Learning to Hash for Large Scale Image Retrieval

Sean Moran

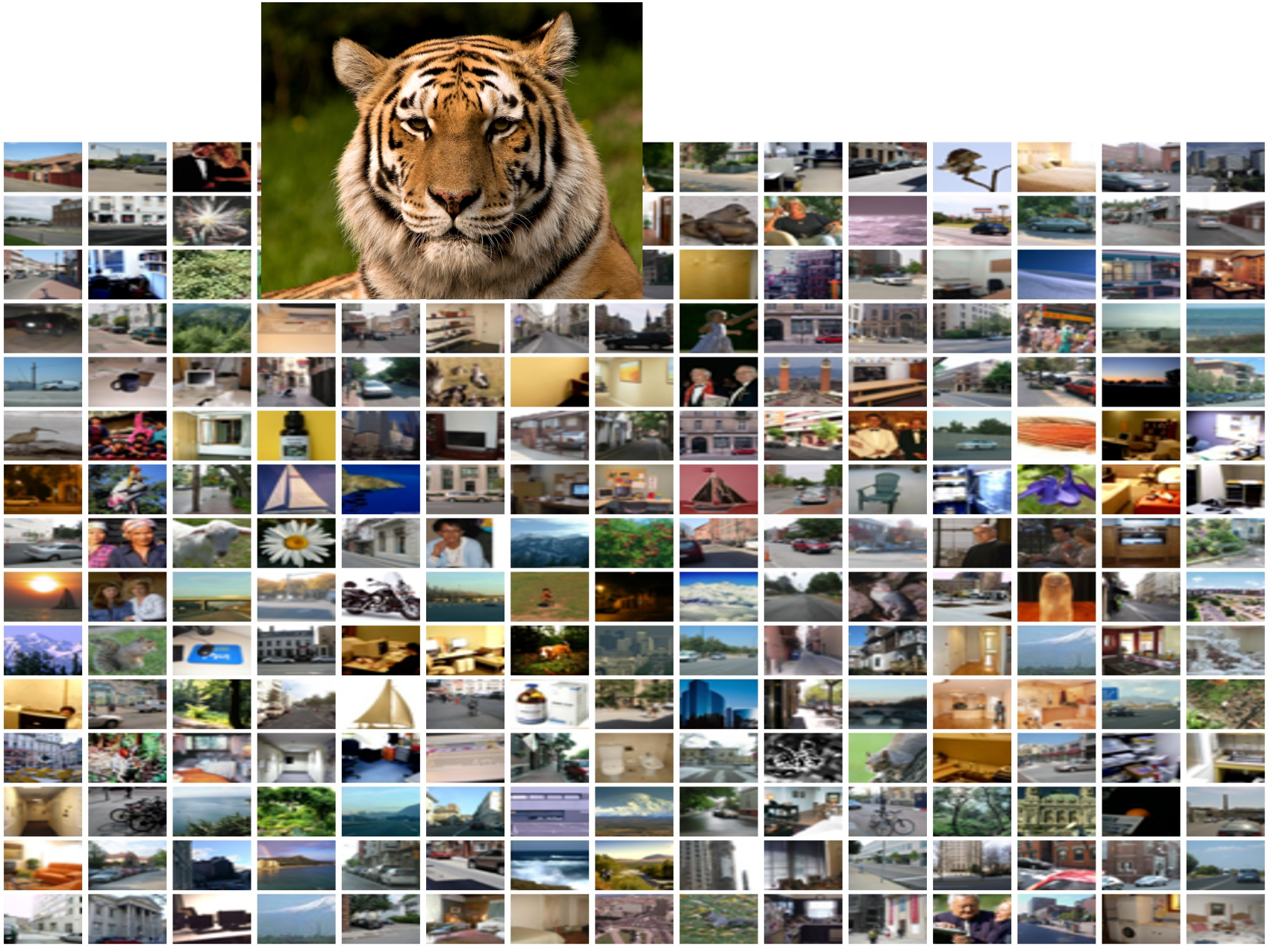
Pre-Viva Talk

27^{th} November 2015

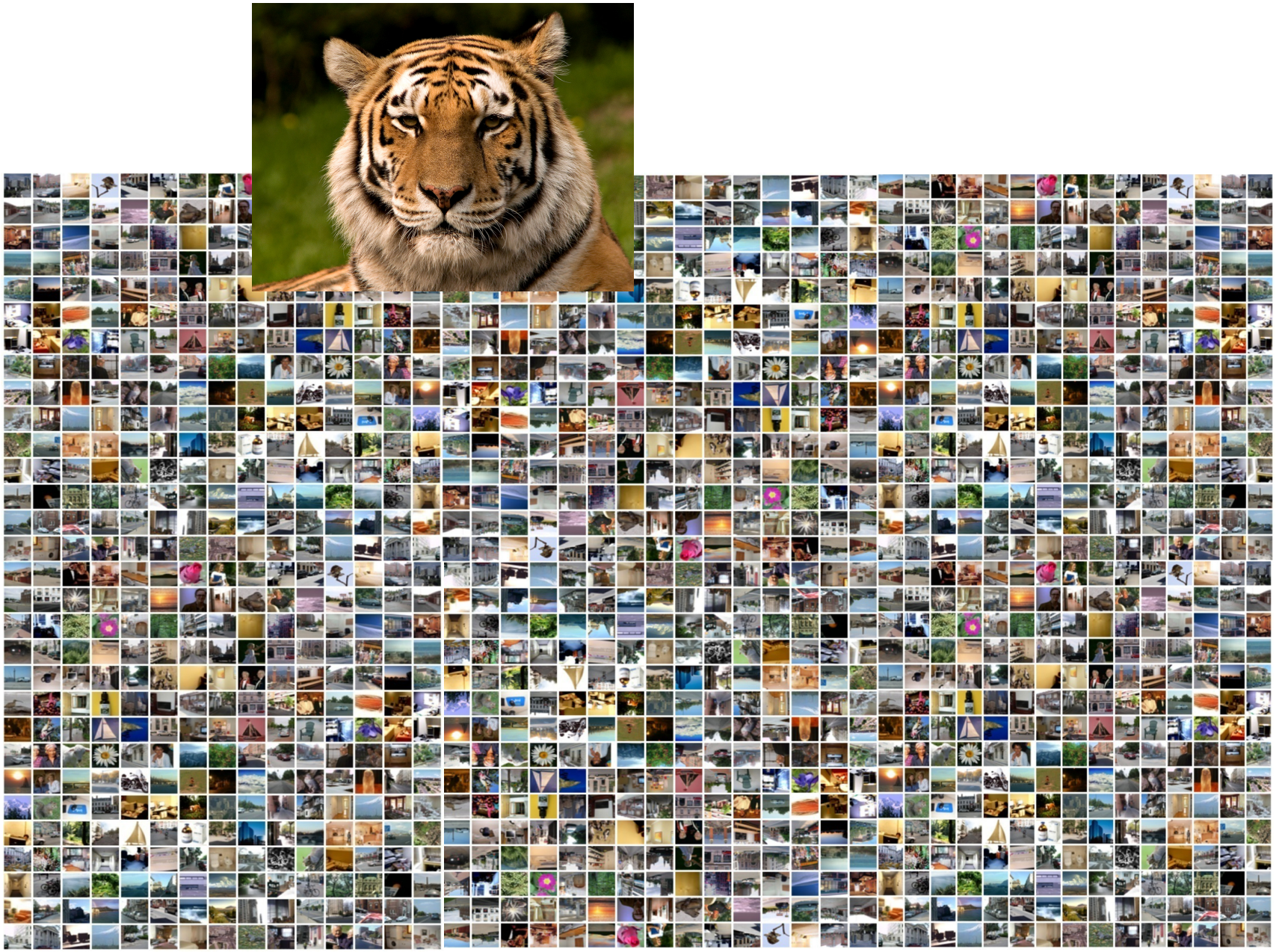
Learning to Hash for Large Scale Image Retrieval

Learning to Hash for Large Scale Image Retrieval

Efficient Retrieval in Large Image Archives

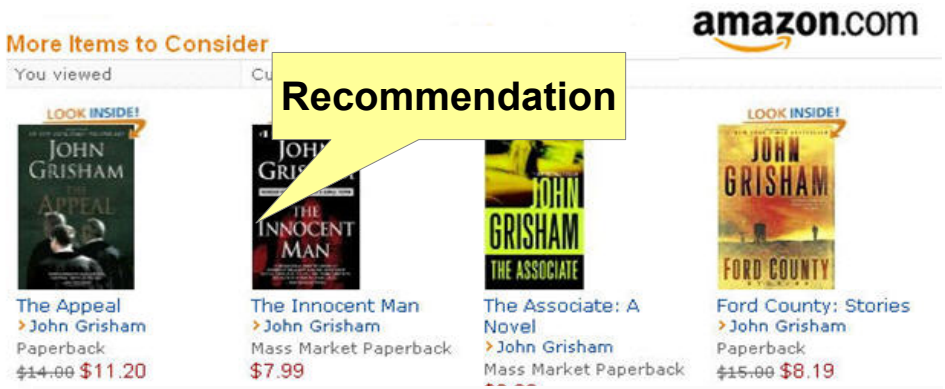


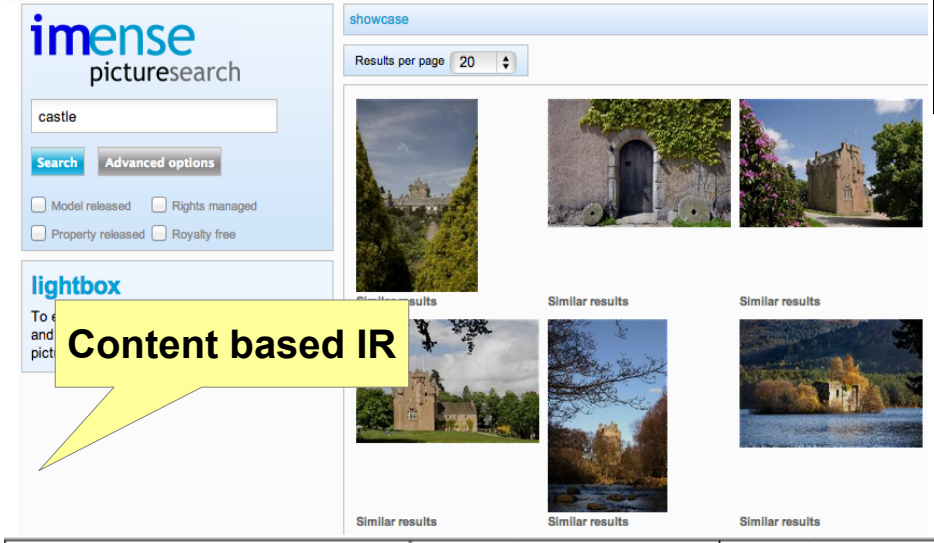
Efficient Retrieval in Large Image Archives

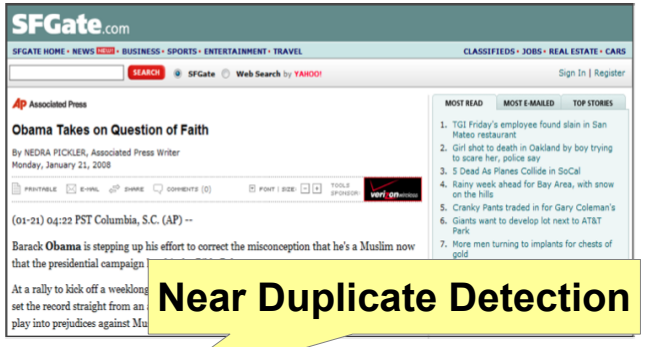


Efficient Retrieval in Large Image Archives

Applications





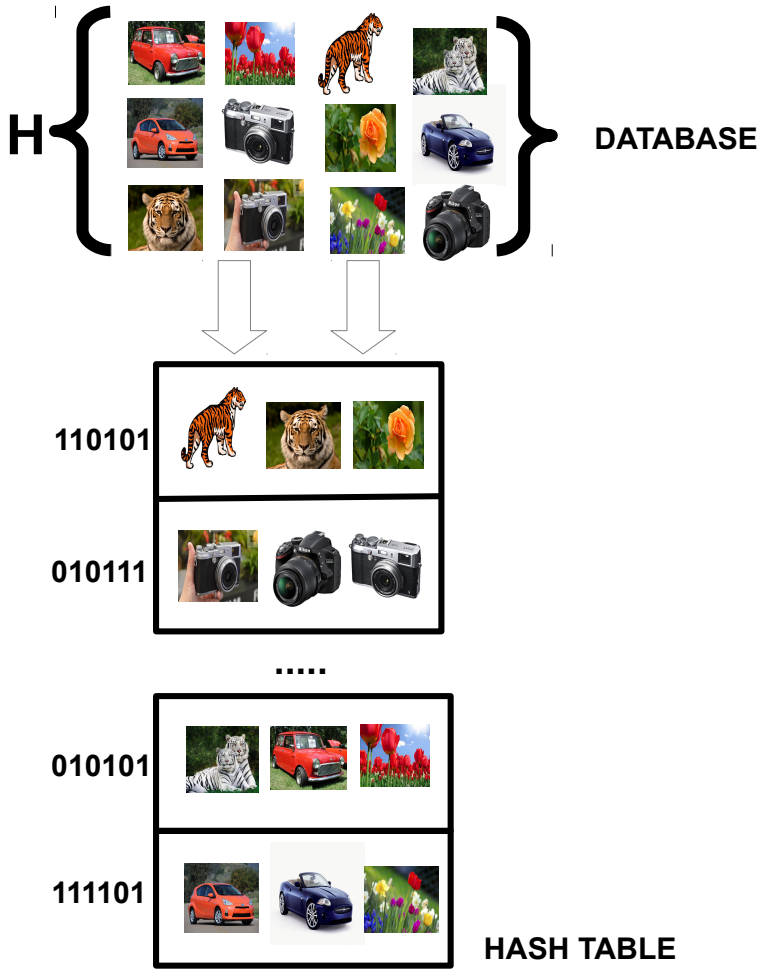


computer science	TSUNAMI	Louis Vuitton
mechanical engineering	tidal wave	PRADA
electrical engineering	LANDSLIDE	Fendi
chemical engineering	EARTH	
Civil Engineering	volcani	
ECONOMICS	HAILSTO	BURBERRY
ENGINEERING	Typhoon	GUCCI

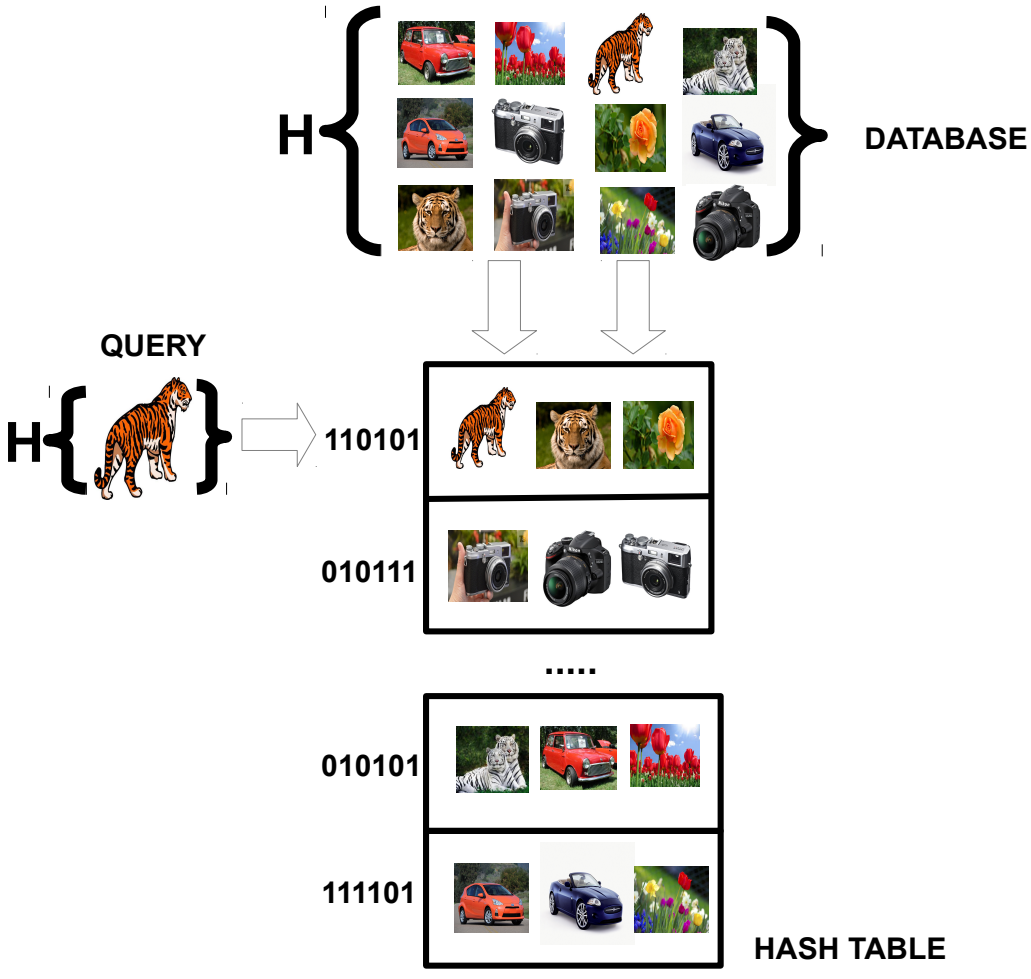
Noun Clustering

Locality Sensitive Hashing

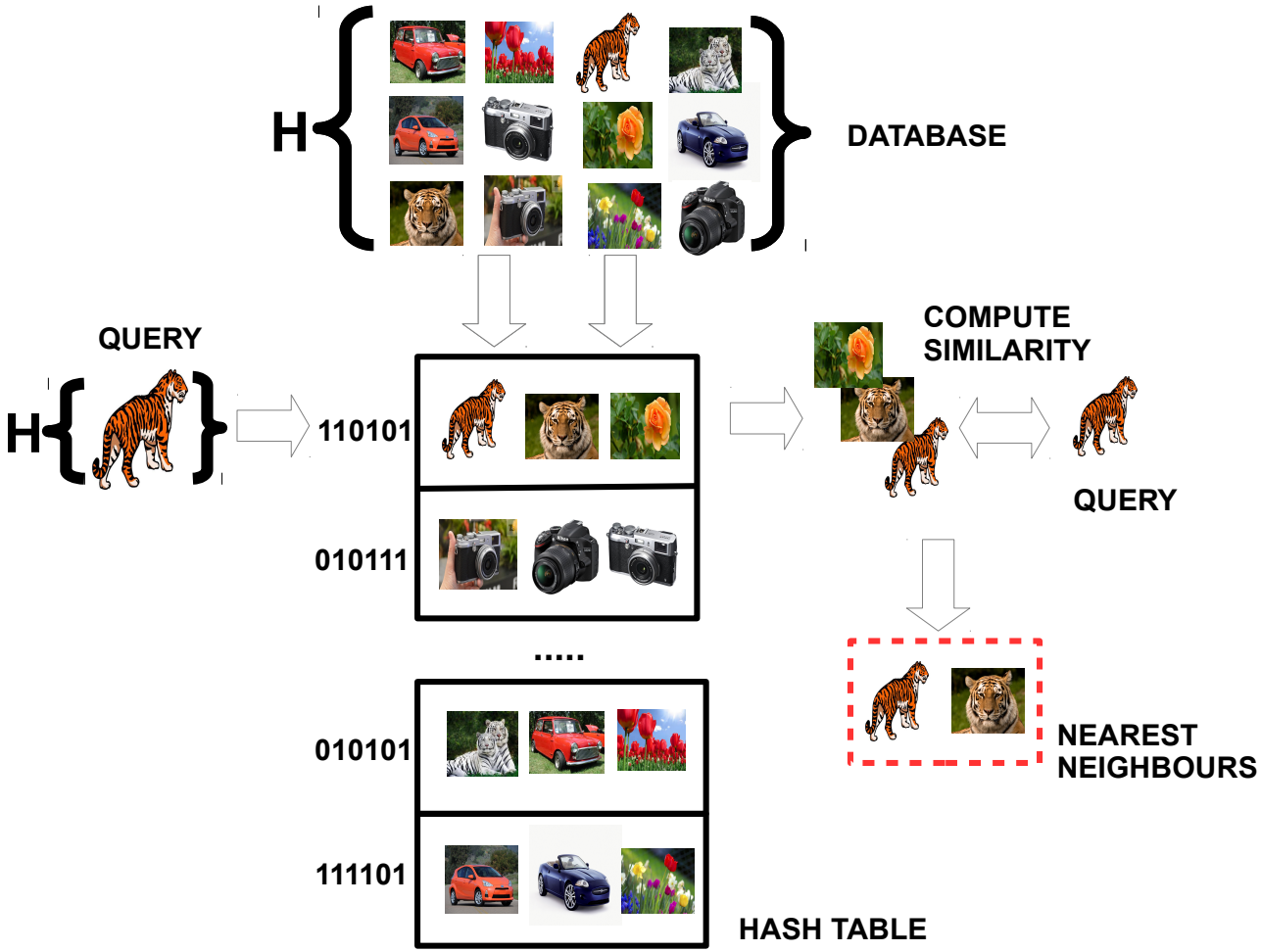
Locality Sensitive Hashing



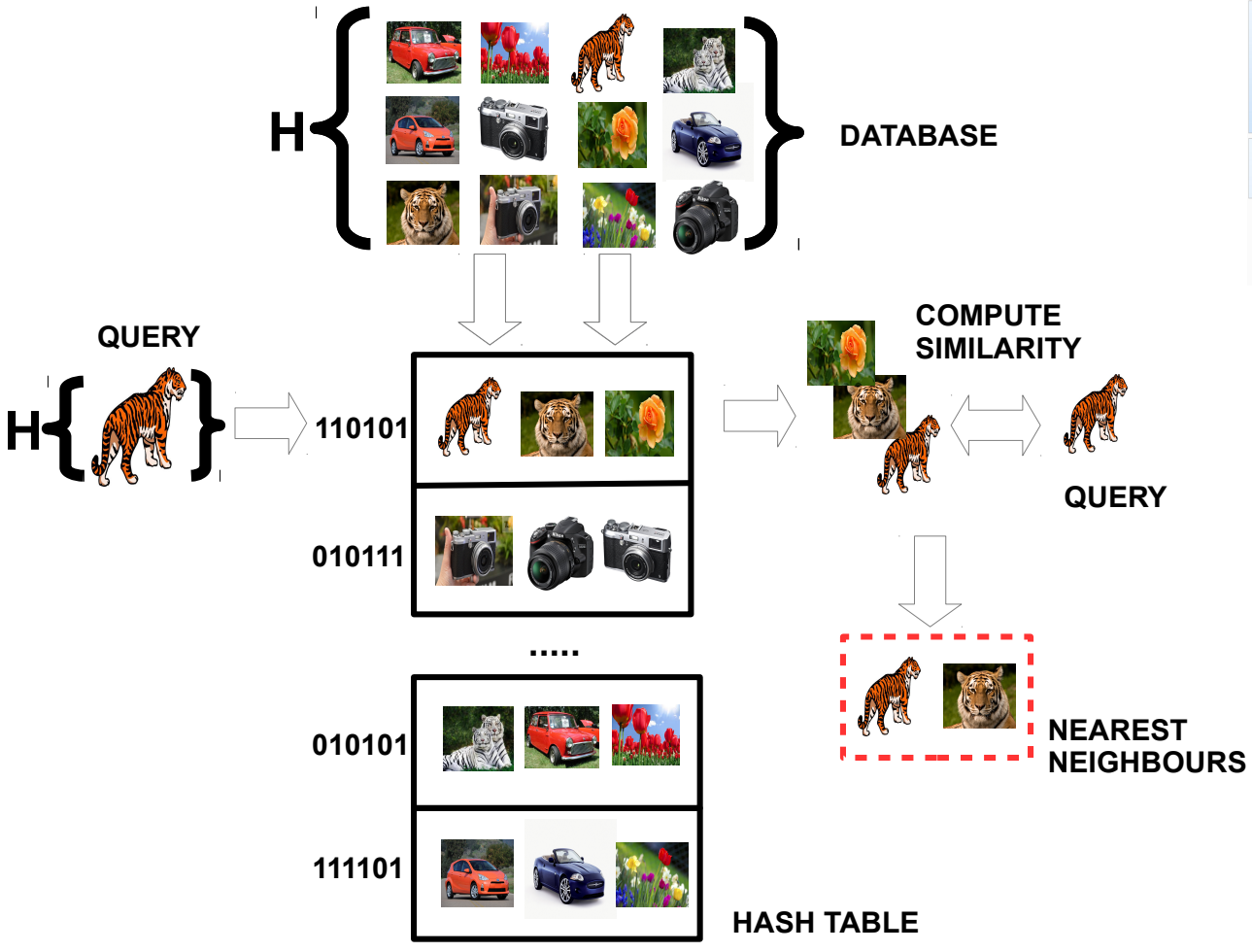
Locality Sensitive Hashing



Locality Sensitive Hashing



Locality Sensitive Hashing



Content Based IR

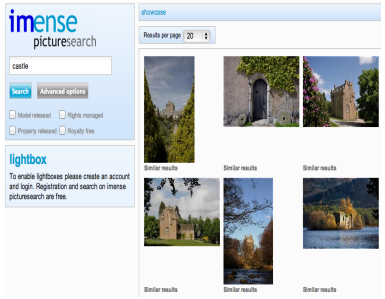


Image: Imense Ltd

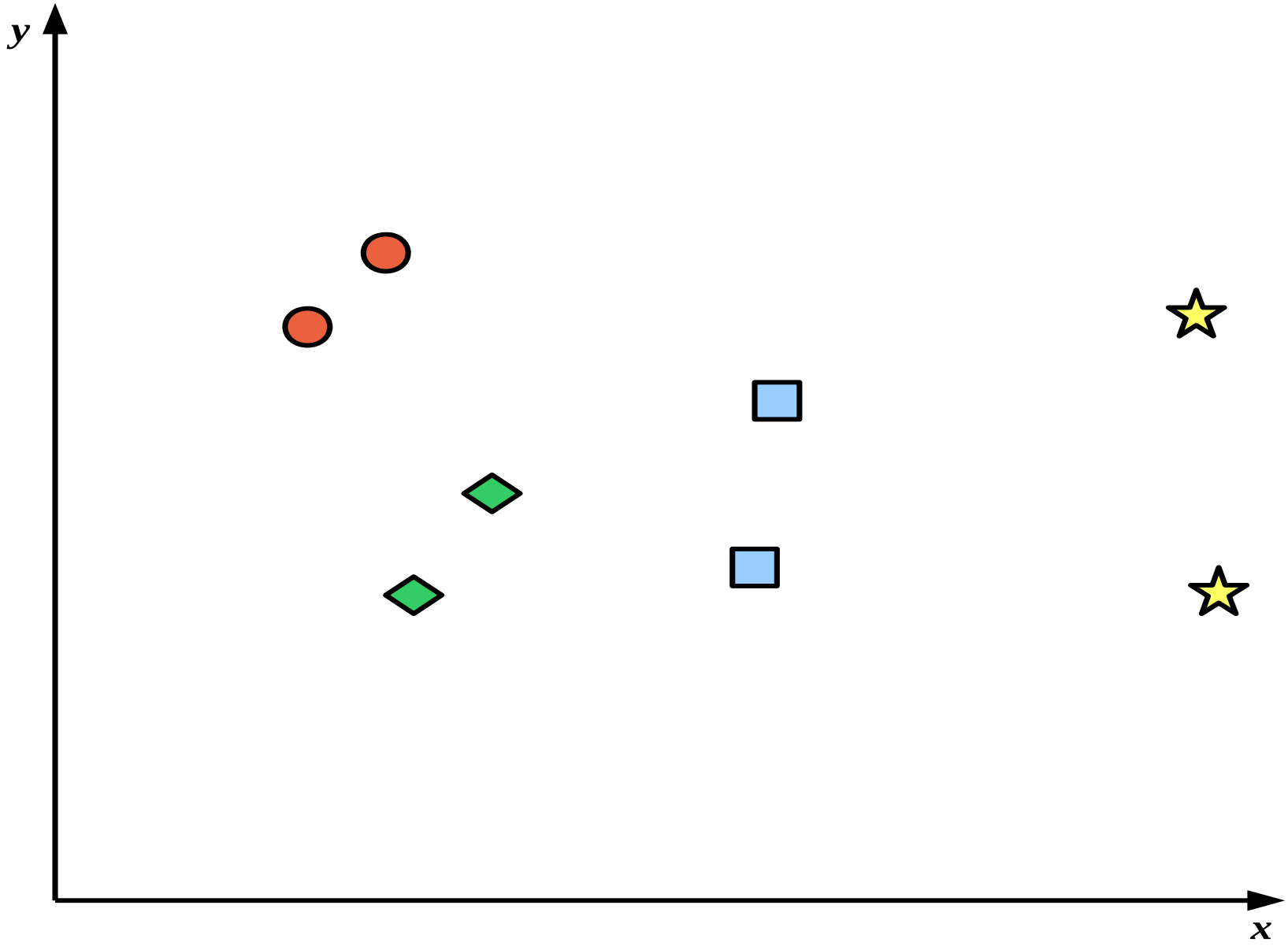
Near duplicate detection

Image: Xu et al.

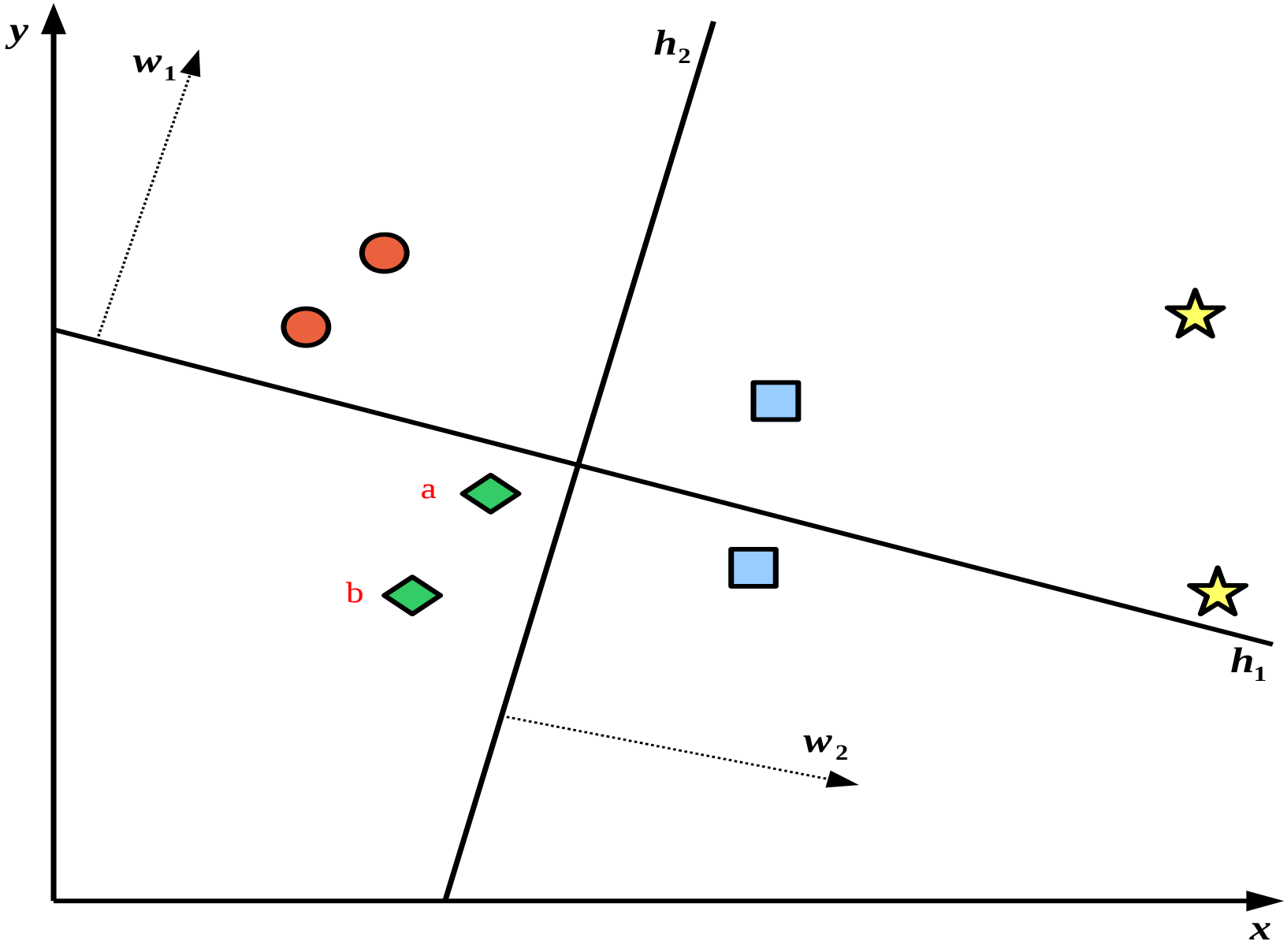
Location Recognition

Image: Doersch et al.

