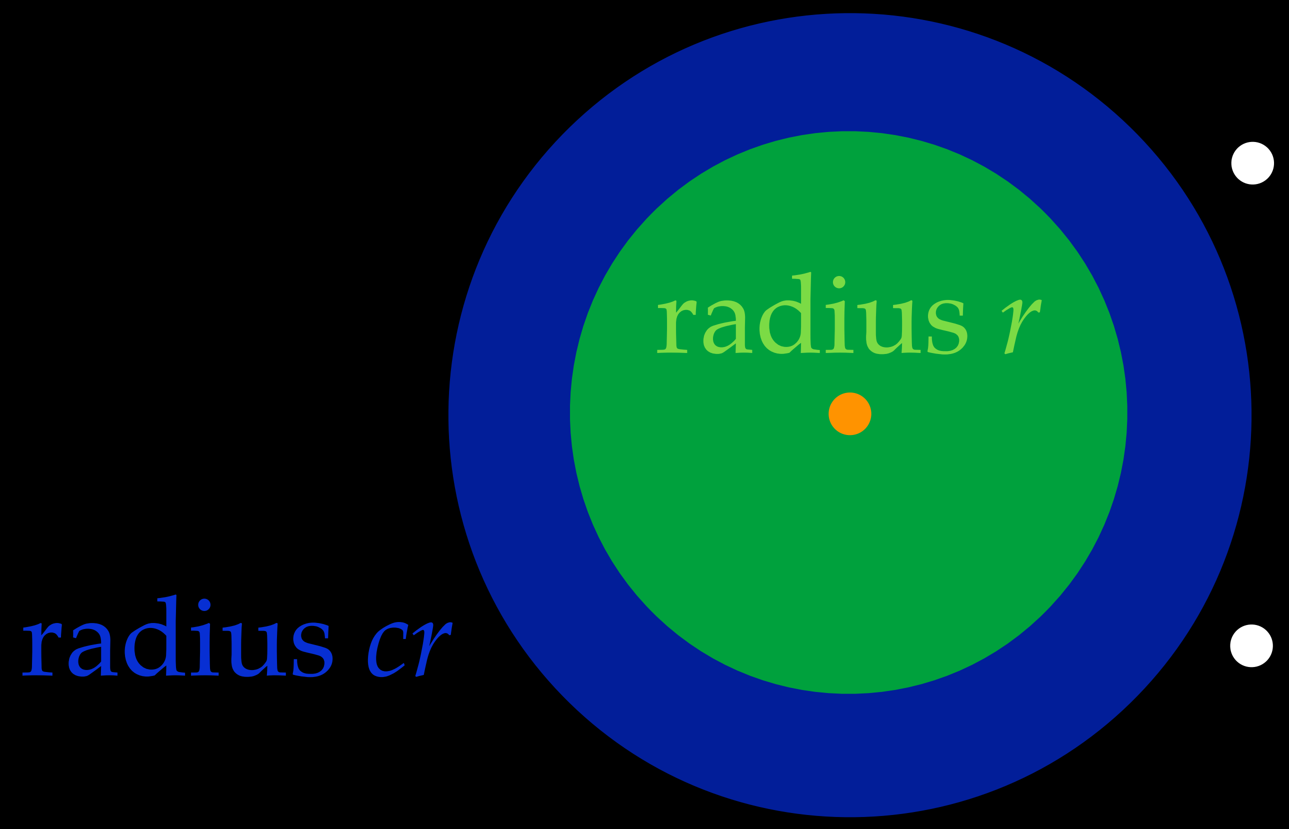
Distance sensitive Bloom filters



- Store collection of S bit vectors
- Given query vector y determine distinguish
  1. exists $x \in S$ within distance r from y, and
  2. all vectors in S have distance at least cr from y

[Kirsch & Mitzenmacher ‘06]

Distance sensitive Bloom filters

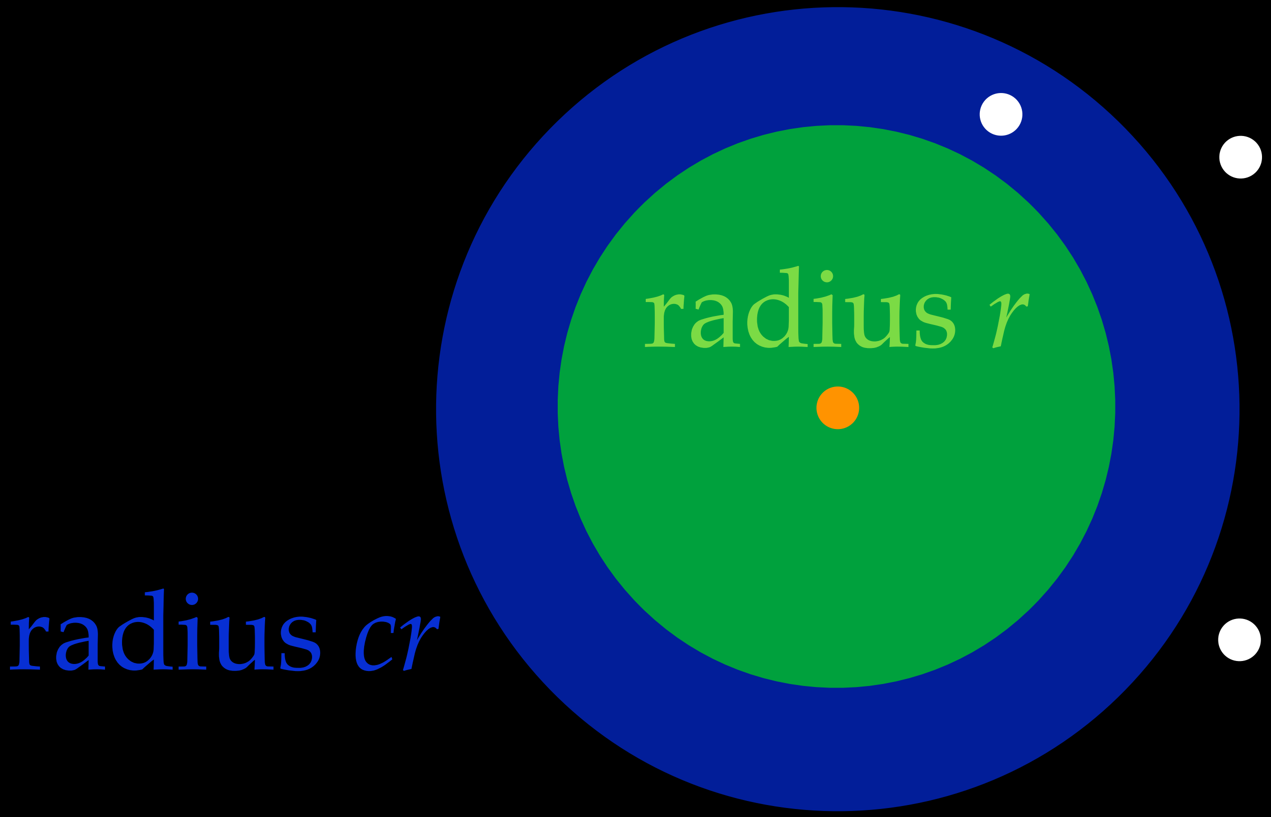


- Store collection of S bit vectors
- Given query vector y determine distinguish

1. exists $x \in S$ within distance r from y, and
2. all vectors in S have distance at least cr from y

