Beyond Hamming Distance:
Exploring Spatial Encoding in Perceptual Hashes

Sean McKeown

Edinburgh Napier University

/in/mckeowns87

s.mckeown@napier.ac.uk

Image Matching with Perceptual Hashes

AKA Semantic Approximate Matching

Distances - Modified Image

astronauts.*png* vs astronauts.*jpg*

Hash	Distance
ahash	0.0
whash	0.0
dhash	0.0
phash	0.0
blockhash	0.0

Image Diff
(compression)

astronauts.png vs astronauts_*edit*.png

Hash	Distance
ahash	0.015625
whash	0.0
dhash	0.046875
phash	0.093750
blockhash	0.023438

Image Diff
(edit)

Distances – Different Images

a.jpg vs b.jpg

Hash	Distance
average_hash	0.703125
whash	0.718750
dhash	0.546875
dhash_vertical	0.609375
phash	0.625000
phash_simple	0.578125
blockhash	0.601563

Calculating Distance

Hamming Distance

- **XOR** bit strings, **count** 1s

Normalised Hamming Distance

- **Divide** by hash length
  - Result is between 0 and 1

- Captures global difference between hashes
  - Positional information is completely lost
  - Often not important, or just not leveraged?

Hamming Distance = 8

A	1	0	0	1	0	1	1	0	0	1	0	0	1	0	0	1
B	0	1	0	1	0	0	1	0	1	1	1	0	0	1	0	0
XOR	1	1	0	0	0	1	0	0	1	0	1	0	1	1	0	1

Hamming Distance = 8

A	1	0	0	1	0	1	0	1	0	1	0	0	1	0	0	1
