Variable Bit Quantisation for LSH

Sean Moran

School of Informatics
University of Edinburgh

TU Graz, August 2013

Variable Bit Quantisation for LSH

Fast search in large-scale datasets

Locality Sensitive Hashing (LSH)

Variable Bit Quantisation (VBQ)

Evaluation

Summary

Variable Bit Quantisation for LSH

Fast search in large-scale datasets

Locality Sensitive Hashing (LSH)

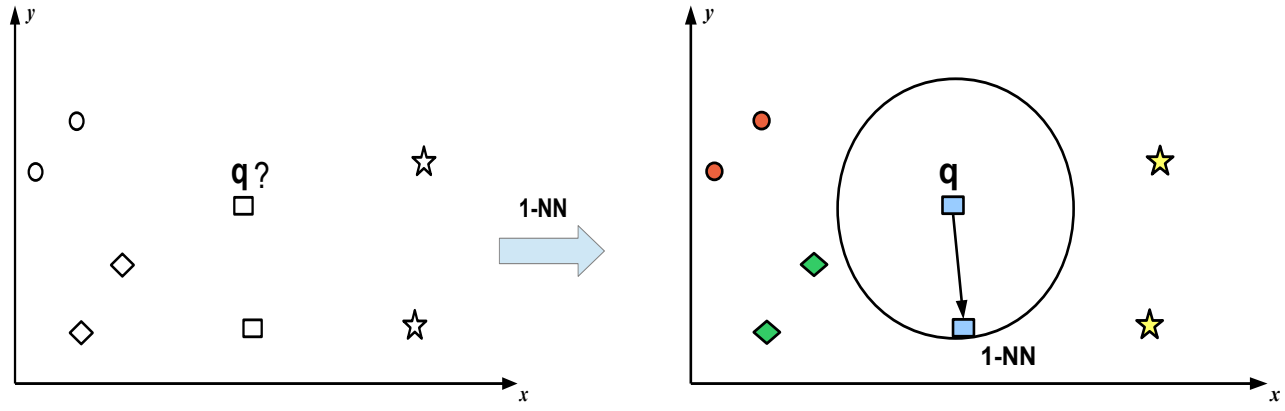
Variable Bit Quantisation (VBQ)

Evaluation

Summary

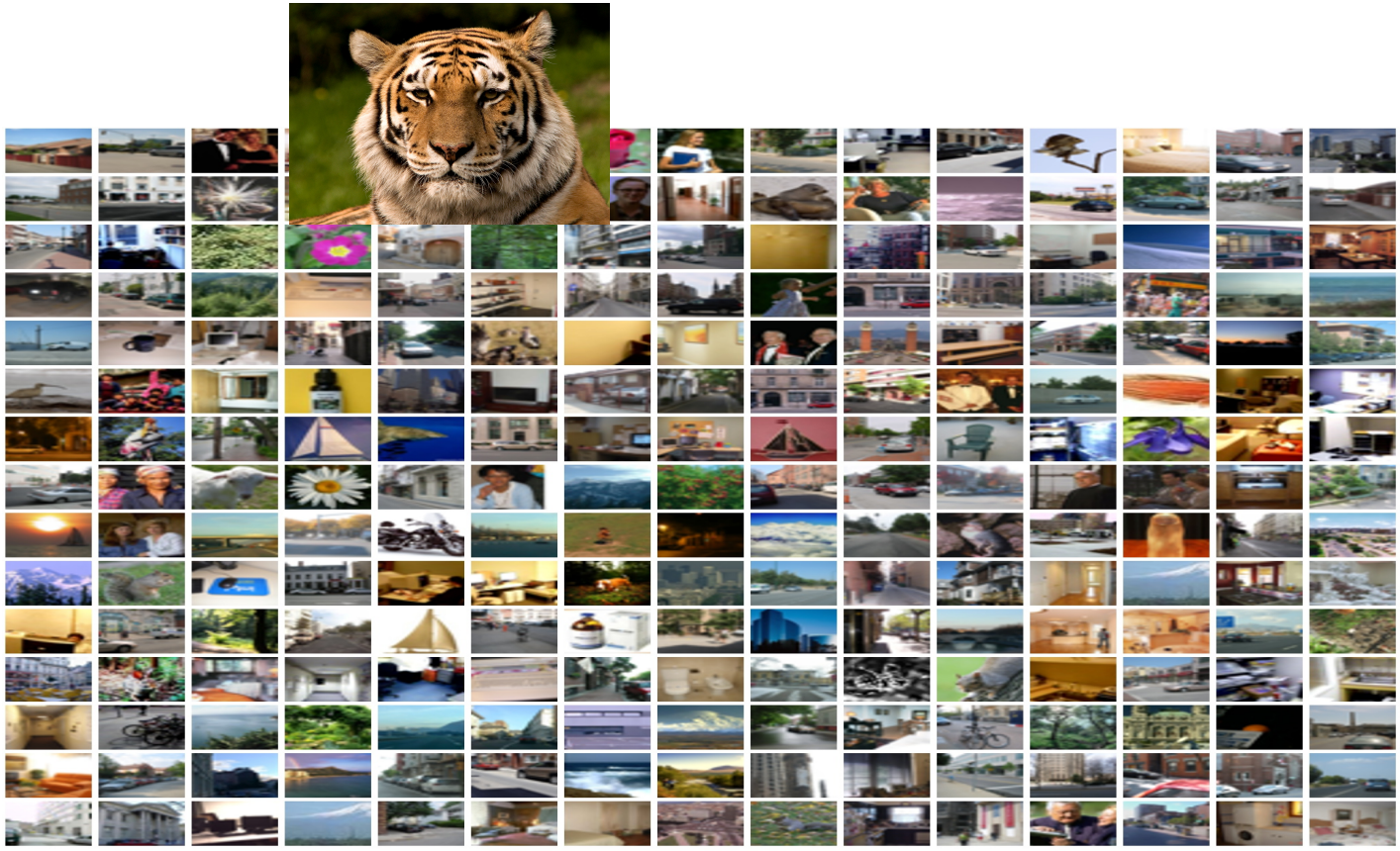
Fast search in large-scale datasets

- Problem: retrieve nearest neighbour(s) to a given query item



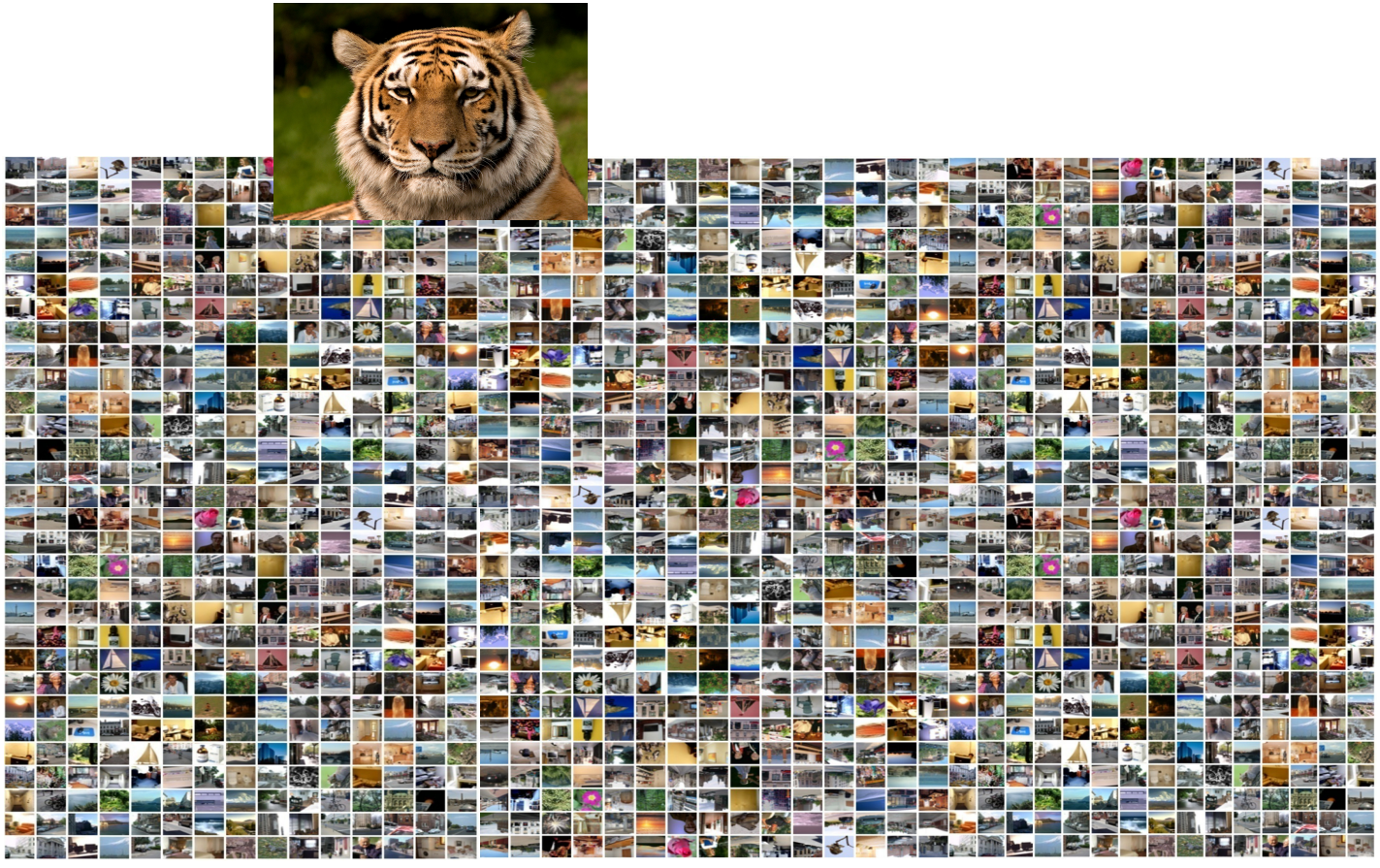
- Naïve approach: compare query to all N database items
  - Scales linearly $\mathcal{O}(N)$ with the size of the database
  - Impractical for all but the smallest of databases

Fast search in large-scale datasets



[1] M. Norouzi and D. Fleet. *Minimal Loss Hashing*. ICML '11.

Fast search in large-scale datasets



[1] M. Norouzi and D. Fleet. *Minimal Loss Hashing*. ICML '11.

Fast search in large-scale datasets

[1] M. Norouzi and D. Fleet. *Minimal Loss Hashing*. ICML '11.