[Kirsch & Mitzenmacher '06]

Distance sensitive Bloom filters

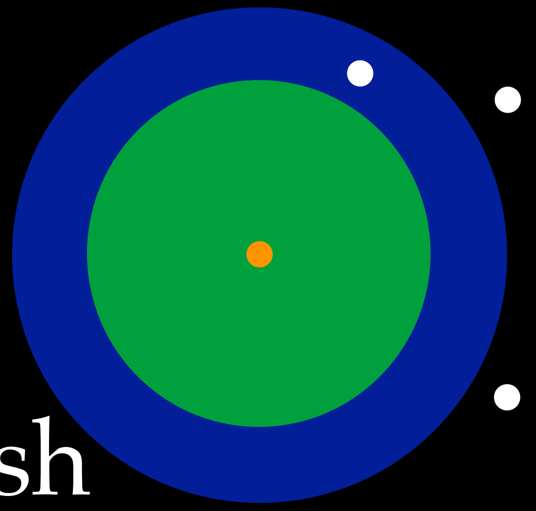


- Store collection of S bit vectors
- Given query vector y determine distinguish

1. exists $x \in S$ within distance r from y, and
2. all vectors in S have distance at least cr from y

[Kirsch & Mitzenmacher ‘06]

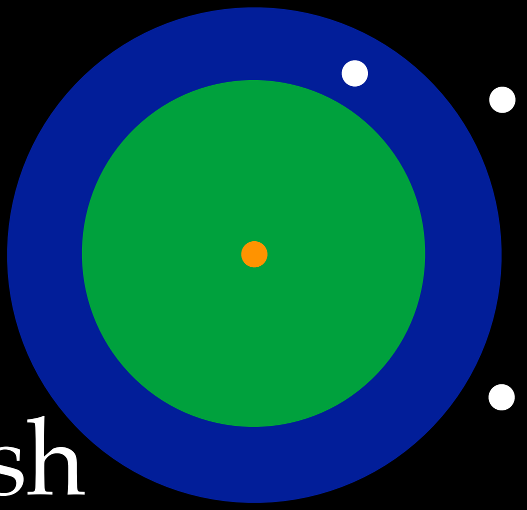
Distance sensitive Bloom filters



- Store collection of S bit vectors
- Given query vector y determine distinguish
  1. exists $x \in S$ within distance r from y, and
  2. all vectors in S have distance at least cr from y
- No requirement outside of these cases; also, in case 2 we allow probability ε of “false positive”

[Kirsch & Mitzenmacher ‘06]

Distance sensitive Bloom filters



- Store collection of S bit vectors
- Given query vector y determine distinguish
  1. exists $x \in S$ within distance r from y, and
  2. all vectors in S have distance at least cr from y
- No requirement outside of these cases; also, in case 2 we allow probability ε of “false positive”

But: Existing solutions also have false negatives…

[Goswami et al. '16]

Distance sensitive Bloom filters

without false negatives

- Consider the single-item case, $S=\{x\}$

- Basic idea: For random $a \in \{-1,+1\}^d$, store $a \cdot x$

  Query y: *If* $|a \cdot x - a \cdot y| \leq r$ answer '1', otherwise '2'

[Goswami et al. ‘16]