B	0	1	1	0	1	0	1	1	0	1	1	0	1	0	0	1
XOR	1	1	1	1	1	1	1	0	0	0	1	0	0	0	0	0

/in/mckeowns87 s.mckeown@napier.ac.uk

Macro Goal: Separate Distance Distributions

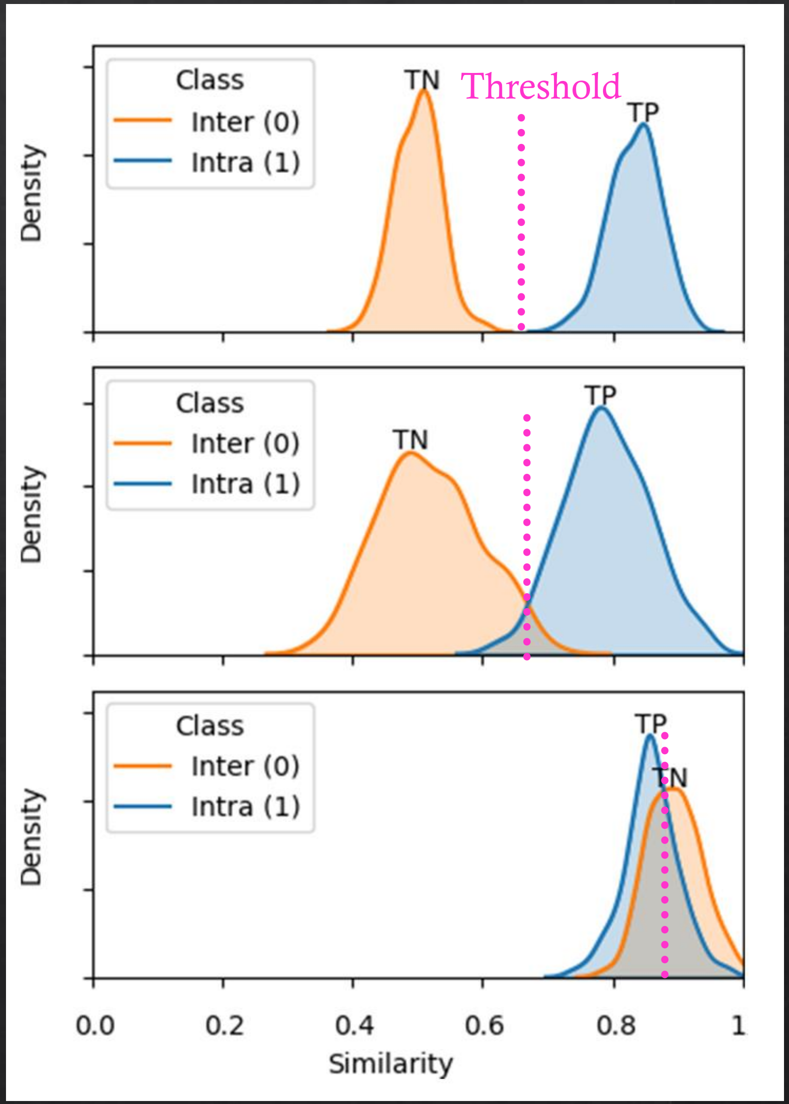
Unrelated Images vs. Modified Originals

- Set a Distance Threshold that lets us determine if an image pair are a:
  - **Match**
  - **No Match**
- Overlap causes **False Positives / False Negatives**
- Focus of prior work is typically the Perceptual Hash, not the <u>Distance Metric</u>

Best Case

Some Errors

Terrible *Flip a Coin*



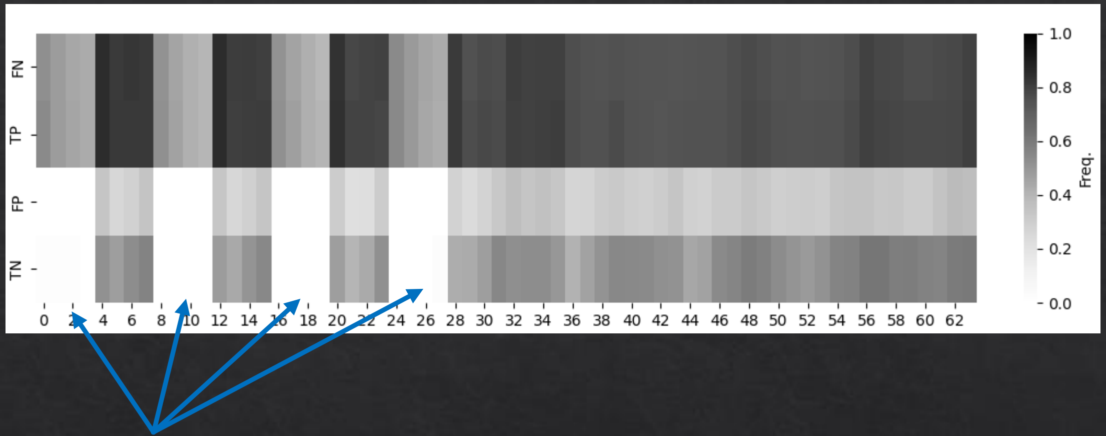
Does Positional Information Matter?

Approach

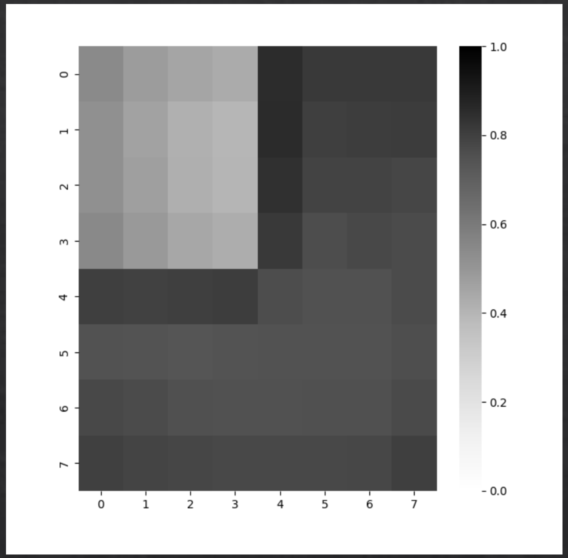
- Determine if redundant data in an image is identifiable in output Perceptual Hashes
  1. Insert constant data at fixed positions (e.g., top-left, border, watermark)
  2. Compare original images to modified images -> aggregate over a dataset
  3. Re-weight hash bits based on their contribution towards correct classifications
     - **Low-weights** = spatially encoded redundant data **(no discriminatory power**)
- Similar approach for transposition (mirroring, rotation), and crop
- Tested on spatial-domain hashes (ahash, dhash) and DCT-based hashes (pHash, PDQ)

