Image Theft Detection with Self-Organising Maps

Philip Prentis, Mats Sjöberg, Markus Koskela, and Jorma Laaksonen

Czech Technical University in Prague,
Helsinki University of Technology
prentphi@fjfi.cvut.cz,
{mats.sjoberg,markus.koskela,jorma.laaksonen}@tkk.fi
http://www.cvut.cz, http://www.tkk.fi

Abstract. In this paper an application of the TS-SOM variant of the self-organising map algorithm on the problem of copyright theft detection for bitmap images is shown. The algorithm facilitates the location of originals of copied, damaged or modified images within a database of hundreds of thousands of stock images. The method is shown to outperform binary decision tree indexing with invariant frame detection.

Keywords: Image theft detection, image retrieval, self-organising maps, TS-SOM, PicSOM.

1 Introduction

Companies that publish large on-line databases of stock images are often faced with the problem of copyright infringement due to image theft. This occurs when people copy images and reuse them for commercial purposes without paying. Locating such images on the web and proving that they are stolen from the company's database may be difficult if the database is very large or if the images have been modified.

One of the few published attempts at solving this problem is presented in [1,2], where Horacek et al describe a method of copyright theft detection that uses stochastic decision trees, which is used as a comparison method in this paper.

Other attempts include *content-based image detection* [3] and alternative means of dealing with image theft such as watermarking (e.g. [4]).

In the past, self-organising maps [5] have been used on a number of image processing and retrieval tasks such as content-based image retrieval [8,9,10,11,12], automatic image annotation [14], colour-based image browsing [15] and others. The use of tree-structured variants, which allow fast logarithmic search [6,7], therefore presents itself for copyright theft detection.

2 Methodology

2.1 Self-Organising Maps

Simply put, a self-organising map (SOM) is a structure resulting from an iterative algorithm that reduces a high-dimensional input-space (set of vectors

C. Alippi et al. (Eds.): ICANN 2009, Part I, LNCS 5768, pp. 495–504, 2009.

from R^N) to a two-dimensional ordered grid of codebook vectors (neurons) $\{n_{ij}|i = 1..N_1, j = 1..N_2; \text{typically} N_1 = N_2\}$ that quantify it. Each input vector is quantised by the codebook vector with the smallest Euclidean distance.

In a successfully adapted map, adjacent grid nodes will quantise similar data, i.e. data points that have a small Euclidean distance. The principle may be seen in Fig. 1, which shows a typical SOM in a 2D input space.

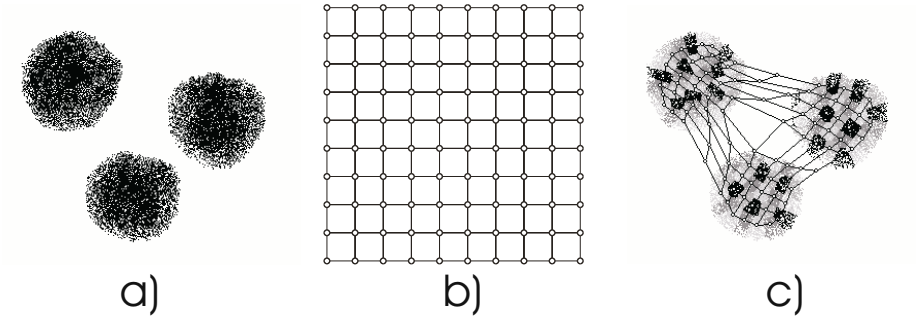

Fig. 1. a) 2-dimensional input vectors in 3 clusters. b) A SOM, topologically ordered in a grid of 10×10 neurons c) The SOM adapts itself to match the input space, each neuron's codebook vector quantifying a set of inputs, denoted by differing shades of grey.

Algorithm. The self-organisation process is achieved as follows:

1. Initialise the codebook vectors $n_{ab}(0)$ at random (usually by setting them to randomly chosen input vectors).
2. Select a random input $i(t)$ and find the best matching neuron (BMN) $n_{best}(t)$ (i.e. the neuron with the closest codebook vector). Every input sample has the same probability of being selected.
3. Move the BMN and its topological neighbours within a certain neighbourhood distance towards the selected input vector. Units located topologically further from BMN are moved less.

$$n_{ab}(t+1) = n_{ab}(t) + \eta(t) \cdot \phi(a,b,t) \cdot [i(t) - n_{ab}(t)], \tag{1}$$

where

$$\eta(t): \ N_0 \rightarrow \ [0;1] \qquad \text{monotonously decreasing}, \tag{2}$$
$$\phi(a,b,t): \ N_0 \times N_0 \times N_0 \rightarrow \ [0;1],$$

ϕ decreases monotonously with the topological distance of n_{ab} from n_{best} and with t. The topological distance is usually calculated as the length of the shortest path from one neuron to the other in the graph (grid) that represents the network's topology.