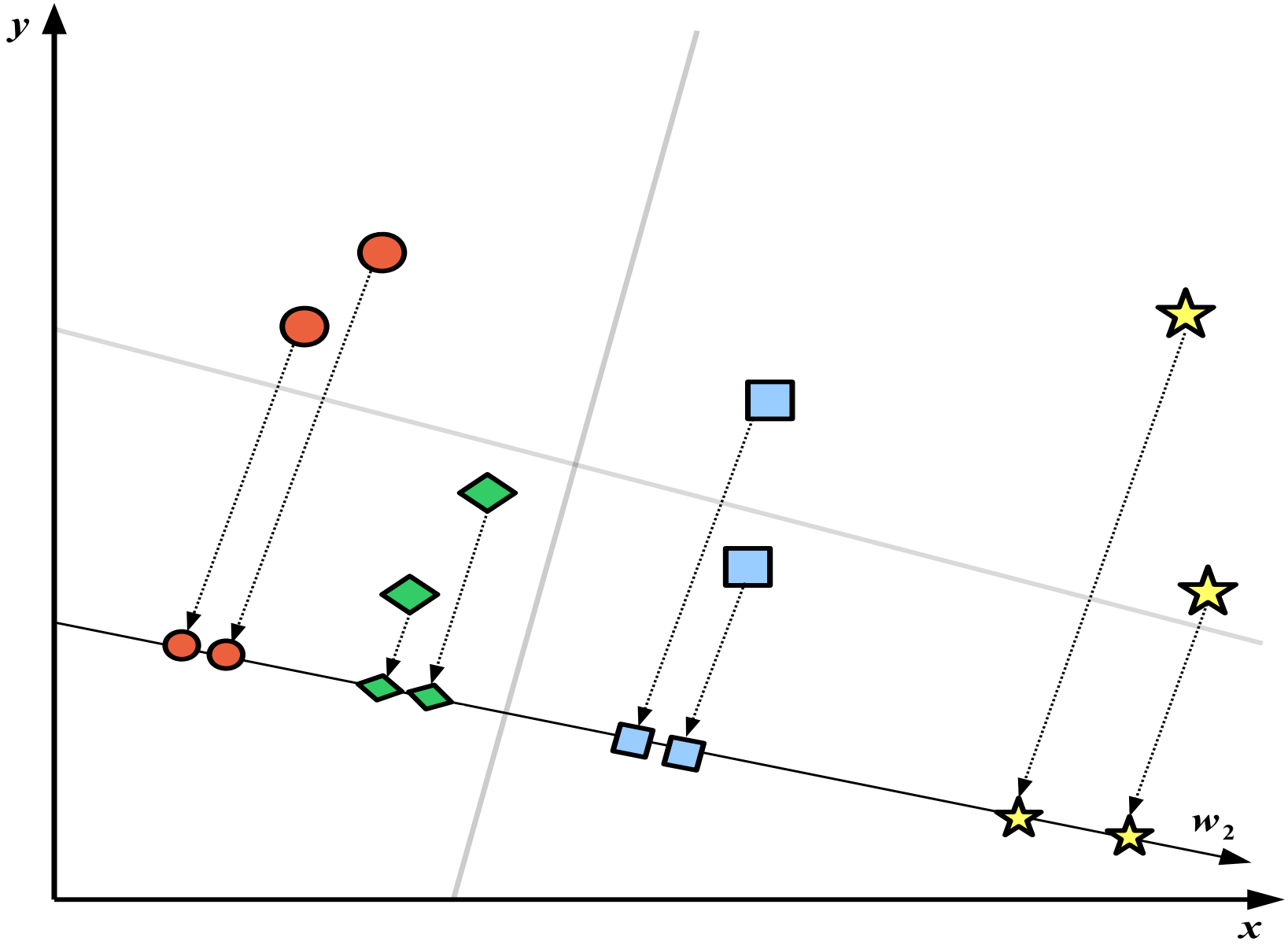
Toy 2D Feature Space



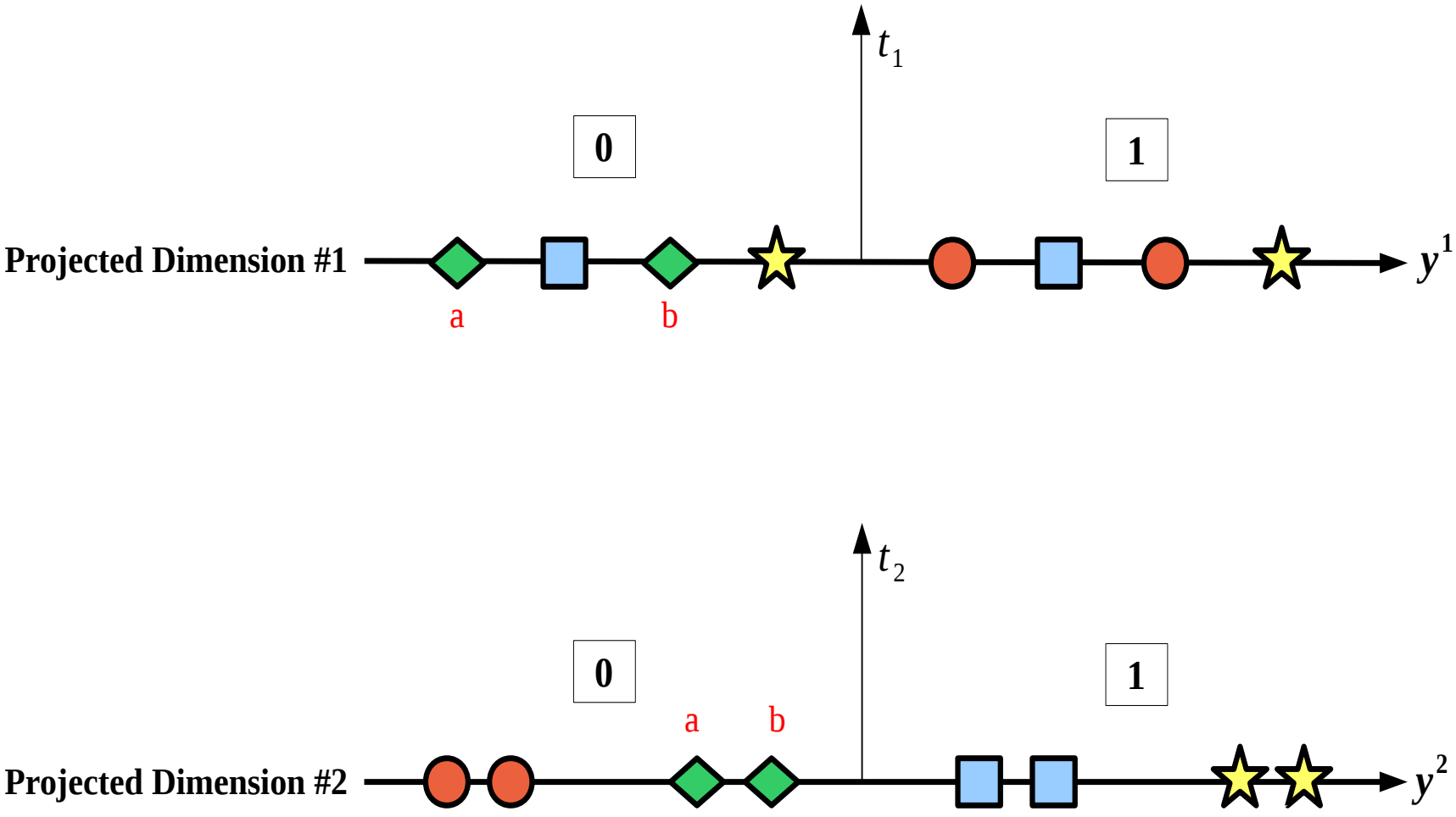
Feature Space Partitioning



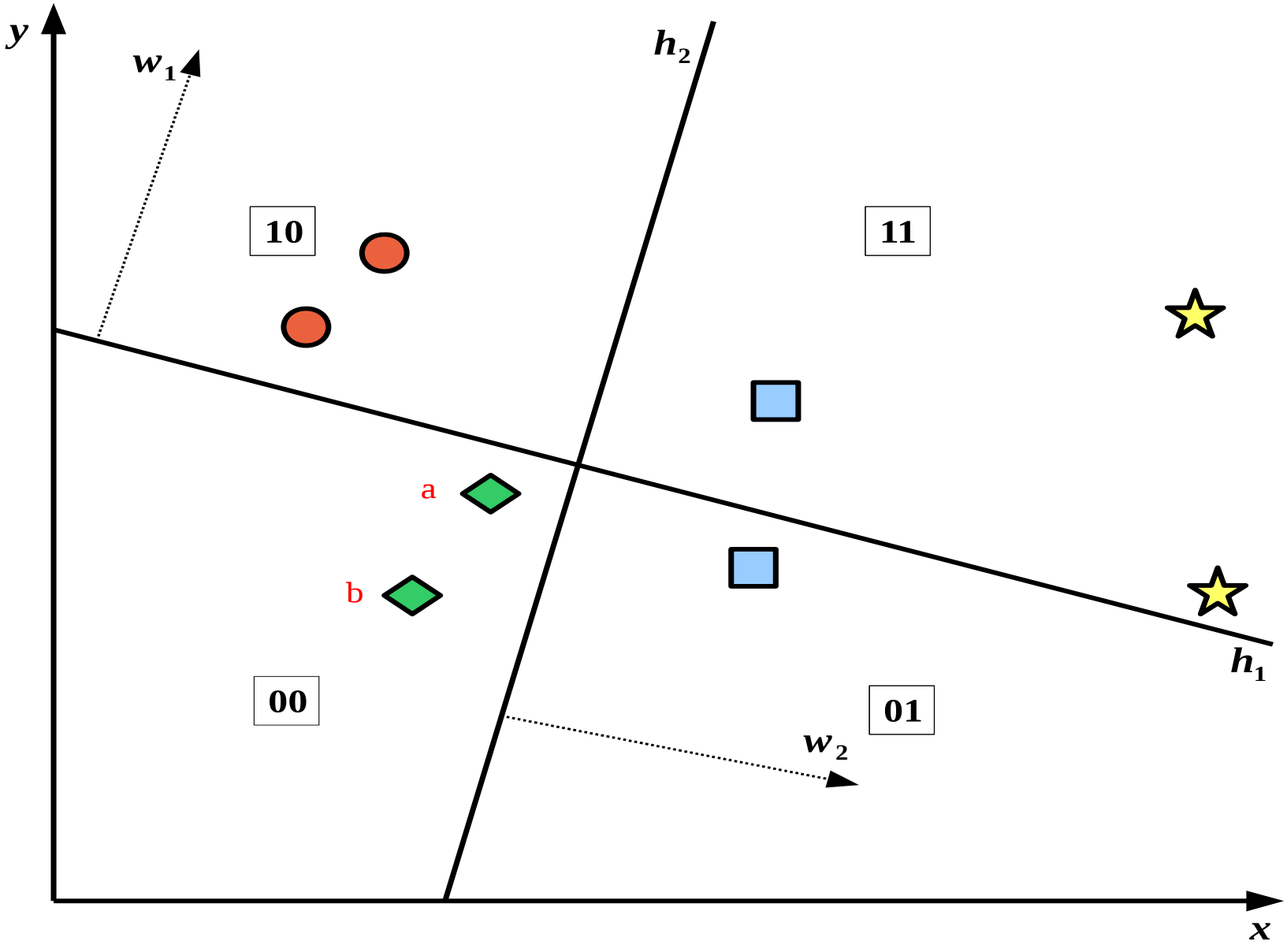
Step 1: Projection onto Normal Vectors



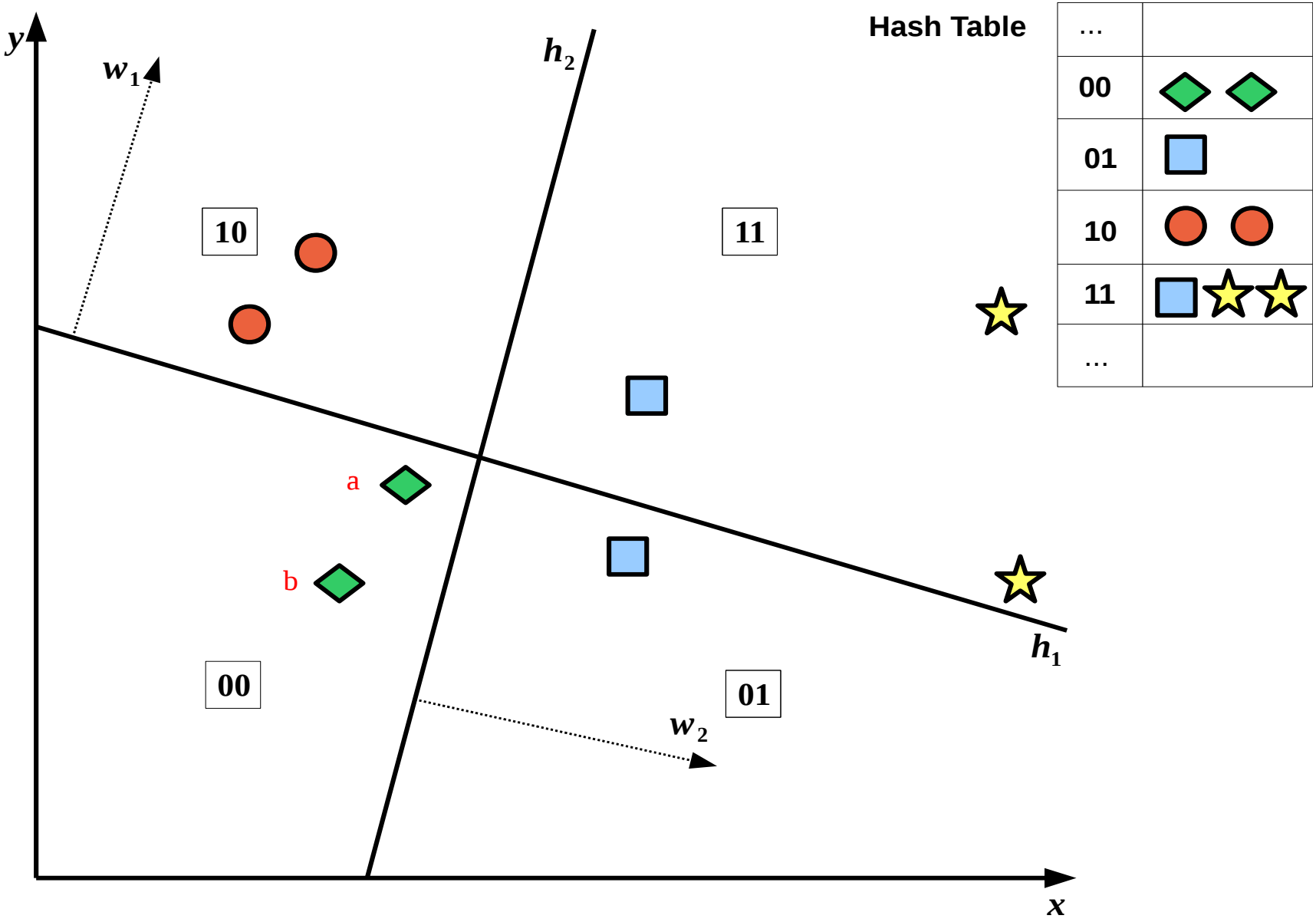
Step 2: Binary Quantisation



2-Bit Hashcodes of Data-Points



2-Bit Hashcodes of Data-Points



Thesis Statement

"The **retrieval effectiveness** *of hashing-based ANN search can be improved by* **learning** *the* **quantisation thresholds** *and* **hashing hyperplanes** *in a manner that is influenced by the* **distribution of the data**"

Relaxing Common Assumptions (A_1-A_3)

- **Three** limiting assumptions permeate literature:
  - A_1: **Single static** threshold placed at **zero**
  - A_2: **Uniform allocation** of thresholds across each dimension
  - A_3: **Linear** hypersurfaces (hyperplanes) positioned **randomly**

Thesis Contributions

- **Five** novel contributions to address the assumptions:
  - Learning **Multiple** Quantisation Thresholds [1]
  - Learning **Variable** Quantisation Thresholds [2]
  - Learning **Unimodal** Hashing Hypersurfaces [3]
  - Learning **Cross-Modal** Hashing Hypersurfaces [4]
  - Learning Hypersurfaces **and** Thresholds together

[1] Sean Moran, Victor Lavrenko, Miles Osborne. **Neighbourhood Preserving Quantisation for LSH**. *In SIGIR'13*.
[2] Sean Moran, Victor Lavrenko, Miles Osborne. **Variable Bit Quantisation for LSH**. *In ACL'13*.
[3] Sean Moran and Victor Lavrenko. **Graph Regularised Hashing**. *In ECIR'15*.
[4] Sean Moran and Victor Lavrenko. **Regularised Cross Modal Hashing**. *In SIGIR'15*.

Thesis Contributions (This Talk)

- **Two** contributions discussed in this talk:
  - **Learning Multiple Quantisation Thresholds** [1]
  - Learning Variable Quantisation Thresholds [2]
  - **Learning Unimodal Hashing Hypersurfaces** [3]
  - Learning Cross-Modal Hashing Hypersurfaces [4]
  - Learning Hypersurfaces and Thresholds together

[1] Sean Moran, Victor Lavrenko, Miles Osborne. **Neighbourhood Preserving Quantisation for LSH**. *In SIGIR'13*.
[2] Sean Moran, Victor Lavrenko, Miles Osborne. **Variable Bit Quantisation for LSH**. *In ACL'13*.
[3] Sean Moran and Victor Lavrenko. **Graph Regularised Hashing**. *In ECIR'15*.
[4] Sean Moran and Victor Lavrenko. **Regularised Cross Modal Hashing**. *In SIGIR'15*.

Evaluation Protocol

Dataset and Features

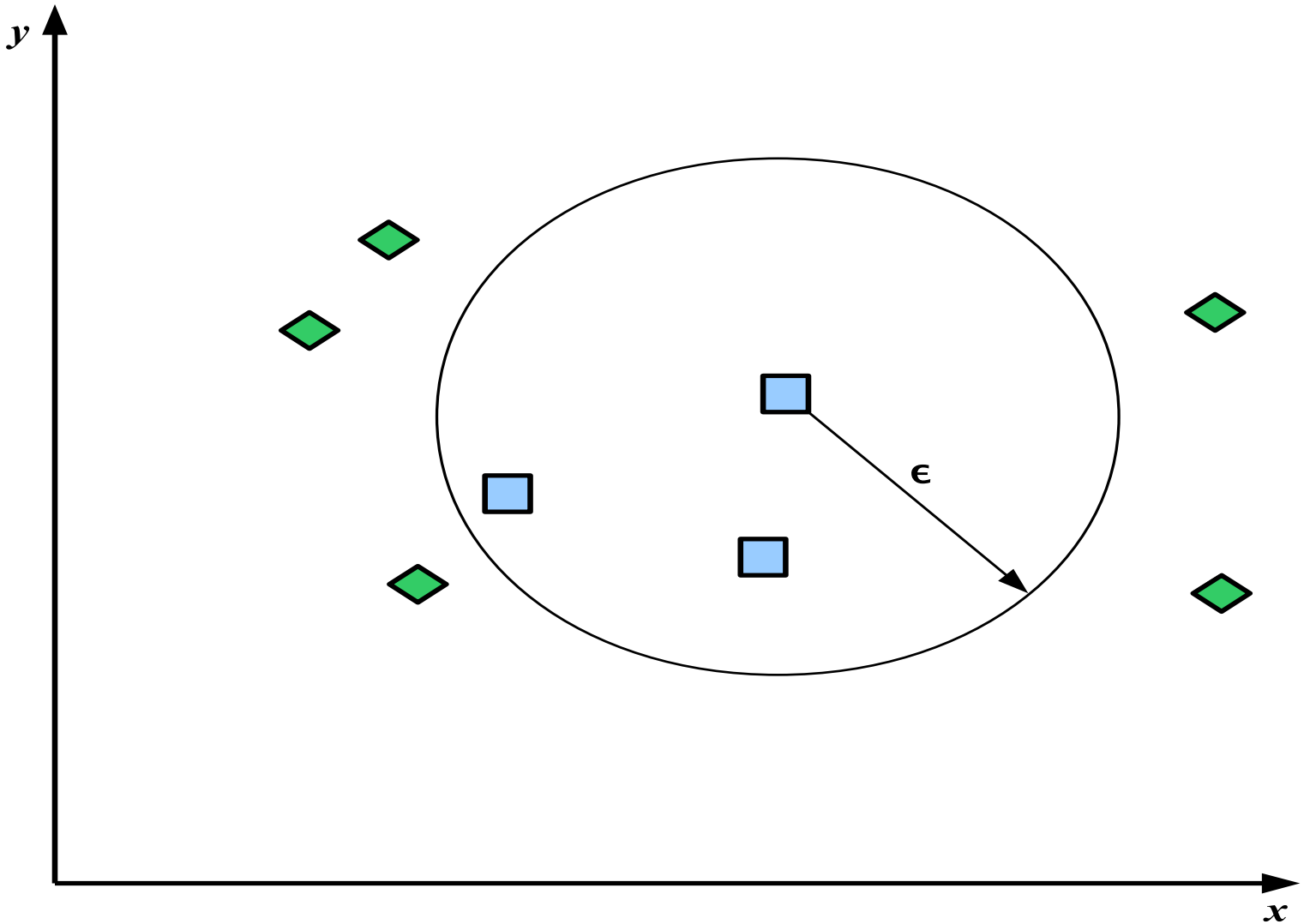
- Standard evaluation datasets [5,6]:
  - **CIFAR-10**: 60,000 images, 512-D GIST descriptors[1]
  - **NUSWIDE**: 270,000 images, 500-D BoW descriptors[2]
  - **SIFT-1M**: 1,000,000 images, 128-D SIFT descriptors[3]

[5] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, Shih-Fu Chang. **Supervised Hashing with Kernels**. *In CVPR'12*.
[6] Yunchao Gong, Svetlana Lazebnik. **Iterative Quantization: A Procrustean Approach to Learning Binary Codes for Large-scale Image Retrieval**. *In CVPR'11*.

[1] http://www.cs.toronto.edu/~kriz/cifar.html
[2] http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm
[3] http://corpus-texmex.irisa.fr/

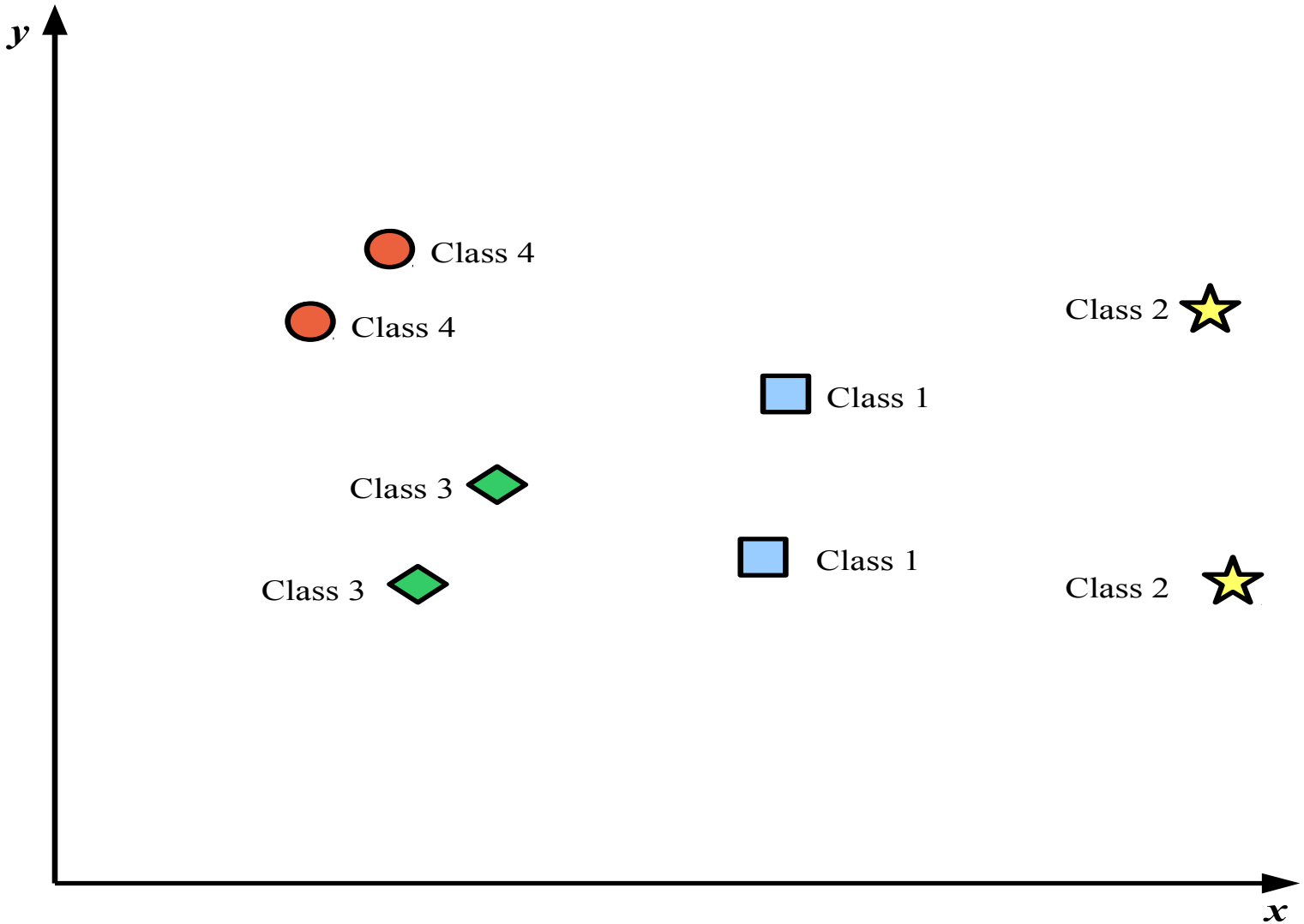
Example NUSWIDE Images [15]

ϵ-Nearest Neighbour Groundtruth [6]



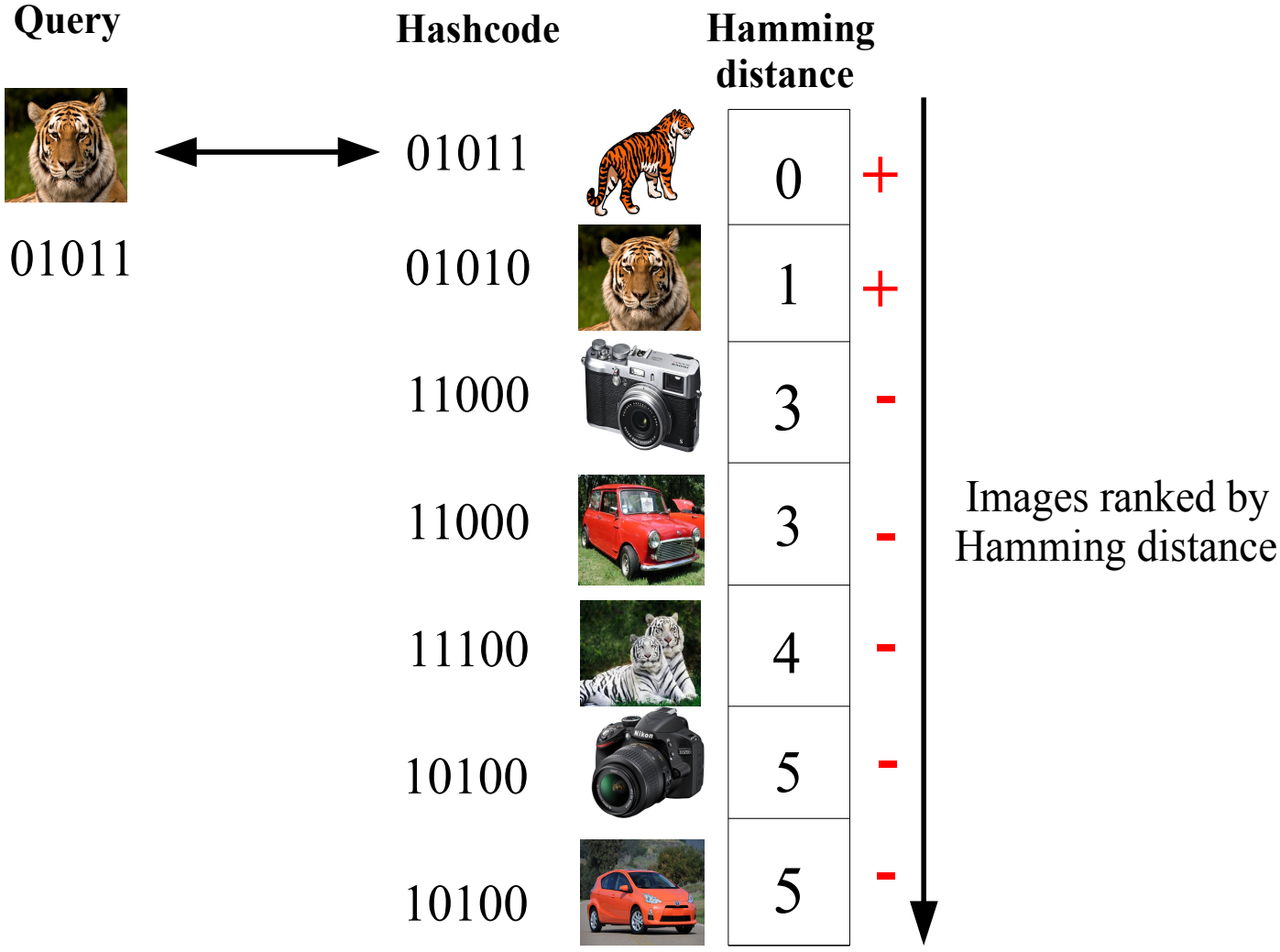
[6] Yunchao Gong, Svetlana Lazebnik. **Iterative Quantization: A Procrustean Approach to Learning Binary Codes for Large-scale Image Retrieval**. *In CVPR'11*.

Class Based Groundtruth [6]



[6] Yunchao Gong, Svetlana Lazebnik. **Iterative Quantization: A Procrustean Approach to Learning Binary Codes for Large-scale Image Retrieval**. *In CVPR'11*.

Hamming Ranking Evaluation Paradigm [5,6]



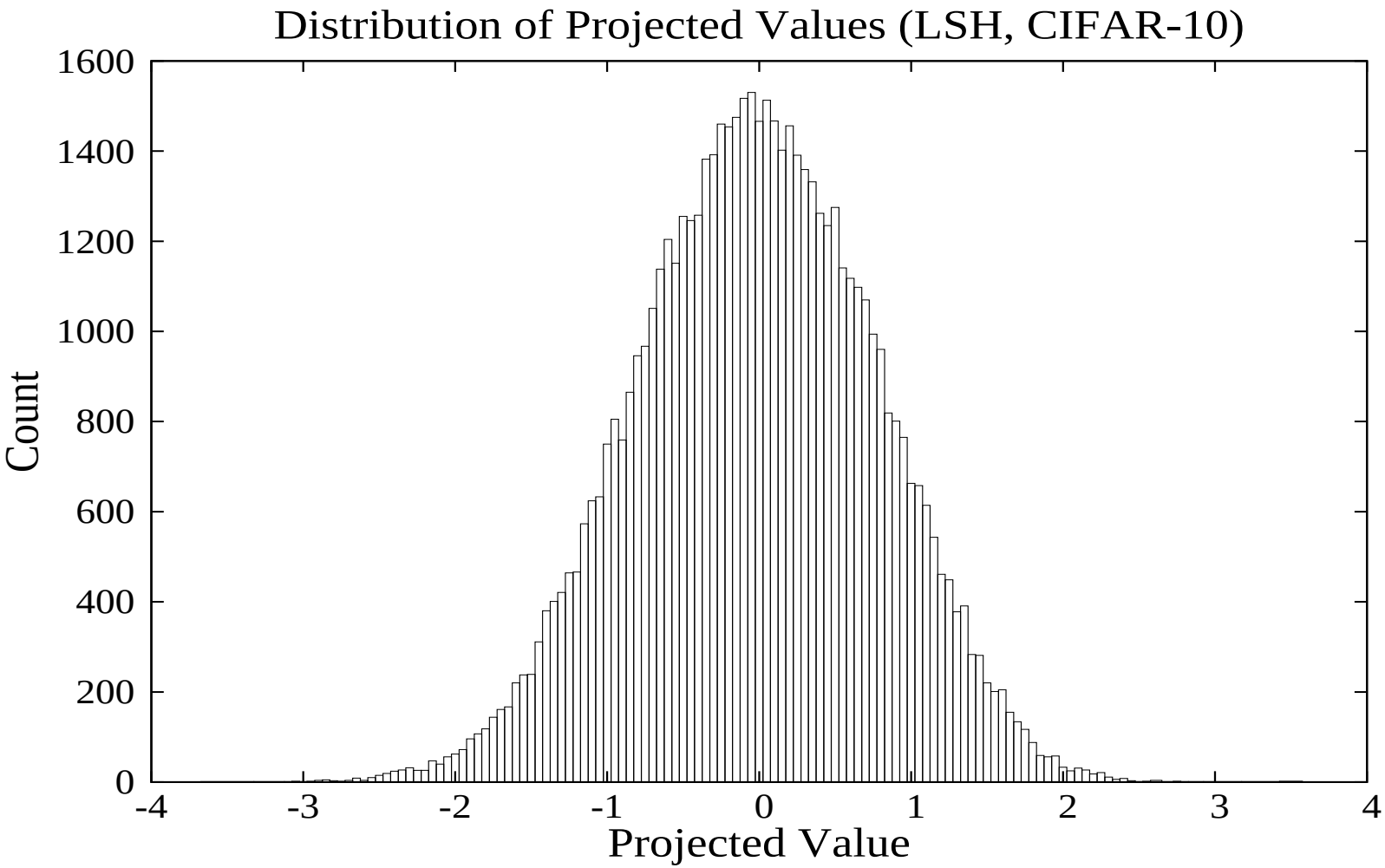
[5] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, Shih-Fu Chang. **Supervised Hashing with Kernels**. *In CVPR'12*.