/in/mckeowns87 s.mckeown@napier.ac.uk

It works: Top-left Redundant Data Insertion



Low-weights indicate these bits are not useful for discrimination in the classification process

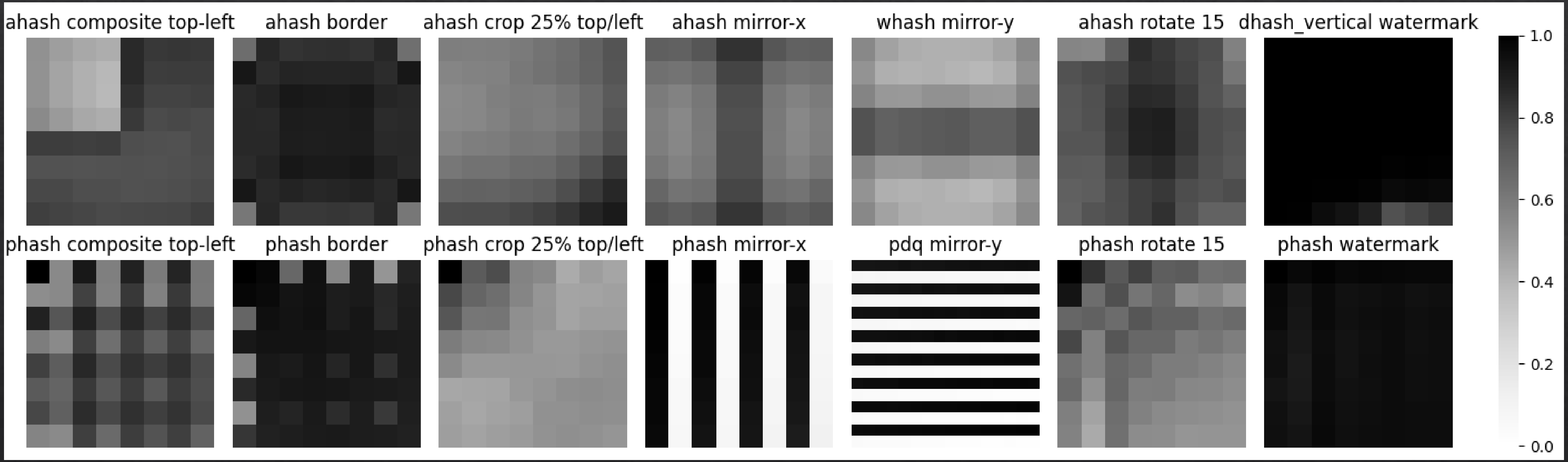


Wrapping to a square (TP only) makes the pattern clearer

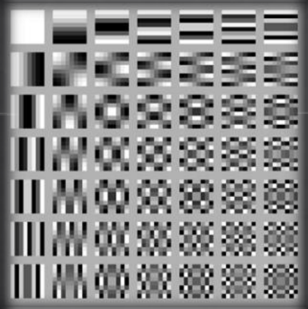
Patterns for a Range of Changes

Spatial-domain

Weight



DCT (Frequency-domain)



Leveraging Hash-bit Positional Encoding

New Metrics!

- Common metrics align strongly with Normalised Hamming Distance
  - Scipy.Spatial.Distance (e.g., L1, Earthmover, Manhattan, Cosine)
- New metrics needed to capture hash "locality"

<u>Normalised Convolution Distance</u>

- Convolutions on XOR of hash matrices

<u>2D N-Grams</u>

- Sliding windows to capture locality, compare windows between hashes

<u>Hathched Matrix</u>

- Take advantage of the row/column pattern for pHash/PDQ

Convolution Distance

1. **XOR** input hash matrices

2. **Convolve**
   - Multiple kernel configurations
   - Settled on (4,4) matrix of ones
   - Captures weighty clusters of change