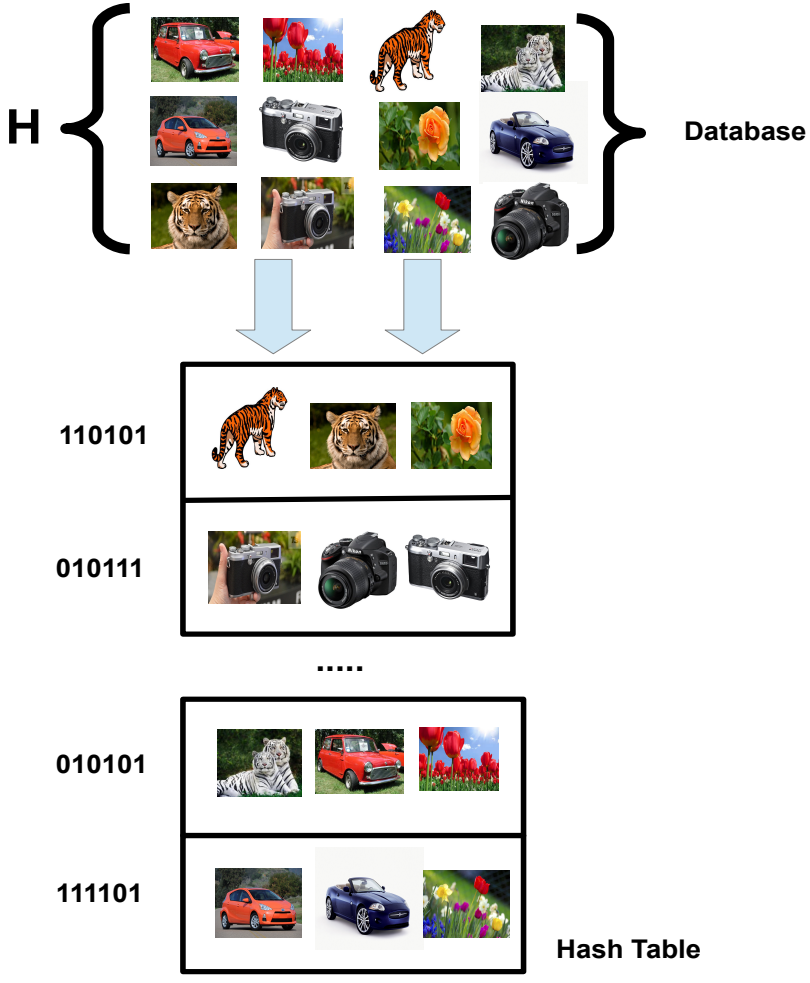
Approximate Nearest Neighbour

- An approximate nearest neighbour may almost be as good as the exact neighbour
- Do not guarantee to return the true nearest neighbour in every case, in return for improved speed or memory savings
- We can solve the nearest neighbour problem by enumerating all approximate nearest neighbours and returning the closest point.