4. Proceed to iteration $t+1$. Repeat 2 and 3 iteratively, reducing the proportion of the distance moved η and the neighbourhood distance ϕ each iteration, until they reach a certain predetermined threshold.

It should be noted that problems may occur if the map does not adapt properly and the topology of the map is not be preserved. In such cases similar data may be classified by topologically distant neurons causing classification errors. For more on self-organising maps, see [5].

2.2 Tree-Structured Self-Organising Maps

With larger maps, search for the BMN slows quadratically with the growing width/height of the topology. The complexity of the search is O(N), where N is the number of neurons in the map (each neuron must be compared to determine the BMN). One solution to this problem is to use tree-structured self-organising maps (TS-SOM), [6,7]. A TS-SOM is a hierarchical structure of SOMs of exponentially increasing size. Each level of the TS-SOM adapts separately, but in the lower levels, the search for the best-matching neuron is limited to those hierarchically connected to the BMN of the previous layer and their neighbours. See Fig. 2.

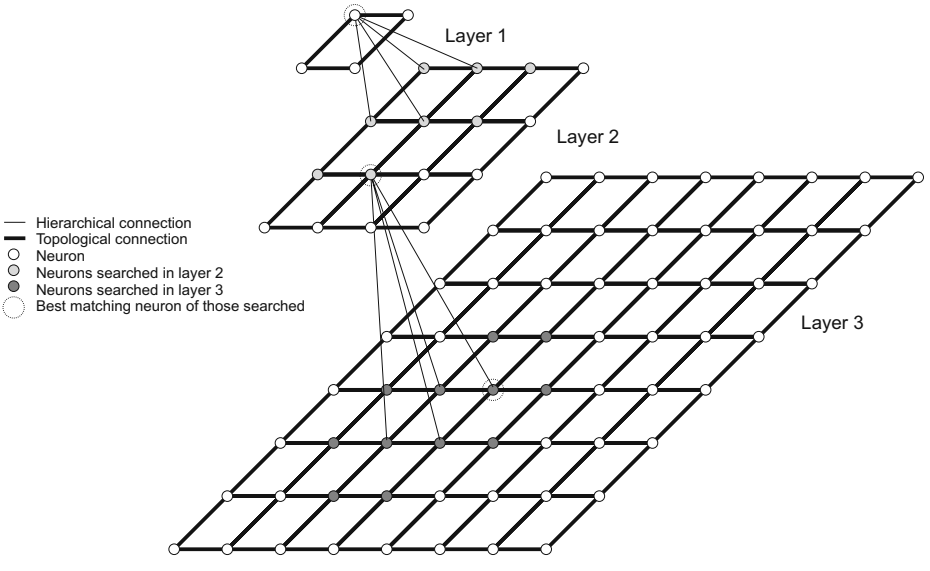

Fig. 2. A 3-layer TS-SOM with 4 neurons at top layer and 64 at the bottom

Algorithm

1. Iterate the SOM algorithm on the top layer until the threshold has been met.
2. Iterate the SOM algorithm on the next layer until the threshold has been met, but limit the search for the BMN to the neurons located under the winning neuron of the previous layer.
3. Repeat 2 until all layers have been updated.

The advantages of such a structure are obvious. Instead of performing a full-search for the BMN at the lower layers, we restrict ourselves to a constant number of neurons per a given layer, thus greatly increasing the adaptation speed. The complexity of the algorithm is $O(\log N)$, where N is the number of neurons on the bottom layer [7]. Also, due to the hierarchical structuring, all the SOMs will be orientated similarly in input space.