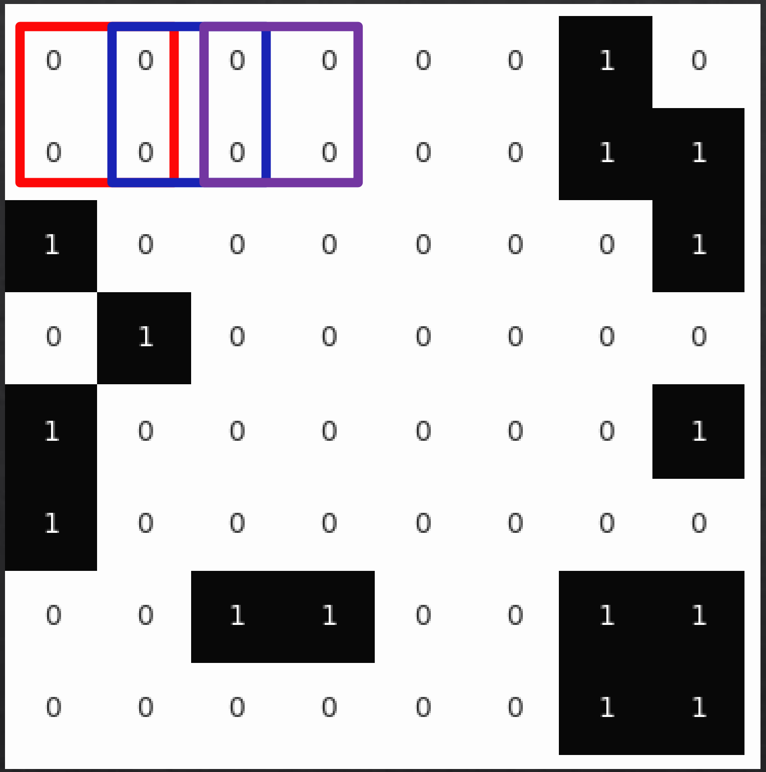
3. **Sum** all matrix elements

4. **Normalise** by maximum possible distance
   1. Dependent on kernel, matrix size

0	0	0	0	0	0	1	0
0	0	0	0	0	0	1	1
1	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0
1	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0
0	0	1	1	0	0	1	1
0	0	0	0	0	0	1	1

0	0	0	0	0	2	3	3
1	1	0	0	0	2	4	4
2	2	1	0	0	1	3	3
3	3	1	0	0	0	2	2
3	3	1	0	0	0	1	1
2	3	2	2	1	1	3	3
1	2	2	2	1	2	4	4
0	1	2	2	1	2	4	4

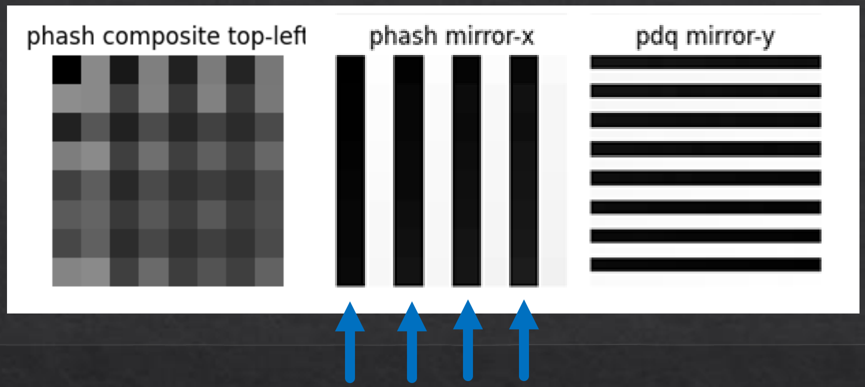
2D N-Grams



1. **Slide a two-dimensional window** over all elements of each hash
   - Overlapping windows
   - Various sizes tested, 2x2 chosen
2. **Flatten** and **concatenate** all windows (single array for each hash)
3. **Calculate Cosine Distance** between both arrays

/in/mckeowns87 s.mckeown@napier.ac.uk

Hatched Matrix Distance



- DCT-based hashes largely generate Row/Column patterns of coefficient weights

1. **Extract rows/columns** for each hash into their own arrays
2. **Compare odd/even rows/columns** (Hamming distance)
3. Take **minimum** value for row/column, then **average**
   - Biases towards similar rows/columns
   - Assumes this isn't by accident!