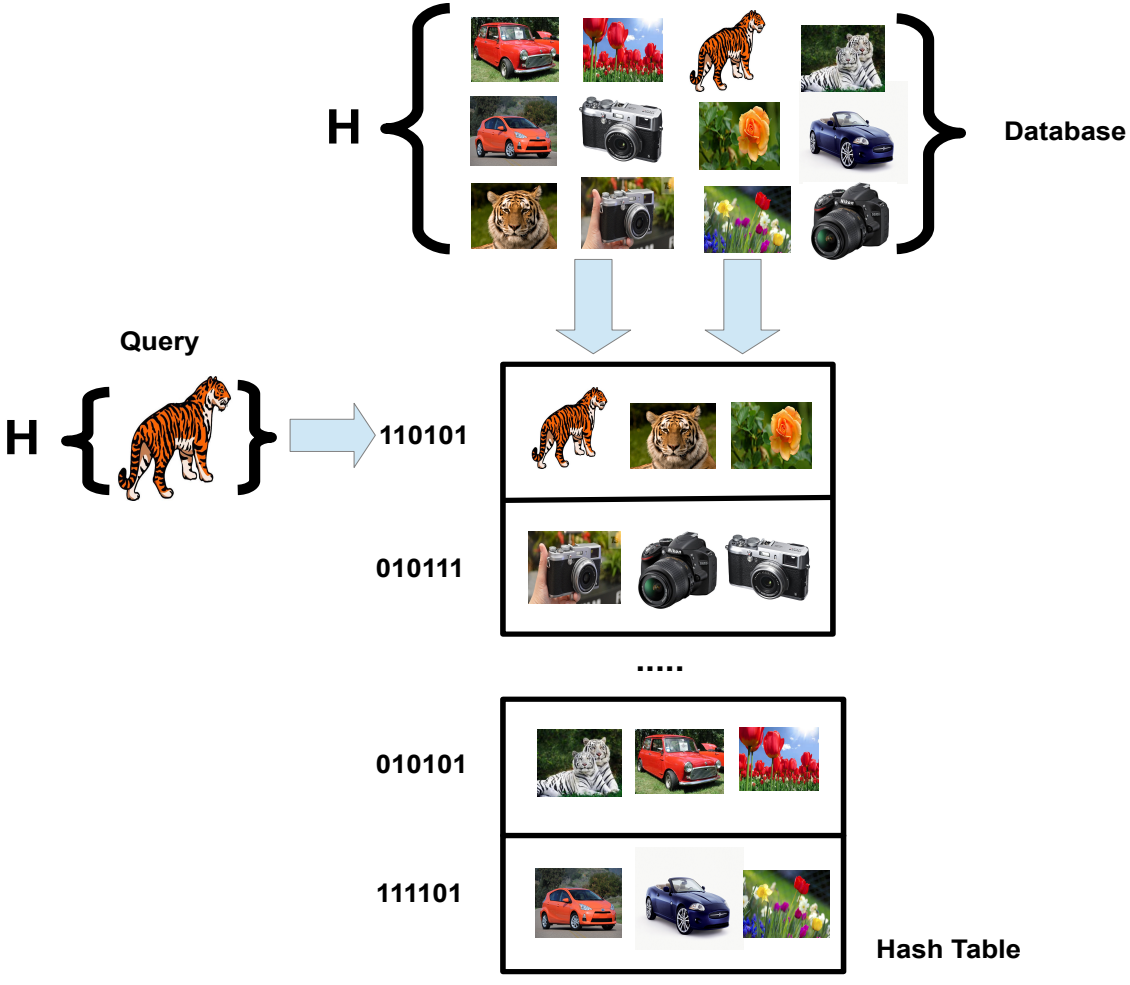
Hashing-based search using binary codes

Database

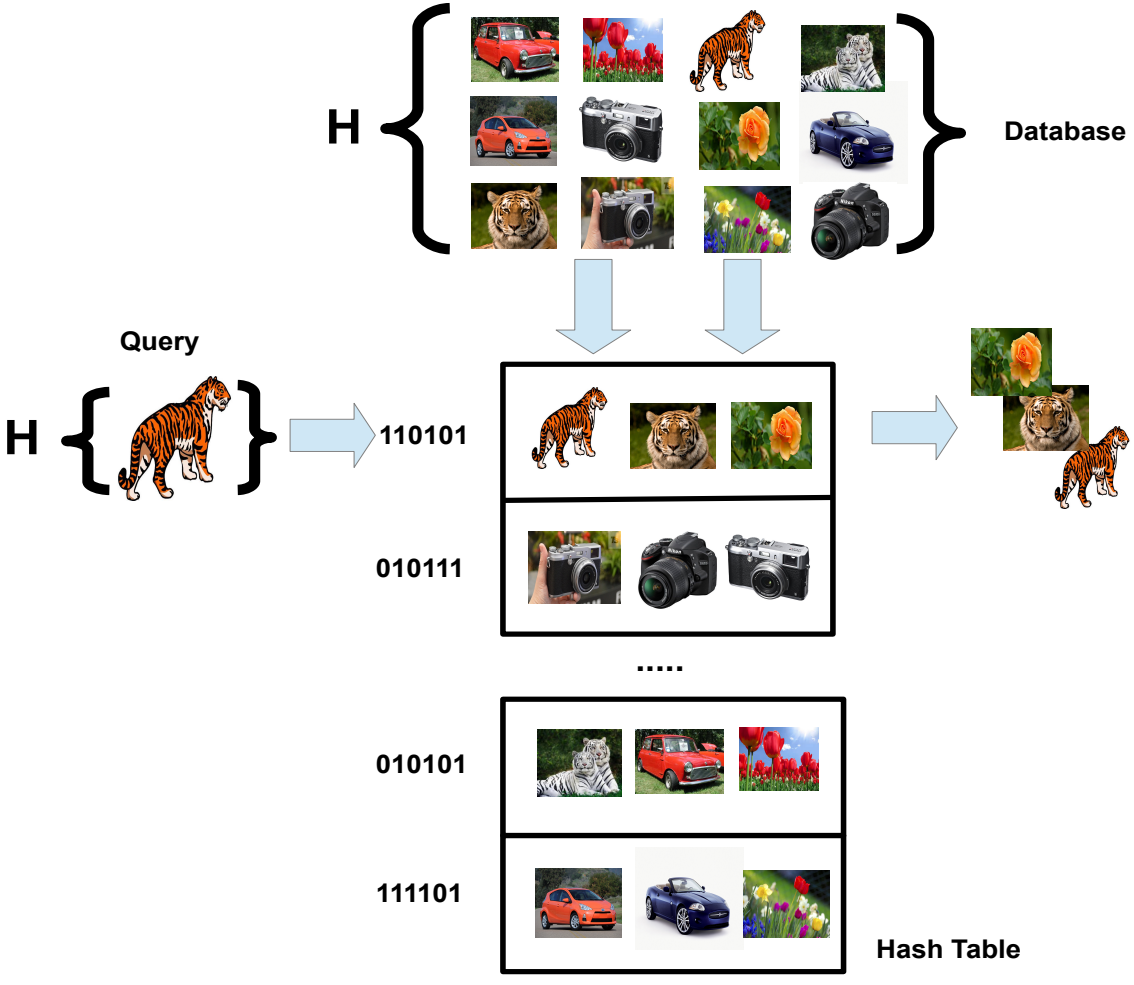
Hashing-based search using binary codes



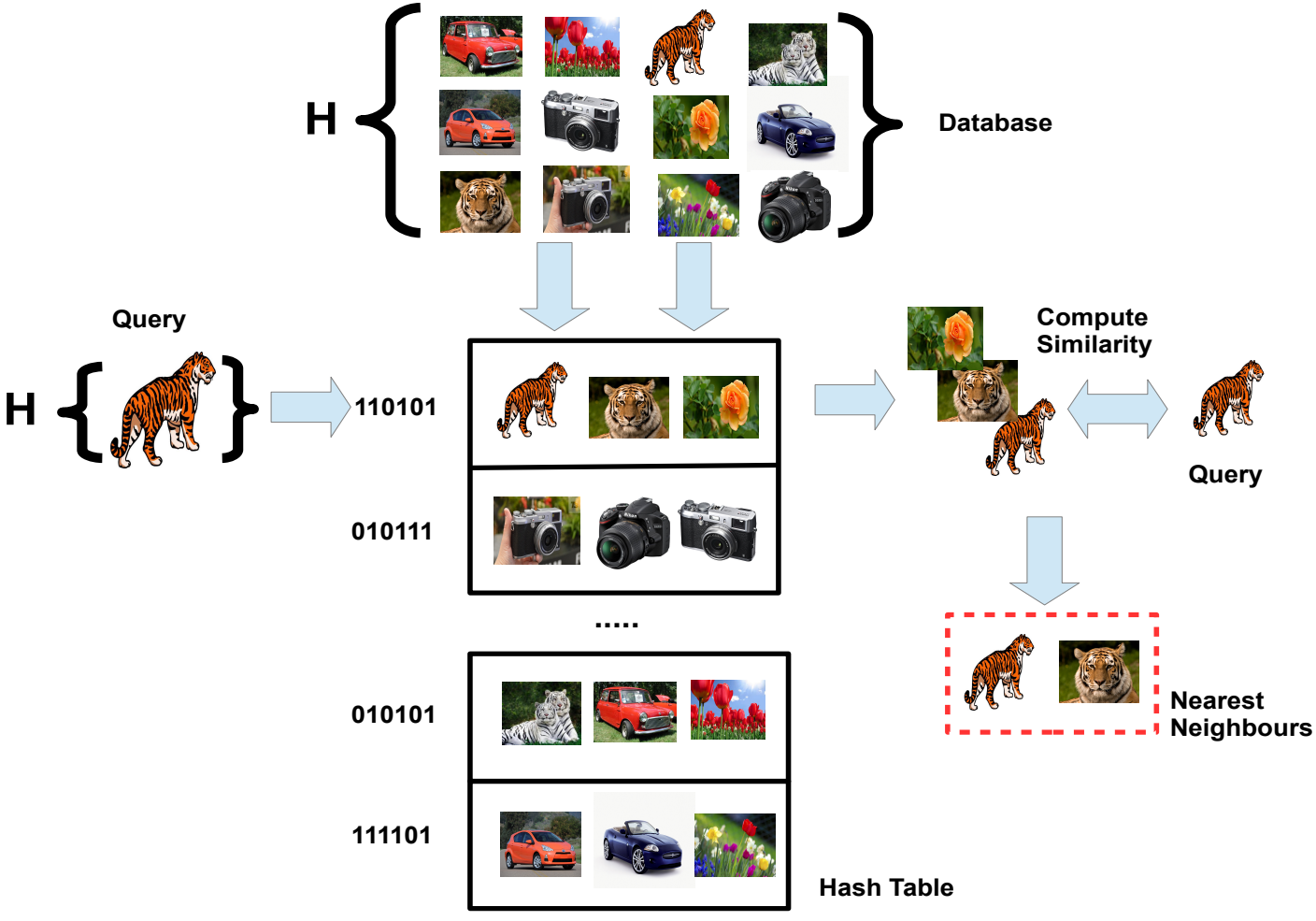
Hashing-based search using binary codes



Hashing-based search using binary codes



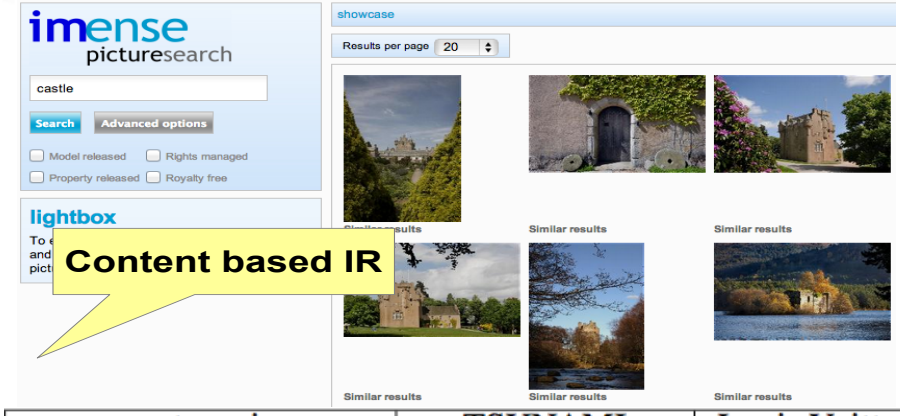
Hashing-based search using binary codes

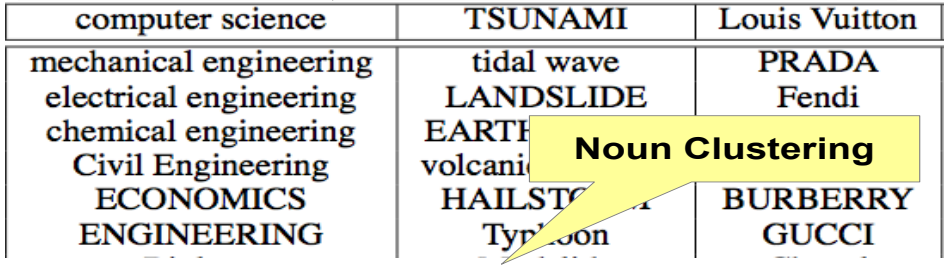


Why transform our data to binary codes?

- Constant O(1) query time with respect to the dataset size.
- Binary code comparison requires few machine instructions (XOR followed by popcount).
- Binary codes are extremely compact e.g. 16Mb to store 1 million, 128 bit encoded data points.

Applications





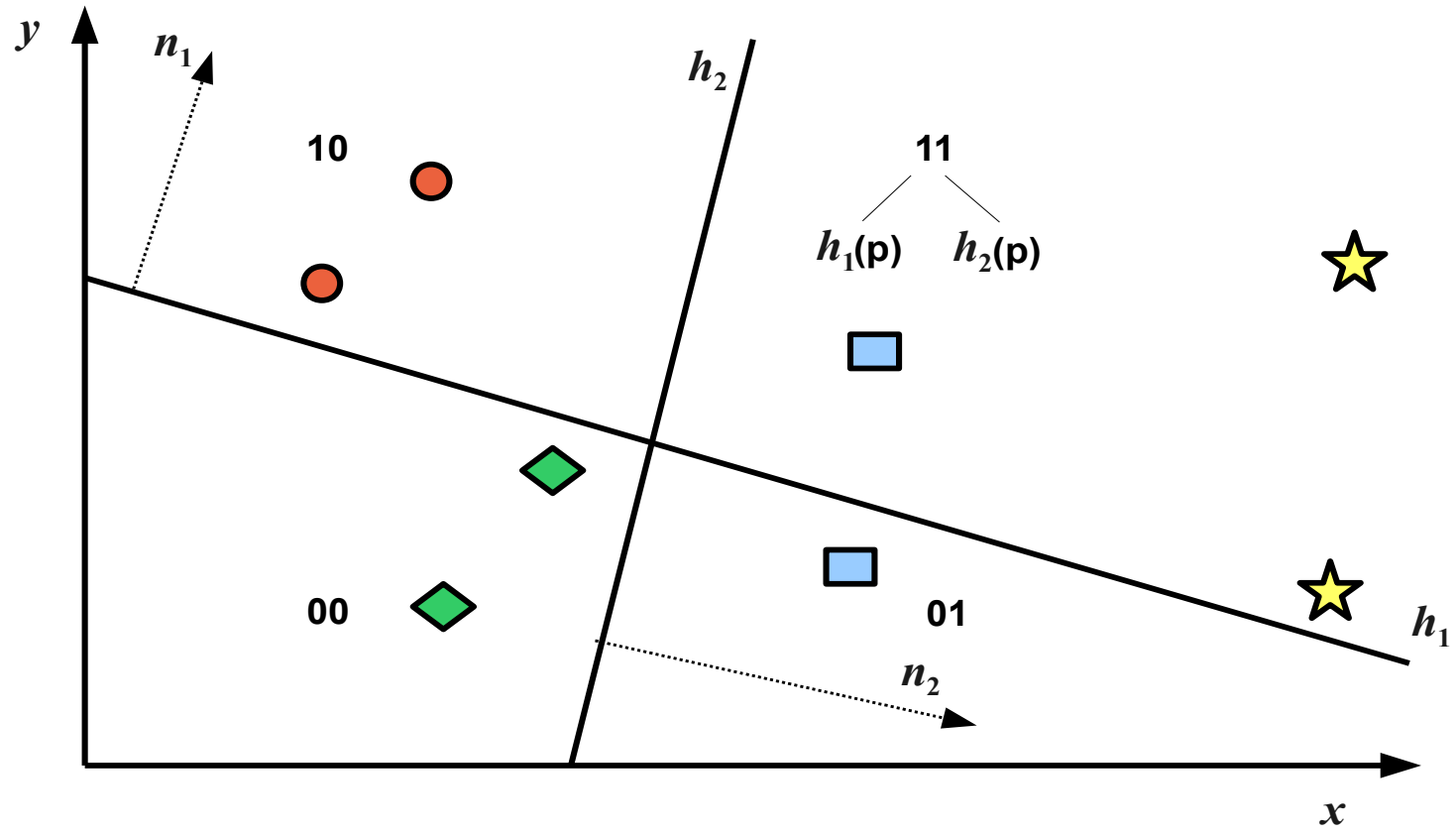
Variable Bit Quantisation for LSH

Computing similarity preserving binary codes

- Locality Sensitive Hashing (LSH) [1]:
  - Randomised algorithm for approximate nearest neighbour search
  - Generates similarity preserving binary codes (fingerprints) for a dataset: data points that are close together are likely to have similar fingerprints.
  - Compare similarity between data points based upon the fingerprints.
  - Preserved similarity depends on the selected hash family

[1] P. Indyk and R. Motwani. *Approximate nearest neighbors: removing the curse of dimensionality*. In STOC '98.

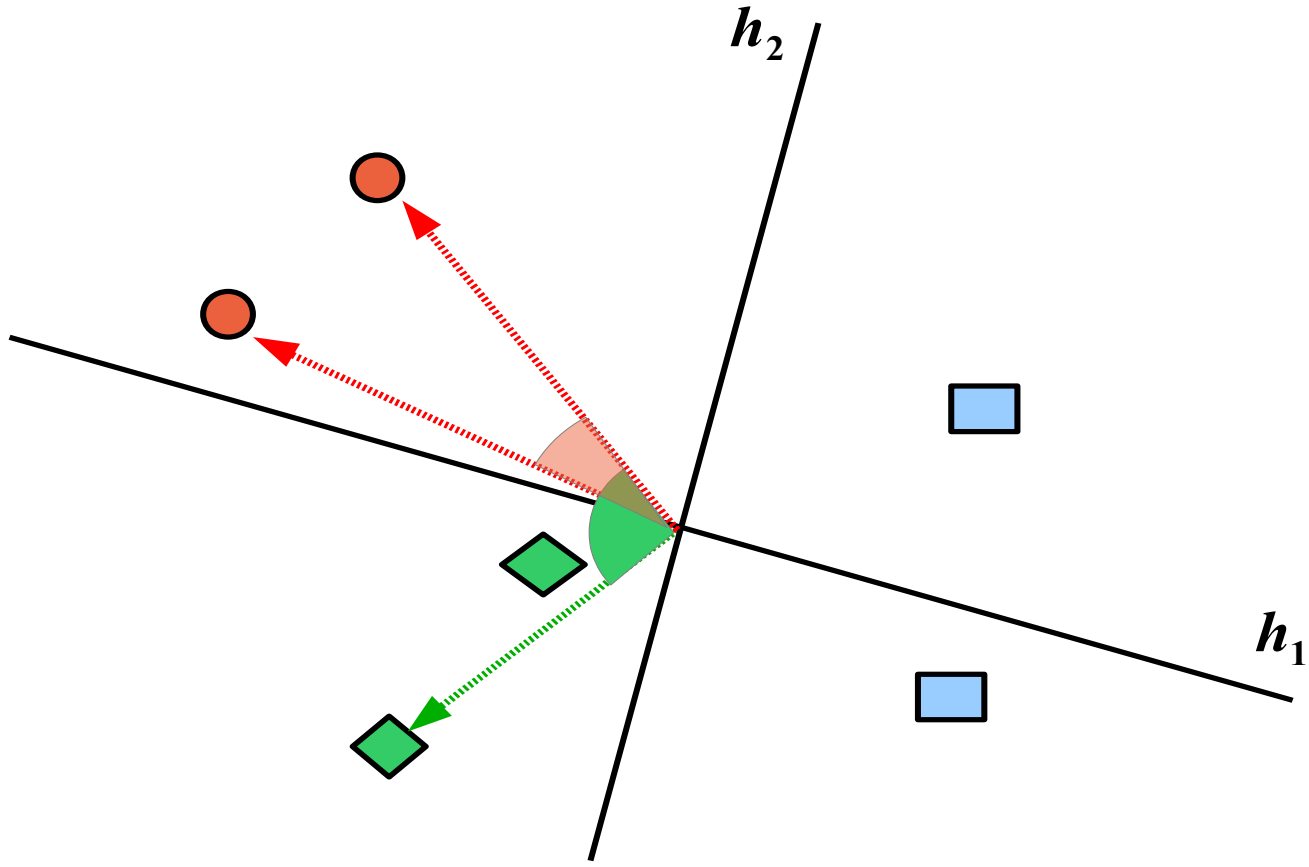
Locality Sensitive Hashing (Cosine Similarity)