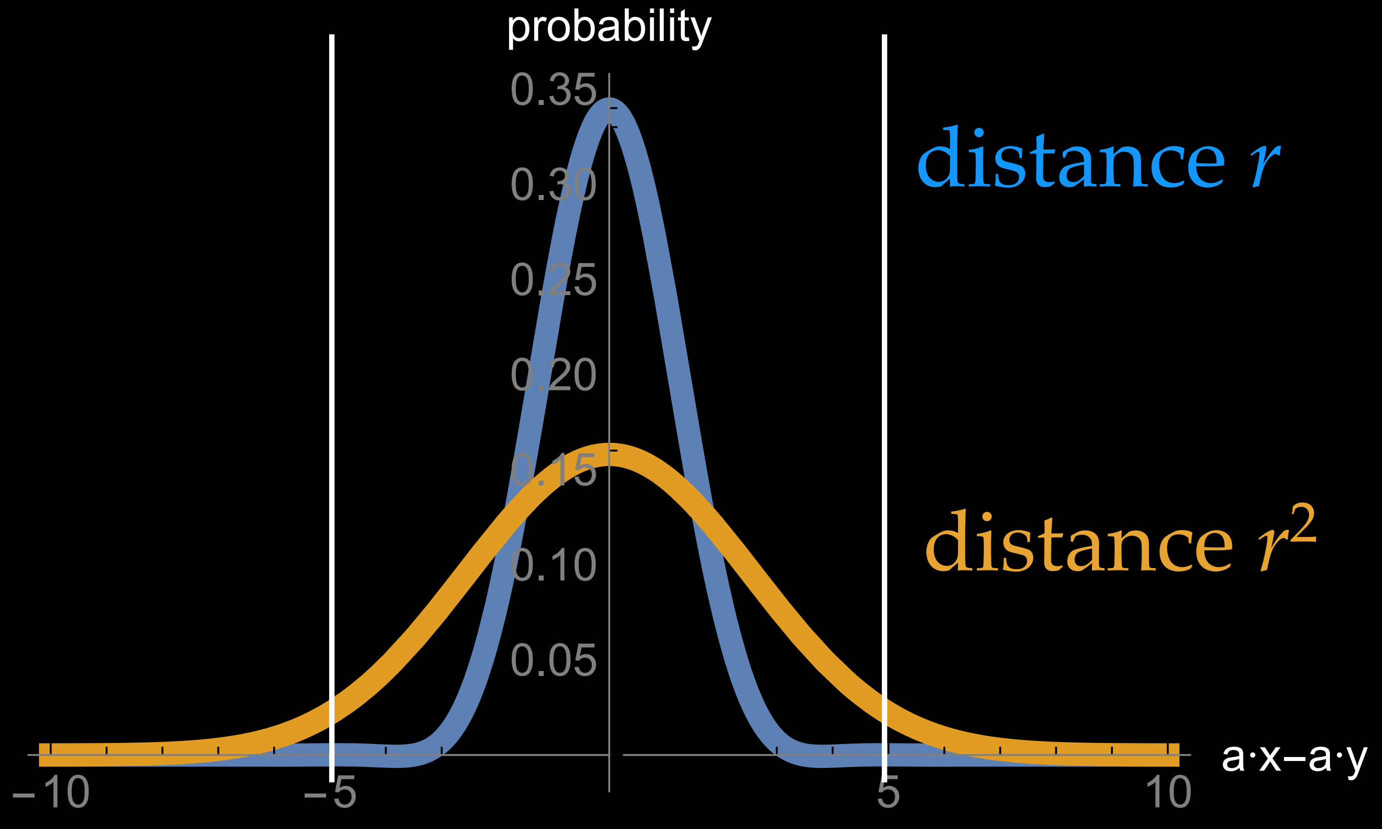
Distance sensitive Bloom filters

without false negatives

- Consider the single-item case, $S=\{x\}$

- Basic idea: For random $a \in \{-1,+1\}^d$, store $a \cdot x$

  Query y: *If* $|a \cdot x - a \cdot y| \le r$ answer ‘1’, otherwise ‘2’



distance r

distance r^2

[Goswami et al. '16]

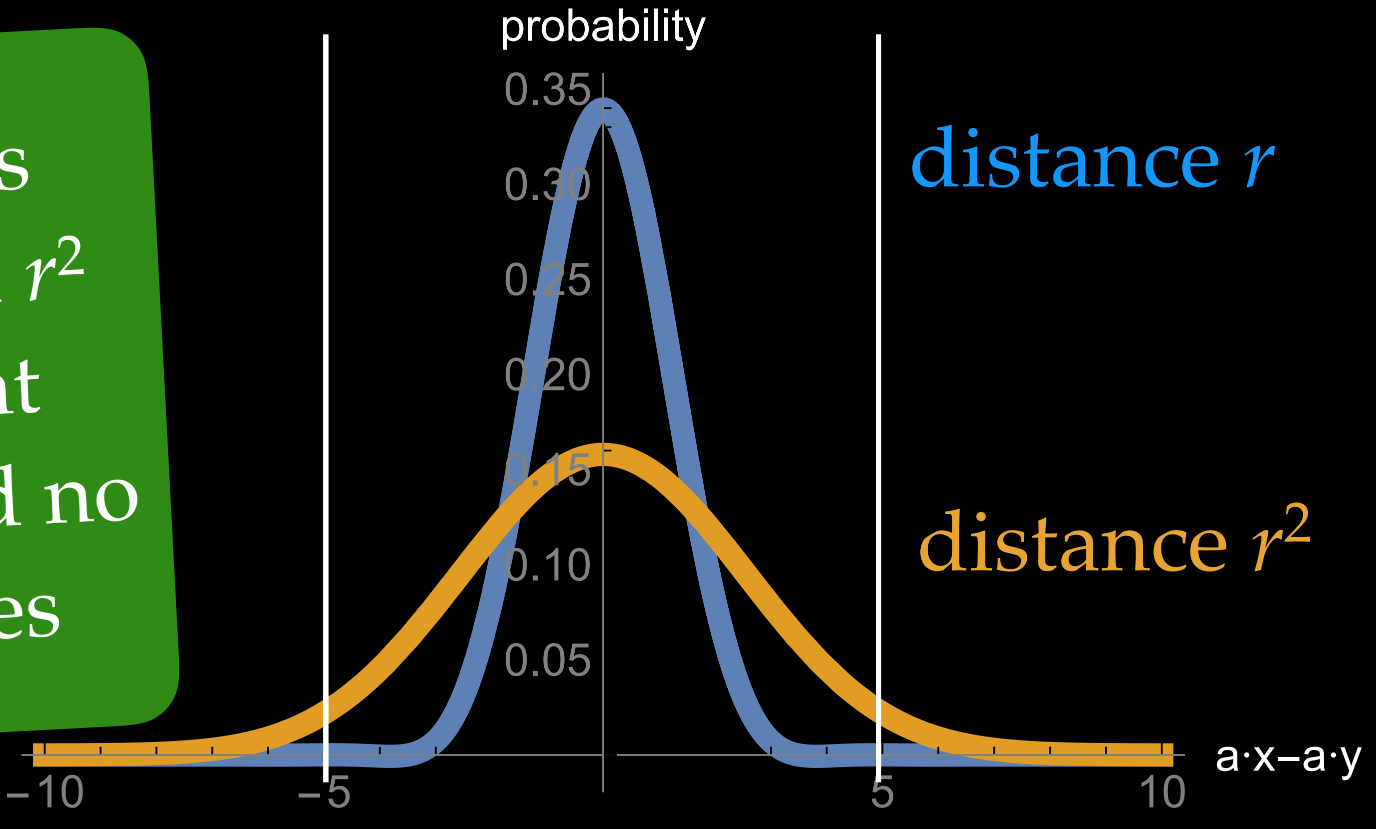
Distance sensitive Bloom filters

without false negatives

- Consider the single-item case, $S=\{x\}$
- Basic idea: For random $a \in \{-1,+1\}^d$, store $a \cdot x$

  Query y: *If* $|a \cdot x - a \cdot y| \leq r$ answer '1', otherwise '2'

Distinguishes distance r and r^2 with constant probability and no false positives