[6] Yunchao Gong, Svetlana Lazebnik. **Iterative Quantization: A Procrustean Approach to Learning Binary Codes for Large-scale Image Retrieval**. *In CVPR'11*.

Learning to Hash for Large Scale Image Retrieval

Why Learn the Quantisation Threshold(s)?

- Threshold at **zero** is in region of highest point density:



Previous work

- **Single Bit Quantisation (SBQ)**: Single static threshold placed at zero (default used in most previous work) [13].
- **Double Bit Quantisation (DBQ)**: Multiple thresholds optimised by minimising squared error objective [7].
- **Manhattan Hashing Quantisation (MHQ)**: Multiple thresholds optimised by k-means [8].

[7] Weihao Kong, Wu-Jun Li.**Double Bit Quantisation**. *In AAAI'12*.
[8] Weihao Kong, Wu-Jun Li. **Manhattan Hashing for Large-Scale Image Retrieval**. *In SIGIR'12*.
[13] Piotr Indyk and Rajeev Motwani. **Approximate nearest neighbors: Towards removing the curse of dimensionality**. *In STOC'98*.

Previous work

Method	Data-Dependent	# Thresholds	Codebook
SBQ [13]		1	0/1
DBQ [7]	✓	2	00/11/10
MHQ [8]	✓	$2^B - 1$	NBC
NPQ [1]	✓	$1,2,2^{B-1}$	Any

B : # *of bits per projected dimension e.g. 1,2,3,4 bits*
NBC: *Natural Binary Code*

[1] Sean Moran, Victor Lavrenko, Miles Osborne. **Neighbourhood Preserving Quantisation for LSH**. *In SIGIR'13*.

Semi-Supervised Objective Function

- **Idea: learn** thresholds by fusing **two sources** of information:

  1. **Supervised:** distances (or labels) of points in the original D-dimensional feature space

  2. **Unsupervised:** distances between pairs of points in the 1D projected space

- **Next few slides:** constructing the semi-supervised objective function

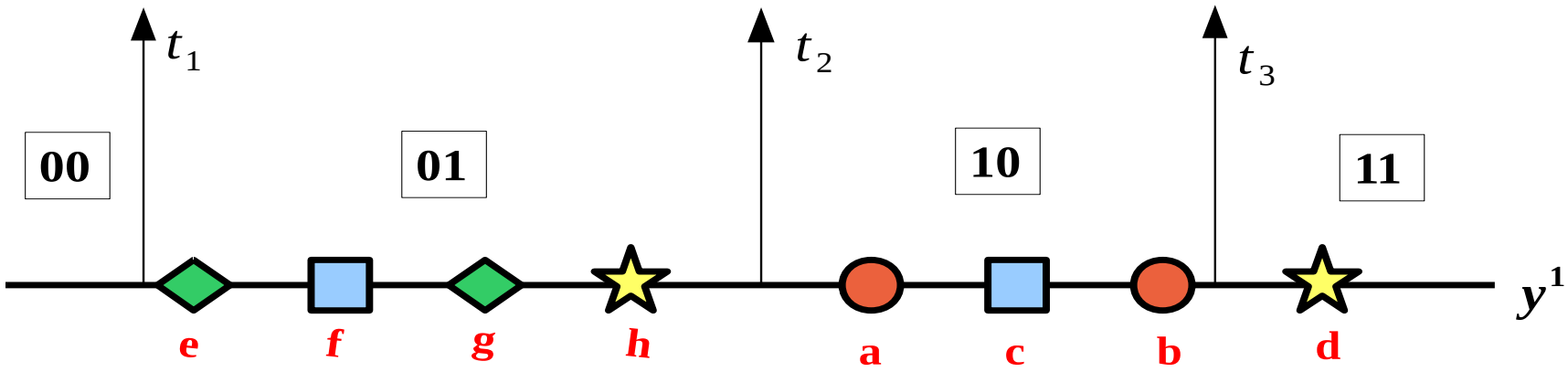
Adjacency Matrix **S**

- **Adjacency matrix S** $\in \{0,1\}^{N_{trd} \times N_{trd}}$:

$$S_{ij} = \begin{cases} 1, & \text{if } \mathbf{x}_i, \mathbf{x}_j \text{ are true NNs} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

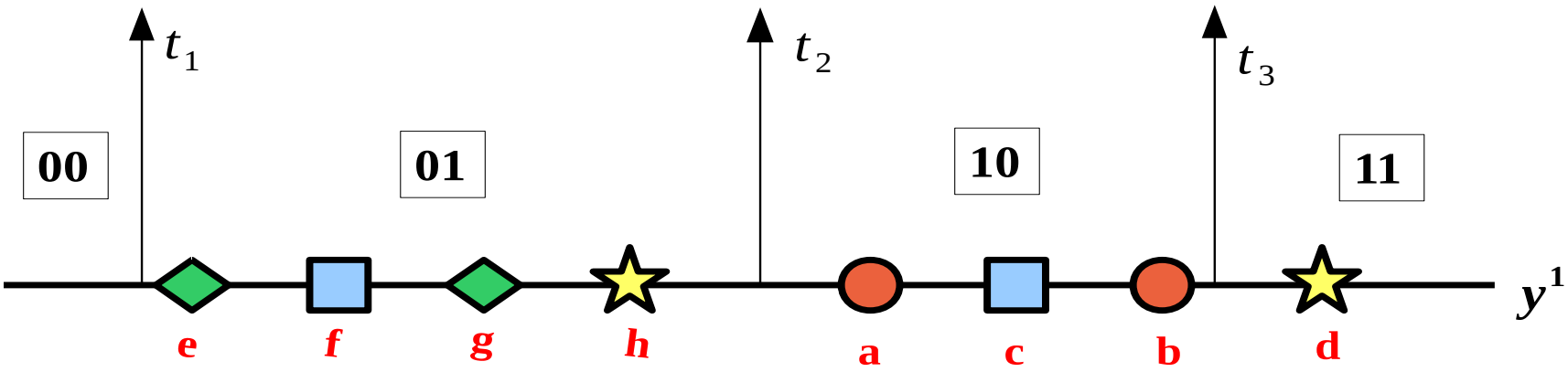
Note: $N_{trd} \ll N$ and matrix **S** highly sparse (1% of elements non-zero)

Adjacency Matrix **S**



$$
\mathbf{S} = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{array}
\begin{array}{c}
\begin{array}{cccccccc} a & b & c & d & e & f & g & h \end{array} \\
\begin{pmatrix}
 & +1 & & & & & & \\
+1 & & & & & & & \\
 & & & & & +1 & & \\
 & & & & & & & +1 \\
 & & & & & & +1 & \\
 & & +1 & & & & & \\
 & & & & +1 & & & \\
 & & & +1 & & & &
\end{pmatrix}
\end{array}
$$

Adjacency Matrix **S**



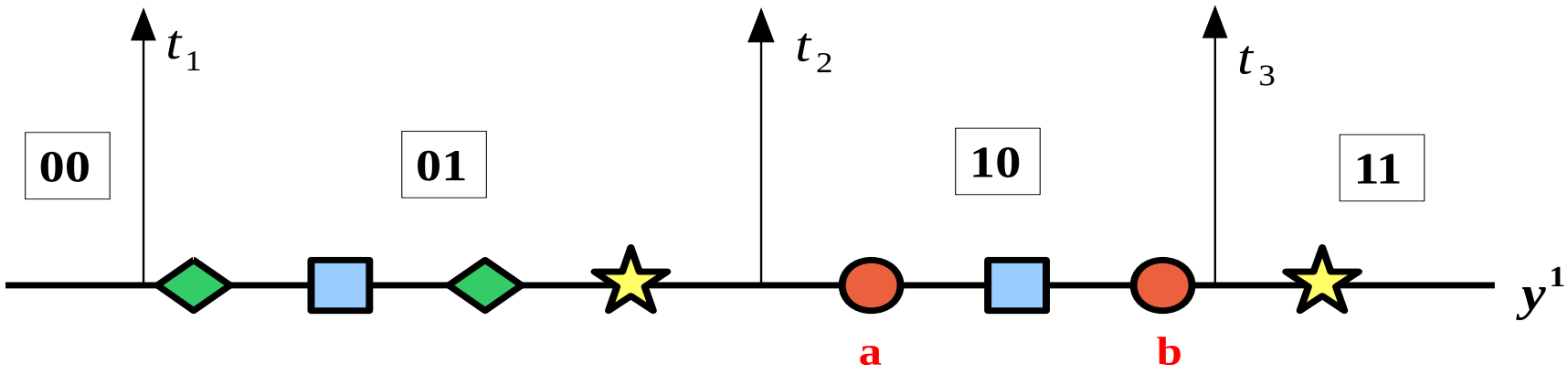
S	a	b	c	d	e	f	g	h
a	0	$+1$	0	0	0	0	0	0
b	$+1$	0	0	0	0	0	0	0
c	0	0	0	0	0	$+1$	0	0
d	0	0	0	0	0	0	0	$+1$
e	0	0	0	0	0	0	$+1$	0
f	0	0	$+1$	0	0	0	0	0
g	0	0	0	0	$+1$	0	0	0
h	0	0	0	$+1$	0	0	0	0

Indicator Matrix **P**

- **Indicator matrix $\mathbf{P} \in \{0,1\}^{N_{trd} \times N_{trd}}$:**

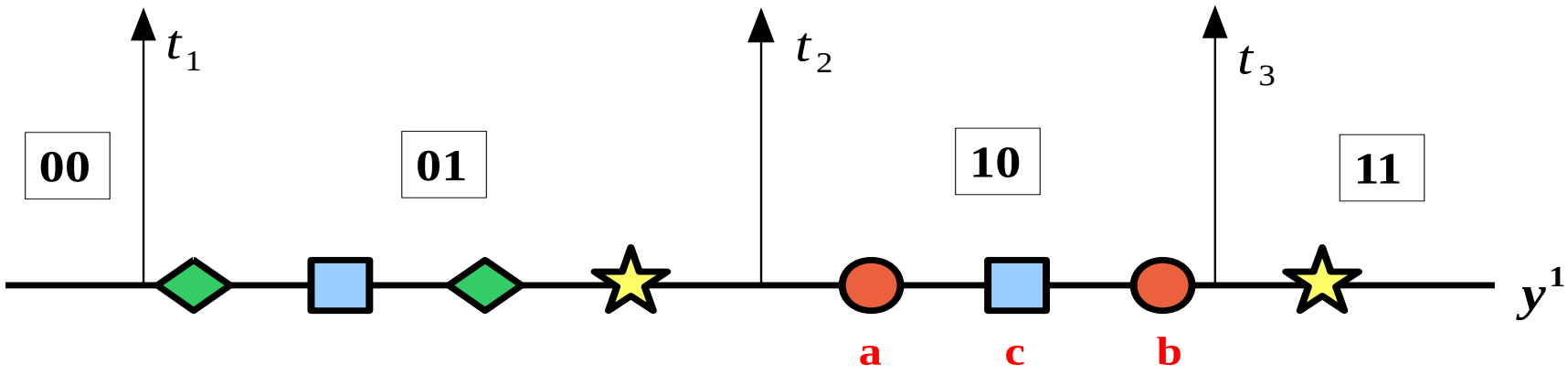
$$P_{ij}^{k} = \begin{cases} 1, & \text{if} \quad \exists_{\gamma} \quad s.t. \quad t_{k\gamma} \leq (y_i^k, y_j^k) \leq t_{k(\gamma+1)} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Indicator Matrix **P**



$$\mathbf{P} \quad \begin{matrix} & a & b & c & d & e & f & g & h \\ a & & +1 & & & & & & \\ b & +1 & & & & & & & \\ c & & & & & & & & \\ d & & & & & & & & \\ e & & & & & & & & \\ f & & & & & & & & \\ g & & & & & & & & \\ h & & & & & & & & \end{matrix}$$

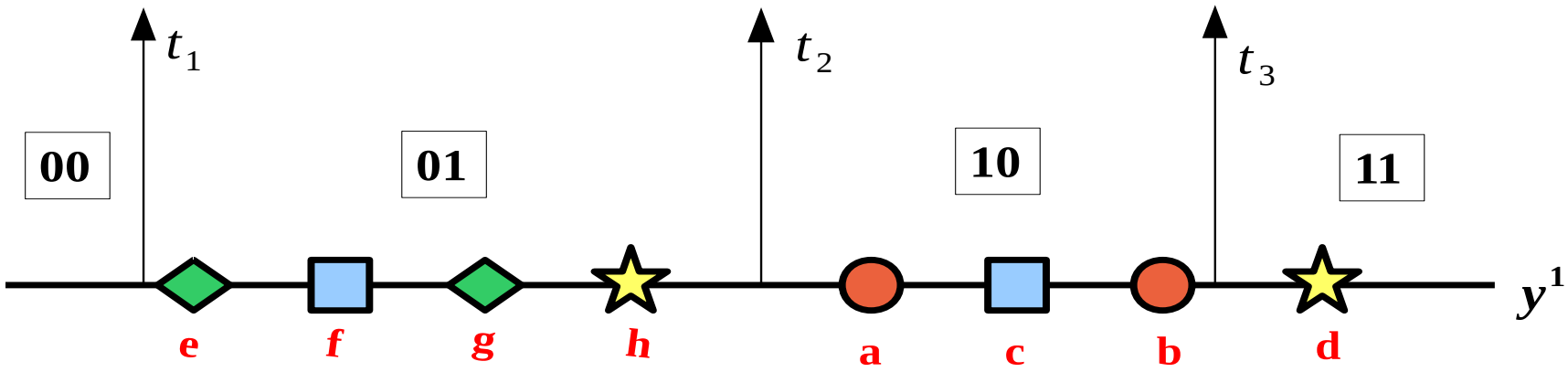
Indicator Matrix **P**



$$\mathbf{P} \quad \begin{matrix} & a & b & c & d & e & f & g & h \end{matrix}$$

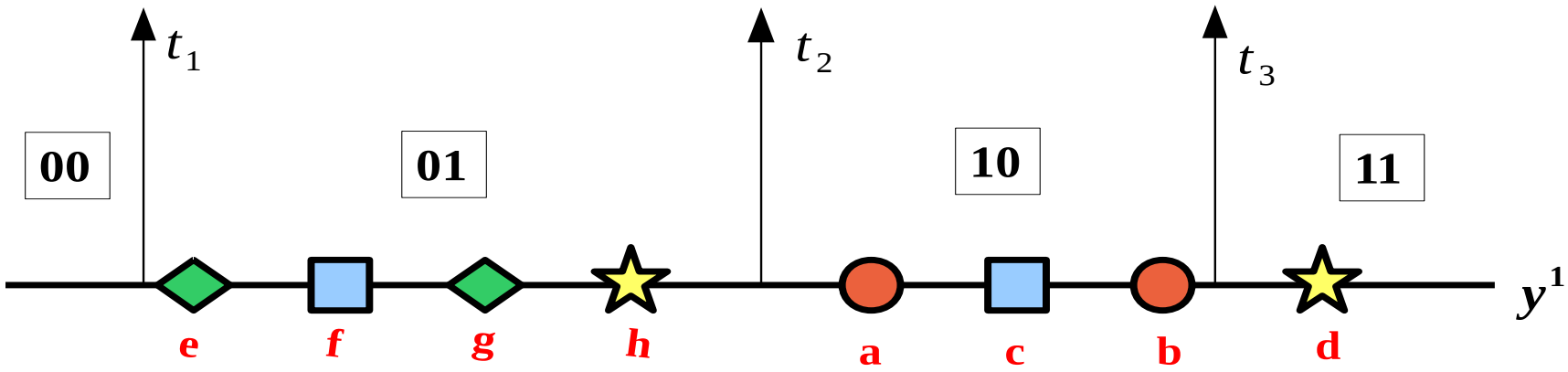
$$\begin{matrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{matrix} \begin{pmatrix} & +1 & +1 & & & & & \\ +1 & & +1 & & & & & \\ +1 & +1 & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \end{pmatrix}$$

Indicator Matrix **P**



$$\mathbf{P} = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{array} \begin{array}{c} \begin{array}{cccccccc} a & b & c & d & e & f & g & h \end{array} \\ \begin{pmatrix} & +1 & +1 & & & & & \\ +1 & & +1 & & & & & \\ +1 & +1 & & & & & & \\ & & & & & & & \\ & & & & & +1 & +1 & +1 \\ & & & & +1 & & +1 & +1 \\ & & & & +1 & +1 & & +1 \\ & & & & +1 & +1 & +1 & \end{pmatrix} \end{array}$$

Indicator Matrix **P**



$$
\mathbf{P} = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{array}
\begin{array}{c}
\begin{array}{cccccccc} a & b & c & d & e & f & g & h \end{array} \\
\left(\begin{array}{cccccccc}
0 & +1 & +1 & 0 & 0 & 0 & 0 & 0 \\
+1 & 0 & +1 & 0 & 0 & 0 & 0 & 0 \\
+1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & +1 & +1 & +1 \\
0 & 0 & 0 & 0 & +1 & 0 & +1 & +1 \\
0 & 0 & 0 & 0 & +1 & +1 & 0 & +1 \\
0 & 0 & 0 & 0 & +1 & +1 & +1 & 0
\end{array} \right)
\end{array}
$$

Supervised Part: F_1-Measure Maximisation

- Integrate supervisory signal by **maximising F_1-measure**:

$$F_1 = \frac{2\,TP}{2\,TP + FP + FN}$$

- True positives (TPs): **true NNs** in **same** regions
- False positives (FPs): **non-NNs** in **same** regions
- False negatives (FNs): **true NNs** in **different** regions

Supervised Part: F_1-Measure Maximisation

- **True Positives (TPs)**:

$$TP = \frac{1}{2}\sum_{ij} P_{ij} S_{ij} = \frac{1}{2}\|\mathbf{P} \circ \mathbf{S}\|_1$$

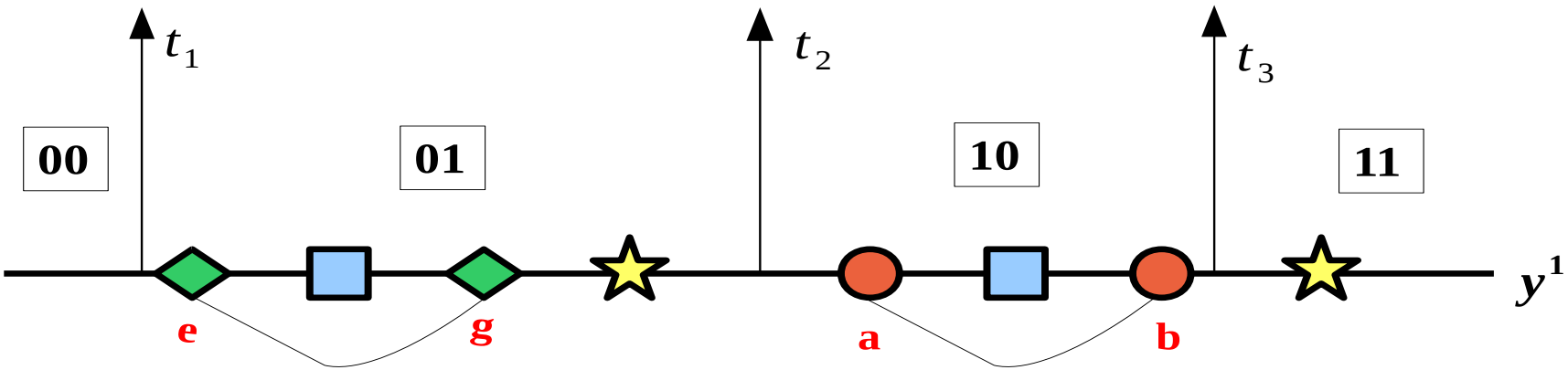
$$\mathbf{P} \circ \mathbf{S} \quad \begin{matrix} a & b & c & d & e & f & g & h \end{matrix}$$

$$\begin{matrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{matrix} \begin{pmatrix} 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 \\ +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & +1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & +1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Supervised Part: F_1-Measure Maximisation

- **True Positives (TPs):**

$$TP = \frac{1}{2}\|(\mathbf{P}) \circ (\mathbf{S})\|_1 = 4/2 = 2$$



Supervised Part: F_1-Measure Maximisation

- **False Negatives (FNs)**:

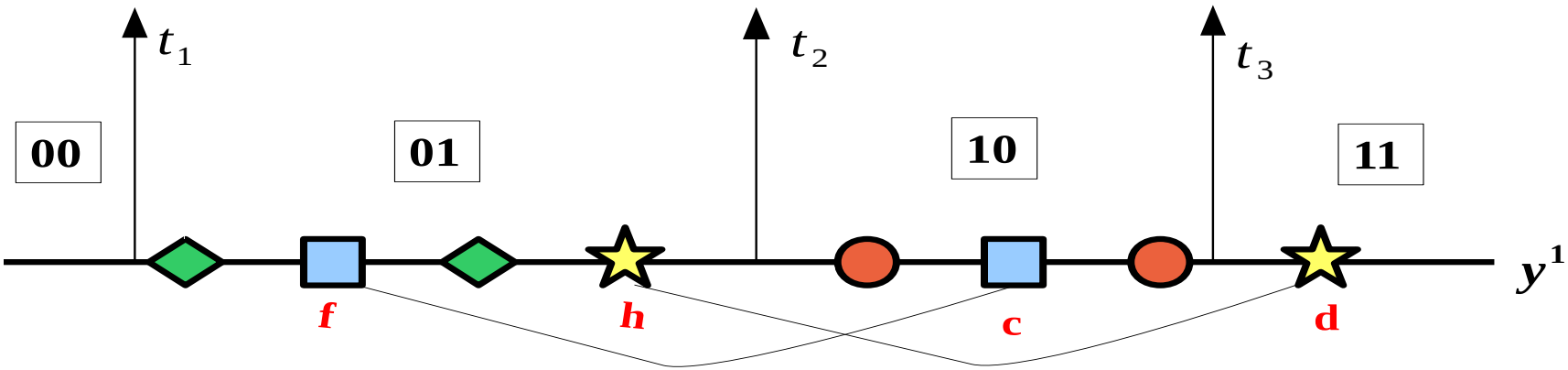
$$FN = \frac{1}{2}\sum_{ij} S_{ij} - TP = \frac{1}{2}\|\mathbf{S}\|_1 - TP$$

$$\begin{array}{c} \mathbf{S} \\ a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{array} \begin{array}{c} \begin{array}{cccccccc} a & b & c & d & e & f & g & h \end{array} \\ \begin{pmatrix} 0 & +1 & 0 & 0 & 0 & 0 & 0 & 0 \\ +1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & +1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & +1 \\ 0 & 0 & 0 & 0 & 0 & 0 & +1 & 0 \\ 0 & 0 & +1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & +1 & 0 & 0 & 0 \\ 0 & 0 & 0 & +1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Supervised Part: F_1-Measure Maximisation

- **False Negatives (FNs)**:

$$FN = \frac{1}{2}\|\mathbf{S}\|_1 - TP = 8/2 - 2 = 2$$



Supervised Part: F_1-Measure Maximisation

- **False Positives (FPs)**:

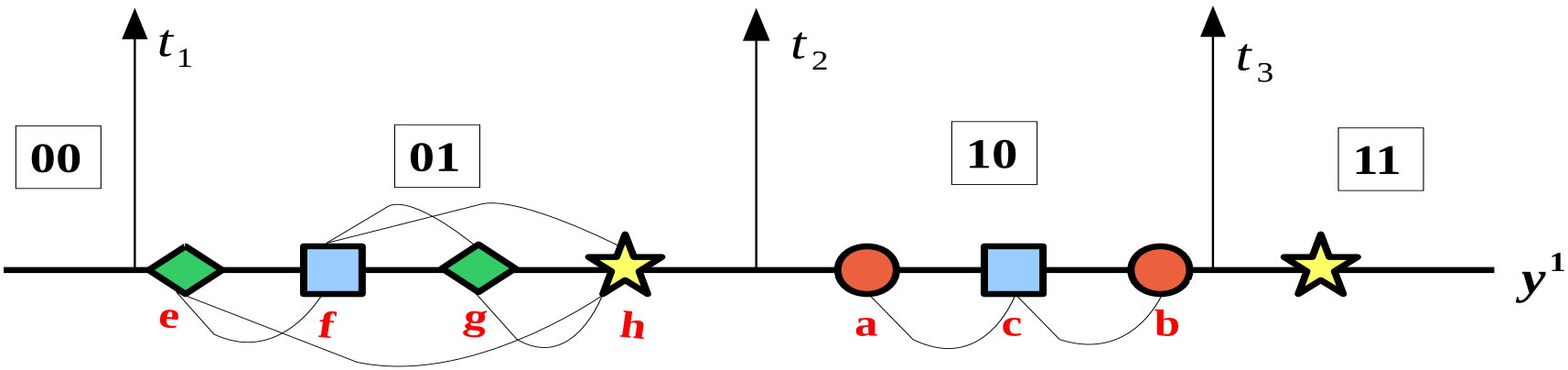
$$FP = \frac{1}{2}\sum_{ij} P_{ij} - TP = \frac{1}{2}\|\mathbf{P}\|_1 - TP$$

$$\begin{array}{c} \mathbf{P} \\ a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{array} \begin{array}{c} \begin{array}{cccccccc} a & b & c & d & e & f & g & h \end{array} \\ \begin{pmatrix} 0 & +1 & +1 & 0 & 0 & 0 & 0 & 0 \\ +1 & 0 & +1 & 0 & 0 & 0 & 0 & 0 \\ +1 & +1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & +1 & +1 & +1 \\ 0 & 0 & 0 & 0 & +1 & 0 & +1 & +1 \\ 0 & 0 & 0 & 0 & +1 & +1 & 0 & +1 \\ 0 & 0 & 0 & 0 & +1 & +1 & +1 & 0 \end{pmatrix} \end{array}$$

Supervised Part: F_1-Measure Maximisation

- **False Positives (FPs)**:

$$FP = \frac{1}{2}\sum_{ij} P_{ij} - TP = 9 - 2 = 7$$



- **Note:** Not implemented like this as **P** may be dense.

Semi-Supervised Objective Function

- F_1-measure term:

$$F_1(\mathbf{t}_k) = \frac{2\|\mathbf{P} \circ \mathbf{S}\|_1}{\|\mathbf{S}\|_1 + \|\mathbf{P}\|_1} = 4/13$$

- Unsupervised term:

$$\Omega(\mathbf{t}_k) = \frac{1}{\sigma_k} \sum_{j=1}^{T+1} \sum_{i:y_i^k \in \mathbf{r}_j} \left\{ y_i^k - \mu_j \right\}^2$$

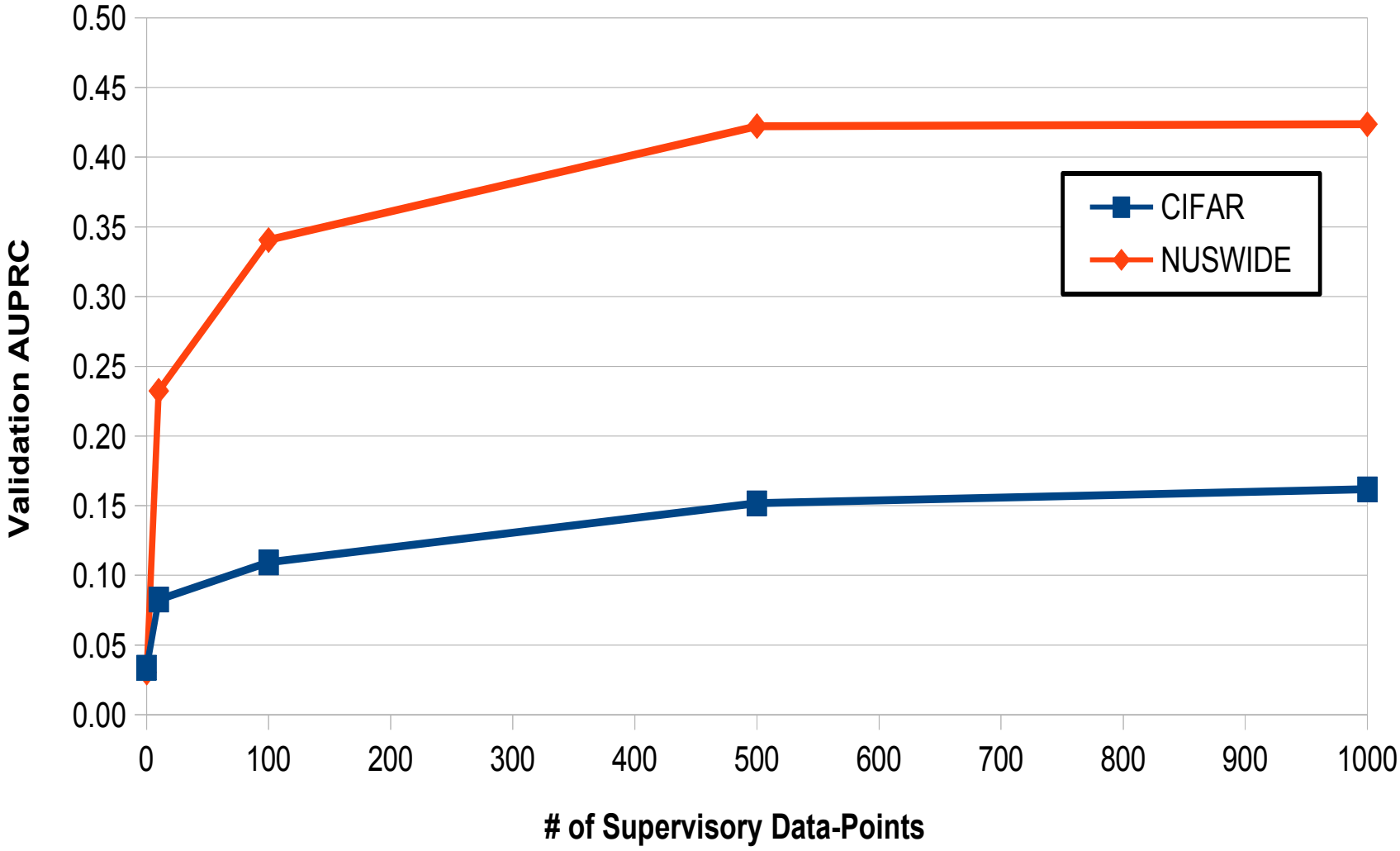
- Maximise objective $\mathcal{J}_{npq}$:

$$\mathcal{J}_{npq}(\mathbf{t}_k) = \alpha F_1(\mathbf{t}_k) + (1-\alpha)(1-\Omega(\mathbf{t}_k))$$

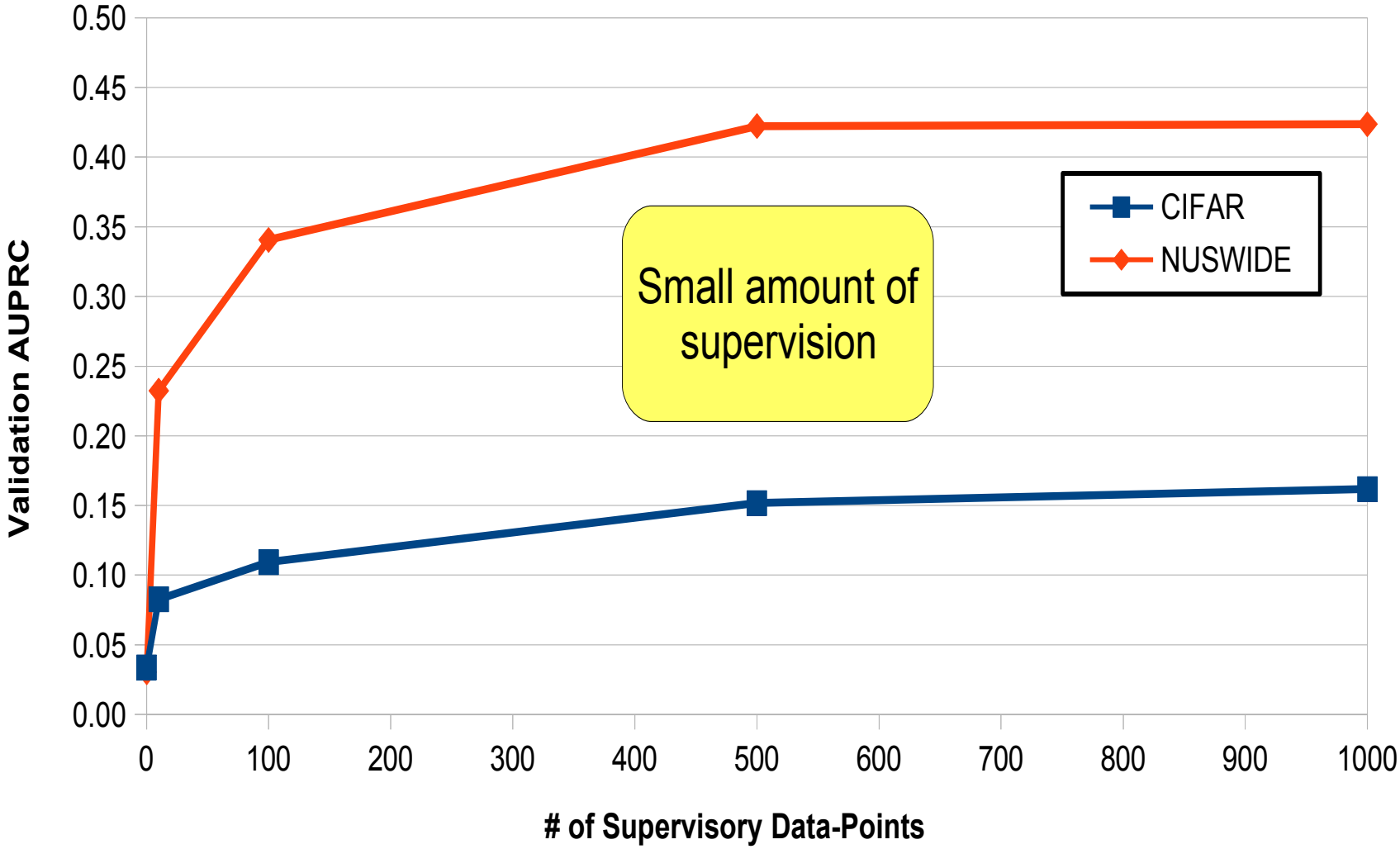
Experimental Results

- **Three** research questions: (more in the thesis):
  - $\mathbf{R}_1$: *Do we outperform a single statically placed threshold?*
  - $\mathbf{R}_2$: *When is more than one threshold beneficial to effectiveness?*
  - $\mathbf{R}_3$: *Can our method generalise to multiple thresholds and many different projection functions?*