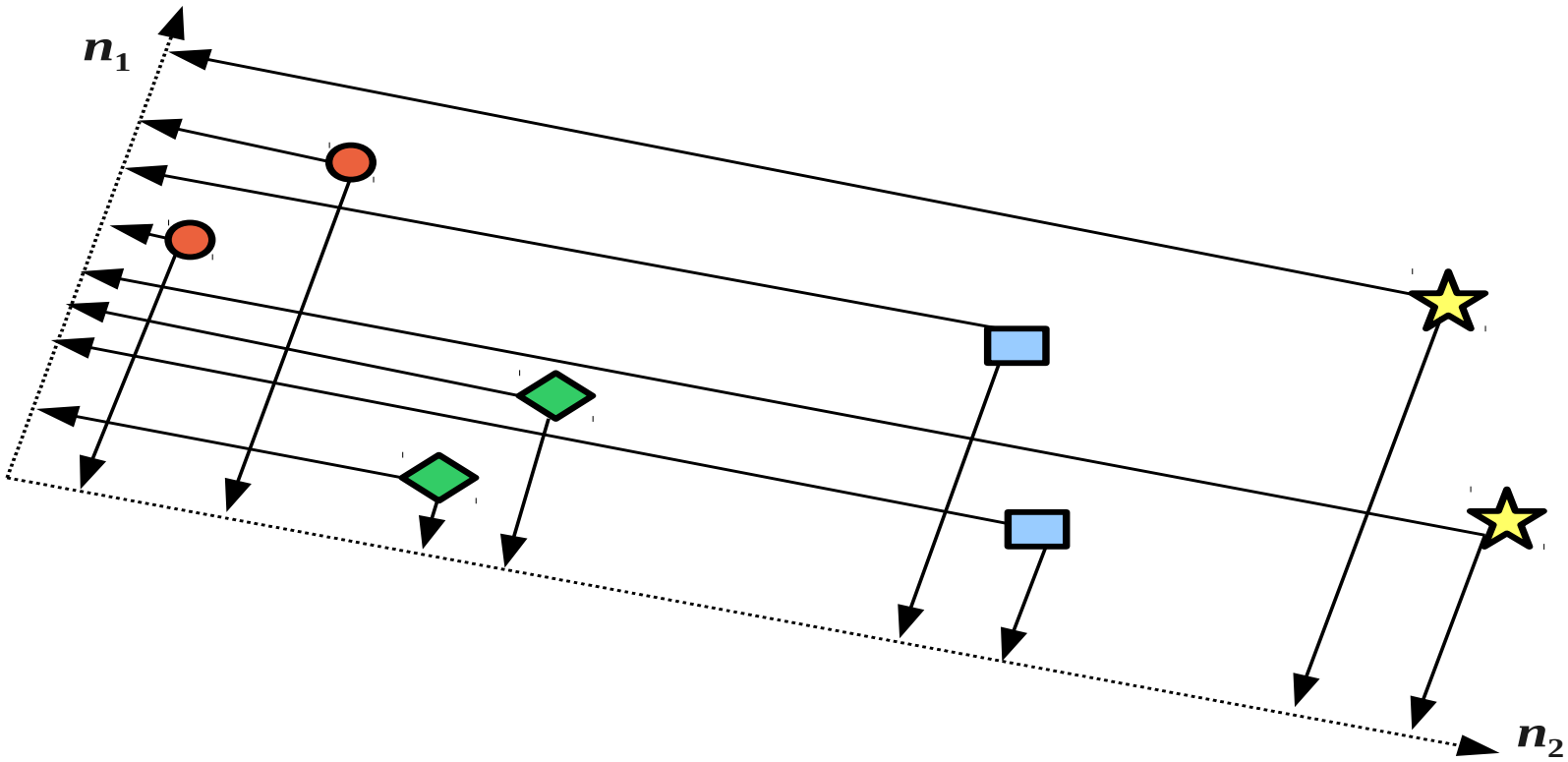
- Vanilla LSH: Each hyperplane gives 1 bit to the bit encoding for a data point

Preserving cosine similarity in the binary representation

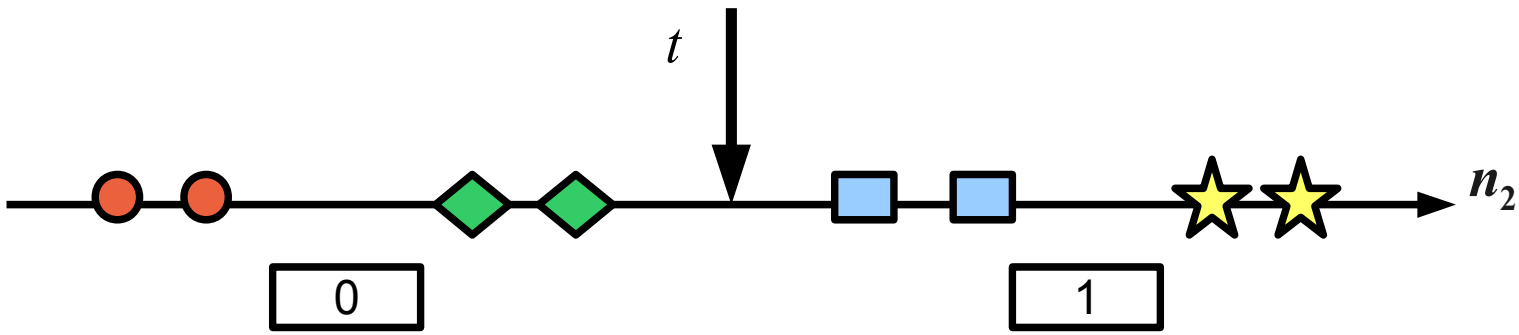
- Two vectors' bits match with probability proportional to the cosine of the angle between them:



Step 1: Low Dimensional Projection



Step 2: Single Bit Quantisation (SBQ)



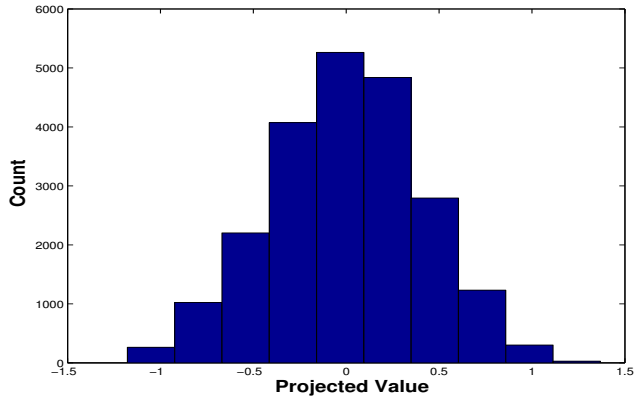
Plus many more...

- Kernel methods [1]
- Spectral methods [2] [3]
- Neural networks [4]
- Loss based methods [5]
- **All use single bit quantisation (SBQ)...**

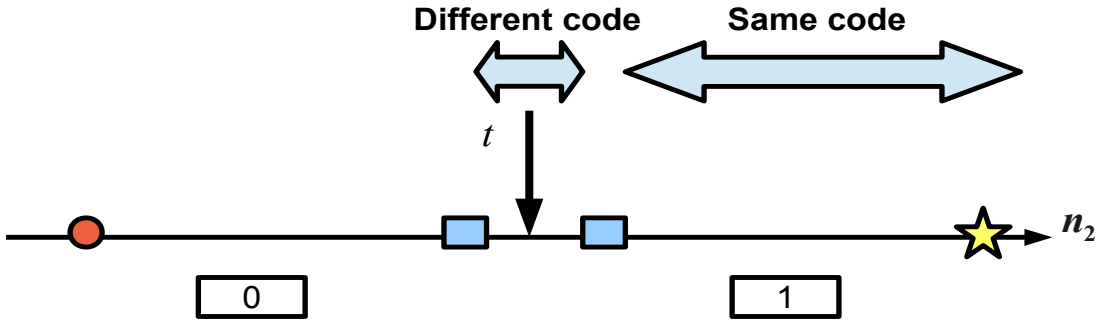
[1] M. Raginsky and S. Lazebnik. *Locality-sensitive binary codes from shift-invariant kernels*. In NIPS '09.
[2] Y. Weiss and A. Torralba and R. Fergus. *Spectral Hashing*. NIPS '08.
[3] J. Wang and S. Kumar and SF. Chang. *Semi-supervised hashing for large-scale search*. PAMI '12.
[4] R. Salakhutdinov and G. Hinton. *Semantic Hashing*. NIPS '08.
[5] B. Kulis and T. Darrell. *Learning to Hash with Binary Reconstructive Embeddings*. NIPS '09.

Problem 1: SBQ leads to high quantisation errors

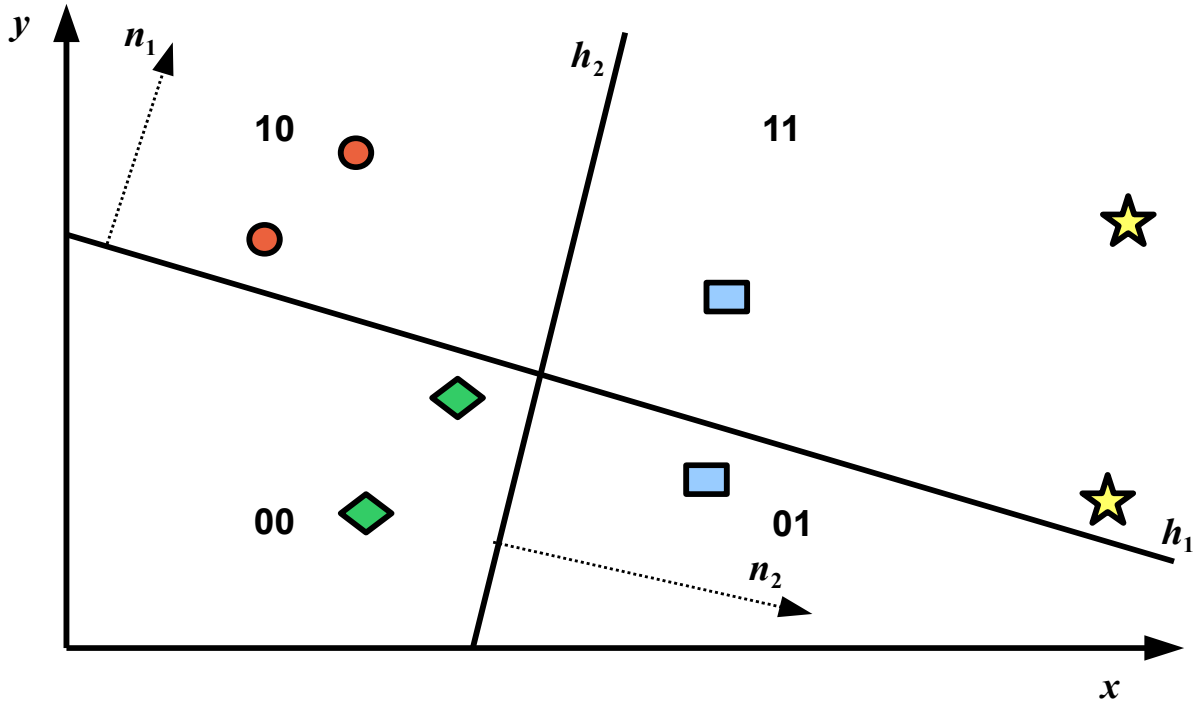
- Highest point density typically occurs around zero:



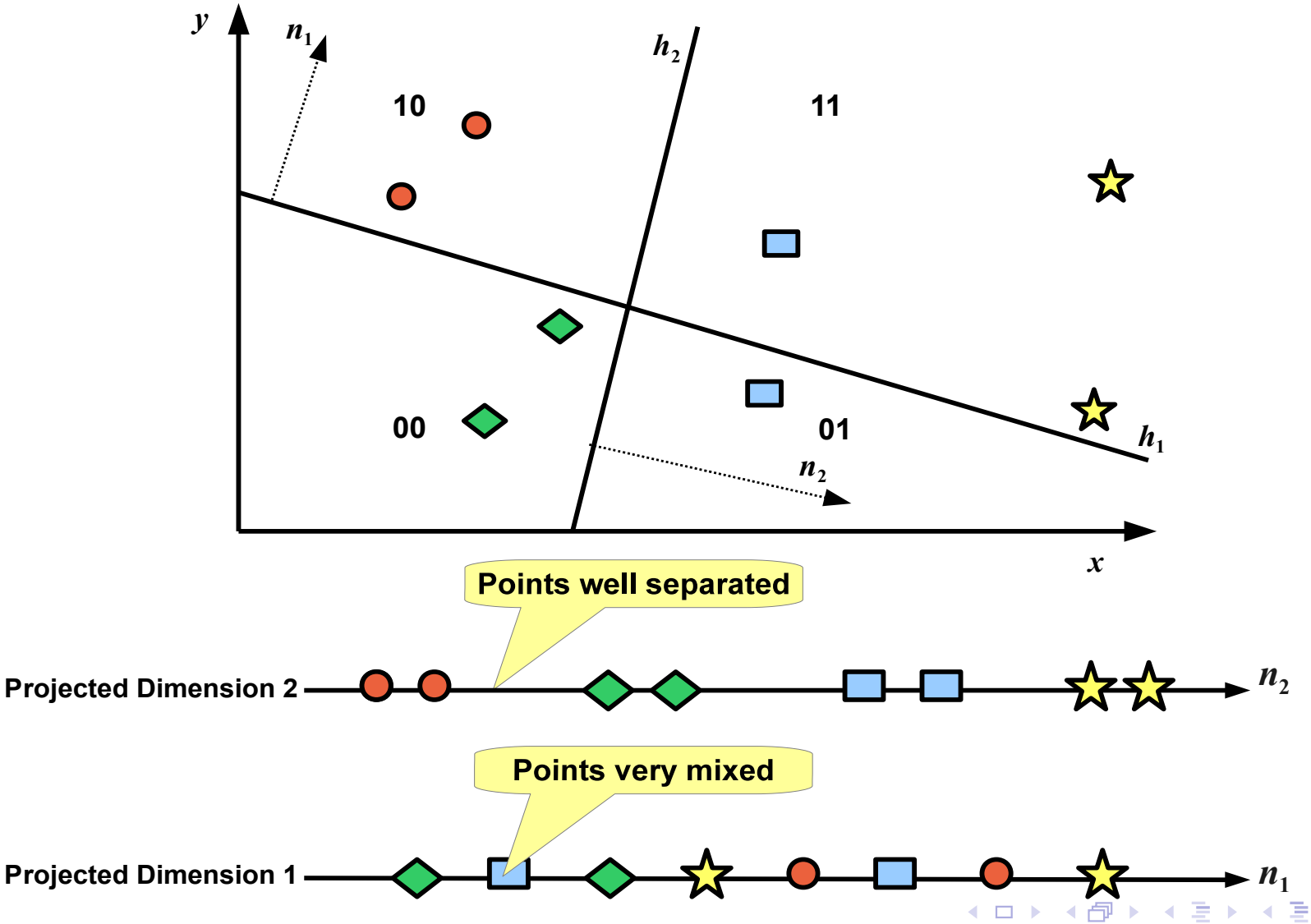
- Closer points can have a greater Hamming distance than distant points:



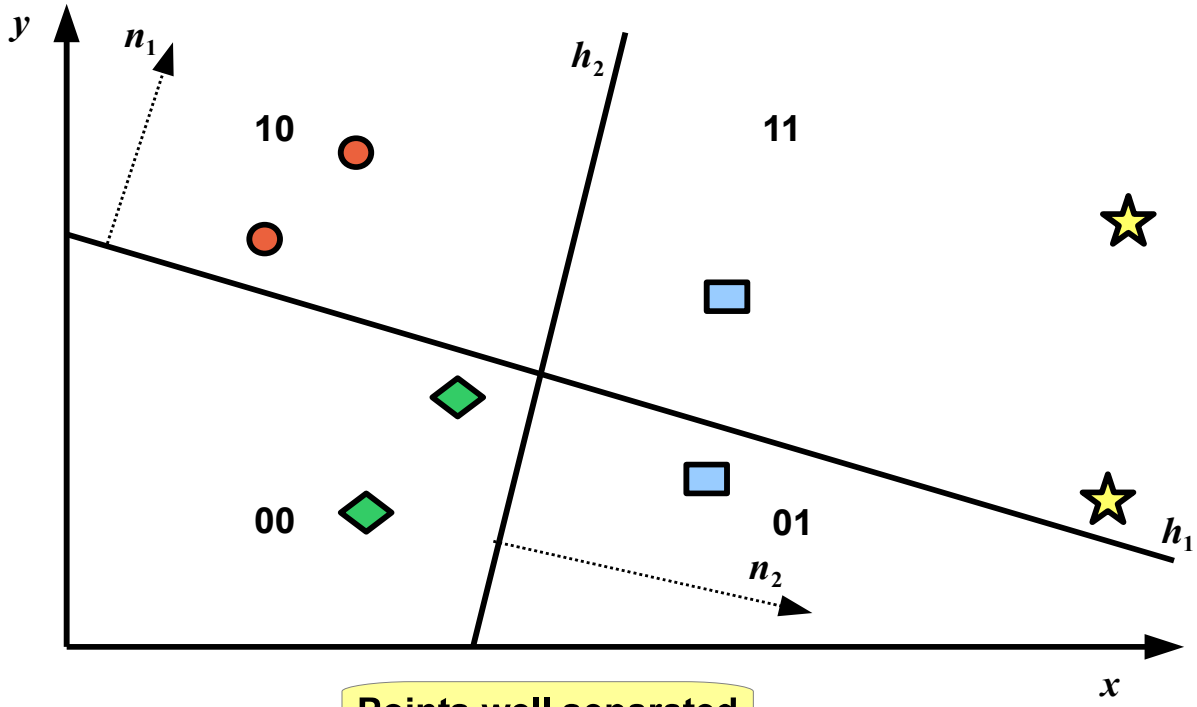
Problem 2: some hyperplanes are better than others

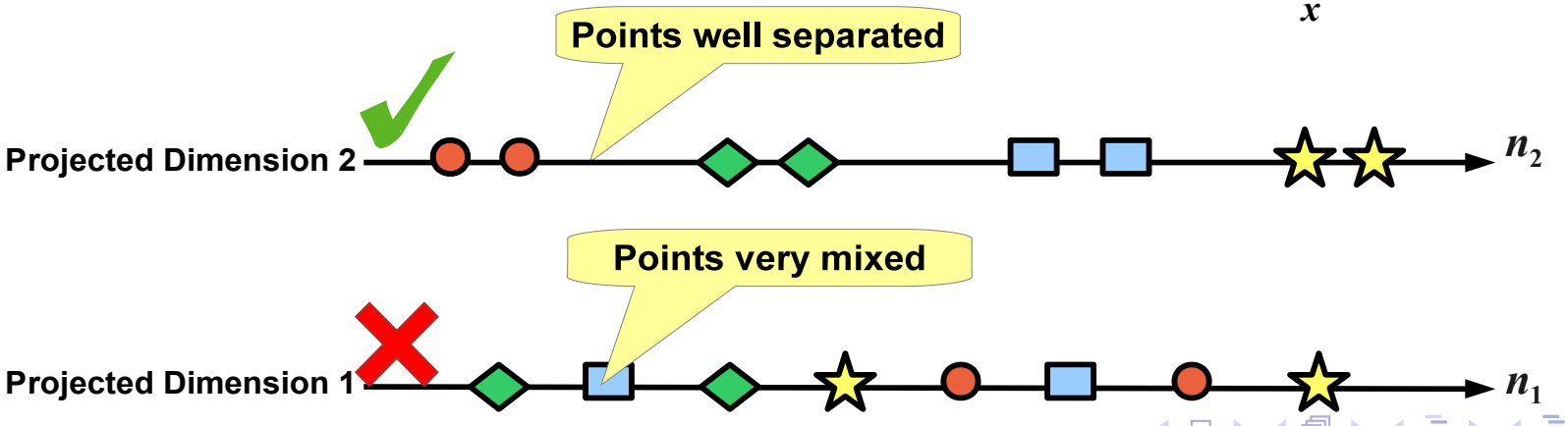


Problem 2: some hyperplanes are better than others



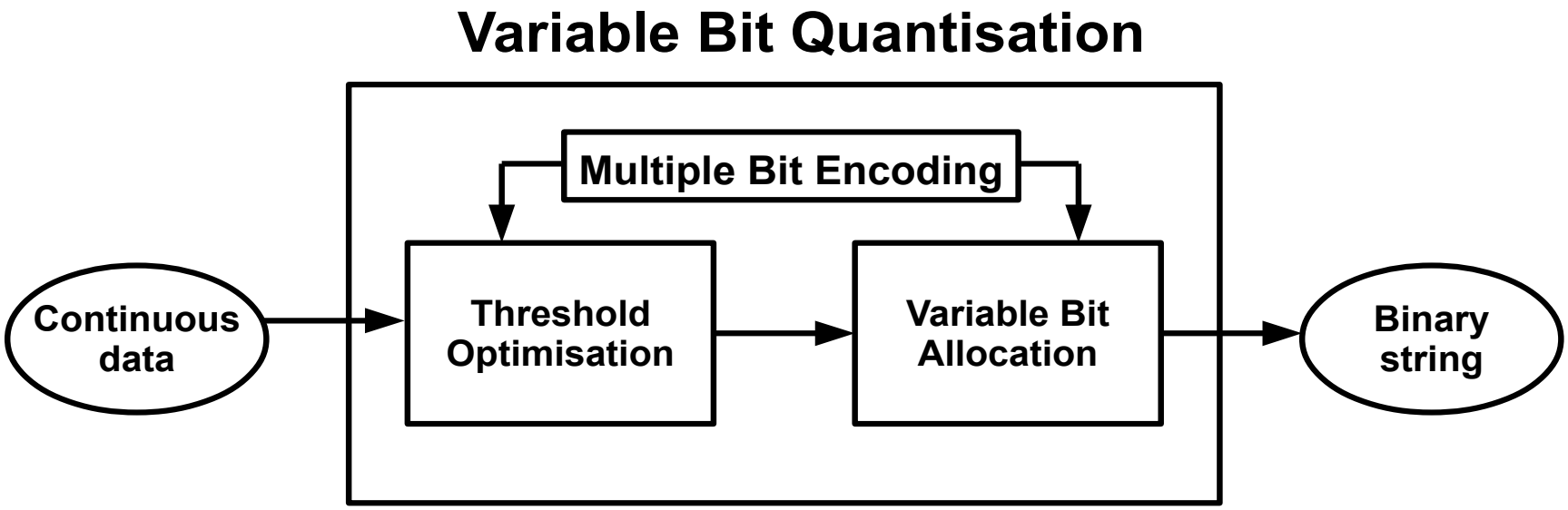
Problem 2: some hyperplanes are better than others