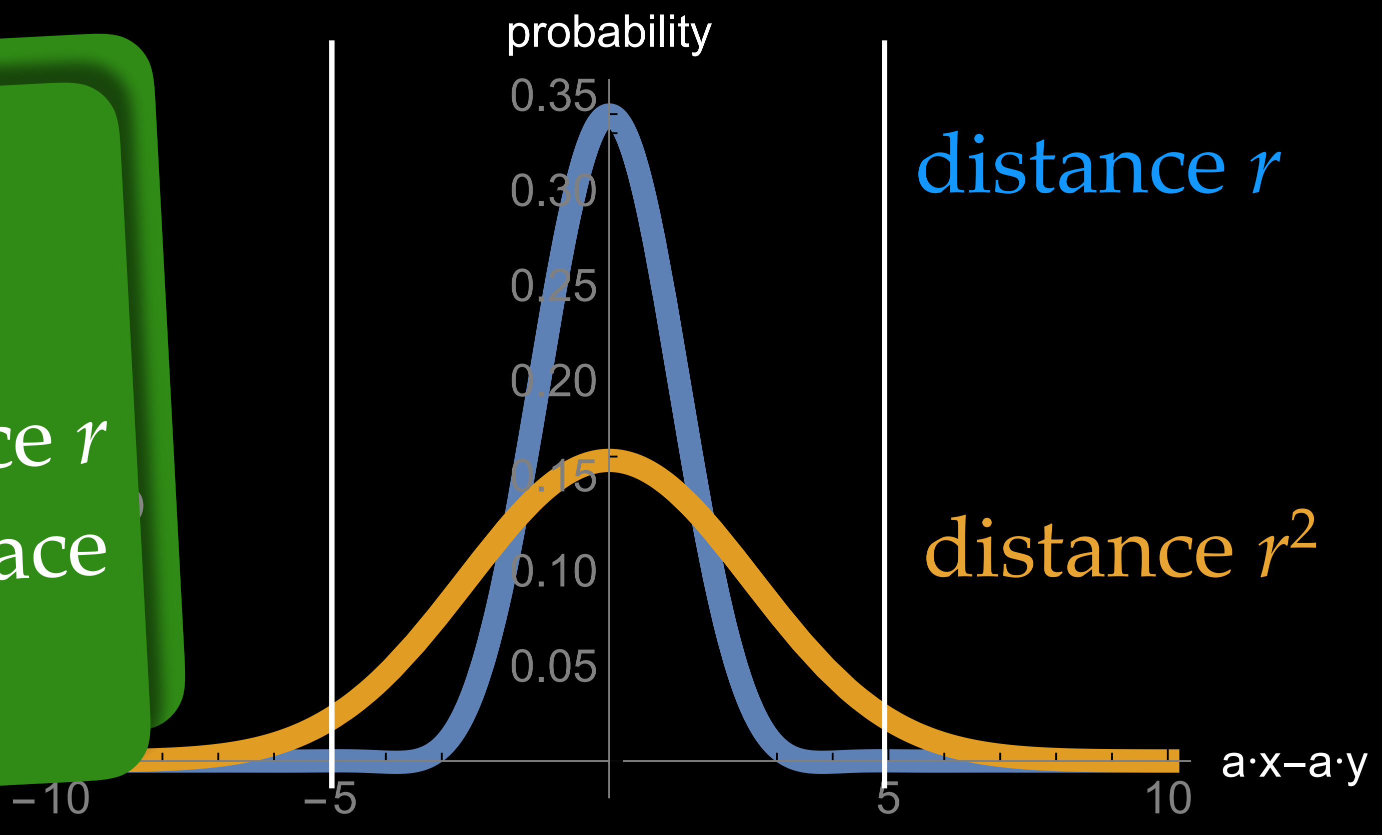
[Goswami et al. '16]

Distance sensitive Bloom filters

without false negatives

- Consider the single-item case, $S=\{x\}$
- Basic idea: For random $a \in \{-1,+1\}^d$, store $a \cdot x$

  Query y: *If* $|a \cdot x - a \cdot y| \leq r$ answer '1', otherwise '2'



In general, distinguish between distance r and cr using space $\tilde{O}(r/(c\text{-}1))$

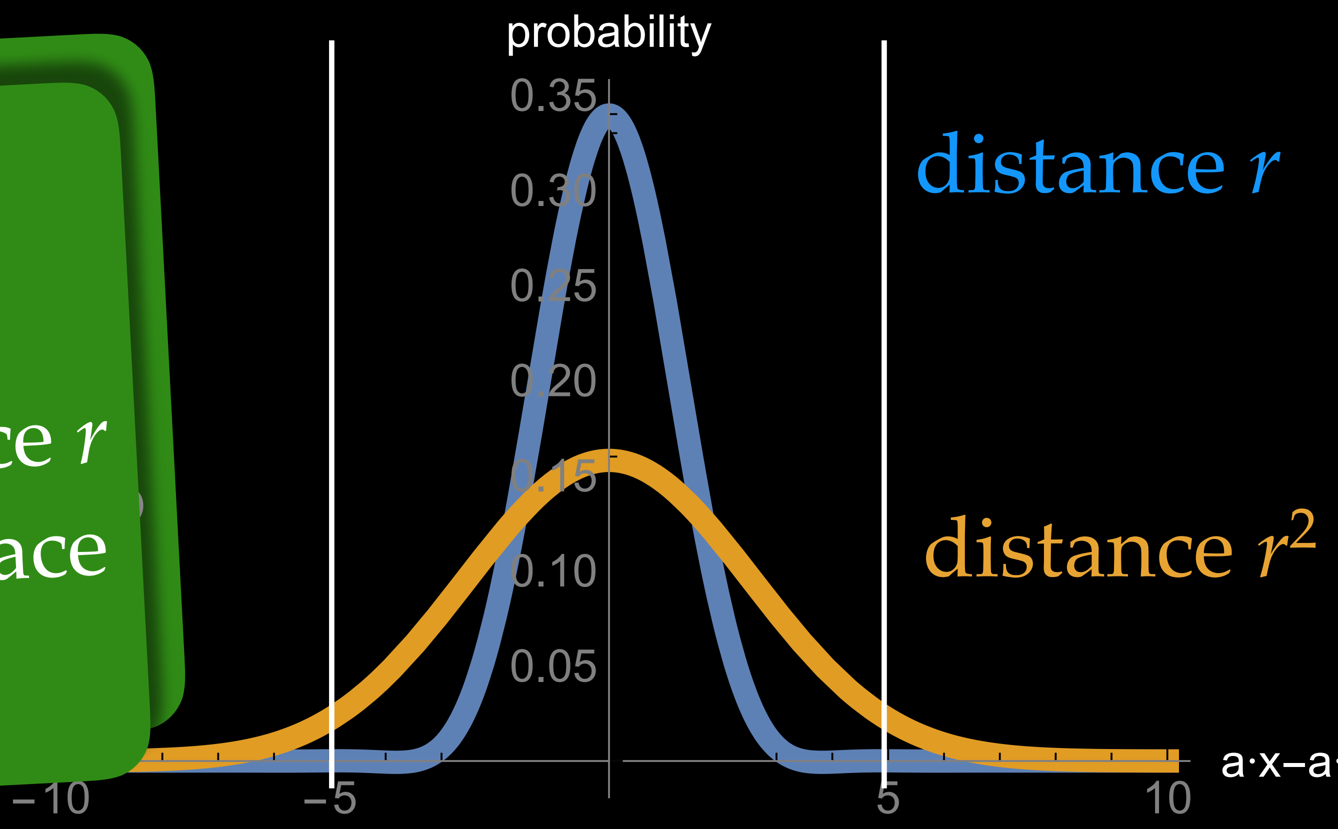
[Goswami et al. '16]