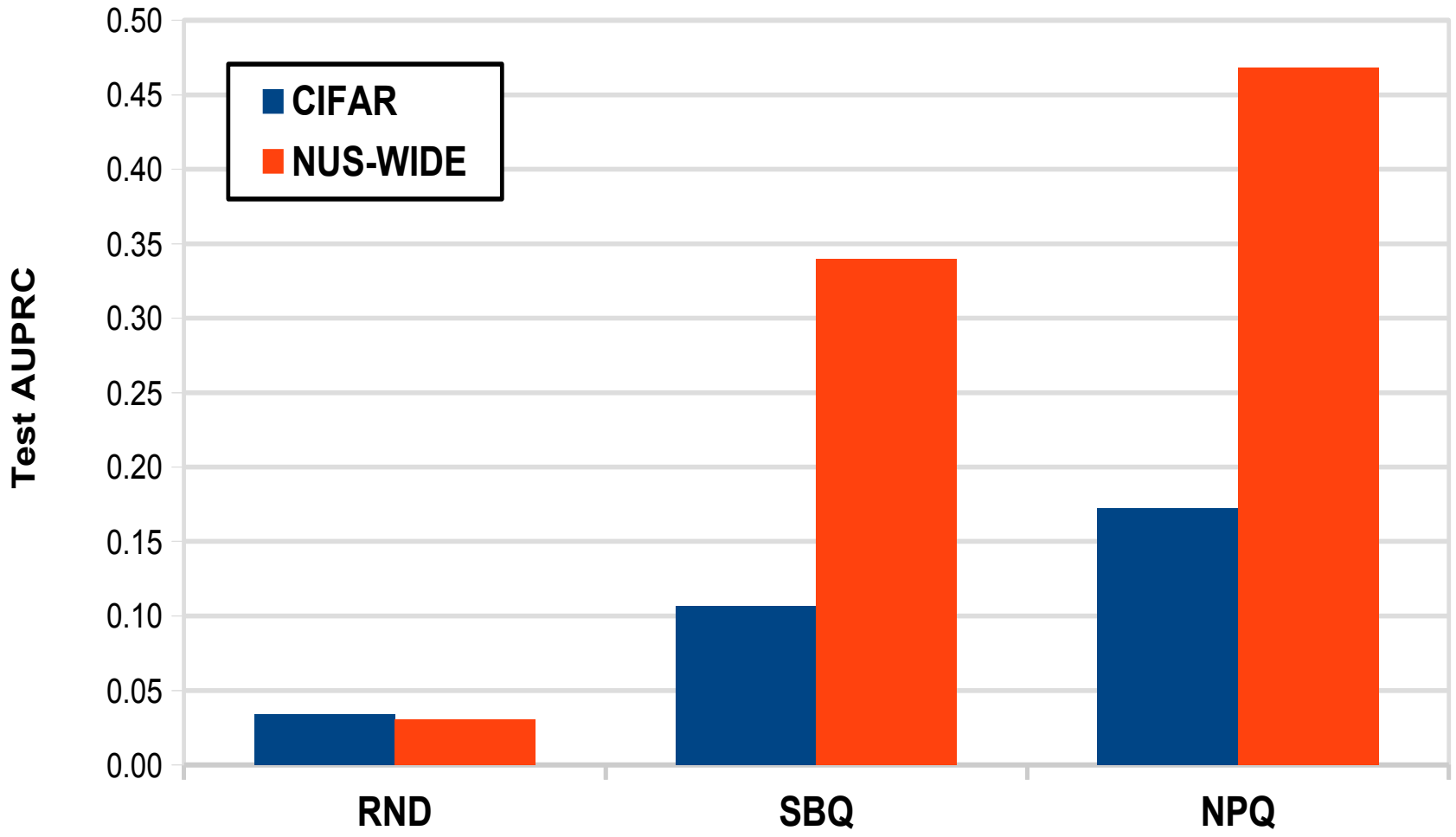
Learning Curve (AUPRC vs. Amount of Supervision)



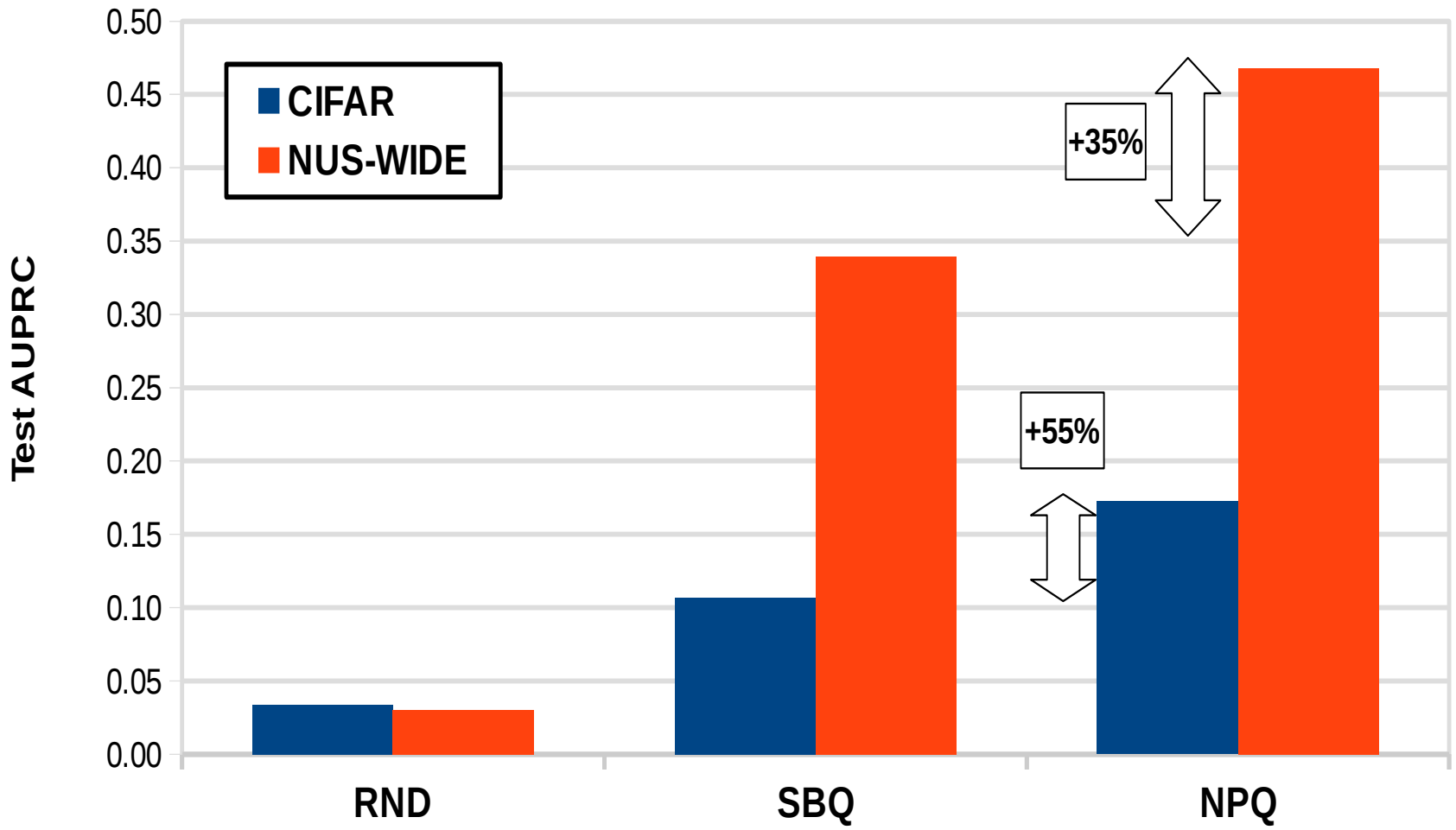
Learning Curve (AUPRC vs. Amount of Supervision)



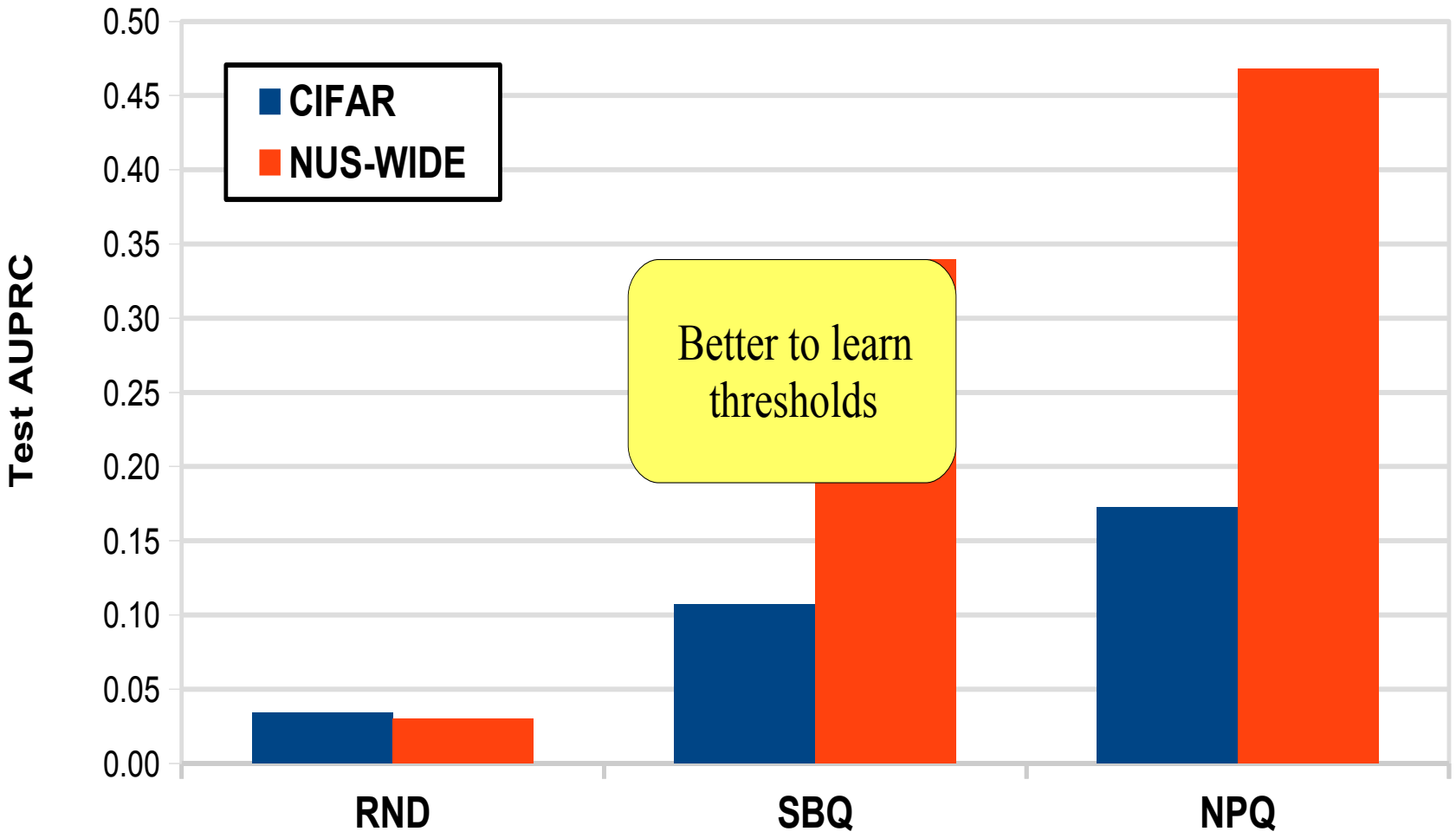
Benefit of Optimising a Single Threshold (R_1)



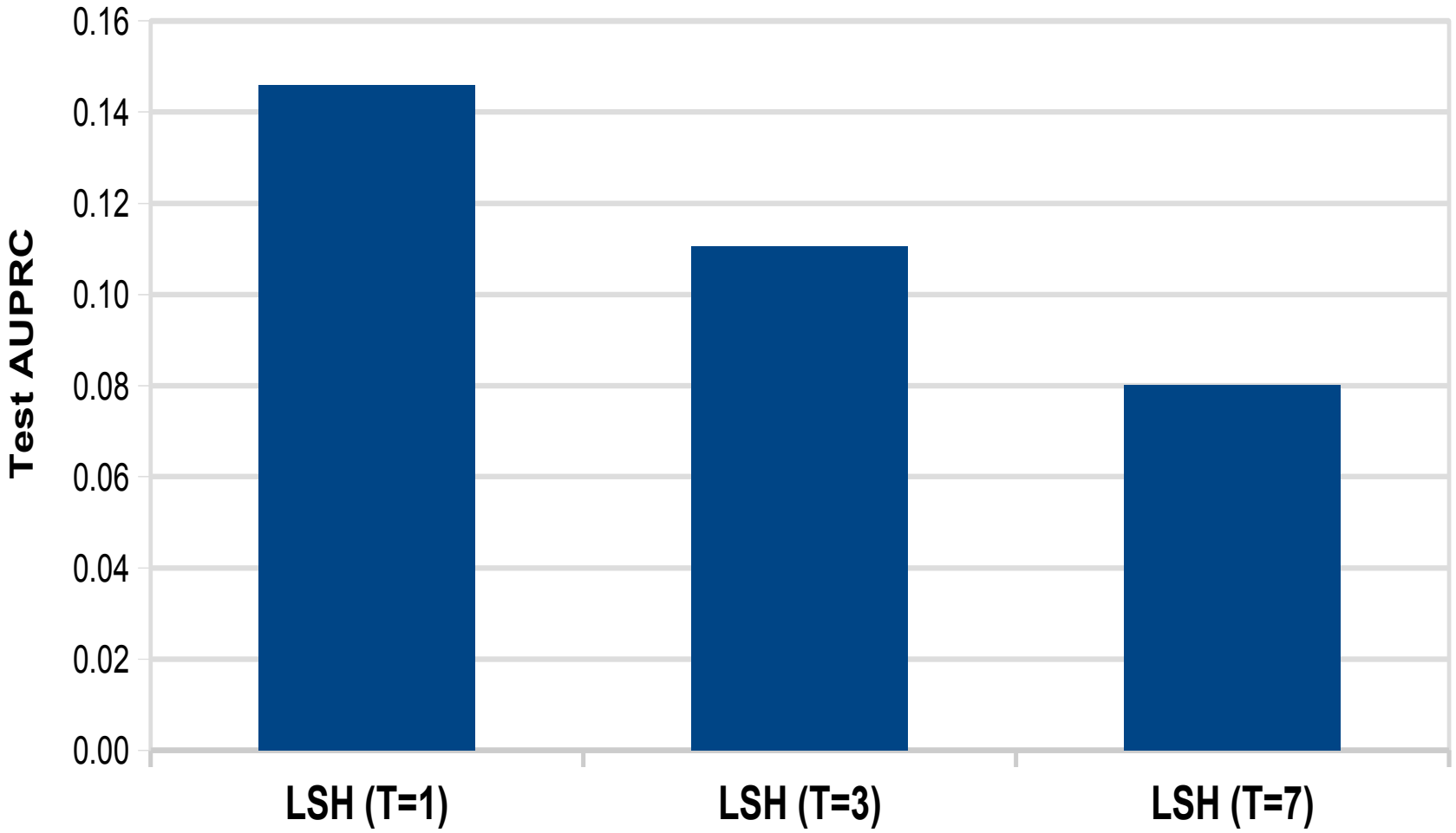
Benefit of Optimising a Single Threshold (R_1)



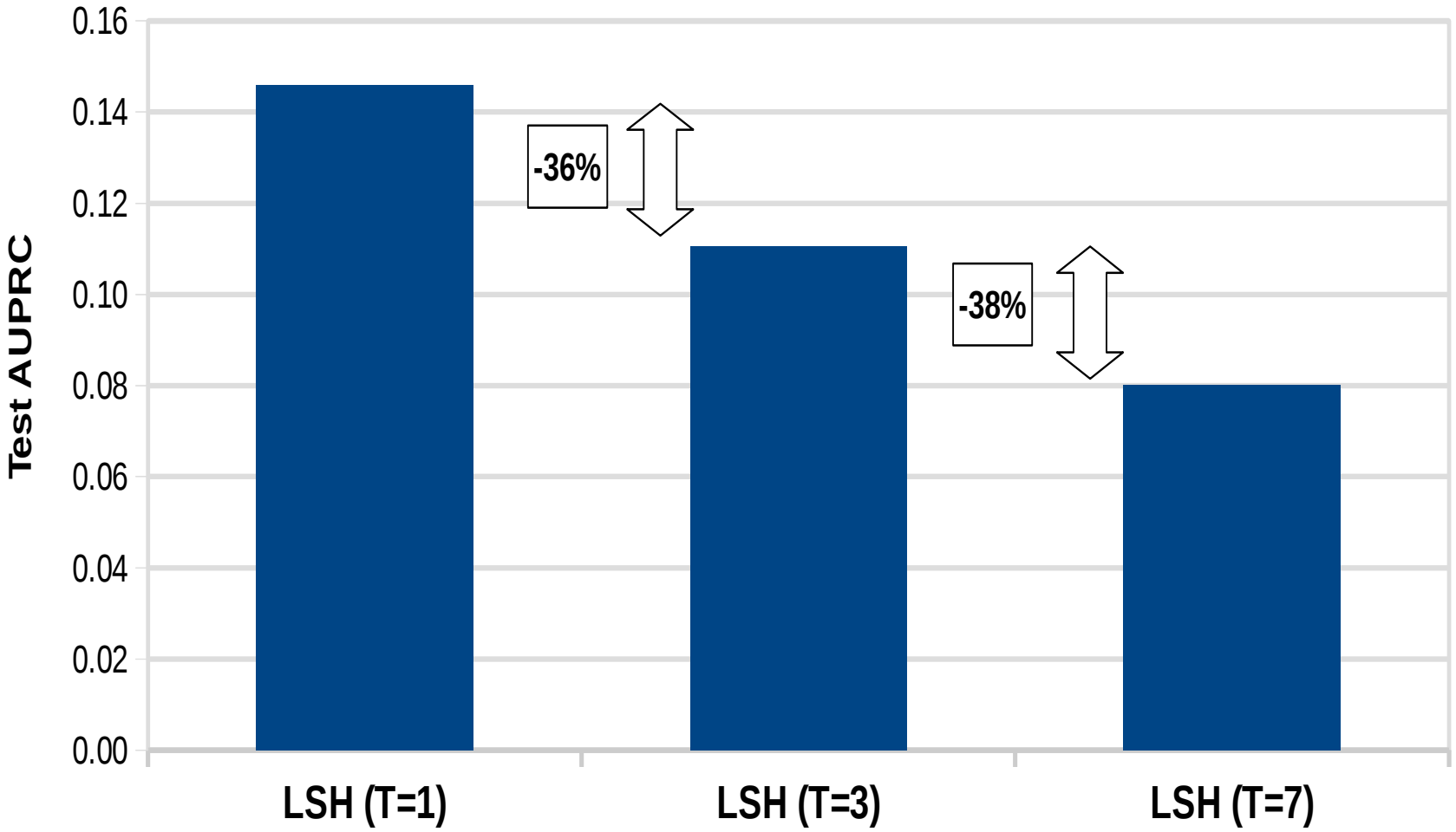
Benefit of Optimising a Single Threshold (R_1)



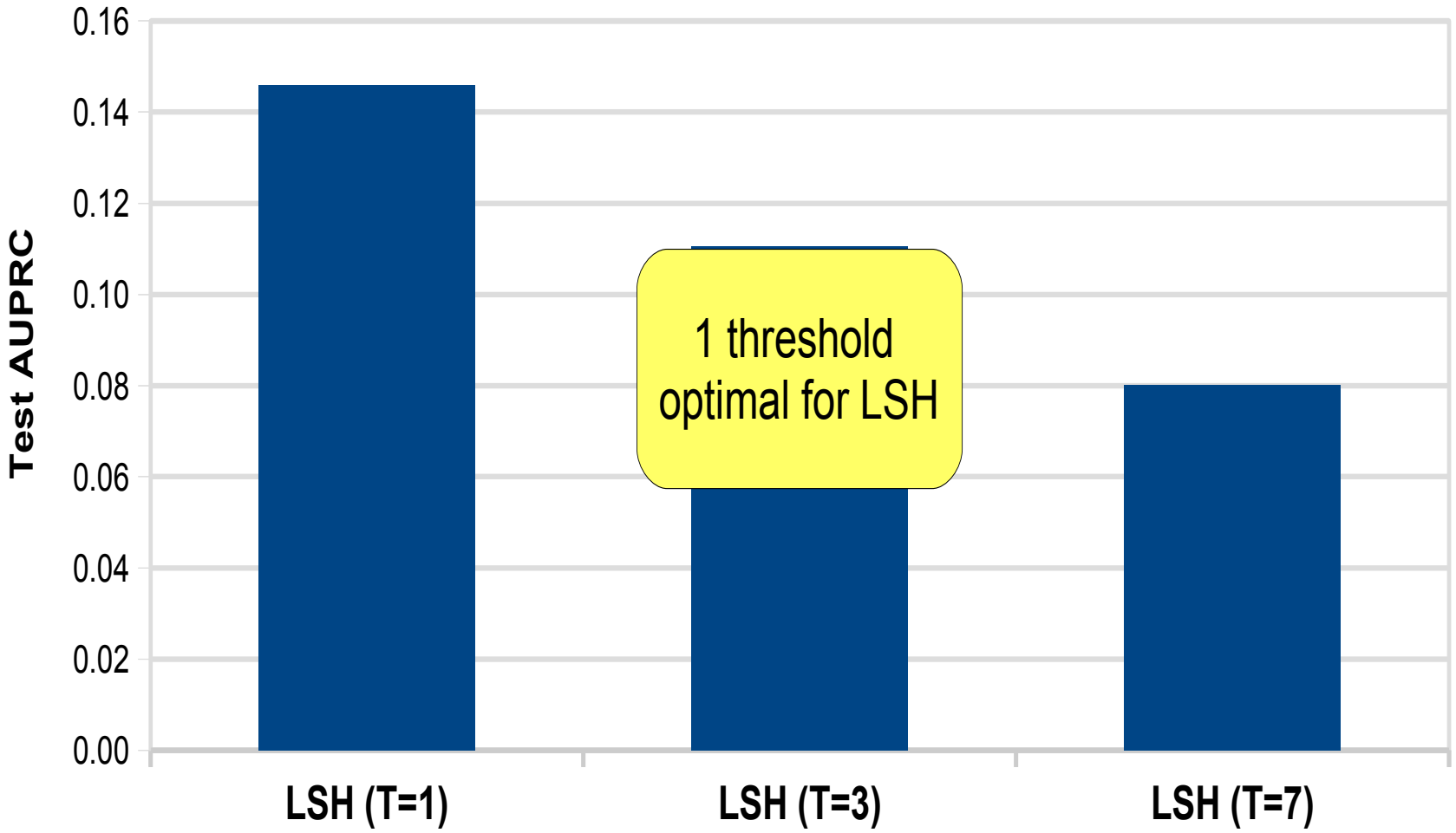
Benefit of Multiple Thresholds (R_2)



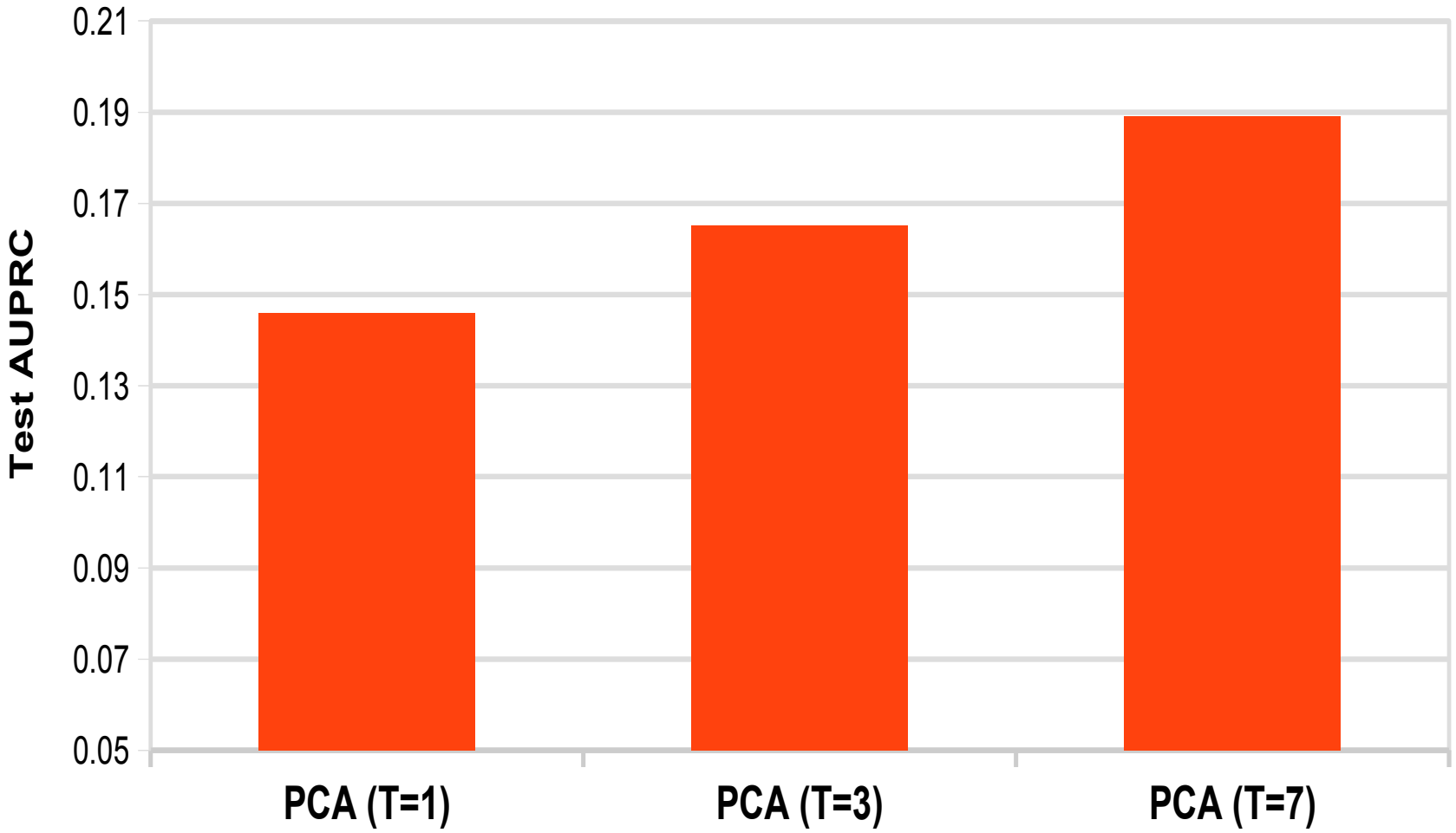
Benefit of Multiple Thresholds (R_2)



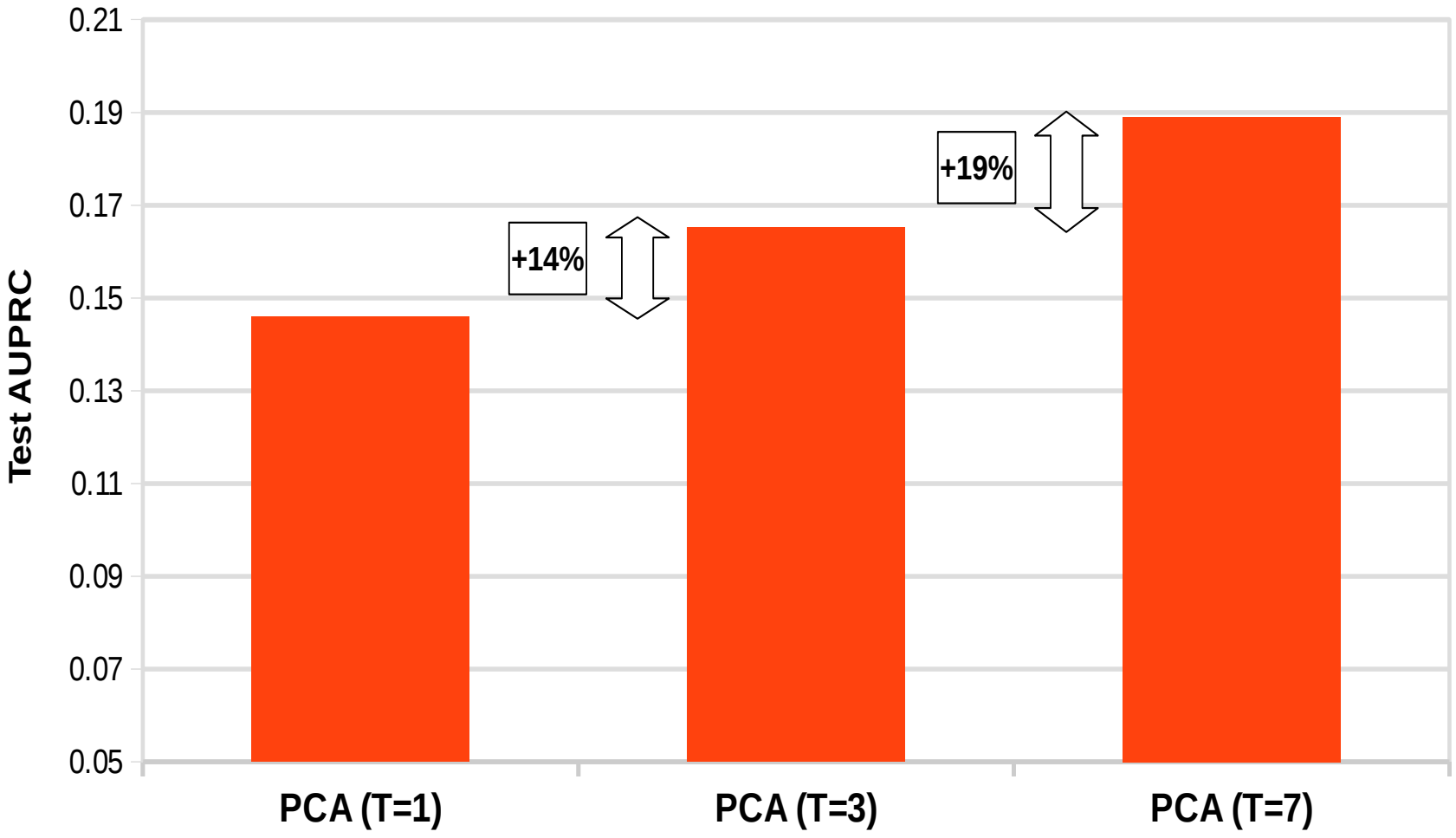
Benefit of Multiple Thresholds (R_2)



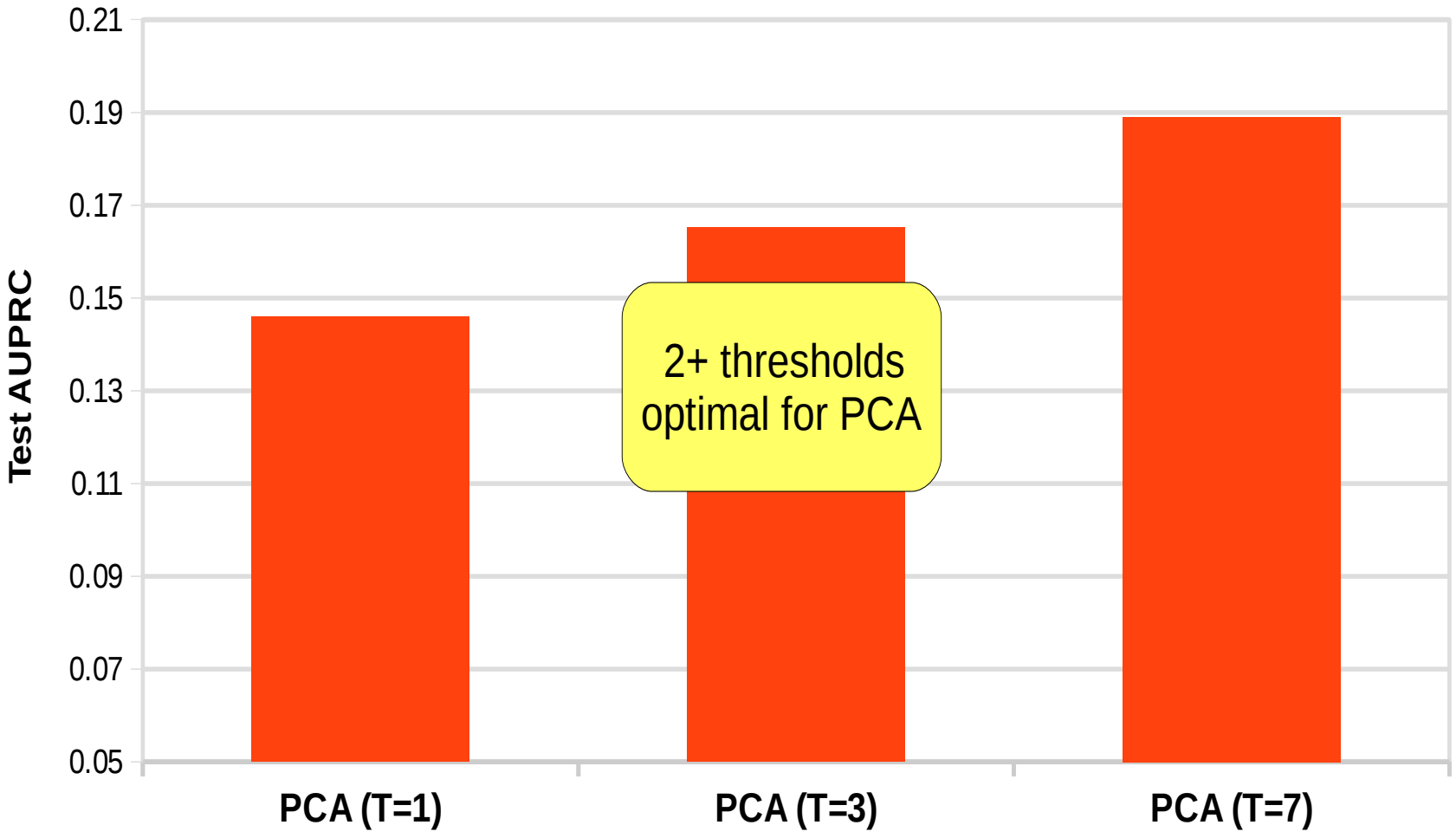
Benefit of Multiple Thresholds (R_2)