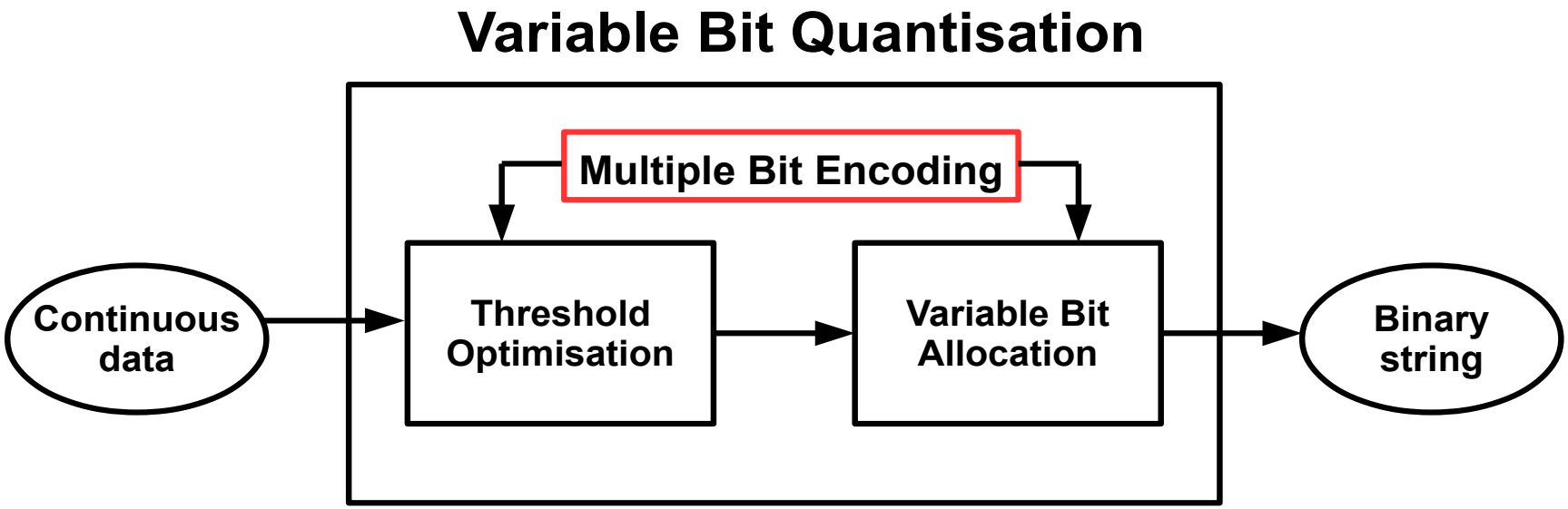


Variable Bit Quantisation for LSH

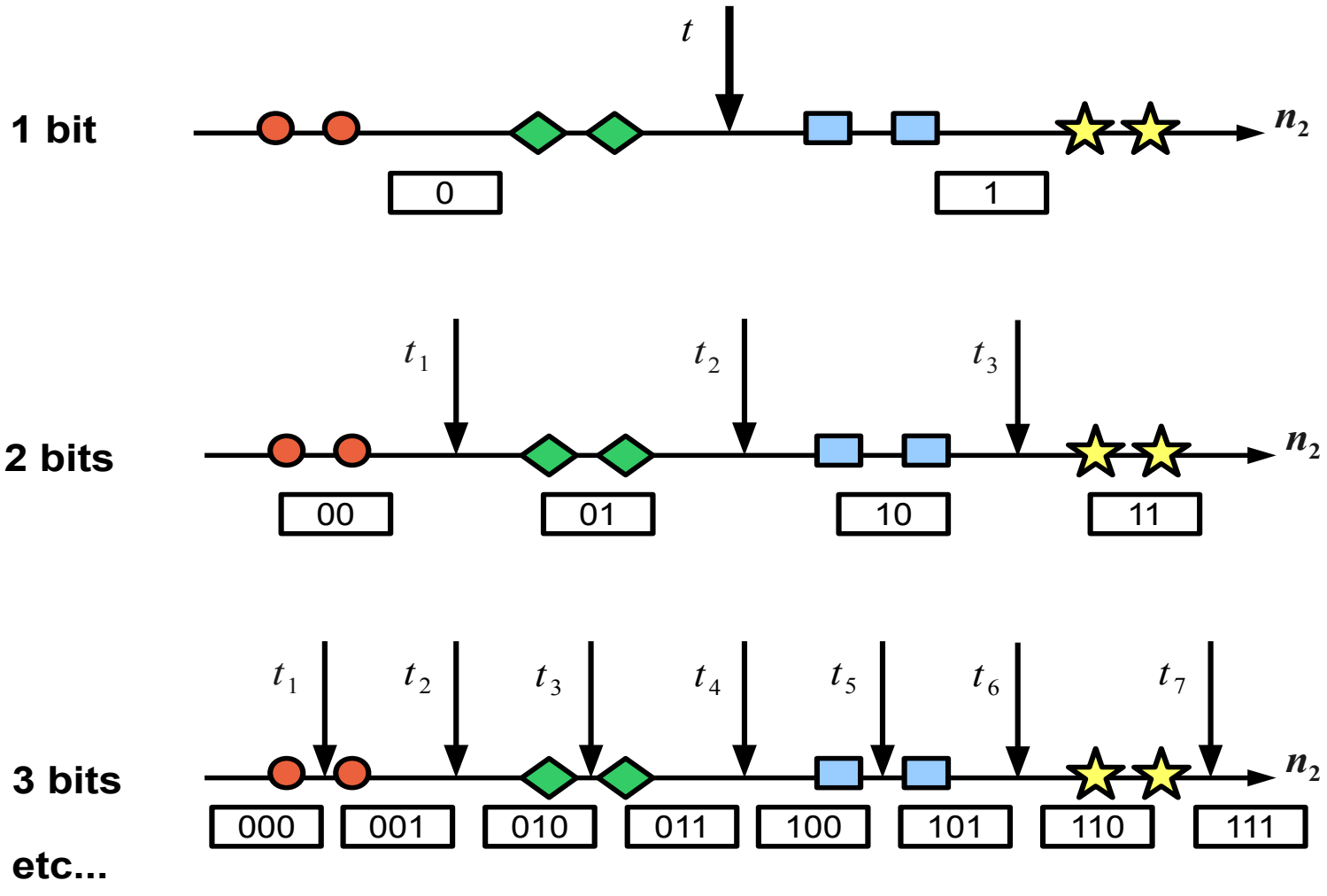
Our Solution: Variable Bit Quantisation



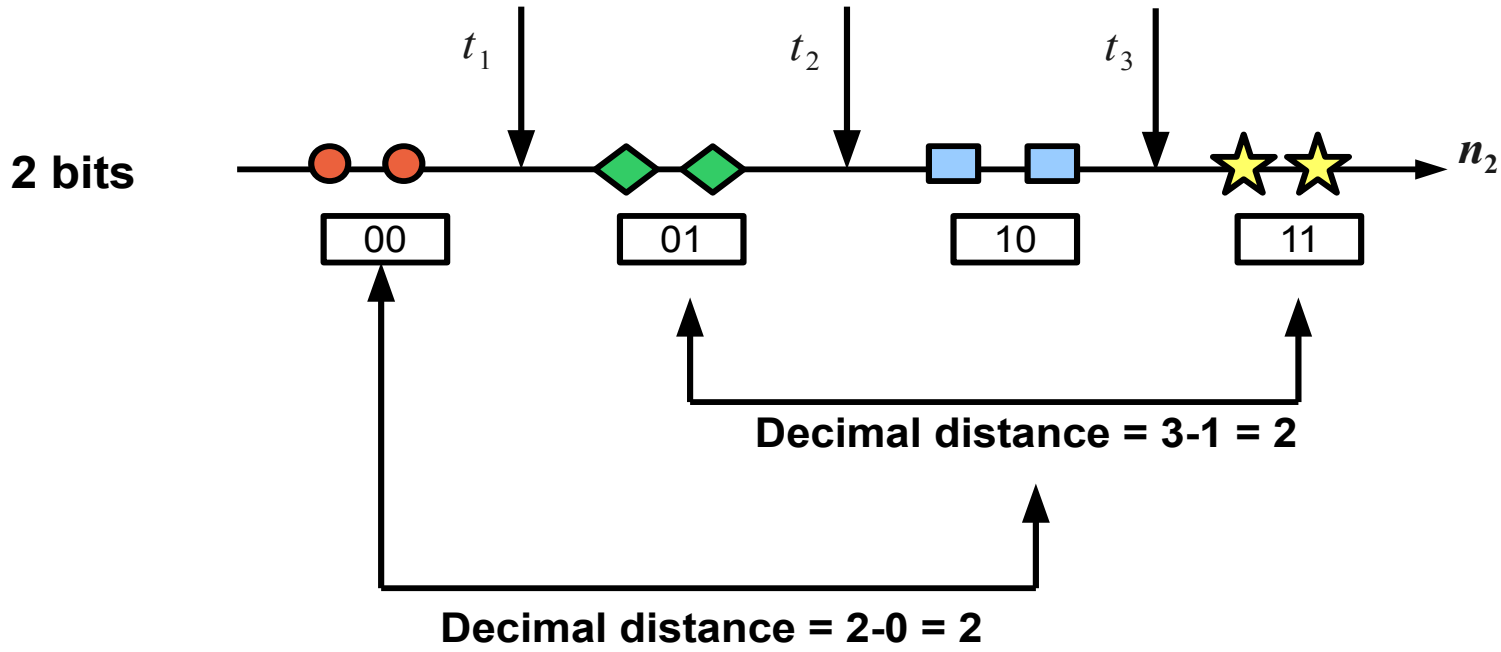
Our Solution: Variable Bit Quantisation



Multiple Bit Encoding: Natural Binary Code

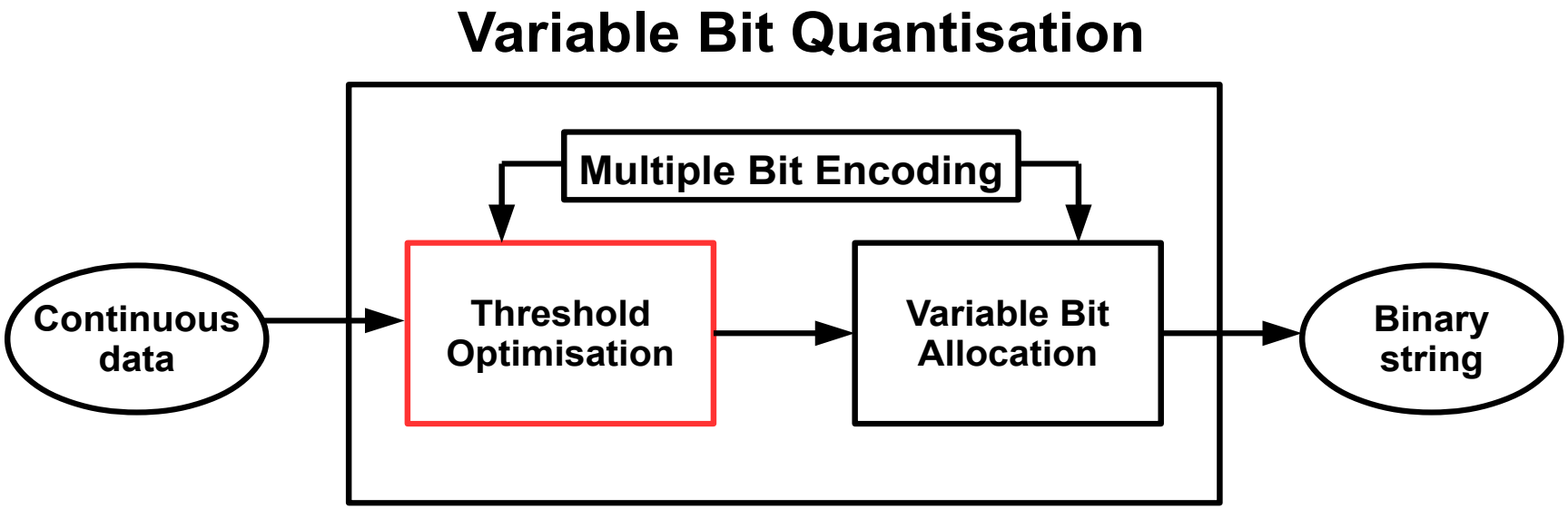


Multiple Bit Encoding [1]



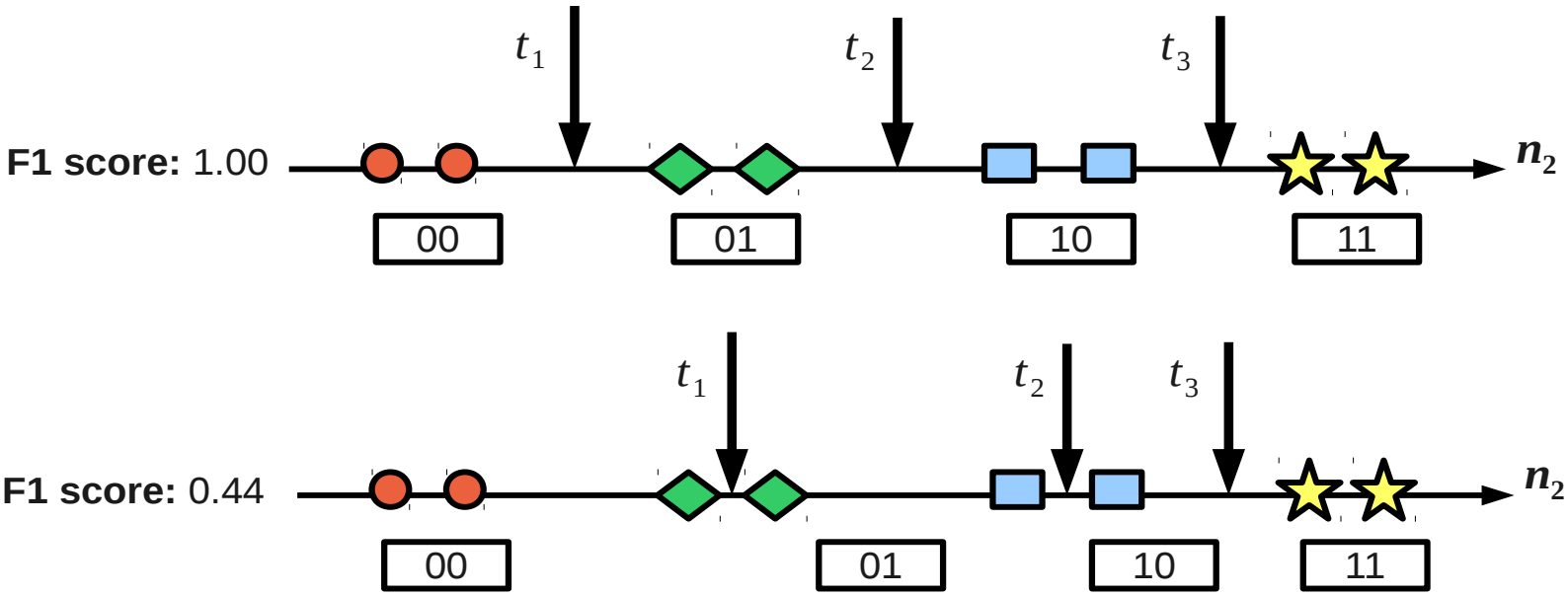
[1] W. Kong and W. Li and M. Guo. *Manhattan hashing for large-scale image retrieval*. SIGIR '12.

Our Solution: Variable Bit Quantisation

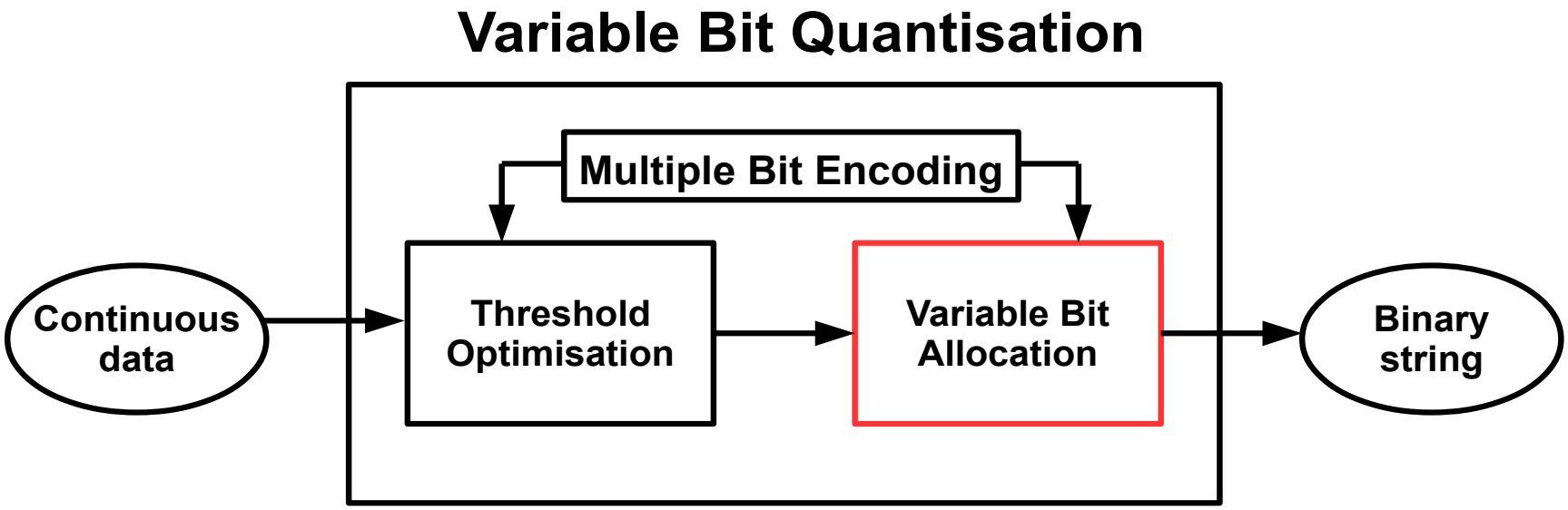


Threshold Optimisation

- Multiple bits per hyperplane requires multiple thresholds.
- F_β-based optimisation using pairwise constraints matrix S: TP: # $S_{ij} = 1$ pairs in the same region. FP: # $S_{ij} = 0$ pairs in the same region. FN: # $S_{ij} = 1$ pairs in different regions.

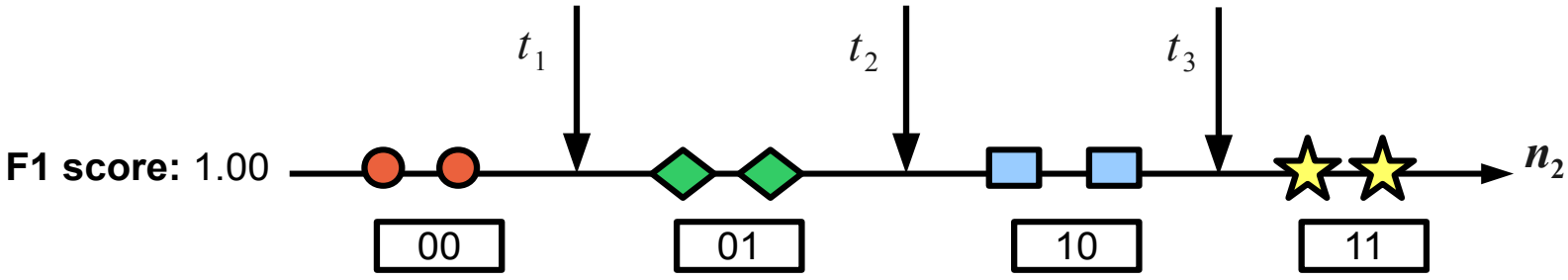


Our Solution: Variable Bit Quantisation



Variable Bit Allocation

- F_β score is a measure of the *neighbourhood preserving* quality of a hyperplane:



2 bits assigned: 3 thresholds perfectly preserve the neighbourhood structure

0 bits assigned: 0 thresholds do just as well as one or more thresholds

- Compute bit allocation that maximises the *cumulative* F_1 score across all hyperplanes subject to a bit budget B.