Distance sensitive Bloom filters

without false negatives

- Consider the single-item case, $S=\{x\}$
- Basic idea: For random $a \in \{-1,+1\}^d$, store $a \cdot x$

  Query y: *If* $|a \cdot x - a \cdot y| \leq r$ answer '1', otherwise '2'



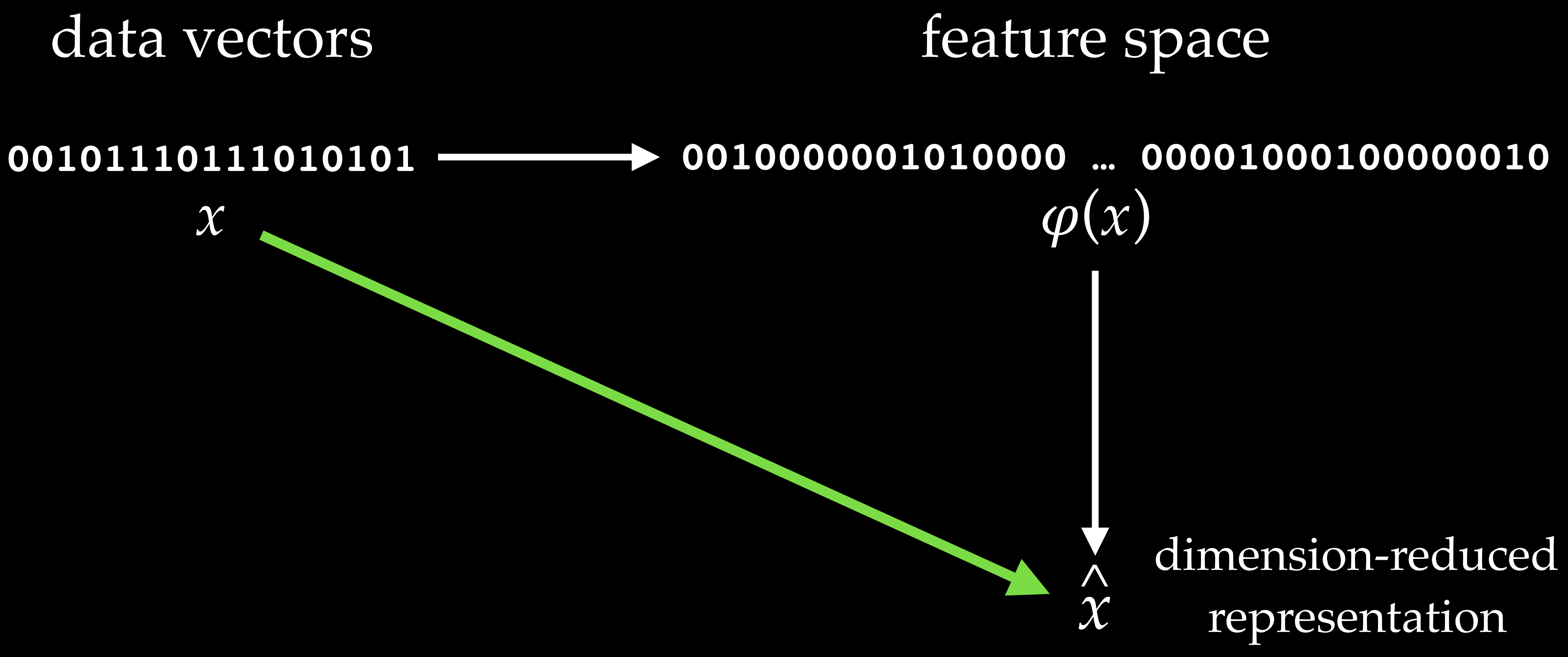
In general, distinguish between distance r and cr using space $\tilde{O}(r/(c-1))$

Space usage essentially optimal

[Karppa et al. '16]