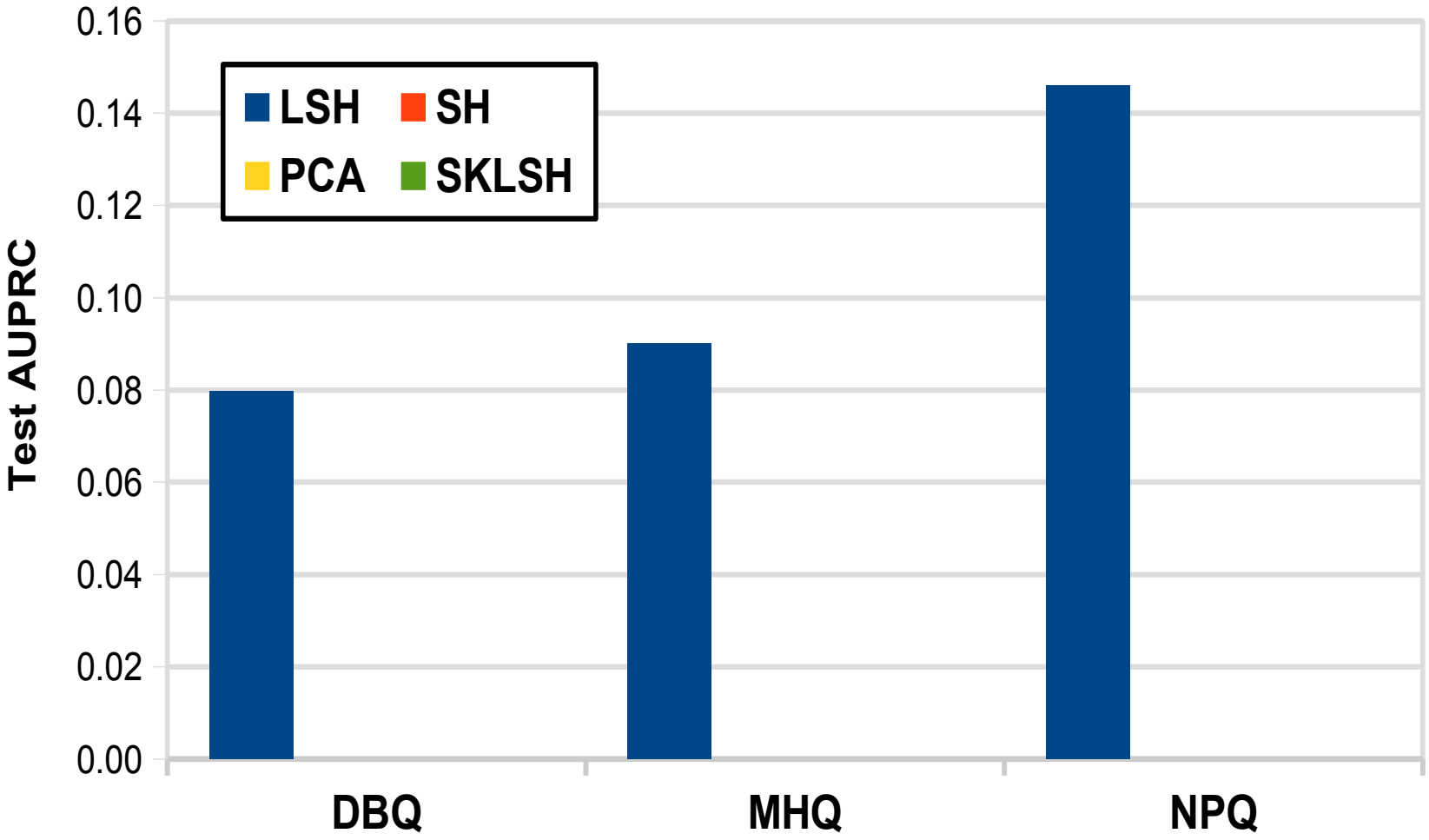
Benefit of Multiple Thresholds (R_2)



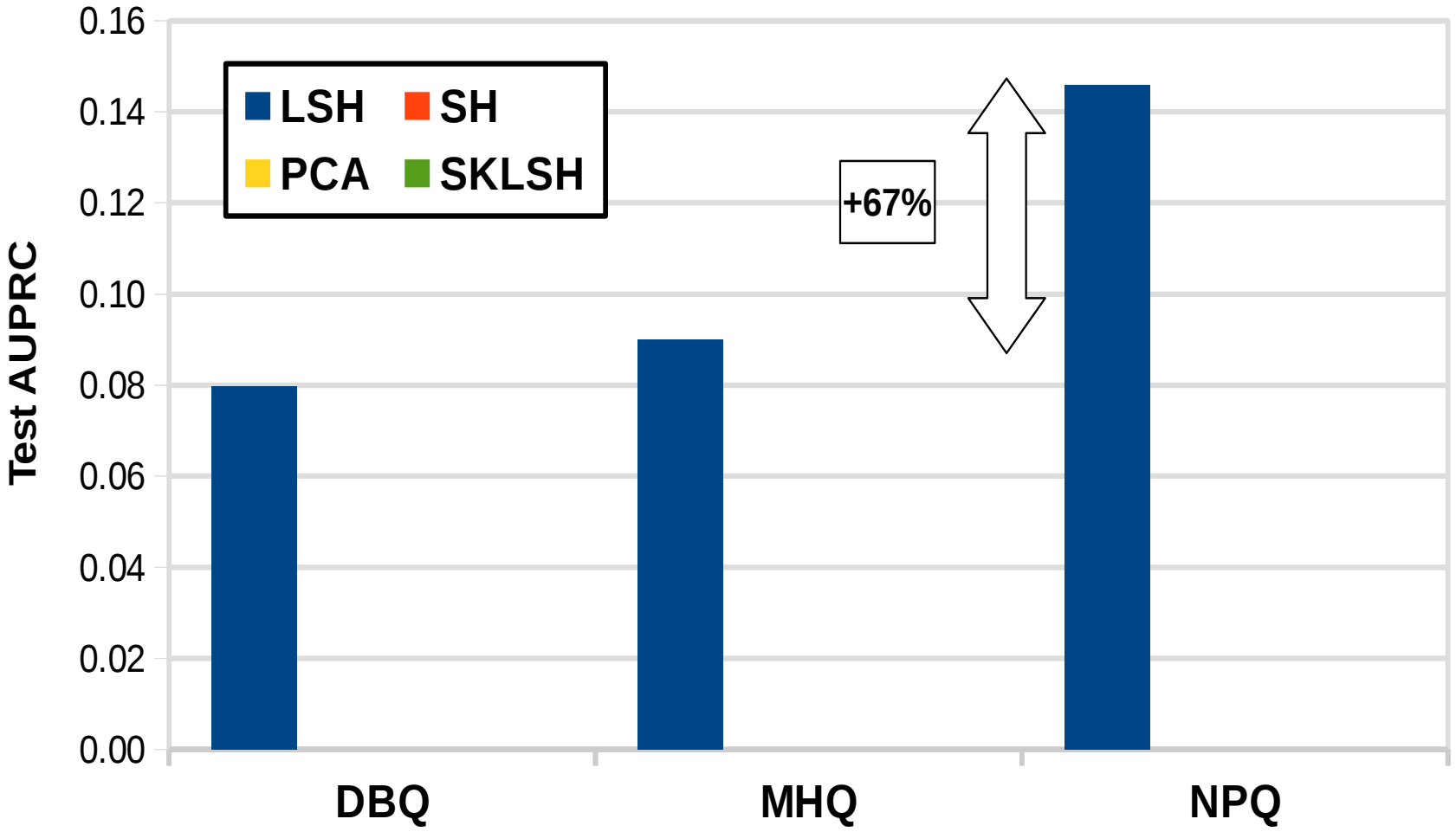
Benefit of Multiple Thresholds (R_2)



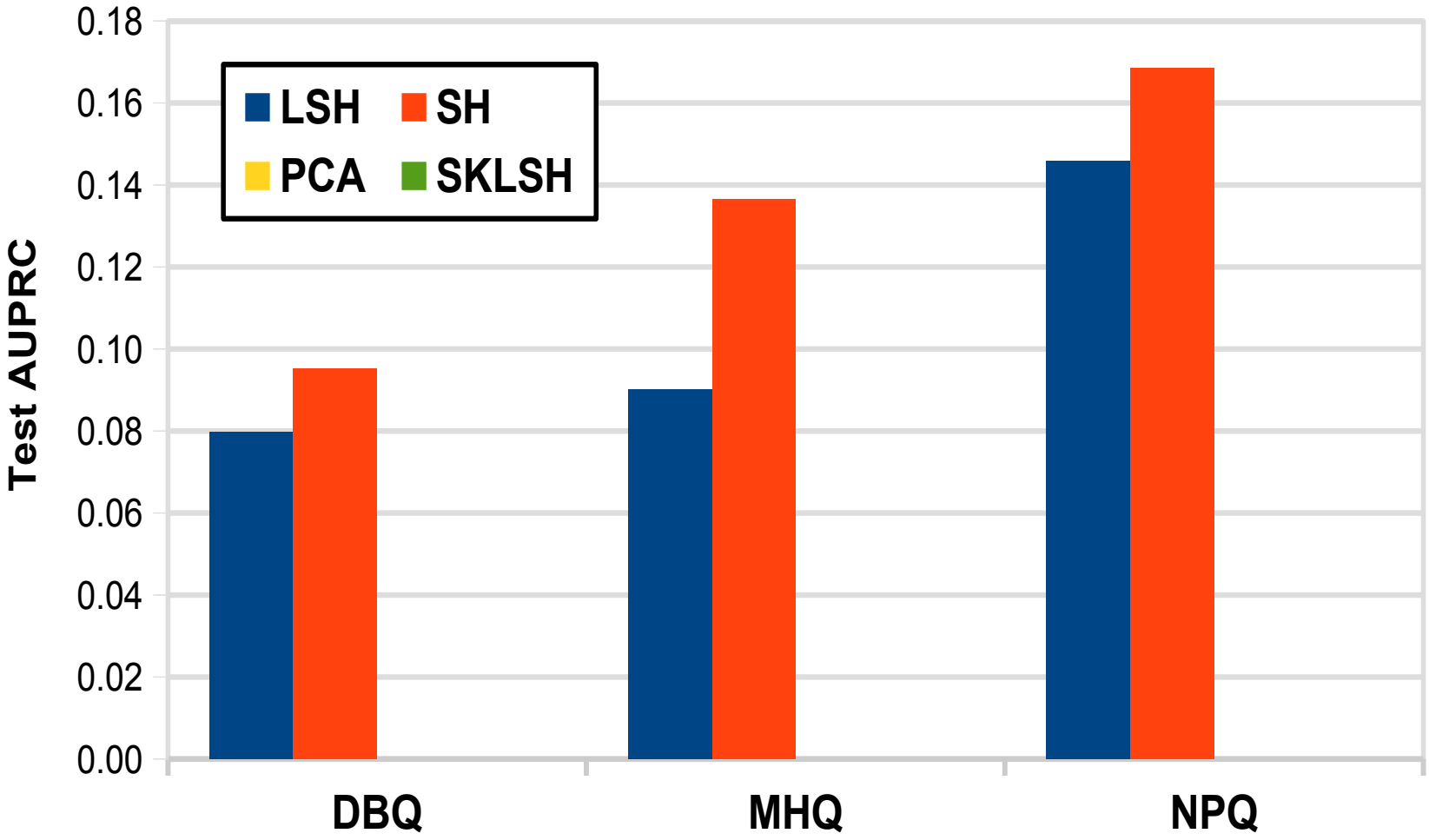
Comparison to the State-of-the-Art (R_3)



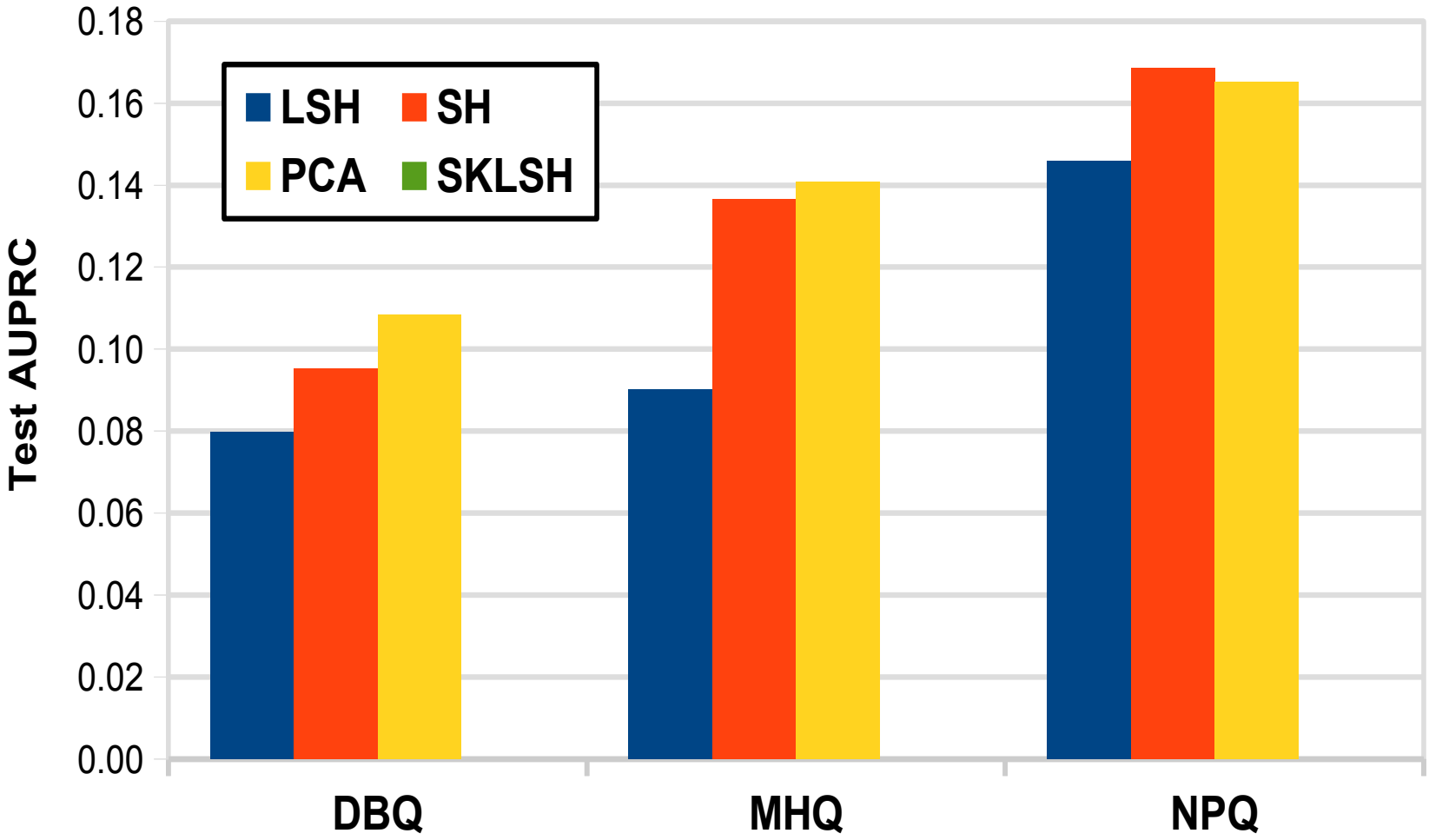
Comparison to the State-of-the-Art (R_3)



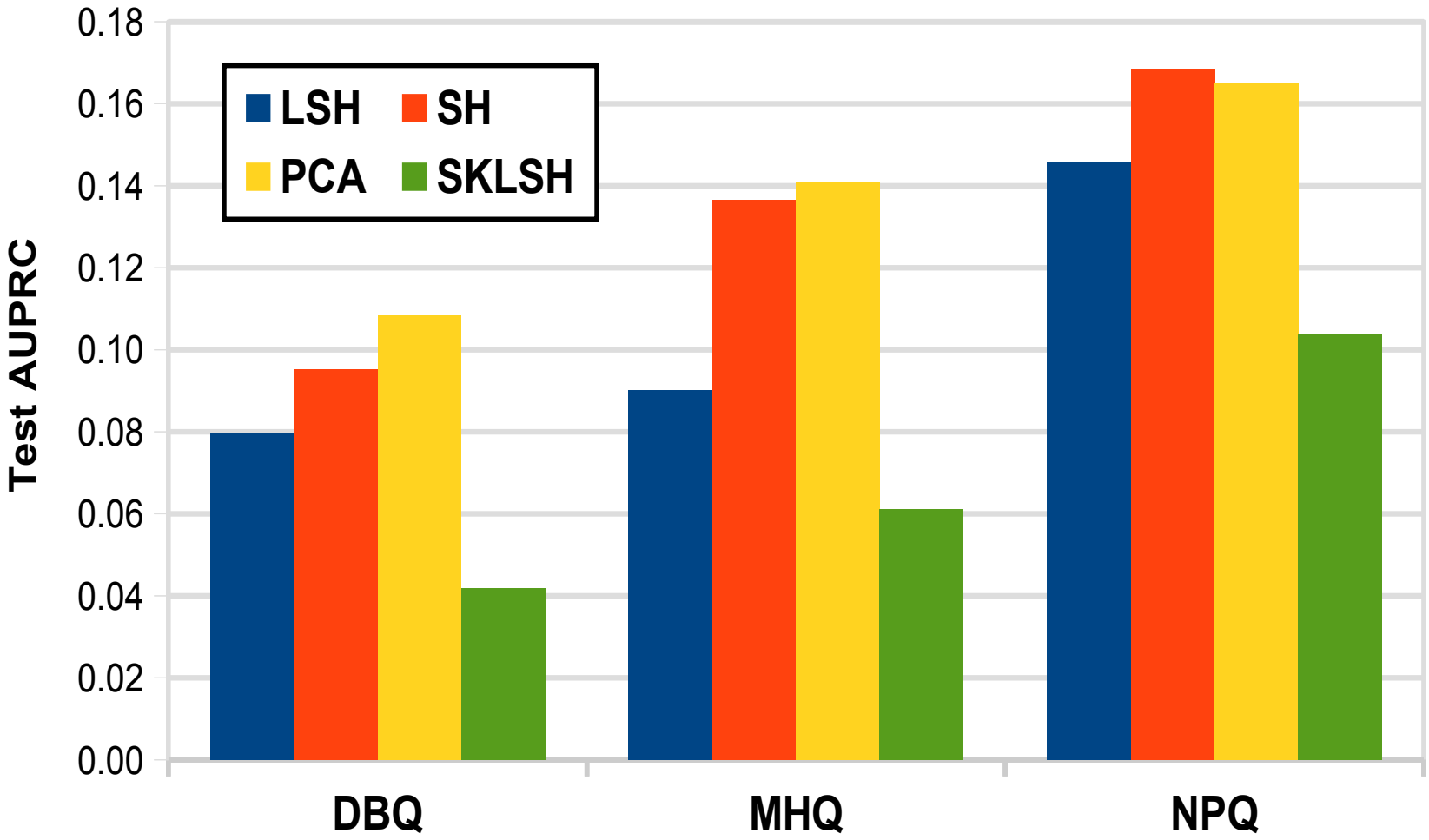
Comparison to the State-of-the-Art (R_3)



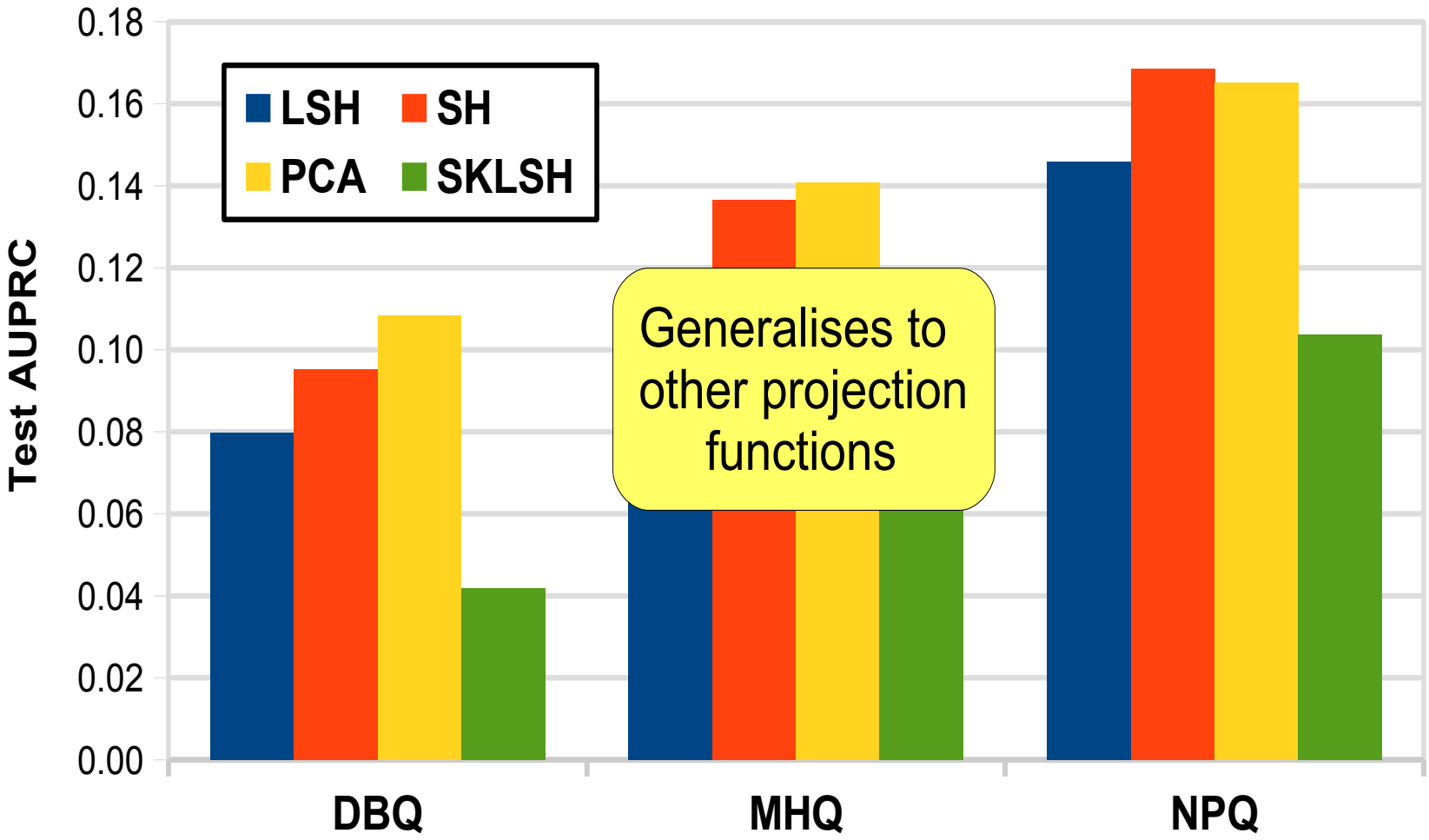
Comparison to the State-of-the-Art (R_3)



Comparison to the State-of-the-Art (R_3)



Comparison to the State-of-the-Art (R_3)

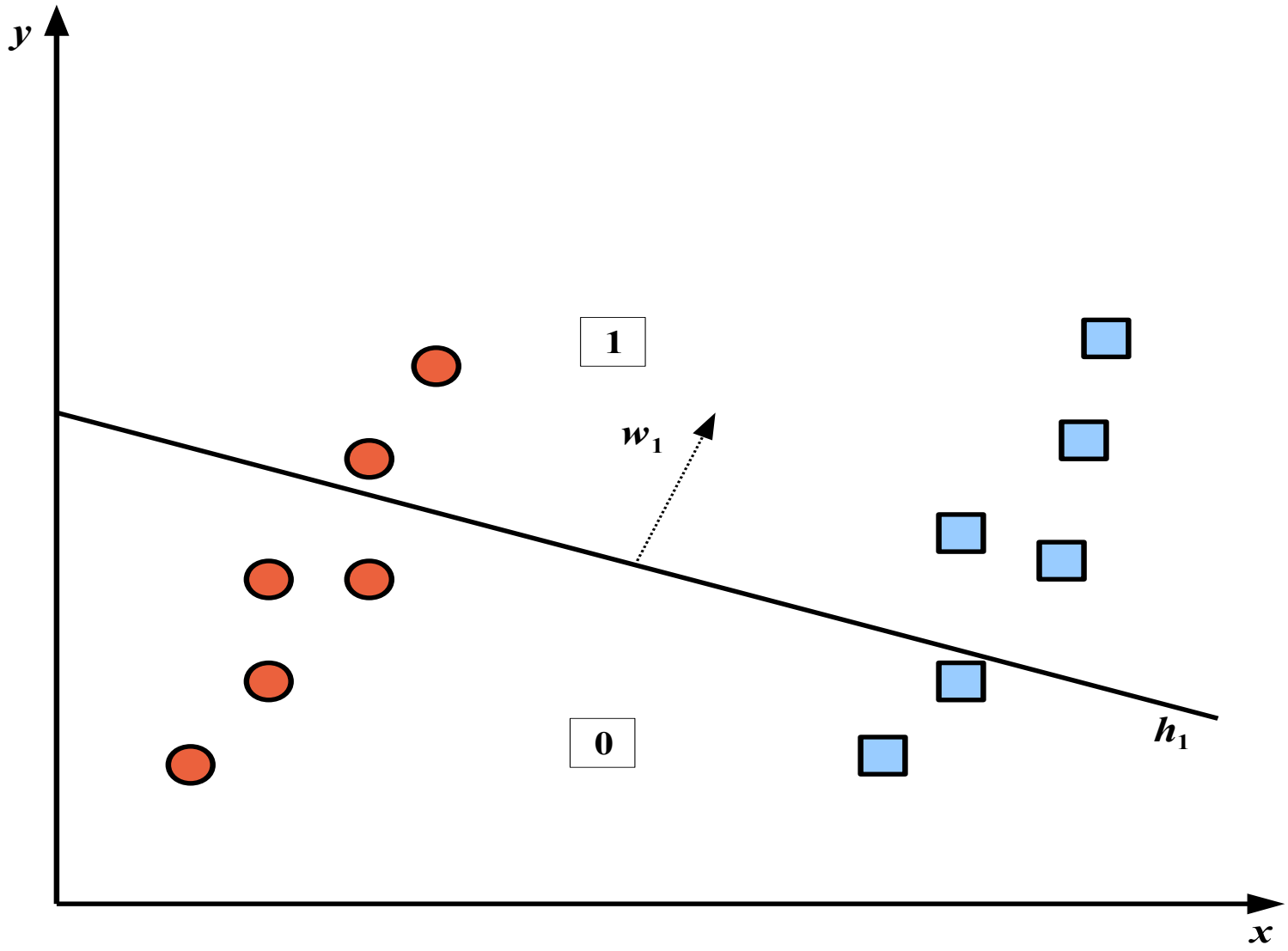


End of Part 1 of Talk

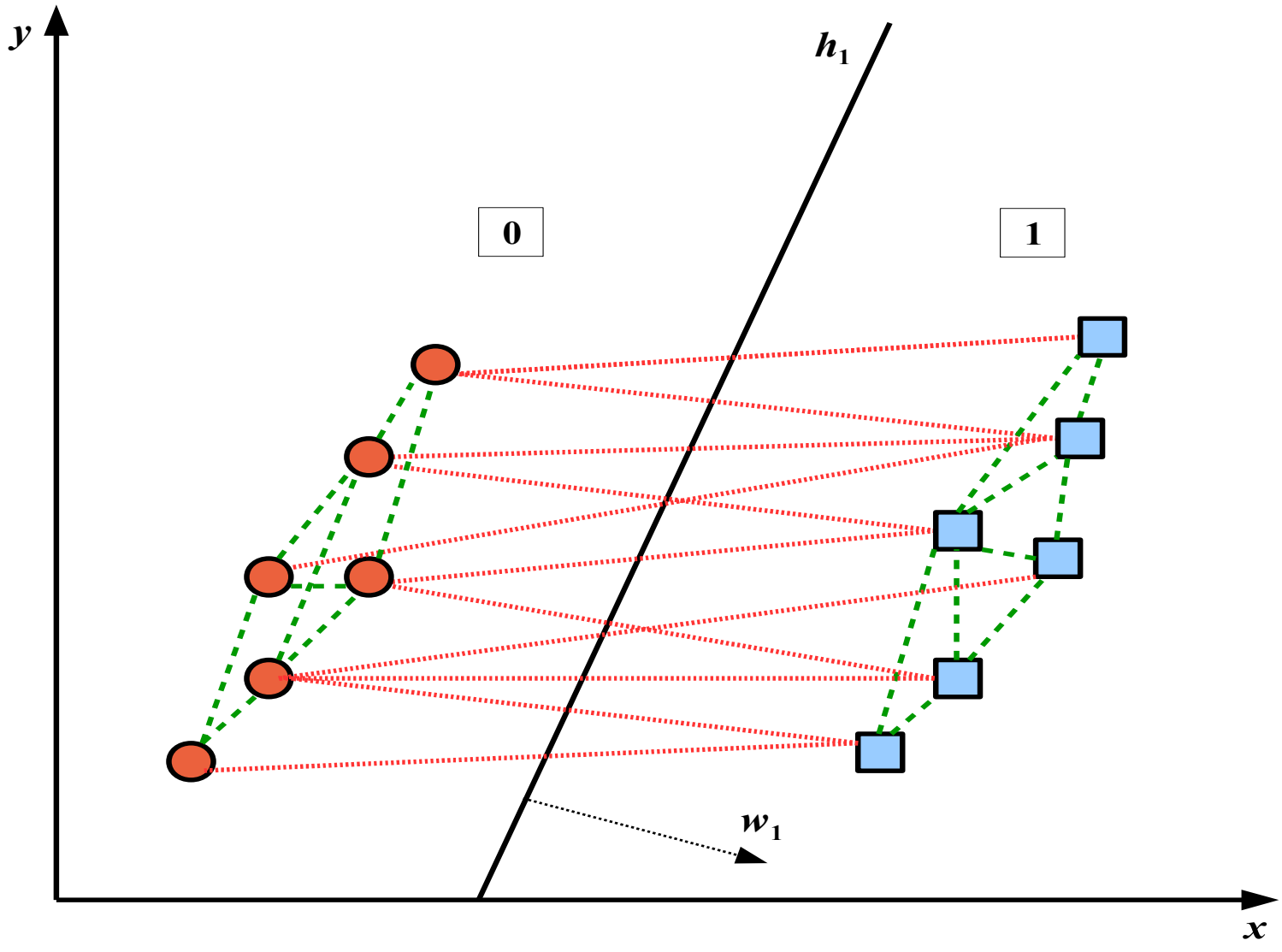
- Takeaway messages:
  - $\mathbf{R}_1$: Optimising 1 threshold is much better than placing the threshold at zero.
  - $\mathbf{R}_2$: ≥ 2 thresholds beneficial for eigendecomposition-based projections.
  - $\mathbf{R}_3$: The benefits of threshold learning generalise to other projection functions.

Learning to Hash for Large Scale Image Retrieval

Why Learn the Hashing Hypersurfaces?



Why Learn the Hashing Hypersurfaces?



Previous work

- **Data-independent:** Locality Sensitive Hashing (LSH) [13]
- **Data-dependent (unsupervised):** Anchor Graph Hashing (AGH) [11], Spectral Hashing (SH) [9]
- **Data-dependent (supervised):** Self Taught Hashing (STH) [12], Supervised Hashing with Kernels (KSH) [5], ITQ + CCA [6], Binary Reconstructive Embedding (BRE) [10]

[5] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, Shih-Fu Chang. **Supervised Hashing with Kernels**. *In CVPR'12*.
[6] Yunchao Gong, Svetlana Lazebnik. **Iterative Quantization: A Procrustean Approach to Learning Binary Codes for Large-scale Image Retrieval**. *In CVPR'11*.
[9] Yair Weiss, Antonio Torralba, Rob Fergus. **Spectral Hashing**. *In NIPS'08*.
[10] Brian Kulis and Trevor Darrell. **Learning to Hash with Binary Reconstructive Embeddings**. *In NIPS'09*.
[11] Wei Liu, Jun Wang, Sanjiv Kumar, Shih-Fu Chang. **Hashing with Graphs**. *In ICML'11*.
[12] Dell Zhang, Jun Wang, Deng Cai, Jinsog Lu. **Self-Taught Hashing for Fast Similarity Search**. *In SIGIR'10*.
[13] Piotr Indyk and Rajeev Motwani. **Approximate nearest neighbors: Towards removing the curse of dimensionality**. *In STOC'98*.