2.3 PicSOM

PicSOM [10] works by taking a number of TS-SOMs, each one mapping a different feature calculated on the given image set. A query image is taken and its BMN is located in each map. (In classification tasks, multiple queries are used, so multiple BMN are located; however, for copyright theft detection, only a single query is used.) For the lowest-level layers of the TS-SOM, matrices of values are calculated, one value for every neuron. The BMN is awarded a value of 1 while the others are given a value of 0. Then a convolution filter with a triangular kernel (not to be confused with the kernel used in the SOM algorithm) is passed over it, increasing the value of those neurons within the vicinity of the BMN. Due to the topology preserving property of the SOM these nearby units can be expected to be similar to the BMN. Finally, the result is normalised. See Fig. 3.

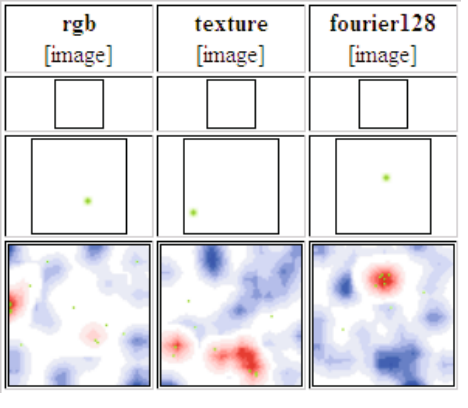

Fig. 3. PicSOM: Three 3-layer TS-SOM with 16 neurons at the top layers and 4096 at the bottom. Value maps are calculated for the bottom layer, low values being coloured blue and high values red. BMN of queries are marked in green.

PicSOM retrieves a desired number of closest matching images from the database according to the sum of their value scores on each map. Therefore, images that score highly on multiple maps (and are therefore similar to the query in multiple features) will be chosen with higher priority than those on fewer maps with lower scores.

3 Experiments

3.1 Data

Image Database. The experiments were run on feature vectors calculated from a data set of 322283 stock images. Each image consisted of a jpeg-encoded true-colour bitmap, scaled to fit in a 640×480 pixel window.

Stolen Images. Stolen images were simulated by taking the first 1000 images in the database and modifying them using 7 different combinations of randomly varying degrees of the following distortions:

- Radiometric distortion (contrast, brightness)
- Scaling
- Cropping
- Frame added
- Logo added

The 7 combinations were as follows:

1. RndLogo1 – random logo
2. RndScale1 – random scaling
3. RndRadiom1 – random radiometric distortion
4. RndCrop1 – random cropping
5. RndLogSca1 – random logo + scaling
6. RndFraLogSca1 – random frame + logo + scaling
7. RndLogCroRad1 – random logo + cropping + radiometric distortion

The 7000 resulting *stolen images* were then used as queries for testing the method's precision.

3.2 Feature Selection

Two experiments were performed. In the first the standard selection of features used by PicSOM [13] was used.

1. rgb = Average rgb colour: average values of Red, Green and Blue channels, calculated in five zones.
2. qgreyrgb = Average grey-scale intensity: as above, but calculated for average grey-scale intensity of zones rather than separate rgb channels.
3. texture = Texture neighbourhood: texture feature based on brightness of neighbouring pixels, calculated in five zones.
4. z1texture = Single zone texture neighbourhood: as above, but counting the entire image as a single zone.
5. colm = Colour moments: values of three first central moments of colour distribution.
6. edgehist = Edge histogram: histogram of four Sobel edge directions, calculated in five zones.

7. edgecoocc = Edge co-occurrence matrix: co-occurrence matrix of four Sobel edge directions, calculated in five zones.
8. colourlayout = Colour layout: DCT coefficients of average colour in 20×20 grid.
9. scalablecolour = Scalable colour: Haar transform of quantised HSV colour histogram.
10. dominantcolour = Dominant colour: CIE Lab coordinates of three dominant colour clusters.
11. edgehistogram = Edge histogram: histogram of five edge types in 4×4 subimages.

Of these, rgb, qgreyrgb, texture, colm, edgehist and edgecoocc are calculated for five separate, equally-sized zones of the given image – a circular zone at the centre, taking up one fifth of the area, plus the top, bottom, left and right portions of the remainder. See Fig. 4

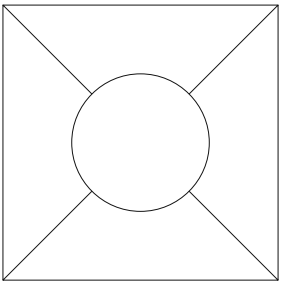

Fig. 4. For the calculation of the rgb, texture, colm, edgehist and edgecoocc features in PicSOM, images are divided up into five separate, equally-sized zones

In the second experiment, only the centre zones of these features were calculated in an attempt to improve the results for images with added frames (with the exception of qgreyrgb, which would have been reduced to a scalar value).