Evaluation

- Compare vs. Baseline: Normalised Hamming Distance
- 250k subset of Flickr 1 Million
- Compare modified image to original: %pt difference to Hamming AUC
- Validated inter-image (no-match) distributions, covered in paper

/in/mckeowns87 s.mckeown@napier.ac.uk

Results

Hatched Matrix

- Mirrored images go from a weakness to a strength for pHash and PDQ (+49 %pts). Small benefit on rotation
- Largely detrimental to spatial hashes

Convolution Distance

- Reasonably large spatial hash benefit for rotation (+2.5 - 8.4 %pts)
- Some gains in top-left insertion (+0.4 – 4.7 %pts)

2D N-Gram

- Disappointing, trade-offs are as large as gains

		AUC	%pt Diff to Hamming		
Hash	**Trans.**	**Hamm.**	**Conv4_4**	**Hatch**	**2gram**
ahash	CropTL	0.829	0.2	0.0	-5.4
	MirrorX	0.766	-0.3	-2.6	-3.6
	Rotate	0.919	**2.5**	-0.1	-0.8
	Border	0.971	0.7	-0.3	0.6
	CompTL	0.527	**3.8**	0.5	-4.8
dhash	CropTL	0.641	-0.6	-0.7	-4.9
	Mirr.X	0.618	-1.1	-3.4	**1.0**
	Rotate	0.808	**7.6**	-1.0	**4.3**
	Border	0.995	0.4	-0.3	0.1
	CompTL	0.992	0.4	-0.4	-2.9
dhash vertical	CropTL	0.646	-0.4	-0.4	-4.3
	MirrorX	0.801	-0.1	-2.9	-6.5
	Rotate	0.780	**8.4**	-1.0	**3.6**
	Border	0.992	0.6	-0.4	0.2
	CompTL	0.989	0.4	-0.6	-5.5
Neural hash	CropTL	0.996	0.0	-0.1	-0.4
	MirrorX	0.930	-0.1	-0.4	-1.0
	Rotate	0.988	0.0	-0.2	-0.4
	Border	0.999	0.0	0.0	0.0
	CompTL	0.844	0.9	-0.7	**2.7**
PDQ	CropTL	0.527	-0.2	-0.1	-0.9
	MirrorX	0.515	0.9	**48.5**	**1.6**
	Rotate	0.502	**1.7**	**3.0**	**2.7**
	Border	1.000	0.0	0.0	0.0
	CompTL	1.000	0.0	0.0	0.0
phash	CropTL	0.586	-2.5	-0.2	-0.3
	MirrorX	0.496	**2.3**	**49.1**	**1.1**
	Rotate	0.675	-0.1	**1.5**	-0.9
	Border	1.000	0.0	0.0	0.0
	CompTL	0.944	**2.4**	0.6	**1.3**
whash	CropTL	0.821	0.3	0.0	-2.1
	MirrorX	0.744	-0.1	-2.9	-1.7
	Rotate	0.904	**3.1**	-0.2	**1.1**
	Border	0.921	**1.4**	-0.5	**3.6**
	CompTL	0.604	**4.7**	0.2	-3.4

Future Work

- Explore alternatives based on observed patterns
  - Crop isn't helped much with the three proposed algorithms
- Frequency domain:
  - DCT low-frequency coefficient weighting (incorporate into Hatched Matrix or other)
- Spatial domain:
  - Centre of the image should take more importance

/in/mckeowns87 s.mckeown@napier.ac.uk

Thank You

Perceptual Hash Evaluation Framework:
https://zenodo.org/records/10363151
https://github.com/AabyWan/PHASER

Prior Work:

S. McKeown and W. J. Buchanan, ‘Hamming distributions of popular perceptual hashing techniques’, *Forensic Science International: Digital Investigation*, vol. 44, p. 301509, 2023.

S. McKeown, P. Aaby, and A. Steyven, ‘PHASER: Perceptual hashing algorithms evaluation and results-An open source forensic framework’, *Forensic Science International: Digital Investigation*, vol. 48, p. 301680, 2024.

/in/mckeowns87

s.mckeown@napier.ac.uk