Previous work

Method	Learnt	Supervised	Scalable	Effectiveness
LSH [13]			✓	Low
SH [9]	✓			Low
STH [12]	✓	✓		Medium
BRE [10]	✓	✓		Medium
ITQ+CCA [6]	✓	✓		Medium
KSH [5]	✓	✓		High
GRH [3]	✓	✓	✓	**High**

[3] Sean Moran and Victor Lavrenko. **Graph Regularised Hashing**. *In ECIR'15*.

Learning the Hashing Hypersurfaces

- Two step *iterative* hashing model:
  - **Step A:** Graph Regularisation

$$\mathbf{L}_m \leftarrow \text{sgn}\left(\alpha\, \mathbf{S}\mathbf{D}^{-1}\mathbf{L}_{m-1} + (1-\alpha)\mathbf{L}_0\right)$$

  - **Step B:** Data-Space Partitioning

$$\begin{aligned} \text{for } k = 1 \ldots K: \quad & \min \ ||\mathbf{w}_k||^2 + C \sum_{i=1}^{N} \xi_{ik} \\ & \text{s.t.} \quad L_{ik}(\mathbf{w}_k{}^{\intercal}\mathbf{x}_i + b_k) \geq 1 - \xi_{ik} \qquad \text{for } i = 1 \ldots N \end{aligned}$$

- Repeat for a set number of iterations (M)

Learning the Hashing Hypersurfaces

- **Step A:** Graph Regularisation [14]

$$\mathbf{L}_m \leftarrow \text{sgn}\left(\alpha\ \mathbf{S}\mathbf{D}^{-1}\mathbf{L}_{m-1} + (1-\alpha)\mathbf{L}_0\right)$$

- **S**: Affinity (adjacency) matrix
- **D**: Diagonal degree matrix
- **L**: Binary bits at specified iteration
- α: Interpolation parameter ($0 \leq \alpha \leq 1$)

[14] Fernando Diaz. **Regularizing query-based retrieval scores**. *In IR'07*.

Learning the Hashing Hypersurfaces

- **Step A:** Graph Regularisation

$$\mathbf{L}_m \leftarrow \text{sgn}\left(\alpha\ \mathbf{S}\mathbf{D}^{-1}\mathbf{L}_{m-1} + (1-\alpha)\mathbf{L}_0\right)$$

- **S**: Affinity (adjacency) matrix
- **D**: Diagonal degree matrix
- **L**: Binary bits at specified iteration
- α: Interpolation parameter ($0 \leq \alpha \leq 1$)

Learning the Hashing Hypersurfaces

- **Step A:** Graph Regularisation

$$\mathbf{L}_m \leftarrow \text{sgn}\left(\alpha\ \mathbf{S}\mathbf{D}^{-1}\mathbf{L}_{m-1} + (1-\alpha)\mathbf{L}_0\right)$$

- **S**: Affinity (adjacency) matrix
- **D**: Diagonal degree matrix
- **L**: Binary bits at specified iteration
- α: Interpolation parameter ($0 \leq \alpha \leq 1$)

Learning the Hashing Hypersurfaces

- **Step A:** Graph Regularisation

$$\mathbf{L}_m \leftarrow \text{sgn}\left(\alpha\ \mathbf{S}\mathbf{D}^{-1}\mathbf{L}_{m-1} + (1-\alpha)\mathbf{L}_0\right)$$

- **S**: Affinity (adjacency) matrix
- **D**: Diagonal degree matrix
- **L**: Binary bits at specified iteration
- α: Interpolation parameter ($0 \leq \alpha \leq 1$)

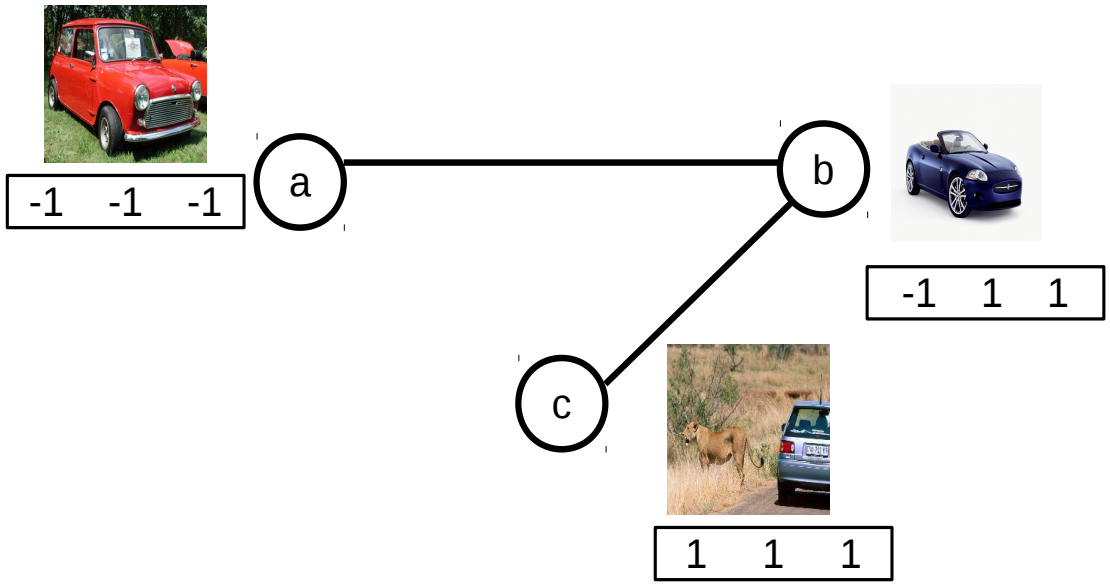
Learning the Hashing Hypersurfaces

- **Step A:** Graph Regularisation

$$\mathbf{L}_m \leftarrow \text{sgn}\left(\alpha\ \mathbf{S}\mathbf{D}^{-1}\mathbf{L}_{m-1} + (1-\alpha)\mathbf{L}_0\right)$$

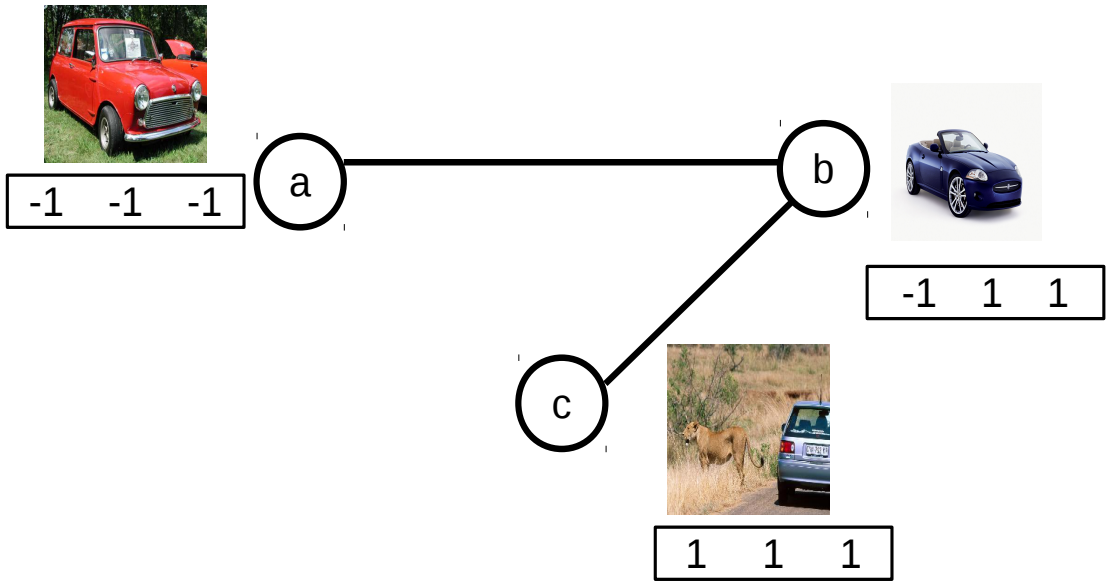
- **S**: Affinity (adjacency) matrix
- **D**: Diagonal degree matrix
- **L**: Binary bits at specified iteration
- α: Interpolation parameter ($0 \leq \alpha \leq 1$)

Learning the Hashing Hypersurfaces



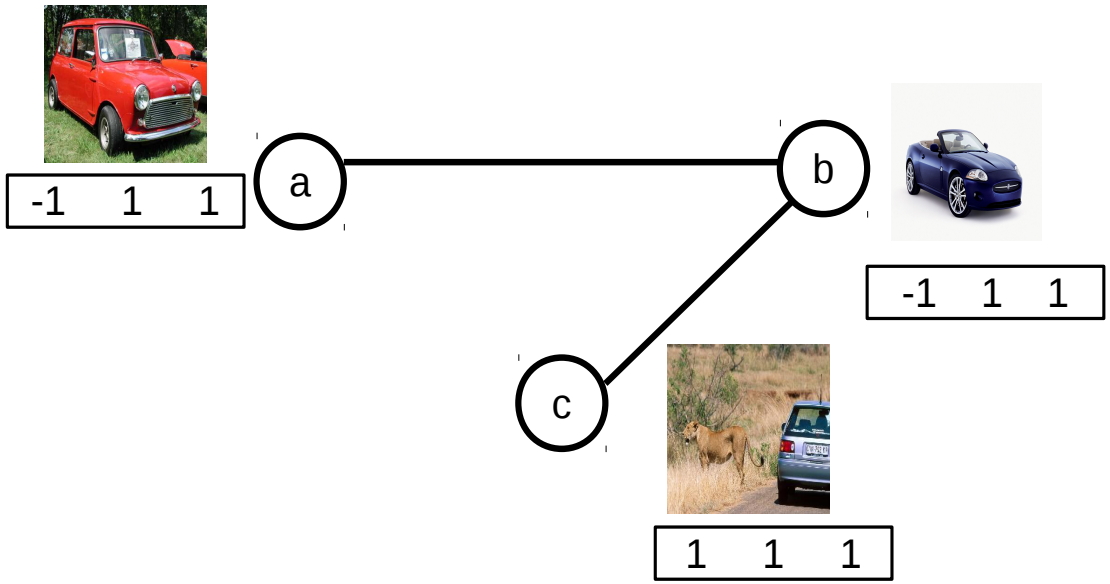
$$\mathbf{S} \quad \begin{matrix} & a & b & c \\ a & 1 & 1 & 0 \\ b & 1 & 1 & 1 \\ c & 0 & 1 & 1 \end{matrix} \qquad \mathbf{D}^{-1} \quad \begin{matrix} & a & b & c \\ a & 0.5 & 0 & 0 \\ b & 0 & 0.33 & 0 \\ c & 0 & 0 & 0.5 \end{matrix} \qquad \mathbf{L}_0 \quad \begin{matrix} & b_1 & b_2 & b_3 \\ a & -1 & -1 & -1 \\ b & -1 & 1 & 1 \\ c & 1 & 1 & 1 \end{matrix}$$

Learning the Hashing Hypersurfaces



$$\mathbf{L}_1 = sgn\left\{ \begin{pmatrix} -1 & 0 & 0 \\ -0.33 & 0.33 & 0.33 \\ 0 & 1 & 1 \end{pmatrix} \right\}$$

Learning the Hashing Hypersurfaces



$$\mathbf{L}_1 = \begin{array}{c} \\ a \\ b \\ c \end{array} \begin{array}{c} \begin{array}{ccc} b_1 & b_2 & b_3 \end{array} \\ \begin{pmatrix} -1 & 1 & 1 \\ -1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \end{array}$$

Learning the Hashing Hypersurfaces

- **Step B:** Data-Space Partitioning

$$\text{for } k = 1 \ldots K: \quad \min \; ||\mathbf{w}_k||^2 + C \sum_{i=1}^{N} \xi_{ik}$$
$$\text{s.t.} \quad L_{ik}(\mathbf{w}_k{}^{\mathsf{T}} \mathbf{x}_i + t_k) \geq 1 - \xi_{ik} \qquad \text{for } i = 1 \ldots N$$

- $\mathbf{w}_k$: Hyperplane normal k
- t_k: bias of hyperplane k
- $\mathbf{x}_i$: data-point i
- L_{ik}: bit k of data-point i

ξ_{ik}: slack variable ik

K: # bits

N: # data-points

Learning the Hashing Hypersurfaces

- **Step B:** Data-Space Partitioning

$$\text{for } k = 1 \ldots K: \quad \min \; ||\mathbf{w}_k||^2 + C \sum_{i=1}^{N} \xi_{ik}$$
$$\text{s.t.} \quad L_{ik}(\mathbf{w}_k{}^{\mathsf{T}} \mathbf{x}_i + t_k) \geq 1 - \xi_{ik} \qquad \text{for } i = 1 \ldots N$$

- $\mathbf{w}_k$: Hyperplane normal k
- t_k: bias of hyperplane k
- $\mathbf{x}_i$: data-point i
- L_{ik}: bit k of data-point i

ξ_{ik}: slack variable ik

K: # bits

N: # data-points

Learning the Hashing Hypersurfaces

- **Step B:** Data-Space Partitioning

$$\text{for } k = 1 \ldots K: \quad \min \; ||\mathbf{w}_k||^2 + C \sum_{i=1}^{N} \xi_{ik}$$
$$\text{s.t.} \quad L_{ik}(\mathbf{w}_k{}^{\mathsf{T}} \mathbf{x}_i + t_k) \geq 1 - \xi_{ik} \qquad \text{for } i = 1 \ldots N$$

- $\mathbf{w}_k$: Hyperplane normal k
- t_k: bias of hyperplane k
- $\mathbf{x}_i$: data-point i
- L_{ik}: bit k of data-point i

ξ_{ik}: slack variable ik

K: # bits

N: # data-points

Learning the Hashing Hypersurfaces

- **Step B:** Data-Space Partitioning

$$\text{for } k = 1 \ldots K: \quad \min \; ||\mathbf{w}_k||^2 + C \sum_{i=1}^{N} \xi_{ik}$$
$$\text{s.t.} \quad L_{ik}(\mathbf{w}_k{}^{\mathsf{T}} \mathbf{x}_i + t_k) \geq 1 - \xi_{ik} \qquad \text{for } i = 1 \ldots N$$

- $\mathbf{w}_k$: Hyperplane normal k
- t_k: bias of hyperplane k
- $\mathbf{x}_i$: data-point i
- L_{ik}: bit k of data-point i

ξ_{ik}: slack variable ik

K: # bits

N: # data-points

Learning the Hashing Hypersurfaces

- **Step B:** Data-Space Partitioning

$$\text{for } k = 1 \ldots K: \quad \min \; ||\mathbf{w}_k||^2 + C \sum_{i=1}^{N} \xi_{ik}$$
$$\text{s.t.} \quad L_{ik}(\mathbf{w}_k{}^{\mathsf{T}} \mathbf{x}_i + t_k) \geq 1 - \xi_{ik} \qquad \text{for } i = 1 \ldots N$$

- $\mathbf{w}_k$: Hyperplane normal k
- t_k: bias of hyperplane k
- $\mathbf{x}_i$: data-point i
- L_{ik}: bit k of data-point i