Deterministic feature mappings

for the polynomial kernel



data vectors

feature space

0010111011101 0101

0010000001010000 … 0000100010000010

x

$\varphi(x)$

$\hat{x}$

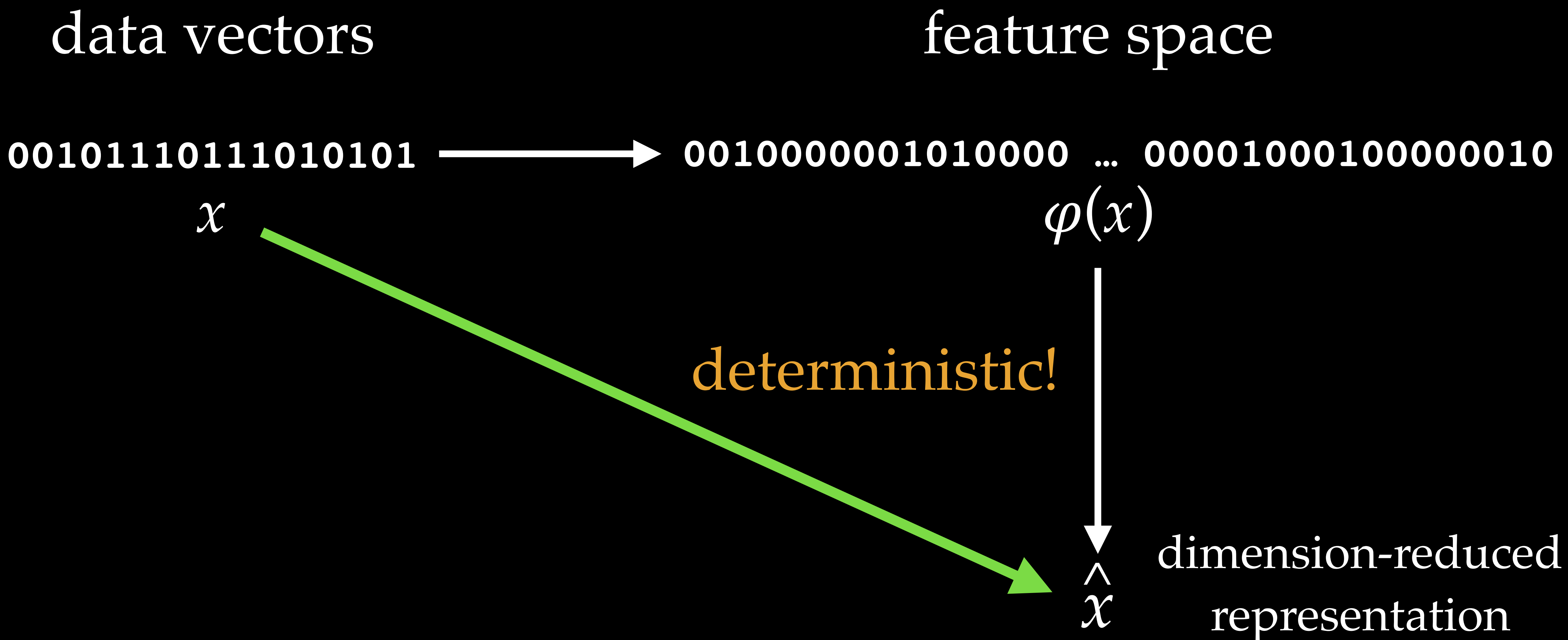
dimension-reduced representation

$$\hat{x} \cdot \hat{y} \approx (x \cdot y)^k$$

[Karppa et al. ‘16]

Deterministic feature mappings

for the polynomial kernel



data vectors

feature space

0010111011101010 1

x

0010000001010000 … 0000100010000010

$\varphi(x)$

deterministic!

$\hat{x}$

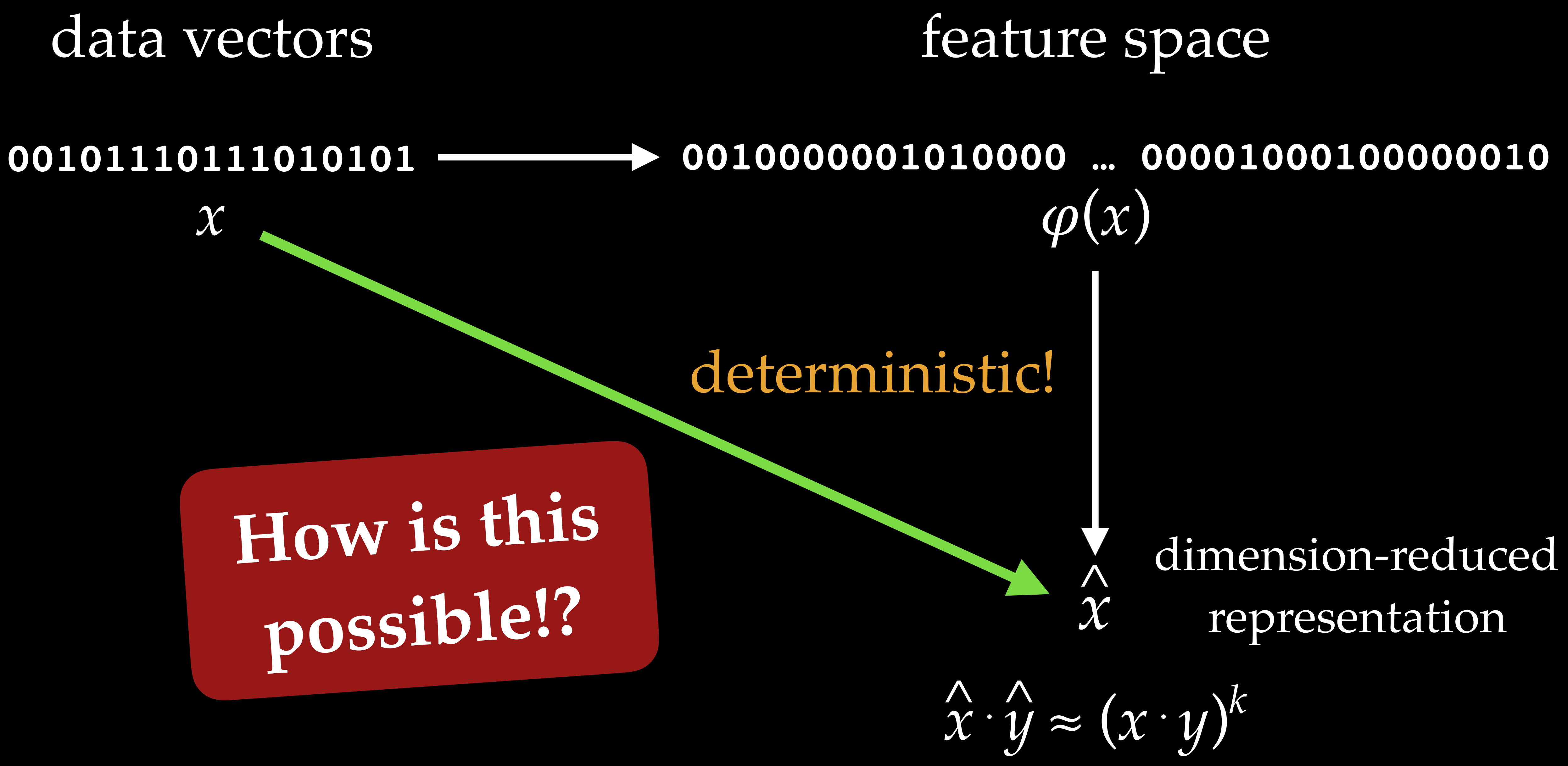
dimension-reduced representation

$$\hat{x} \cdot \hat{y} \approx (x \cdot y)^k$$

[Karppa et al. '16]

Deterministic feature mappings

for the polynomial kernel



data vectors

feature space

0010111011101010 1 → 0010000001010000 … 0000100010000010

x $\varphi(x)$

deterministic!

How is this possible!?