3.3 Precision Measure

Precision was calculated for each of the 7 combinations of image distortions separately as the percentage of originals located by the method. An image was considered to be located by the method if it appeared within the first 20 images returned by PicSOM. It is assumed that such a small number of images could be easily visualised at once as thumbnails and identified by a human operator or safely verified using a computationally heavier method such as phase-correlation, as in [1].

3.4 TS-SOM Parameters

The TS-SOMs used for mapping the feature spaces describing the data were made of 4 levels of 4×4, 16×16, 64×64 and 256×256 neurons, respectively. Each neuron on the top three layers was hierarchically connected with the 4×4 neurons located directly beneath it on the next layer with the search for the BMN being conducted on an area of 10×10 neurons beneath it – i.e. a frame 3 neurons wide surrounding the 16 hierarchically connected ones (the area was 7×10 for edge neurons and 7×7 for corner ones, as the map topology did not loop).

Each map was trained by performing 100000 iterations on each level in succession. To prevent overlearning, the training parameters were not fine-tuned to match the data, but simply taken from previous unpublished optimisation performed on other data.

3.5 Training Time

Feature Vectors. The average calculation time of the individual features ranged from 0.17s (e.g. 11 – Edge histogram) to 0.67s (e.g. 10 – Dominant colour) per image, depending on the complexity of the features used. The average total calculation time for all features per a single image was 3.51s. Therefore, with parallelization, feature vectors for the entire database of 323 thousand images could be calculated in approximately 60 hours. If calculated in series, it would take about 2 weeks.

TS-SOM Training. TS-SOM training time varied from a couple of hours to several days, depending on the dimensionality of the feature vectors.

Search Time. Due to the fast logarithmic search for the BMN used by the TS-SOM, the search time for a single *stolen image* is insignificant when compared with the feature vector calculation time (3.51s), which precedes it.

3.6 Comparison

The results were compared with those of the binary decision tree (BDT) method used in [1], which were calculated on the same 7000 "stolen images" with originals in an unspecified subset of 100000 of the 322283 images in the database. Their precision scores are equal to the percentage of located images. An image was considered located if it was one of the top twenty returned by the BDT indexing system and had been successfully verified with phase correlation.

As there were only *features* calculated from the original images available (i.e. the actual images were not available), it was not possible to incorporate verification process into PicSOM. However, cases of incorrect phase correlation verification and subsequent loss of precision were sufficiently rare as to render them insignificant.

3.7 Results

Table 1 shows the average precision scores calculated on the 1000 images of a given combination of transformations for both experiments (PicSOM with standard features and PicSOM with centre-zone only features) as well as the comparison scores of the Binary Decision Tree Indexing system published in [1]. As can be seen, PicSOM achieved better results than BDT for almost all combinations of image distortion, with the exception of framed images where BDT scored marginally higher. The most significant difference was for cropped images, where PicSOM greatly outperformed BDT.

Furthermore, limiting the calculation of the features to the centre zones of the images greatly improved the results for framed images (outperforming BDT by 17.6%), while achieving similar albeit slightly reduced precision scores for the other distortion types.

Table 1. PicSOM – PicSOM retrieval with standard feature set; PicSOM-C – PicSOM retrieval using centre-zones of 5-zone features; BDT – Binary Decision Tree with invariant frame detection (comparison). Highest average precision for given transformation combination marked in bold, lowest in italics.

Average Precision									
Stolen image set		Method			Transformations used on stolen images				
No.	Name	PicSOM	PicSOM-C	BDT	Radiom.	Scale	Crop	Frame	Logo
1	RndLogo1	0.998	**0.999**	*0.982*					Y
2	RndScale1	**0.998**	**0.998**	*0.946*		Y			
3	RndRadiom1	**0.792**	0.770	*0.712*	Y				
4	RndCrop1	**0.762**	0.721	*0.450*			Y		
5	RndLogSca1	**0.997**	0.993	*0.934*		Y			Y
6	RndFraLogSca1	*0.300*	**0.534**	0.358		Y		Y	Y
7	RndLogCroRad1	**0.365**	0.327	*0.182*	Y		Y		Y

4 Conclusions and Discussion

It should be noted that despite the results being skewed in favour of the BDT comparison method (due to the different database sizes), PicSOM still managed to outperform it in all cases without recourse to any specialised feature selection (the centre-zone features were simply a subset of the data provided by the standard feature set). No high-level features (i.e. object detection, etc.) or more sophisticated calculations such as invariant frame detection were used.