ξ_{ik}: slack variable ik

K: # bits

N: # data-points

Learning the Hashing Hypersurfaces

a

b

d

c

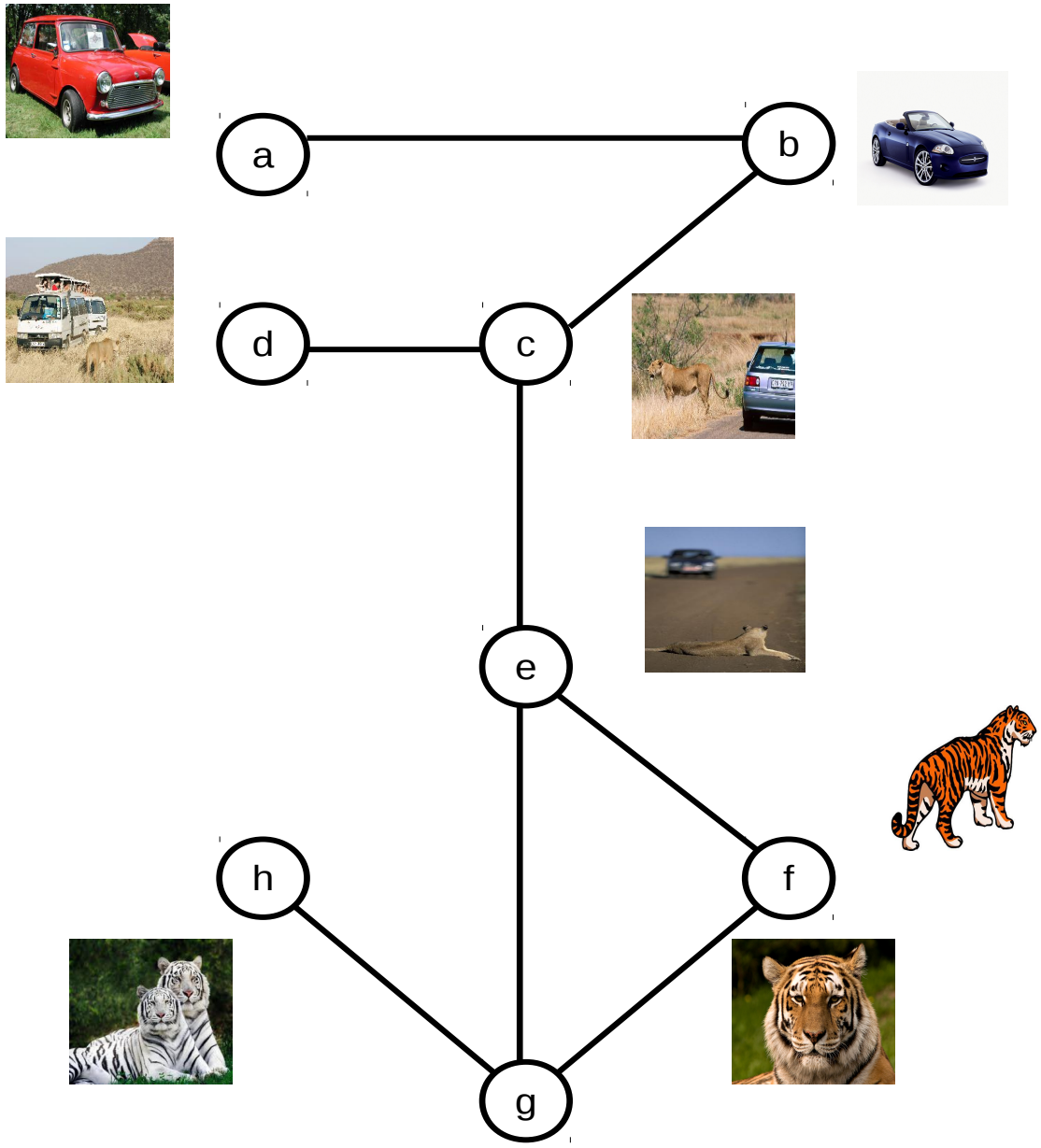
e

h

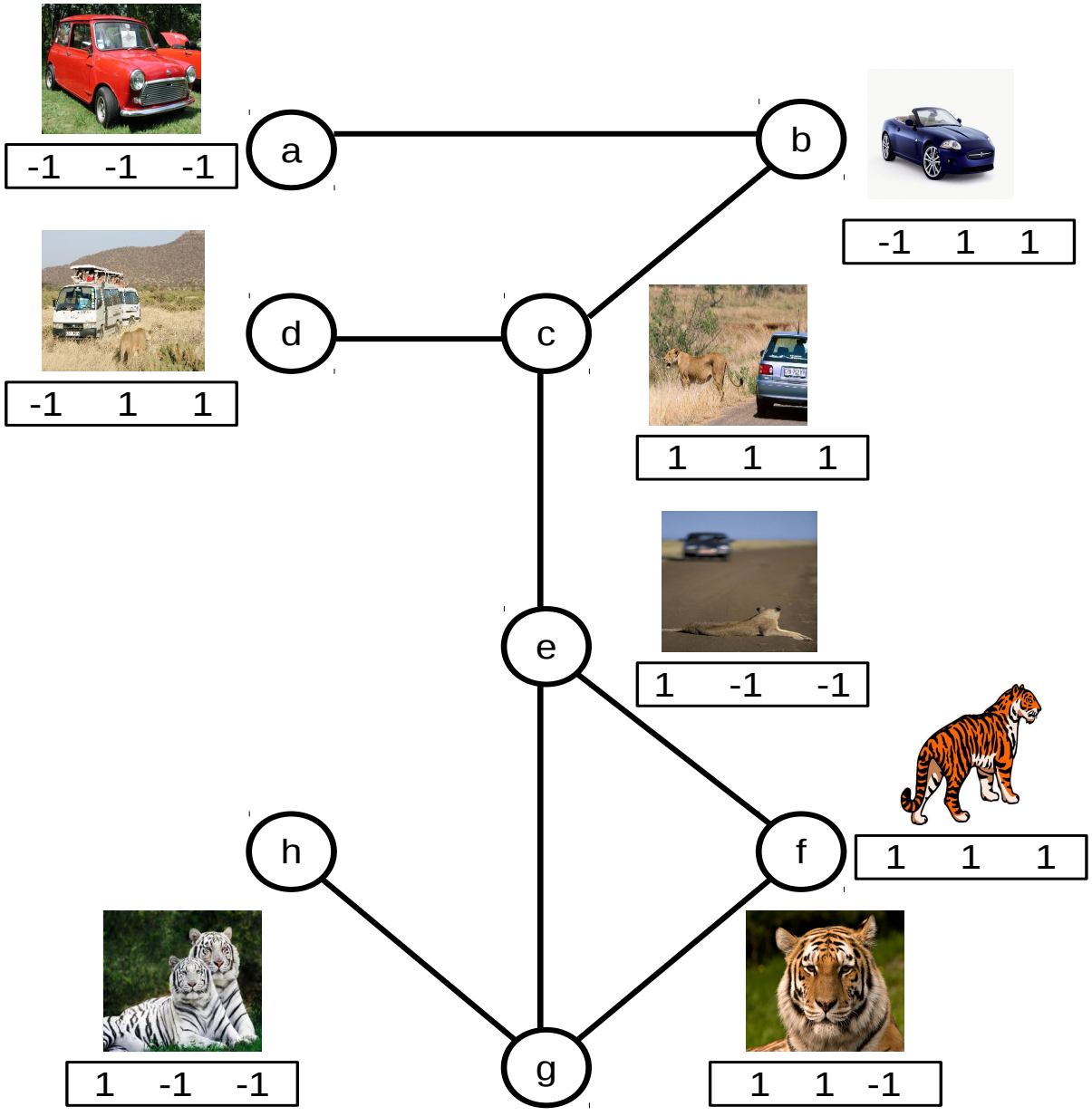
f

g

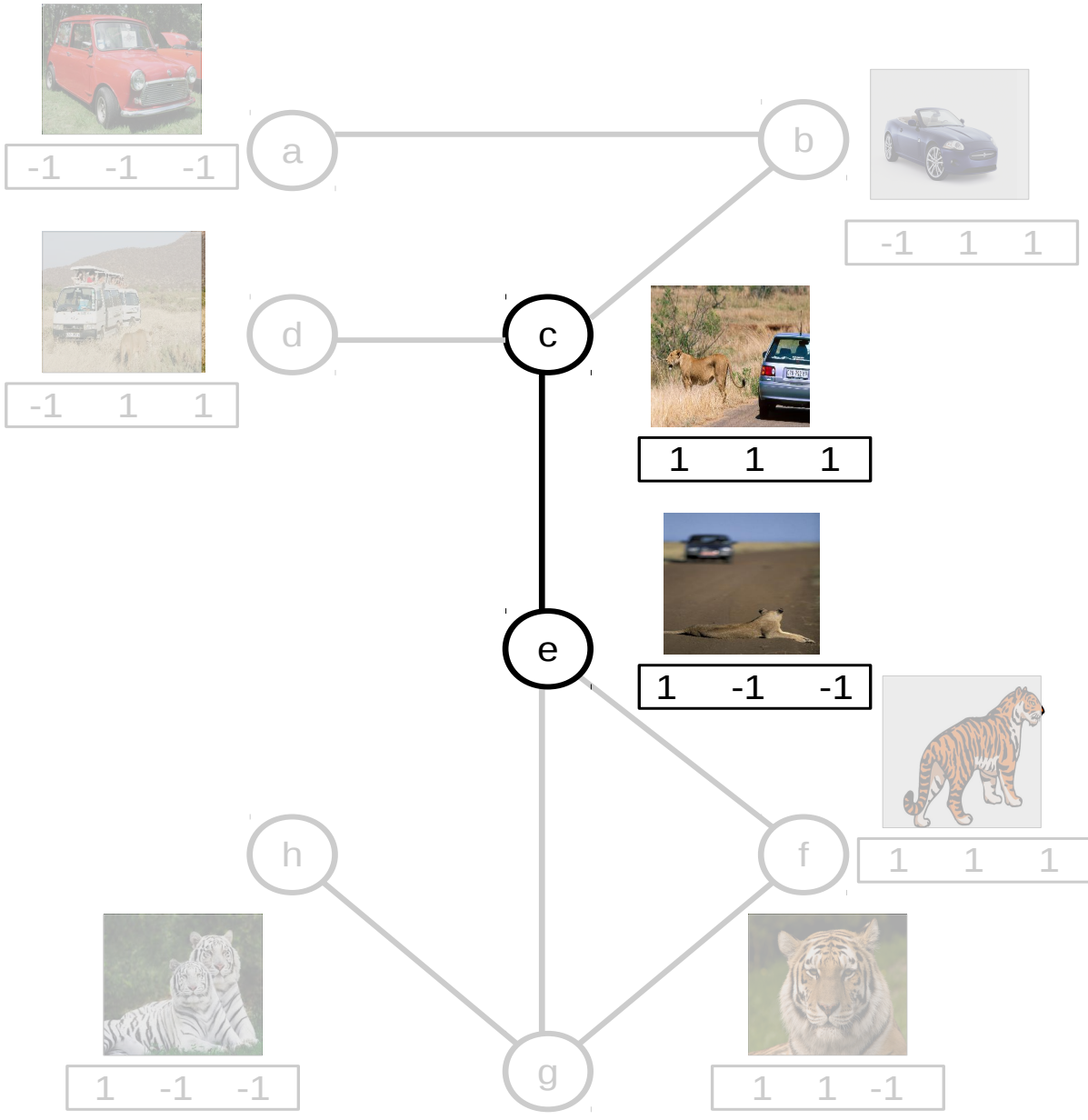
Learning the Hashing Hypersurfaces



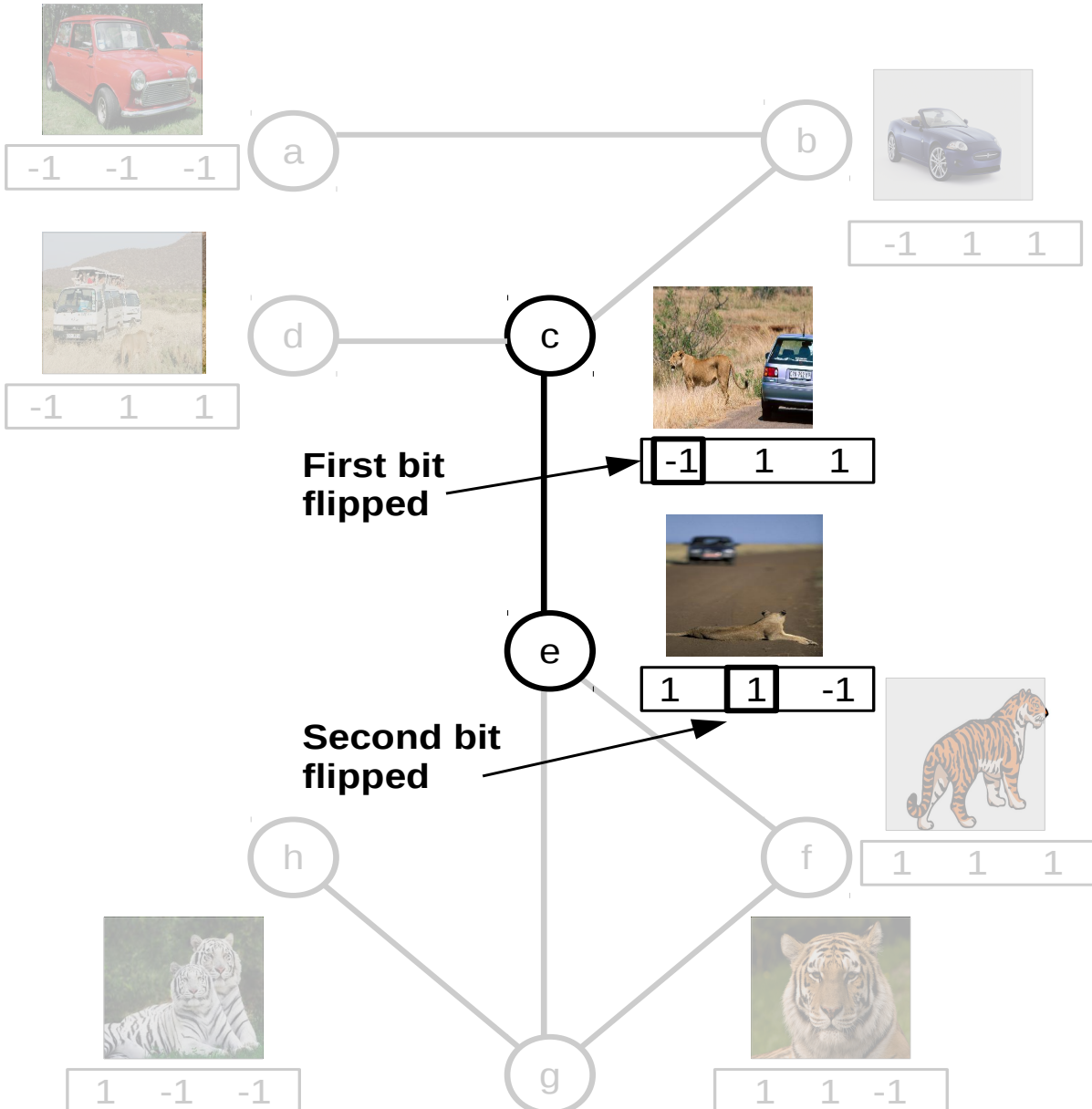
Learning the Hashing Hypersurfaces



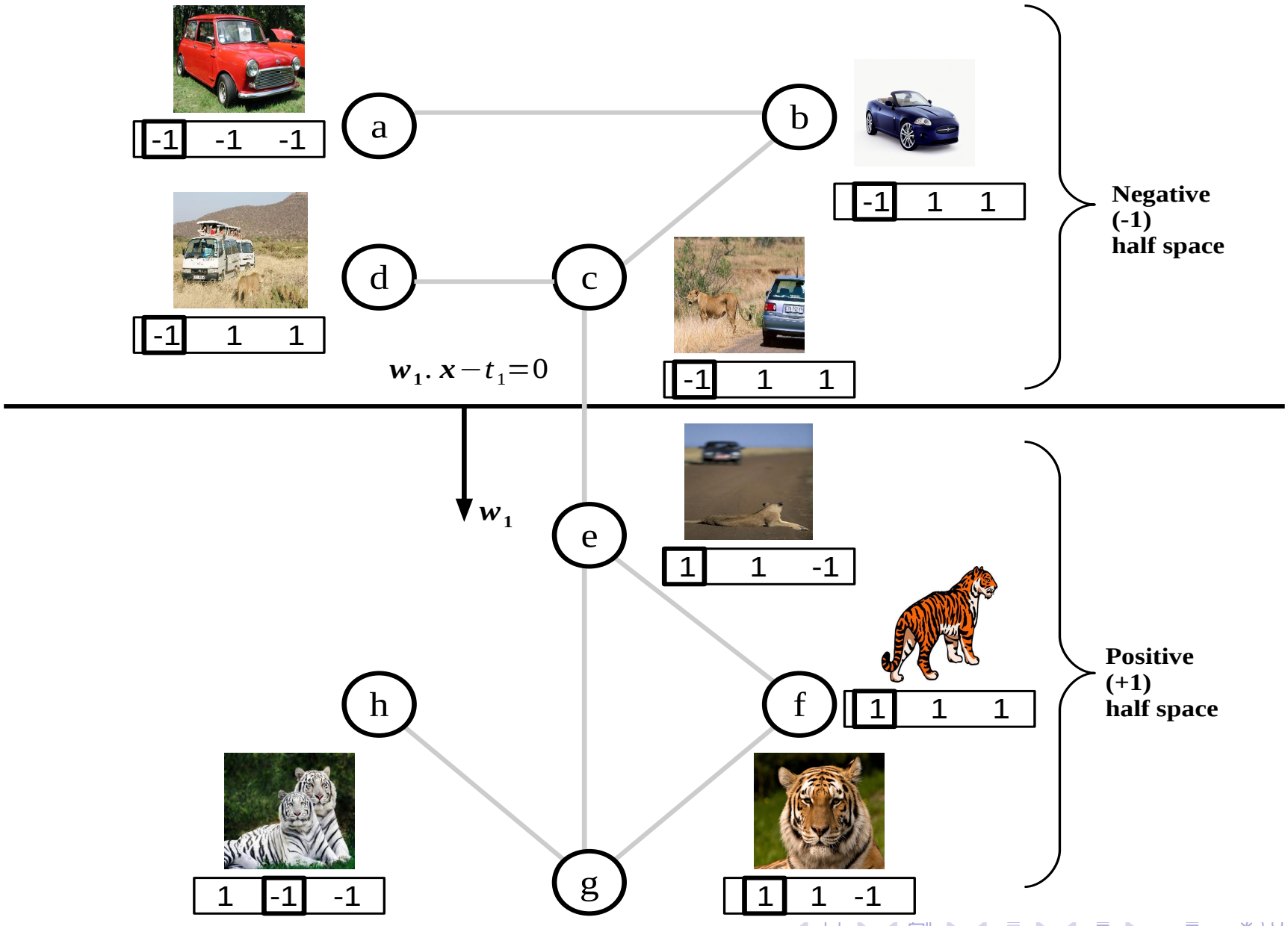
Learning the Hashing Hypersurfaces



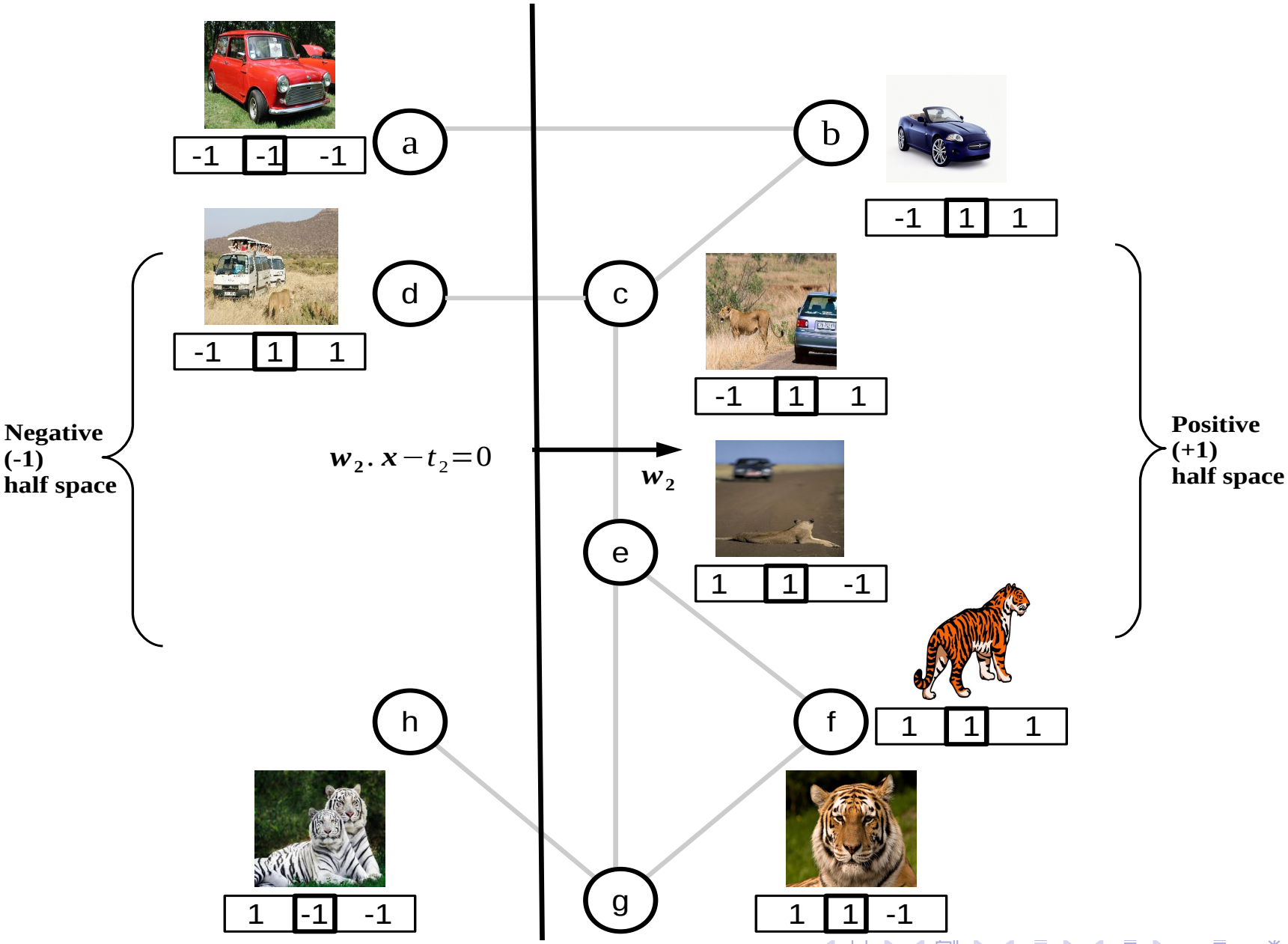
Learning the Hashing Hypersurfaces



Learning the Hashing Hypersurfaces



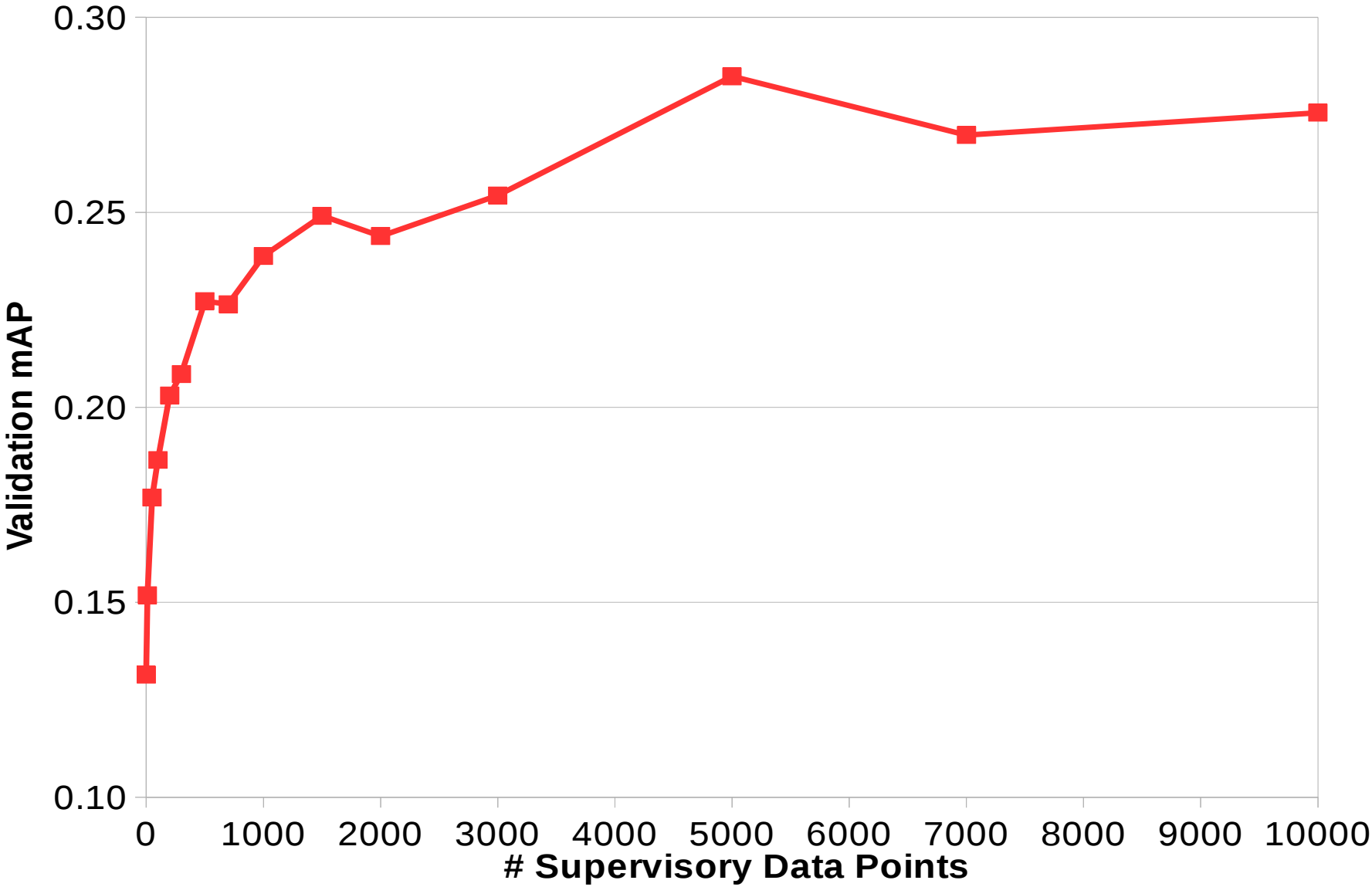
Learning the Hashing Hypersurfaces



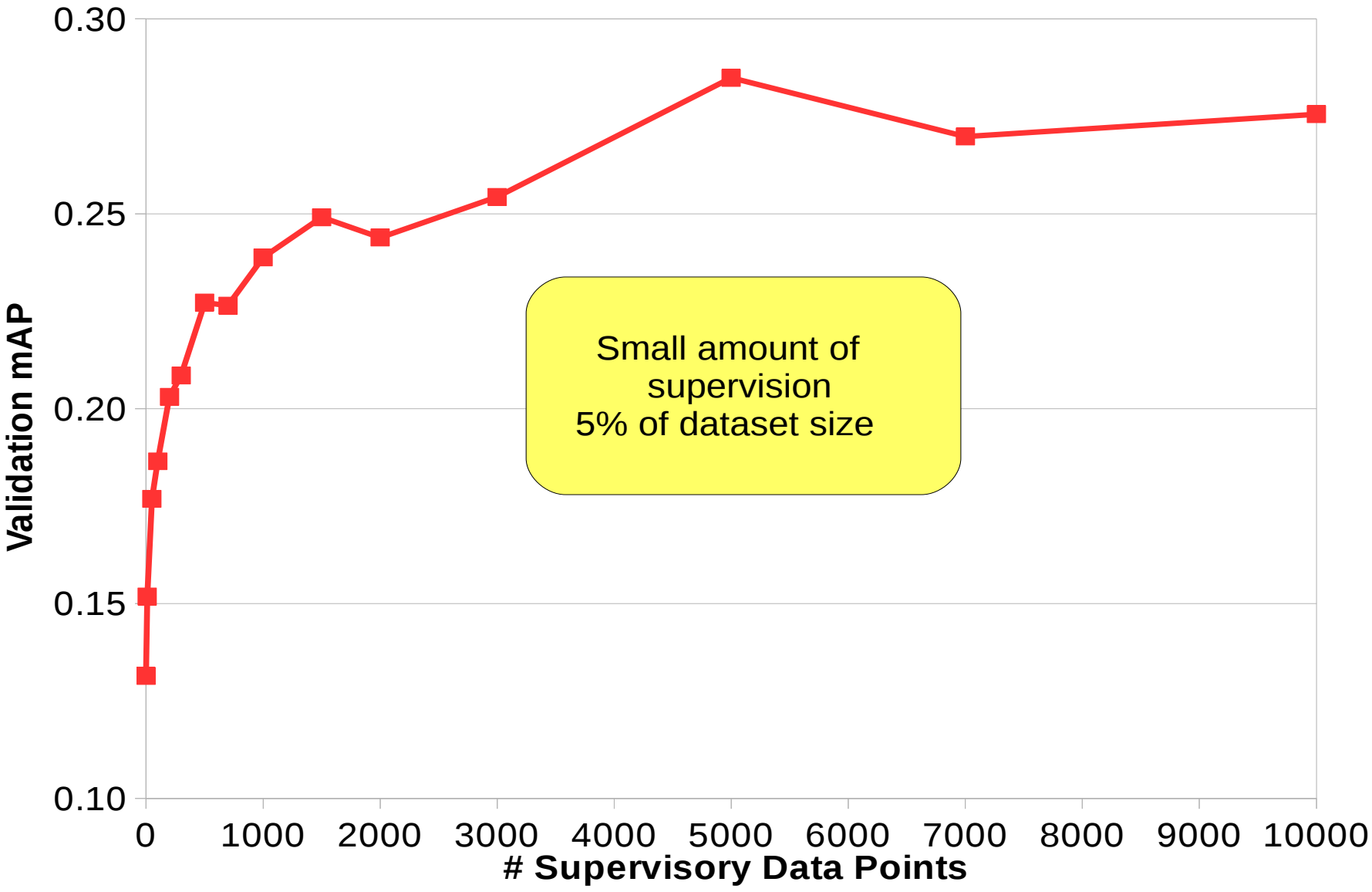
Experimental Results

- **Three** research questions: (more in the thesis):
  - $\mathbf{R}_1$: *Does graph regularisation step help us learn more effective hashcodes?*
  - $\mathbf{R}_2$: *Is it necessary to solve an eigenvalue problem in order to learn effective hashcodes?*
  - $\mathbf{R}_2$: *Do non-linear hypersurfaces provide a more effective bucketing of the space compared to linear hypersurfaces?*

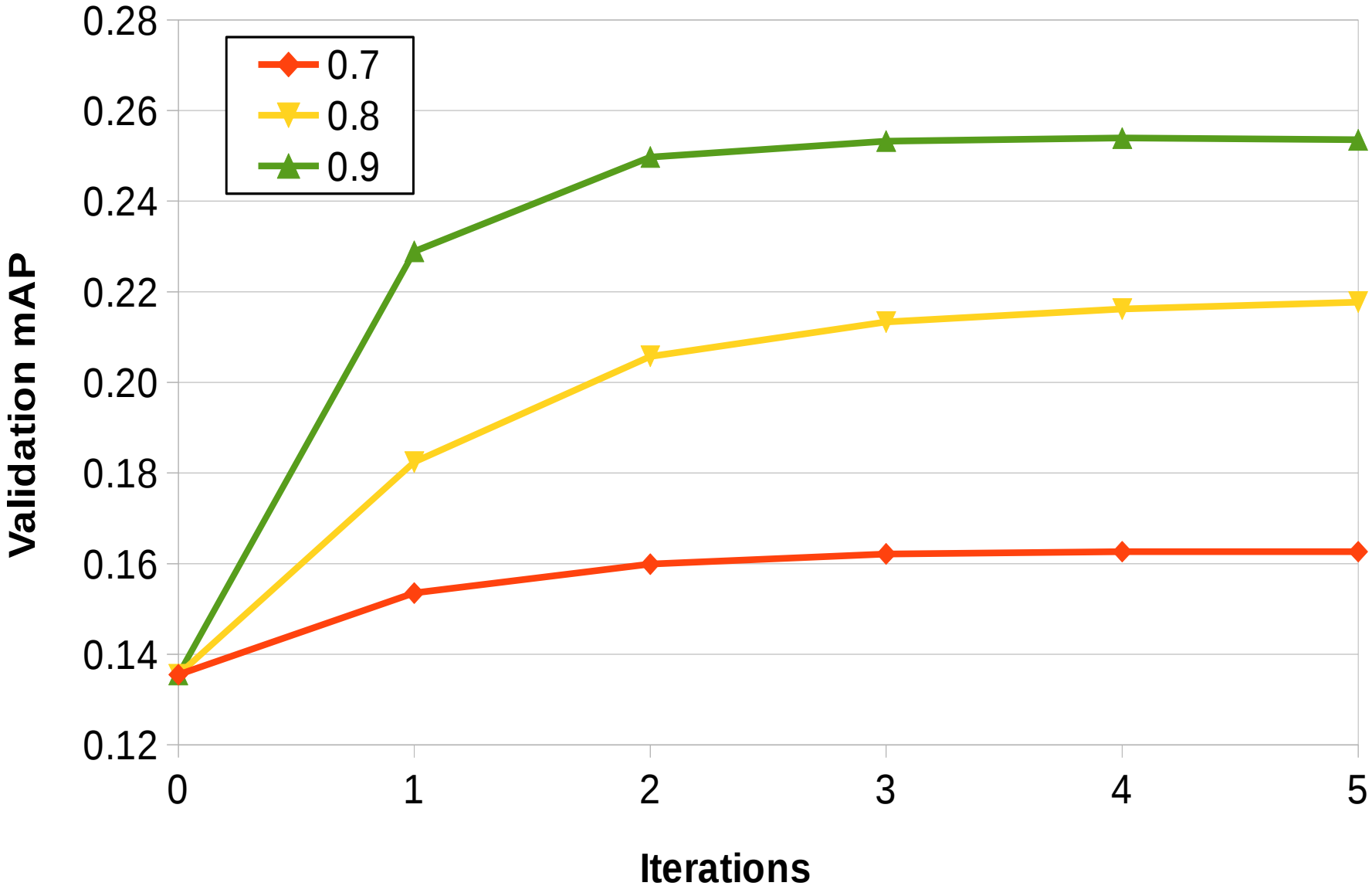
Learning Curve (mAP vs. Amount of Supervision)



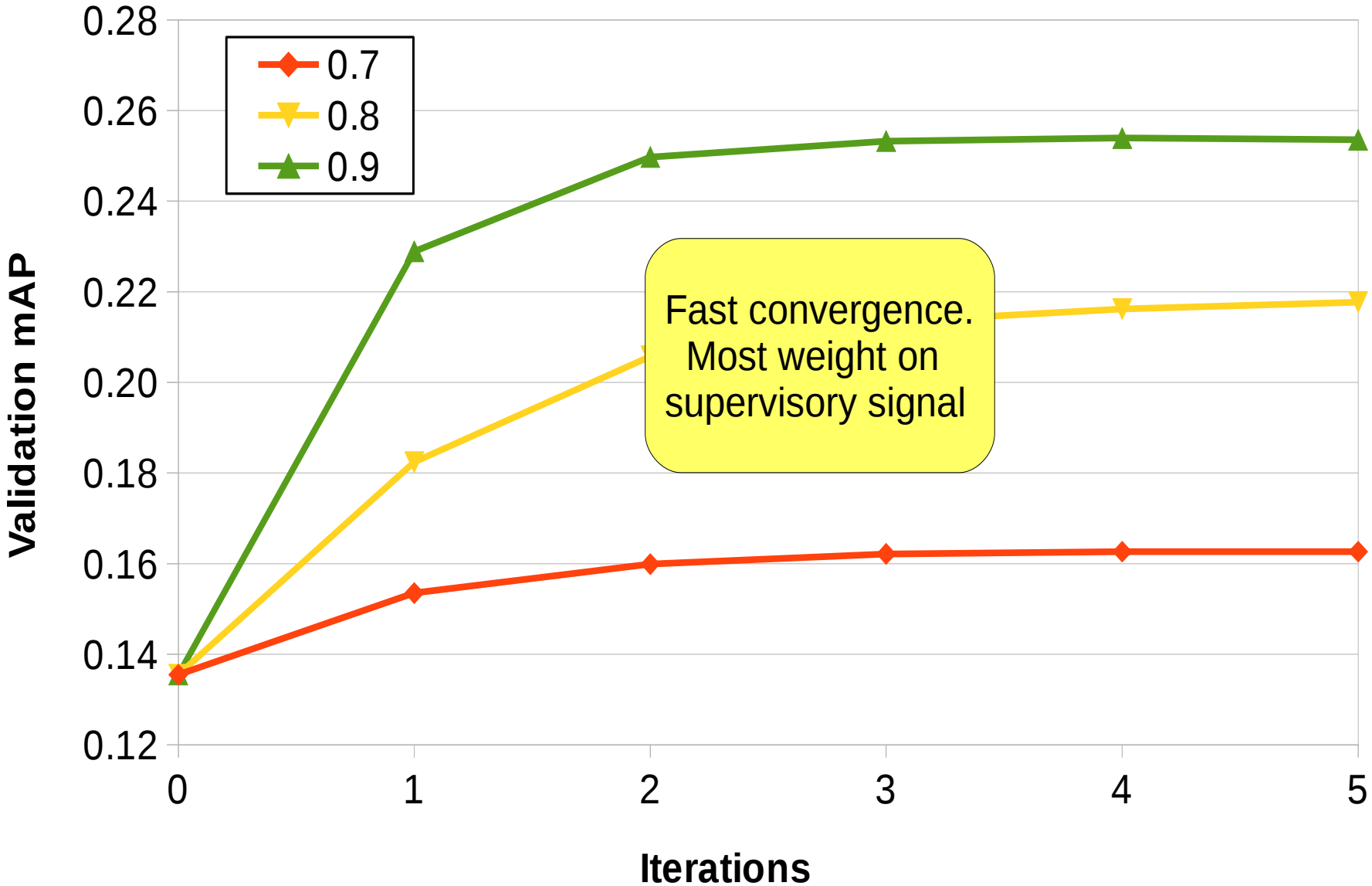
Learning Curve (mAP vs. Amount of Supervision)



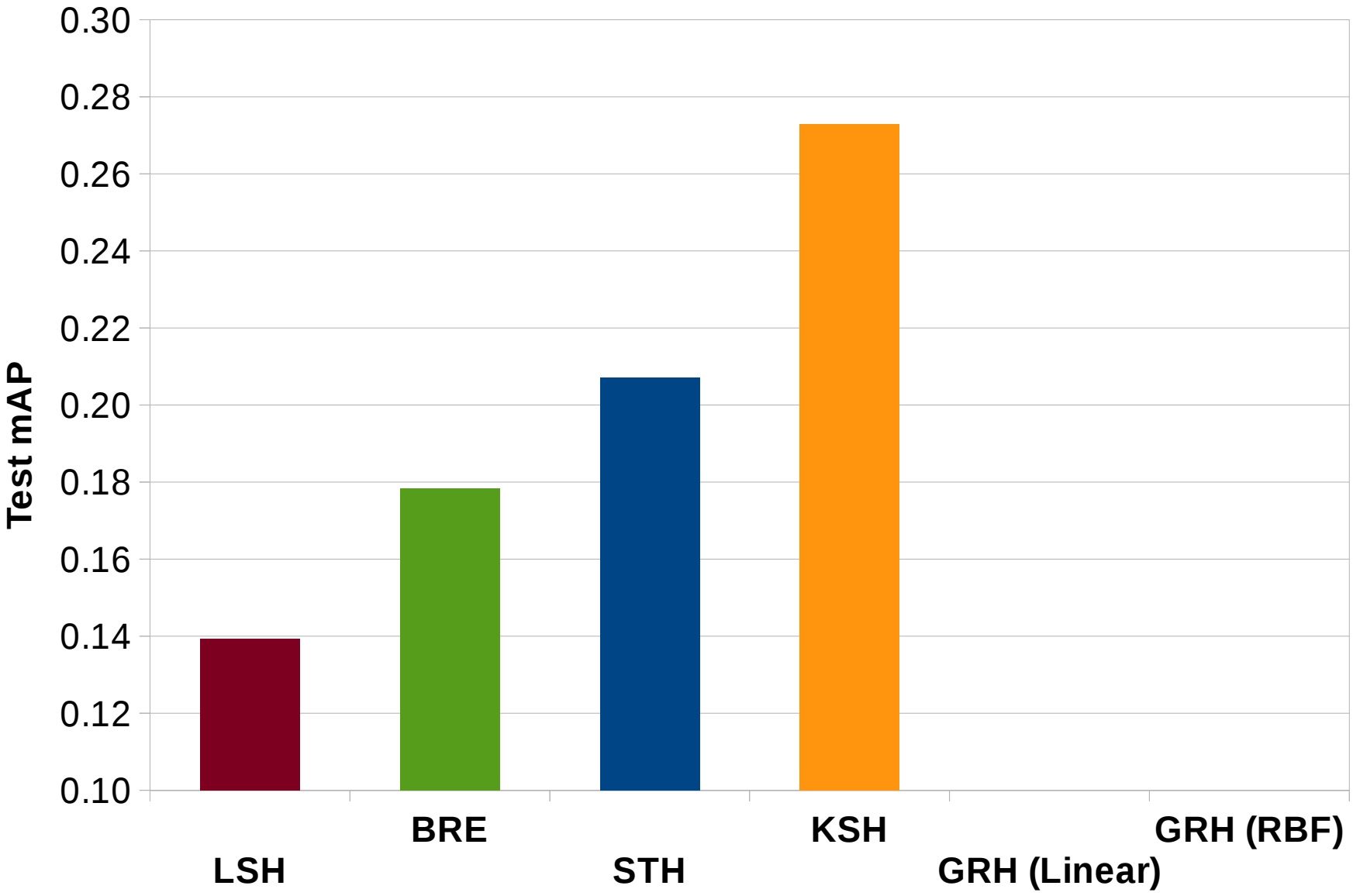
Effect of α and Iterations (CIFAR-10 @ 32 bits)



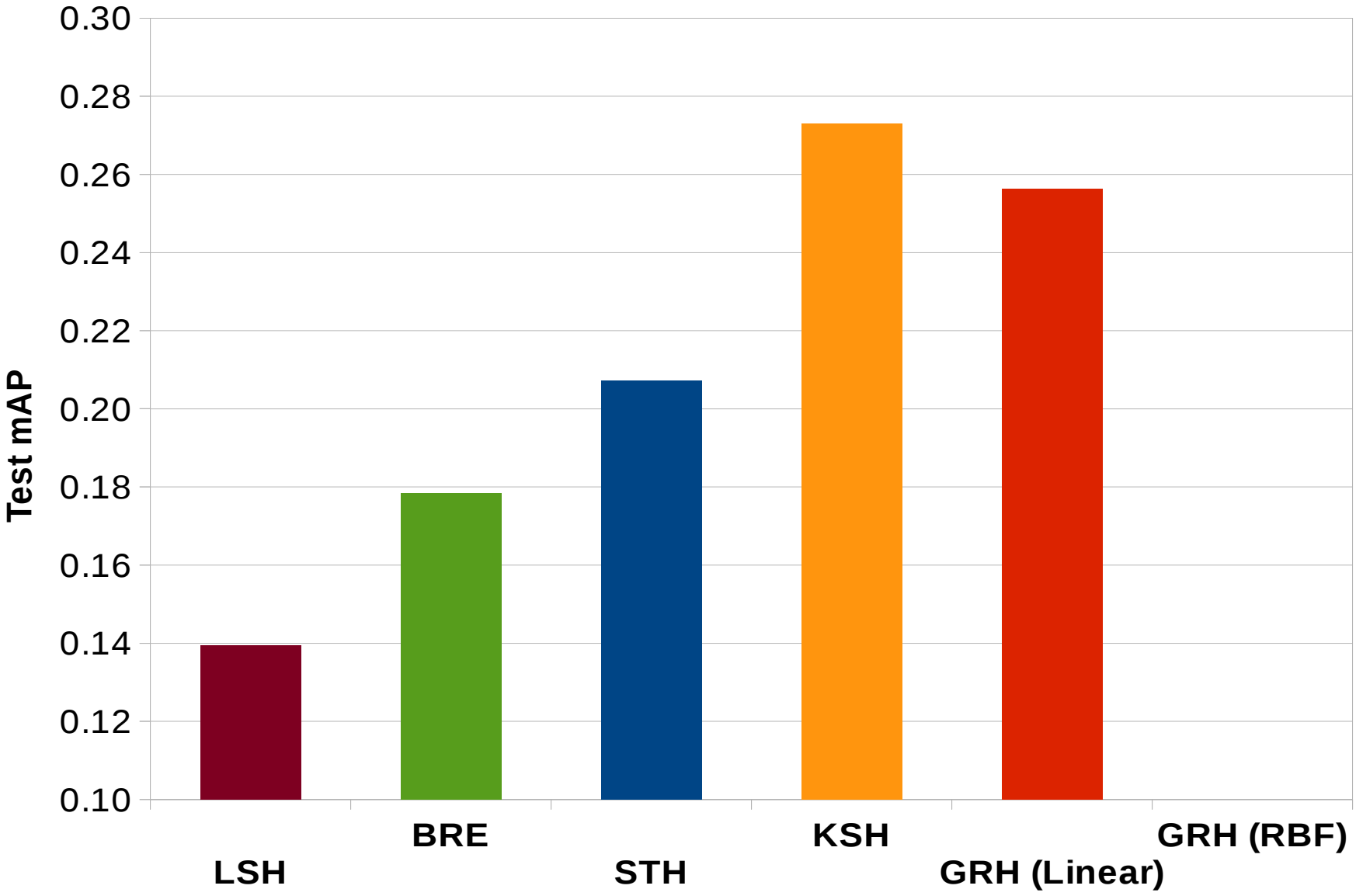
Effect of α and Iterations (CIFAR-10 @ 32 bits)



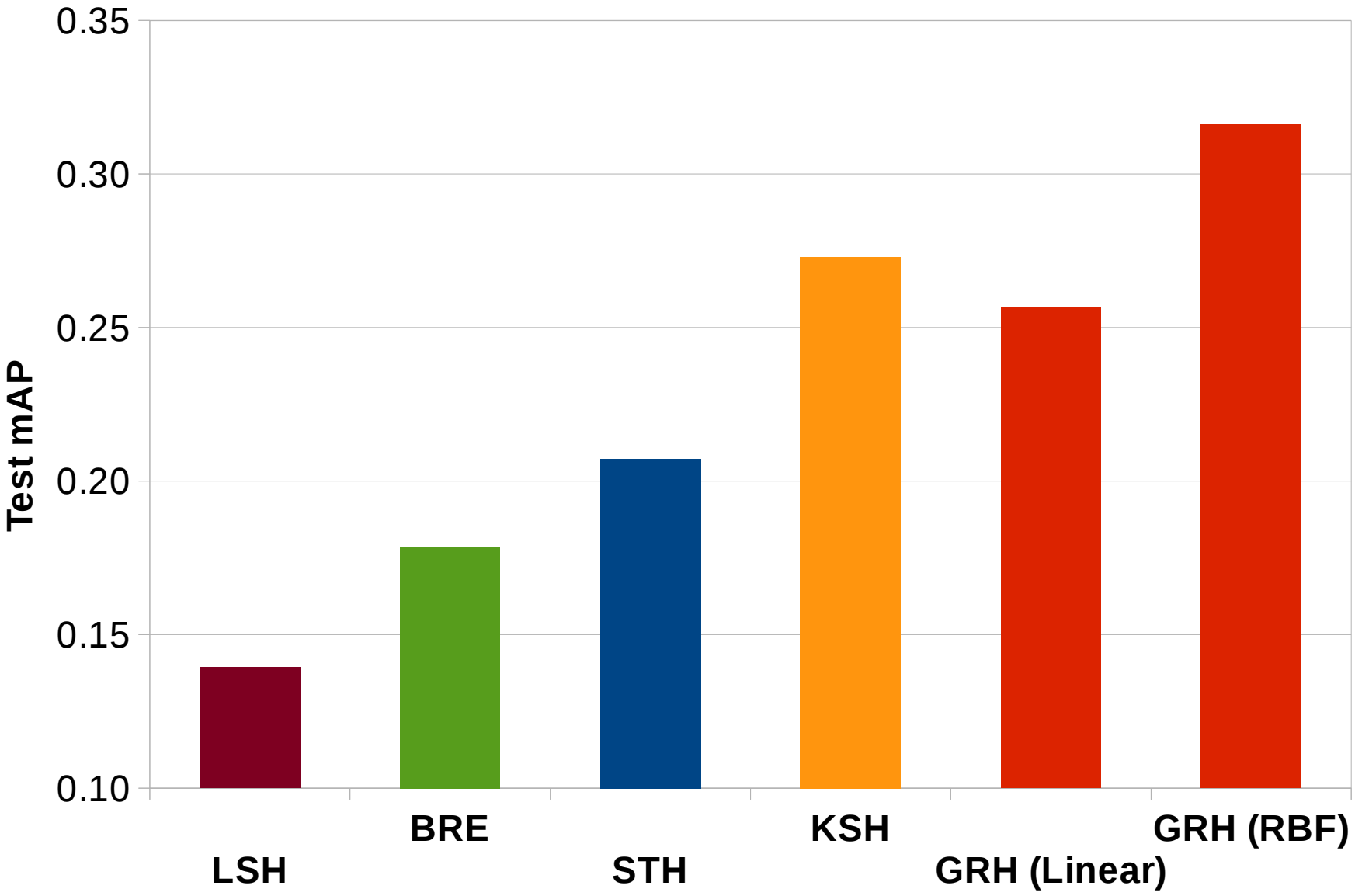
GRH vs Literature (CIFAR-10 @ 32 bits)