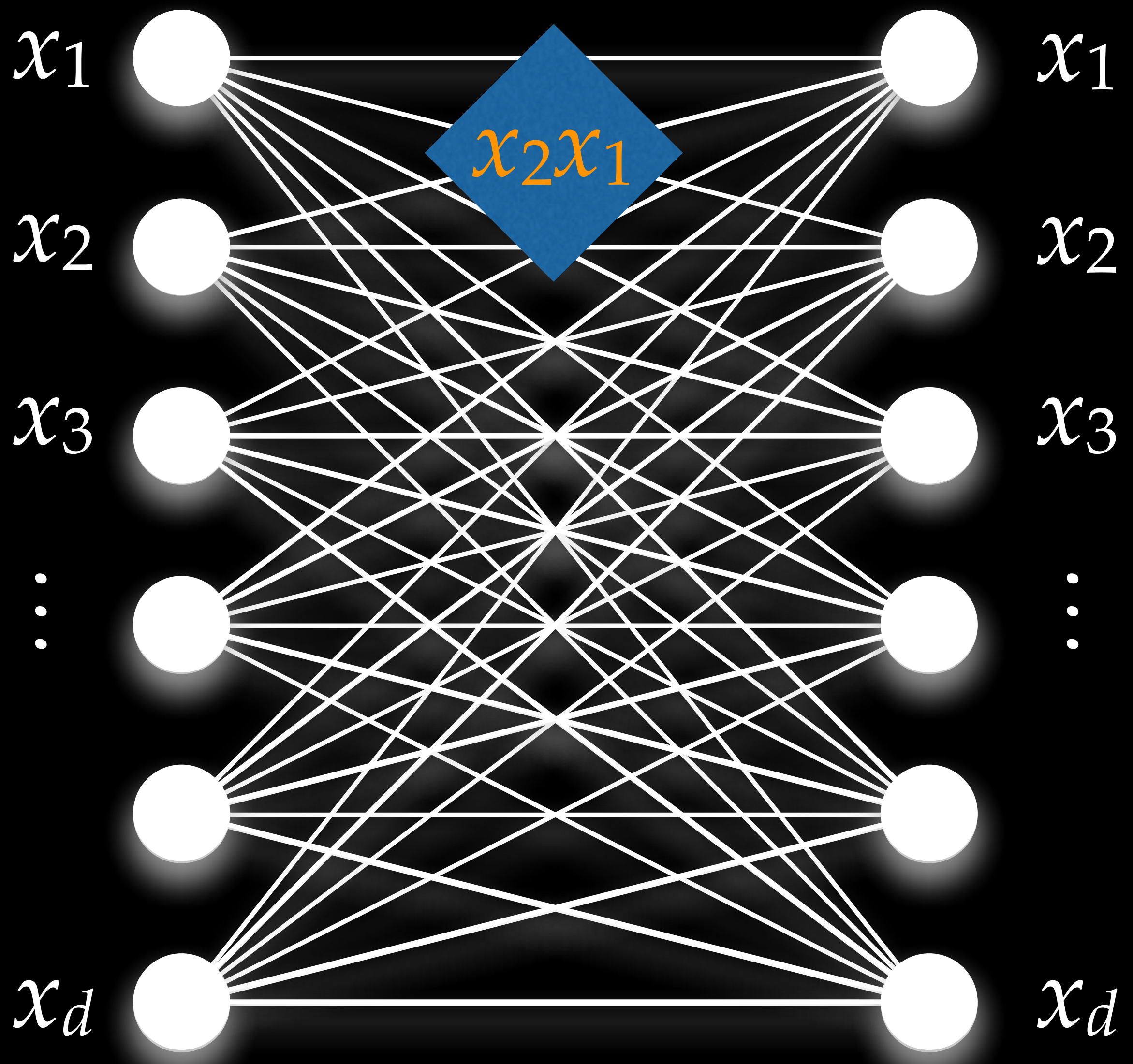
$\hat{x}$ dimension-reduced representation

$$\hat{x} \cdot \hat{y} \approx (x \cdot y)^k$$

[Karppa et al. '16]

How it works

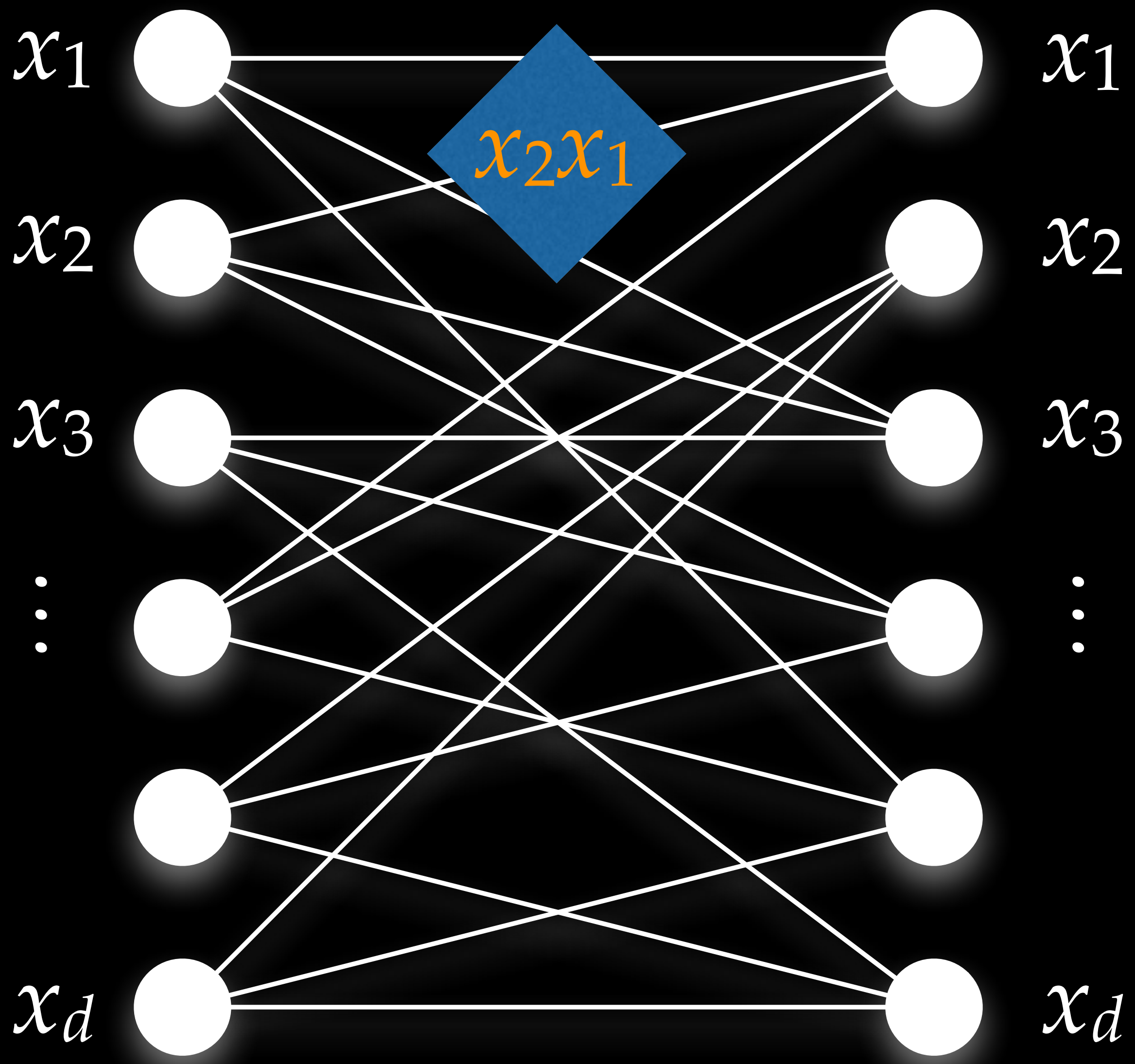
- Considers the kernel $k(x,y) = (x \cdot y)^2$
- Feature space: 1 feature per edge in biclique