- Bit allocation solved as a binary integer linear program (BILP).

Variable Bit Allocation

$$\begin{aligned} \max \quad & \|\mathbf{F} \circ \mathbf{Z}\| \\ \text{subject to} \quad & \|\mathbf{Z}_h\| = 1 \qquad h \in \{1 \dots B\} \\ & \|\mathbf{Z} \circ \mathbf{D}\| \leq B \\ & \mathbf{Z} \quad \text{is binary} \end{aligned}$$

- $\mathbf{F}$ contains the F_β scores per hyperplane, per bit count
- $\mathbf{Z}$ is an indicator matrix specifying the bit allocation
- $\mathbf{D}$ is a constraint matrix
- B is the bit budget
- $\|.\|$ denotes the Frobenius L_1 norm
- $\circ$ the Hadamard product

Variable Bit Allocation

$$\begin{aligned} \max \quad & \|\mathbf{F} \circ \mathbf{Z}\| \\ \text{subject to} \quad & \|\mathbf{Z}_h\| = 1 \qquad h \in \{1 \dots B\} \\ & \|\mathbf{Z} \circ \mathbf{D}\| \leq B \\ & \mathbf{Z} \quad \text{is binary} \end{aligned}$$

$$\mathbf{F} \quad \begin{matrix} \\ b_0 \\ b_1 \\ b_2 \end{matrix} \begin{matrix} h_1 & h_2 \\ \end{matrix} \begin{pmatrix} 0.25 & 0.25 \\ 0.35 & 0.50 \\ 0.40 & 1.00 \end{pmatrix} \qquad \mathbf{D} \begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 2 & 2 \end{pmatrix} \qquad \mathbf{Z} \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}$$

- Sparse solution possible: lower quality hyperplanes can be discarded.