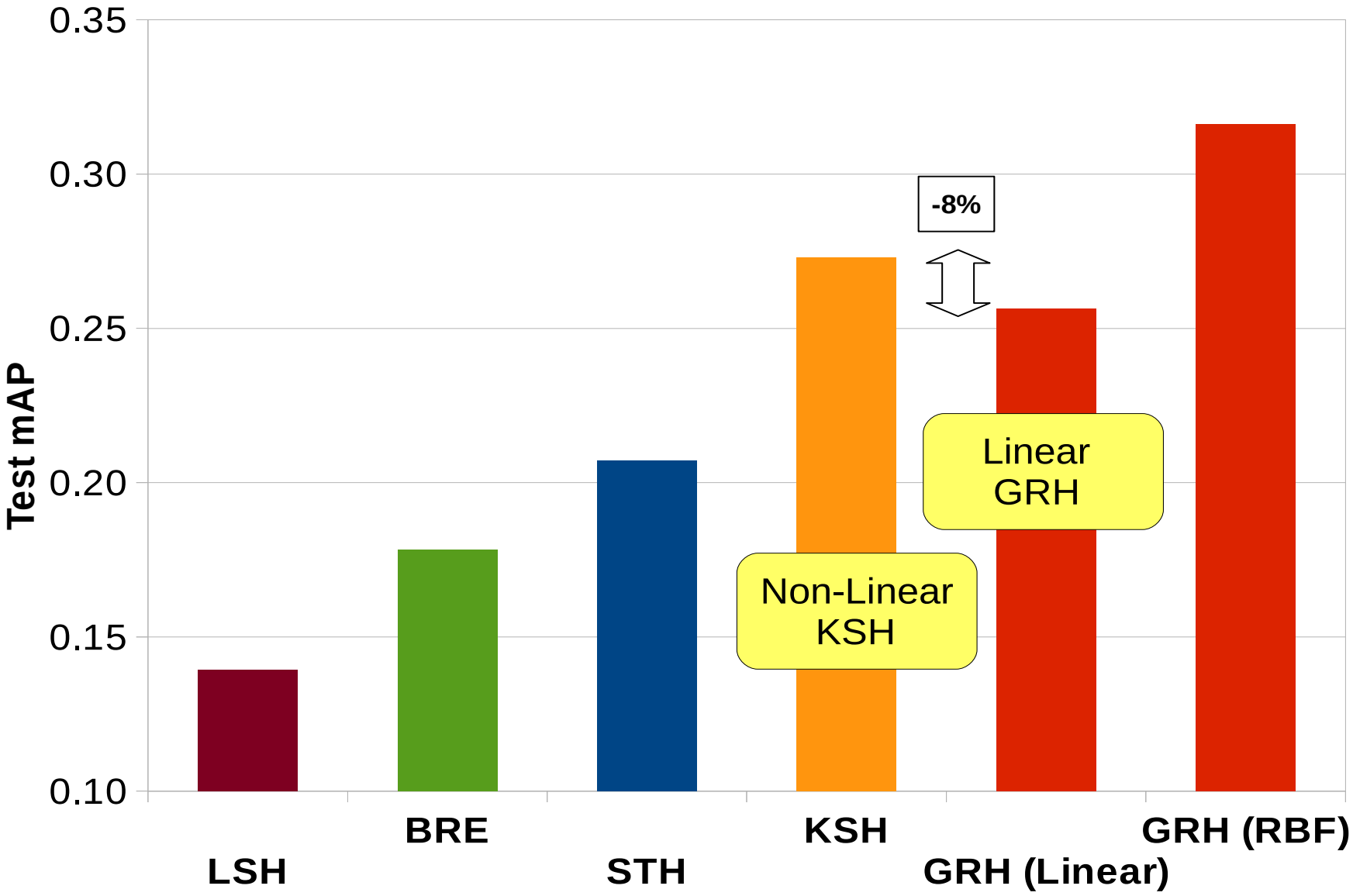
GRH vs Literature (CIFAR-10 @ 32 bits)



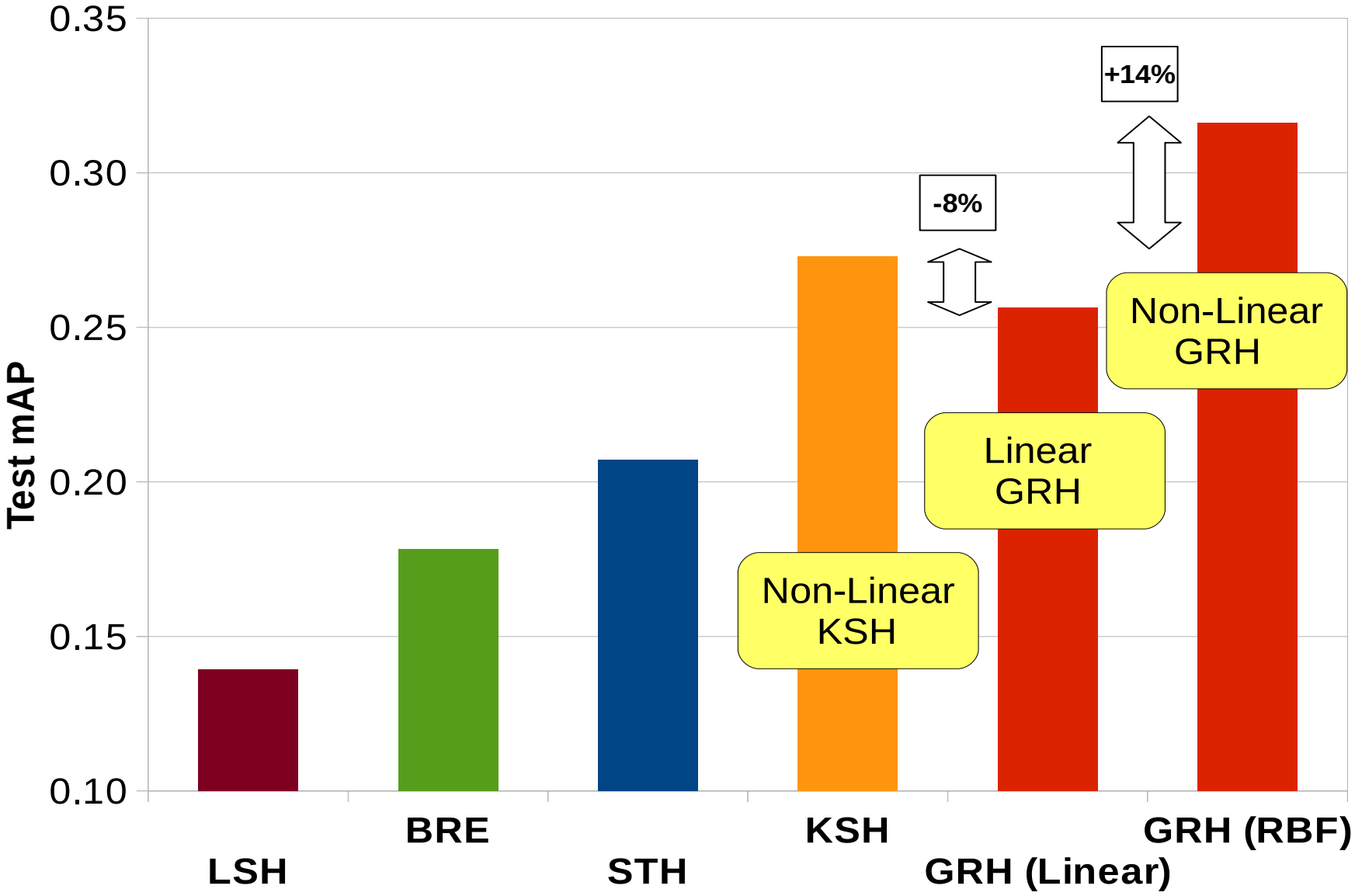
GRH vs Literature (CIFAR-10 @ 32 bits)



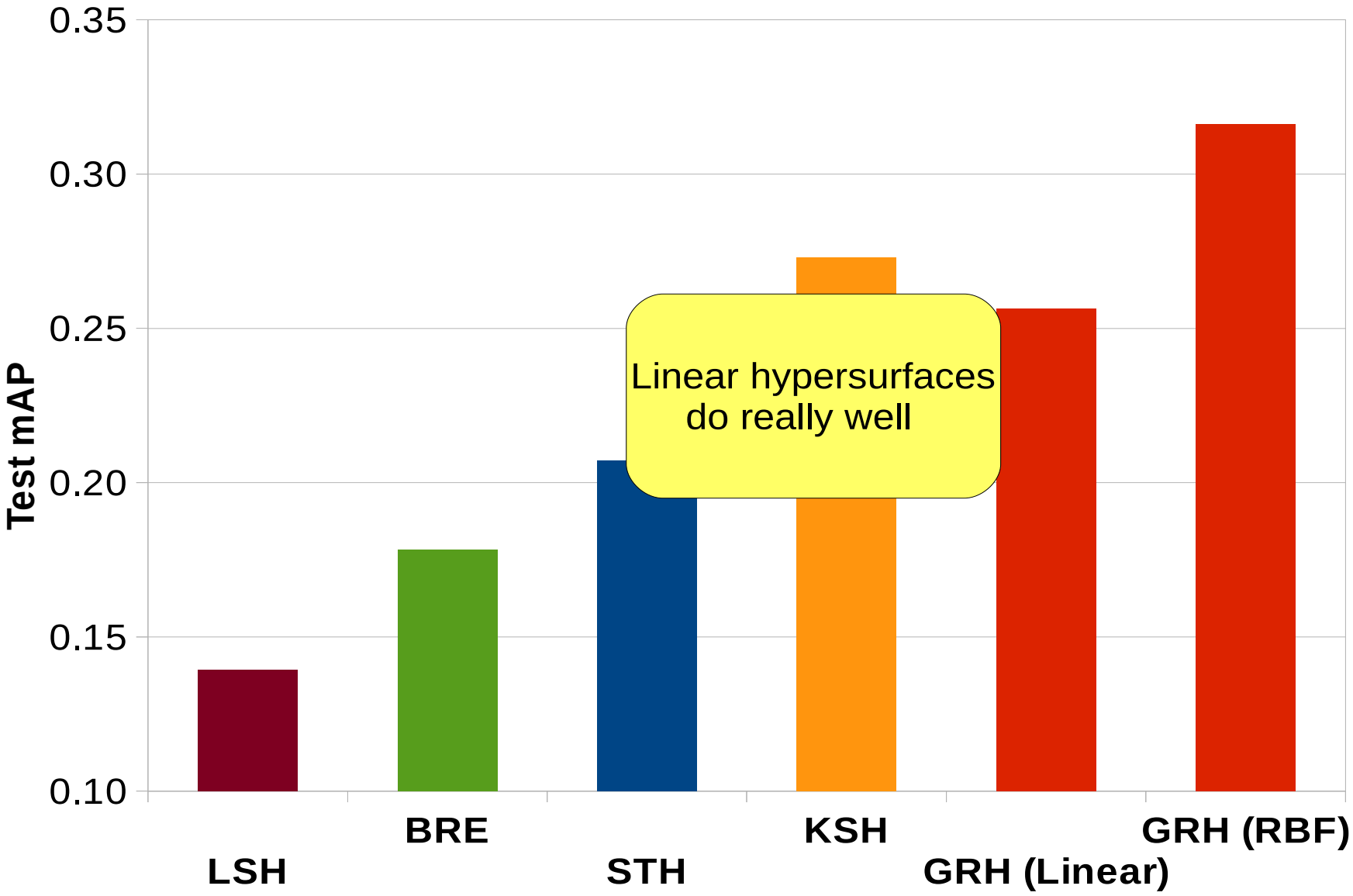
GRH vs Literature (CIFAR-10 @ 32 bits)



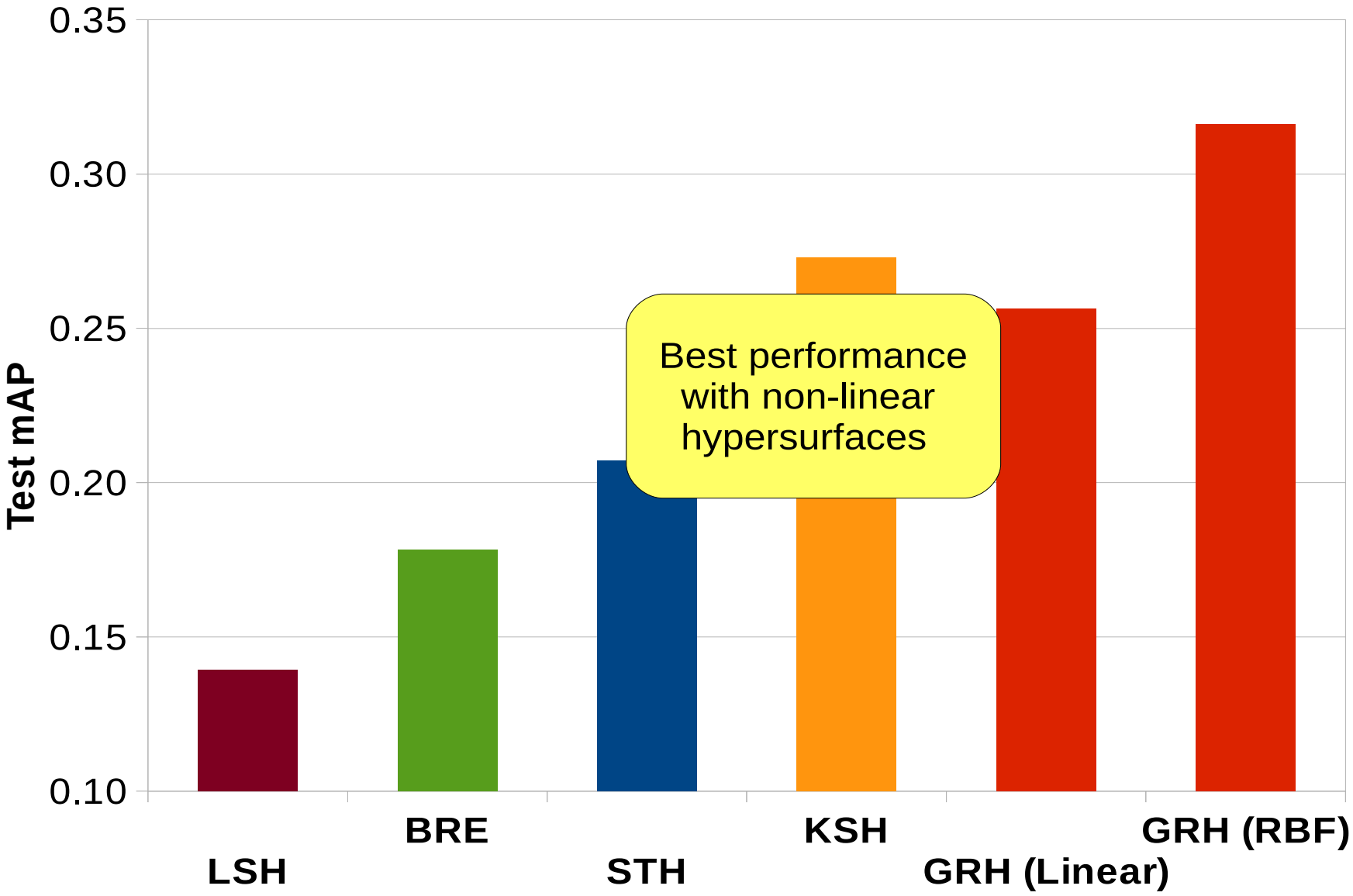
GRH vs Literature (CIFAR-10 @ 32 bits)



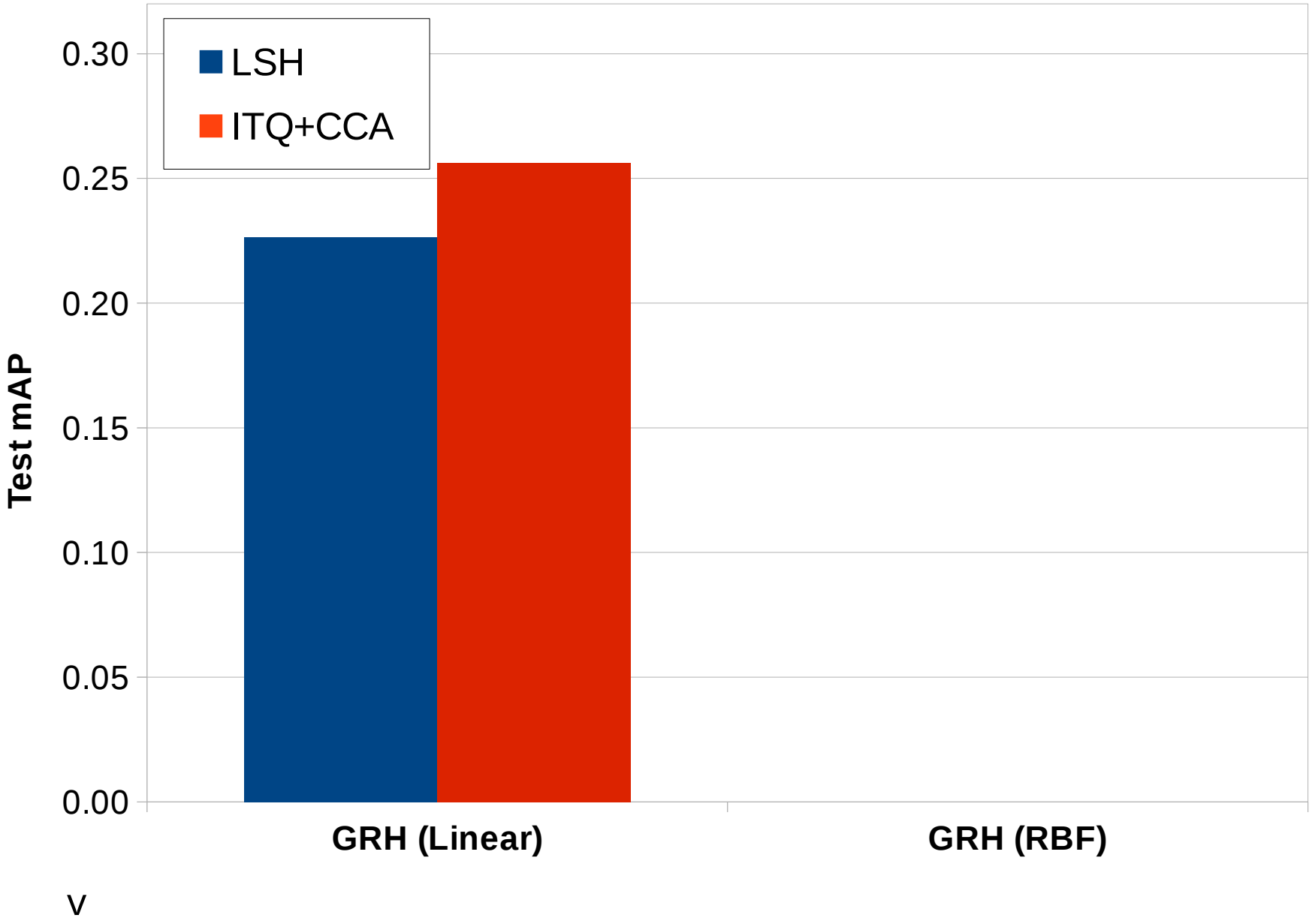
GRH vs Literature (CIFAR-10 @ 32 bits)



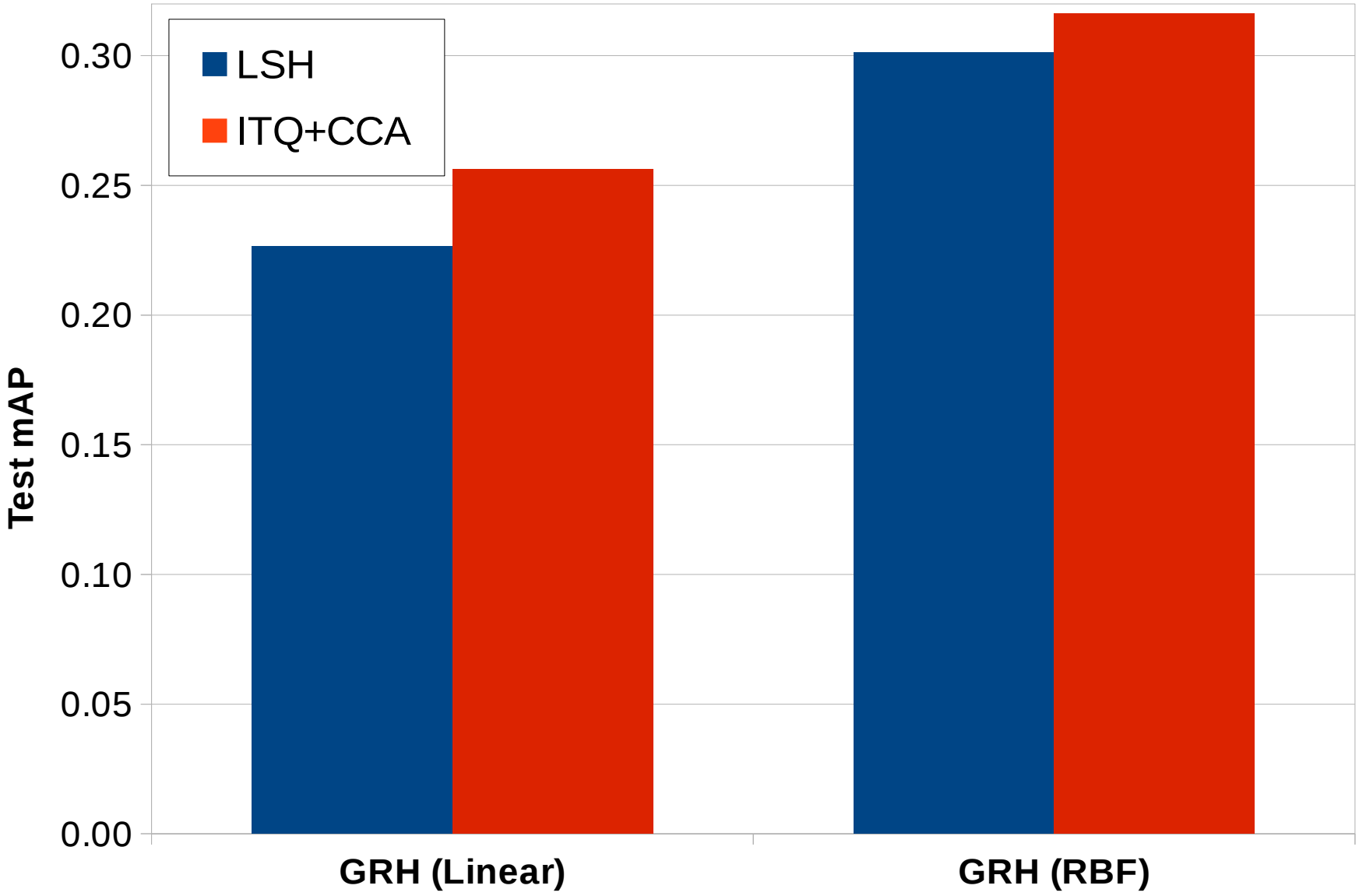
GRH vs Literature (CIFAR-10 @ 32 bits)



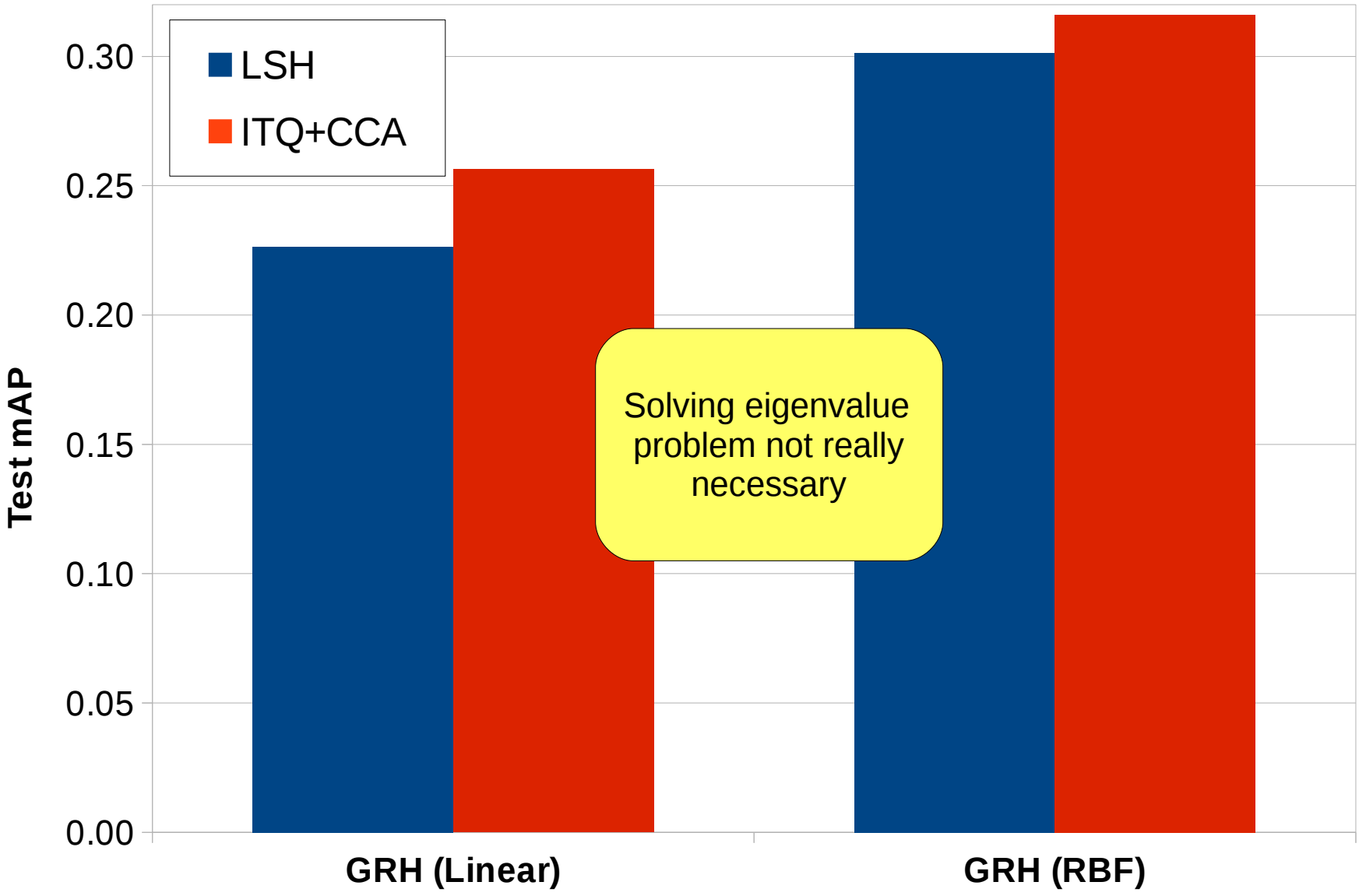
GRH vs. Initialisation Strategy (CIFAR-10 @ 32 bits)



GRH vs. Initialisation Strategy (CIFAR-10 @ 32 bits)



GRH vs. Initialisation Strategy (CIFAR-10 @ 32 bits)



GRH Timing (CIFAR-10 @ 32 bits)

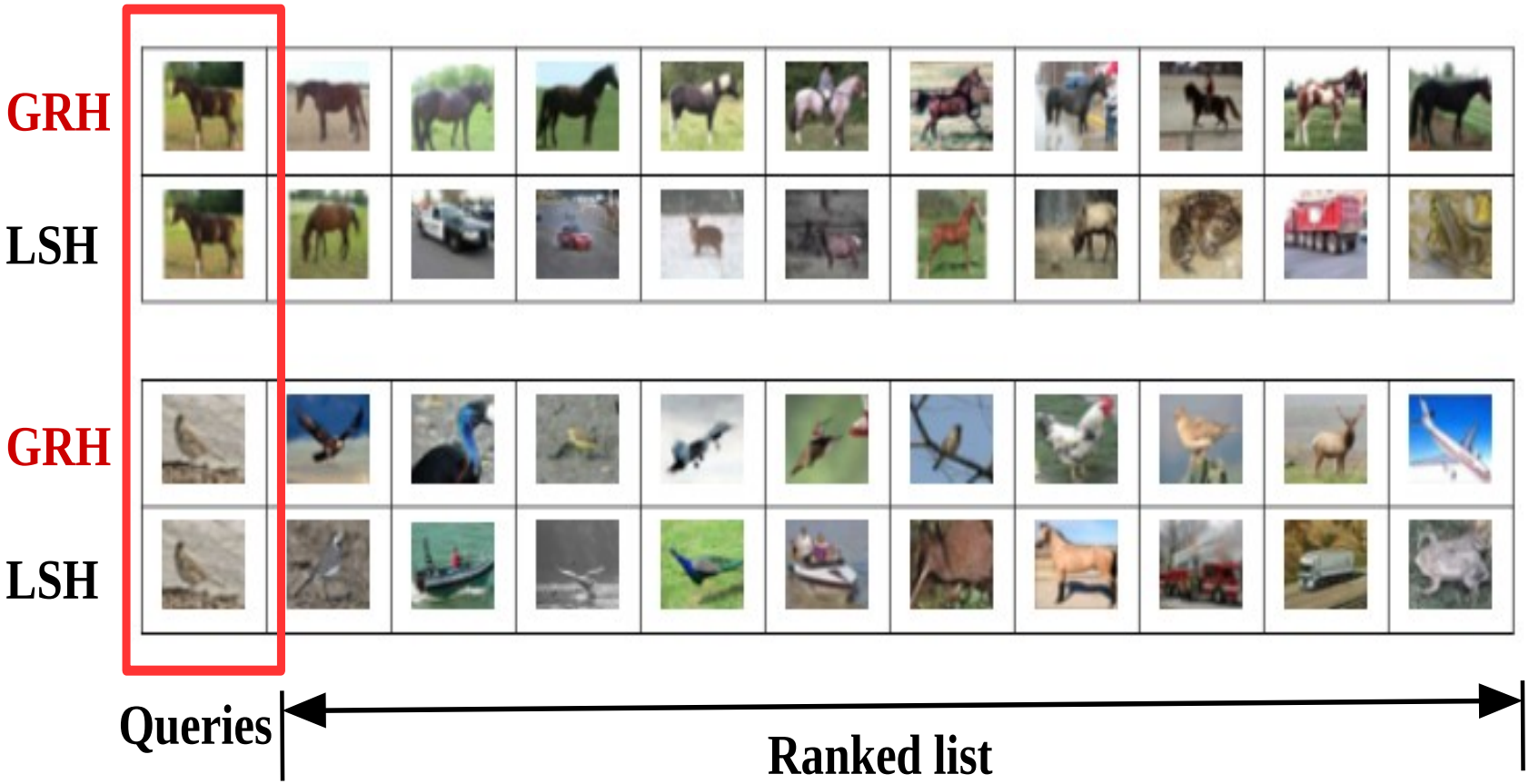
Timings (s)*				
Method	**Train**	**Test**	**Total**	**% rel cng vs. KSH**
GRH	**8.008**	**0.033**	**8.041**	-90%
KSH [5]	74.024	0.103	82.27	–
BRE [10]	227.841	0.370	231.4	+181%

** All times obtained an idle Intel 2.7GHz, 16Gb RAM machine and averaged over 10 runs.*

[5] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, Shih-Fu Chang. **Supervised Hashing with Kernels**. *In CVPR'12*.

[10] Brian Kulis and Trevor Darrell. **Learning to Hash with Binary Reconstructive Embeddings**. *In NIPS'09*.

Example of Top 10 Retrieved Images



End of Part 2 of Talk

- Takeaway messages:
  - $\mathbf{R}_1$: Regularising bits over a graph is effective (and efficient) for hashcode learning.
  - $\mathbf{R}_2$: An intermediate eigendecomposition step is not necessary.
  - $\mathbf{R}_2$: Non-linear hypersurfaces yield a more effective bucketing of the space than hyperplanes.

Learning to Hash for Large Scale Image Retrieval

Thesis Statement (Recap)

"The **retrieval effectiveness** *of hashing-based ANN search can be improved by* **learning** *the* **quantisation thresholds** *and* **hashing hyperplanes** *in a manner that is influenced by the* **distribution of the data**"

Conclusions

- Learning the thresholds and hypersurfaces:
  - Semi-supervised clustering algorithm for learning quantisation thresholds.
  - EM-like algorithm for positioning hashing hypersurfaces based on a supervisory signal.
- Additional contributions in the thesis:
  1. Learning **multi-modal** hashing hypersurfaces.
  2. Learning **variable** quantisation thresholds.
  3. **Combining** threshold and hypersurface learning.

Future Work

- Extend the threshold and hypersurface learning to an **online streaming data** scenario e.g. event detection in Twitter.
- Leverage **Deep Learning** to optimise feature representation and hash functions end-to-end rather than using hand crafted features (e.g. SIFT).

Thank you for your attention!

References

- [1] Sean Moran, Victor Lavrenko, Miles Osborne. **Neighbourhood Preserving Quantisation for LSH**. *In SIGIR'13*.
- [2] Sean Moran, Victor Lavrenko, Miles Osborne. **Variable Bit Quantisation for LSH**. *In ACL'13*.
- [3] Sean Moran and Victor Lavrenko. **Graph Regularised Hashing**. *In ECIR'15*.
- [4] Sean Moran and Victor Lavrenko. **Regularised Cross Modal Hashing**. *In SIGIR'15*.

References

- [5] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, Shih-Fu Chang. **Supervised Hashing with Kernels**. *In CVPR'12*.
- [6] Yunchao Gong, Svetlana Lazebnik. **Iterative Quantization: A Procrustean Approach to Learning Binary Codes for Large-scale Image Retrieval**. *In CVPR'11*.
- [7] Weihao Kong, Wu-Jun Li.**Double Bit Quantisation**. *In AAAI'12*.
- [8] Weihao Kong, Wu-Jun Li. **Manhattan Hashing for Large-Scale Image Retrieval**. *In SIGIR'12*.

References

- [9] Yair Weiss, Antonio Torralba, Rob Fergus. **Spectral Hashing**. *In NIPS'08*.
- [10] Brian Kulis and Trevor Darrell. **Learning to Hash with Binary Reconstructive Embeddings**. *In NIPS'09*.
- [11] Wei Liu, Jun Wang, Sanjiv Kumar, Shih-Fu Chang. **Hashing with Graphs**. *In ICML'11*.
- [12] Dell Zhang, Jun Wang, Deng Cai, Jinsog Lu. **Self-Taught Hashing for Fast Similarity Search**. *In SIGIR'10*.

References

- [13] Piotr Indyk and Rajeev Motwani. **Approximate nearest neighbors: Towards removing the curse of dimensionality**. *In STOC'98*.
- [14] Fernando Diaz. **Regularizing query-based retrieval scores**. *In IR'07*.
- [15] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yan-Tao Zheng. *NUS-WIDE: A Real-World Web Image Database from National University of Singapore*. *In ICIVR'09*.