As both methods use logarithmic search (binary tree vs. TS-SOM) and the constant time for the preprocessing of a single query image (feature calculation vs. invariant frame detection) is around 3s in both cases, neither method may be concluded to be objectively faster.

It is interesting to note how the reduction of the feature set to centre zones dramatically improves the precision for framed images without significantly damaging the other results. Otherwise, framed images were the only case where – thanks

to the invariant frame detection – the BDT indexing method would have outperformed PicSOM.

In summary, in this paper we have described the problem of image copyright theft detection and proposed a solution using tree-structured self-organising maps (TS-SOM). We have shown that TS-SOM using low-level features can outperform binary decision trees using invariant frame detection. Further research may include experimentation with more specialised features on larger databases and/or real world data.

Acknowledgments

We would like to thank the Institute of Information Theory and Automation of the Academy of Sciences of the Czech Republic for kindly providing access to features calculated on their database of stock images.

Mats Sjöberg has been supported by a grant from the Nokia Foundation.

References

1. Horáček, O., Bican, J., Kamenický, J., Flusser, J.: Image Retrieval for Image Theft Detection. In: The 5th International Conference on Computer Recognition Systems, Warsaw (2007)
2. Horáček, O., Bican, J., Kamenický, J., Flusser, J.: Image Retrieval for Image Theft Detection. In: Computer Recognition Systems 2. Advances in Soft Computing, vol. 45, pp. 44–51. Springer, Heidelberg (2008)
3. Kim, C.: Content-based image copy detection. Signal Processing: Image Communication 18(3), 169–184 (2003)
4. Craver, S.: Zero Knowledge Watermark Detection. In: Pfitzmann, A. (ed.) IH 1999. LNCS, vol. 1768, pp. 101–116. Springer, Heidelberg (2000)
5. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer, Berlin (2001)
6. Koikkalainen, P., Oja, E.: Self-organizing hierarchical feature maps. In: The International Joint Conference on Neural Networks, San Diego, California, vol. II, pp. 279–284 (1990)
7. Koikkalainen, P.: Progress with the tree-structured self-organizing map. In: The 11th European Conference on Artificial Intelligence (1994)
8. Laaksonen, J., Koskela, M., Laakso, S., Oja, E.: Self-Organizing Maps as a Relevance Feedback Technique in Content Based Image Retrieval. Pattern analysis & Applications 4(2-3), 140–152 (2001)
9. Laaksonen, J., Koskela, M., Oja, E.: Application of Self-Organizing Maps in Content Based Image Retrieval. In: The 9th International Conference on Neural Networks, Edinburgh, (1999).
10. Laaksonen, J., Koskela, M., Oja, E.: PicSOM - Self-Organizing Image Retrieval With MPEG-7 Content Descriptors. IEEE Transactions on Neural Networks (2002)
11. Laaksonen, J., Koskela, M., Laakso, S., Oja, E.: PicSOM - content-based image retrieval with self-organizing maps. Pattern Recognition Letters 21 (2000)
12. Laaksonen, J., Koskela, M., Laakso, S., Oja, E.: The PicSOM Retrieval System: Description and Evaluations. In: Proceedings of CIR-2000, Brighton, UK (2000)

13. Laaksonen, J., Oja, J., Koskela, M., Brandt, S.: Analyzing Low-level Visual Features Using Content-Based Image Retrieval. In: The 7th International Conference on Neural Information Processing, Taejon, Korea (2000)
14. Viitaniemi, V., Laaksonen, J.: Keyword-detection approach to automatic image annotation. In: Proceedings of 2nd European Workshop on the Integration of Knowledge, Semantic and Digital Media Technologies, London, UK (2005)
15. Prentis, P.: GalSOM - Colour-Based Image Browsing and Retrieval with Tree-Structured Self-Organising Maps. In: The 6th International Workshop on Self-Organizing Maps, Bielefeld, Germany (2007)