[Karppa et al. '16]

How it works

- Considers the kernel $k(x,y) = (x \cdot y)^2$
- Feature space: 1 feature per ~~edge in biclique~~ edge in constant degree expander graph

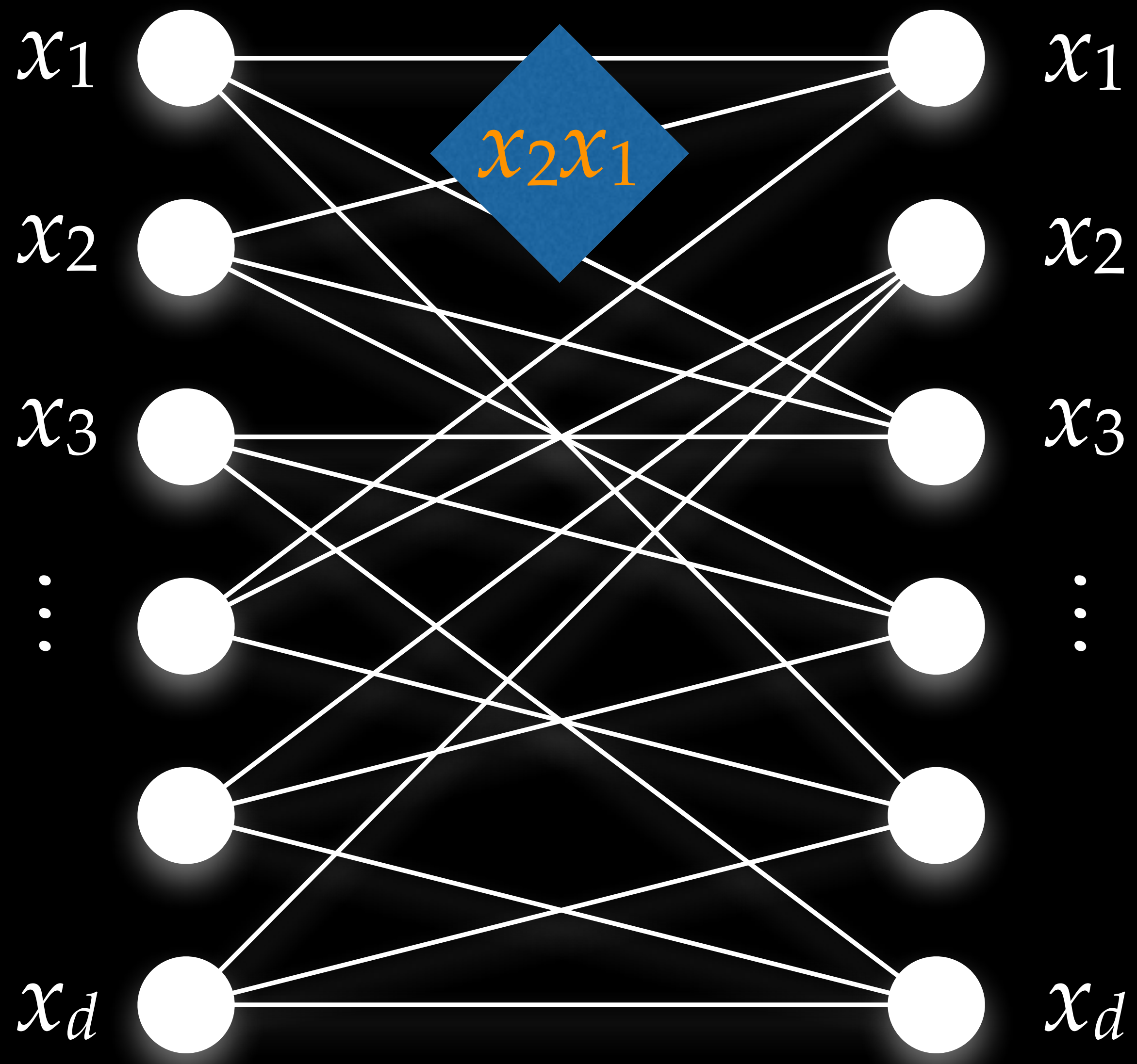


[Karppa et al. '16]

How it works

Proofs only for vectors in $\{-1,+1\}^d$

- Considers the kernel $k(x,y) = (x \cdot y)^2$
- Feature space: 1 feature per ~~edge in biclique~~ edge in constant degree expander graph



Some open questions

What machine learning or knowledge discovery aided decisions can be made "explainable"?

Some open questions

What machine learning or knowledge discovery aided decisions can be made "explainable"?

What ML/KDD algorithms can be sped up by RandNLA methods?

Some open questions

What machine learning or knowledge discovery aided decisions can be made "explainable"?

What ML/KDD algorithms can be sped up by RandNLA methods?

What distance/kernel approximations are possible with *one-sided* error?

Some open questions

What machine learning or knowledge discovery aided decisions can be made "explainable"?

What ML/KDD algorithms can be sped up by RandNLA methods?

What kernels expansions have efficient *deterministic* approximations?

What distance/kernel approximations are possible with *one-sided* error?

Thank you for your attention!

Acknowledgement of economic support:

erc

European Research Council

Established by the European Commission

Scalable Similarity Search

PS. Seeking a post-doc to start in 2017!