Variable Bit Quantisation for LSH

Evaluation Objectives

1. Quantify performance of the threshold optimisation procedure:
   - How does assigning a constant 2 bits per hyperplane based upon our adaptive thresholding scheme affect retrieval performance?
2. Quantify performance of the variable bit assignment routine:
   - How does assigning more bits to higher quality hyperplanes affect retrieval performance?

Evaluation Protocol

- **Task:** Text and image retrieval on three standard datasets: *CIFAR-10*, *100k TinyImages* and *Reuters-21578*.
- **Projections:** LSH [1], Shift-invariant kernel hashing (SIKH) [2], Spectral Hashing (SH) [3] and PCA-Hashing (PCAH) [4].
- **Baselines:** Single Bit Quantisation (SBQ), Manhattan Hashing (MQ)[5], Double-Bit quantisation (DBQ) [6].
- **Hamming Ranking:** how well do we retrieve the $\epsilon-$NN of query points?

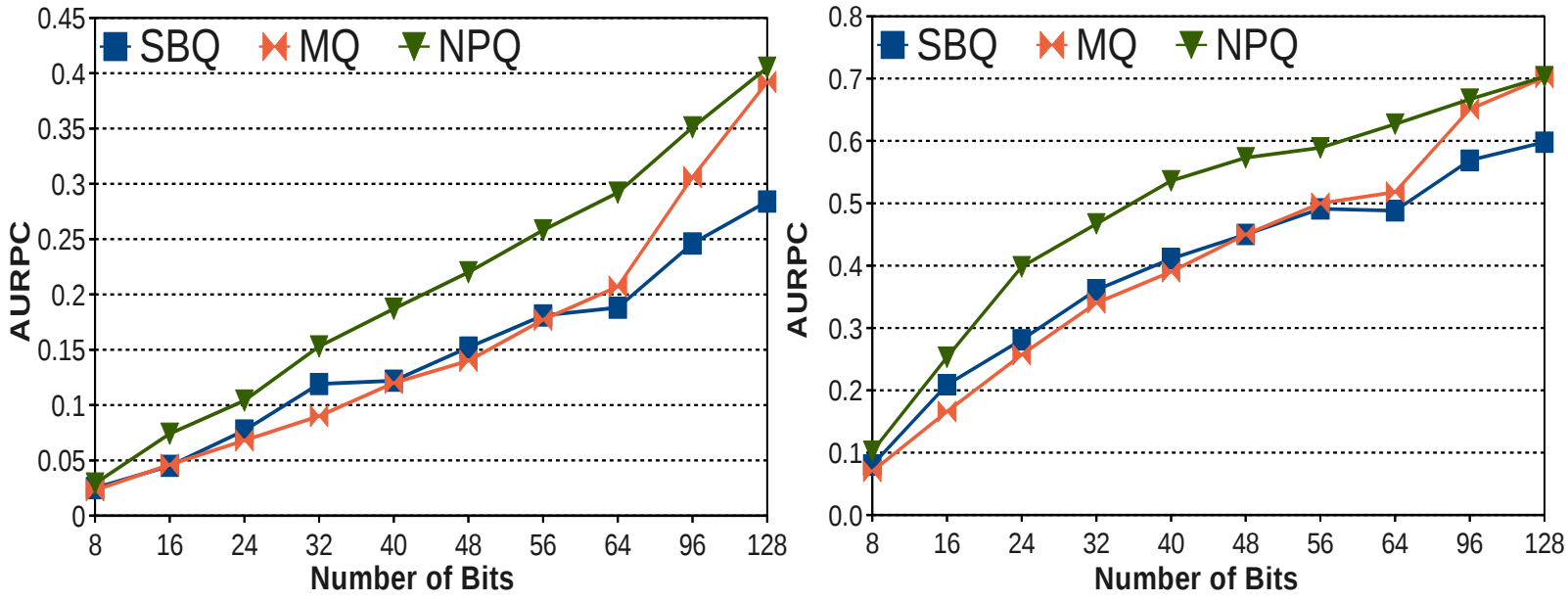
[1] P. Indyk and R. Motwani. *Approximate nearest neighbors: removing the curse of dimensionality*. In STOC '98.
[2] M. Raginsky and S. Lazebnik. *Locality-sensitive binary codes from shift-invariant kernels*. In NIPS '09.
[3] Y. Weiss and A. Torralba and R. Fergus. *Spectral Hashing*. NIPS '08.
[4] J. Wang and S. Kumar and SF. Chang. *Semi-supervised hashing for large-scale search*. PAMI '12.
[5] W. Kong and W. Li and M. Guo. *Manhattan hashing for large-scale image retrieval*. SIGIR '12.
[6] W. Kong and W. Li. *Double Bit Quantisation for Hashing*. AAAI '12.

AUPRC across different projections (constant # bits)

Dataset	**CIFAR-10 (32 bits)**				**Tiny Images (32 bits)**			
	SBQ	MQ	DBQ	**NPQ**	SBQ	MQ	DBQ	**NPQ**
SIKH	0.042	0.063	0.047	**0.090**	0.135	0.221	0.182	**0.365**
LSH	0.119	0.093	0.066	**0.153**	0.361	0.340	0.285	**0.464**
SH	0.051	0.135	0.111	**0.167**	0.117	0.237	0.136	**0.356**
PCAH	0.036	0.137	0.107	**0.153**	0.046	0.257	0.295	**0.312**

- Assigning multiple bits via our adaptive thresholding scheme yields substantial gains.
- NPQ and a cheap projection (e.g. LSH) can outperform SBQ and an expensive projection (e.g. PCA).

AUPRC for LSH across a broad bit range



(a) CIFAR-10 **(b) 100k TinyImages**

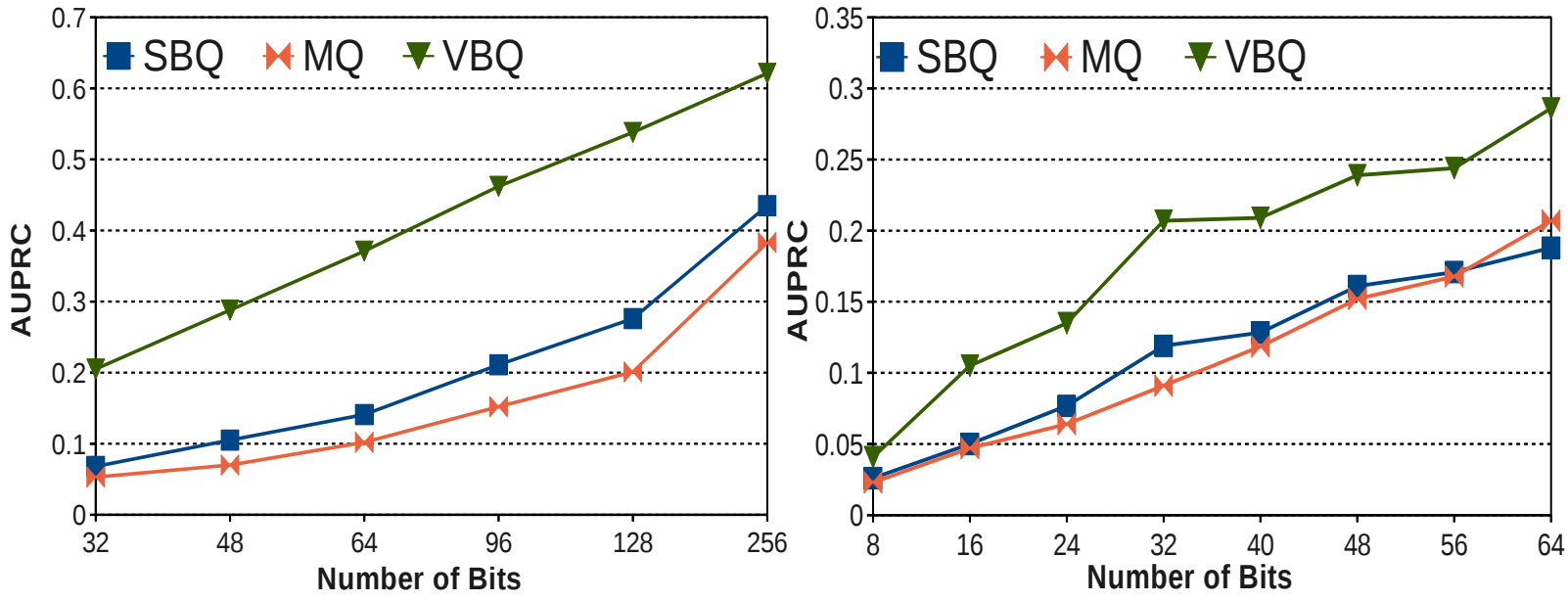
- NPQ is effective across a wide range of bits.

AUPRC across different projections (variable # bits)

Dataset	CIFAR-10 (32 bits)				Reuters-21578 (128 bits)			
	SBQ	MQ	DBQ	**VBQ**	SBQ	MQ	DBQ	**VBQ**
SIKH	0.042	0.046	0.047	**0.161**	0.102	0.112	0.087	**0.389**
LSH	0.119	0.091	0.066	**0.207**	0.276	0.201	0.175	**0.538**
SH	0.051	0.144	0.111	**0.202**	0.033	0.028	0.030	**0.154**
PCAH	0.036	0.132	0.107	**0.219**	0.095	0.034	0.027	**0.154**

- Varying the number of bits based on hyperplane quality gives additional gains in retrieval accuracy.
- VBQ is an effective quantisation scheme for both image and text datasets.

AUPRC for LSH across a broad bit range



(a) Reuters-21578 **(b) CIFAR-10**

- VBQ is also effective across a wide range of bits.

Variable Bit Quantisation for LSH

Summary

- Proposed a general data-driven technique to adaptively assign variable bits per LSH hyperplane
- Hyperplanes better preserving the neighbourhood structure are afforded more bits from the bit budget
- VBQ substantially increased retrieval performance across standard text and image datasets
- Future work: evaluate with an LSH system that uses hash tables for fast retrieval

Publications

- **Neighbourhood Preserving Quantisation for LSH**. Moran, S., Lavrenko, V., Osborne, M. (2013), *Special Interest Group on Information Retrieval (SIGIR)*, Dublin, Ireland.

- **Variable Bit Quantisation for LSH**. Moran, S., Lavrenko, V., Osborne, M. (2013), *Association for Computational Linguistics (ACL)*, Sofia, Bulgaria.

Thank you for your attention

Sean Moran
University of Edinburgh

sean.moran@ed.ac.uk
www.seanjmoran.com