Regularised Cross-Modal Hashing

Sean Moran
School of Informatics
The University of Edinburgh
EH8 9AB, Edinburgh, UK
sean.moran@ed.ac.uk

Victor Lavrenko
School of Informatics
The University of Edinburgh
EH8 9AB, Edinburgh, UK
vlavrenk@inf.ed.ac.uk

ABSTRACT

In this paper we propose Regularised Cross-Modal Hashing (RCMH) a new cross-modal hashing model that projects annotation and visual feature descriptors into a common Hamming space. RCMH optimises the hashcode similarity of related data-points in the annotation modality using an iterative three-step hashing algorithm: in the first step each training image is assigned a K-bit hashcode based on hyperplanes learnt at the previous iteration; in the second step the binary bits are smoothed by a formulation of graph regularisation so that similar data-points have similar bits; in the third step a set of binary classifiers are trained to predict the regularised bits with maximum margin. Visual descriptors are projected into the annotation Hamming space by a set of binary classifiers learnt using the bits of the corresponding annotations as labels. RCMH is shown to consistently improve retrieval effectiveness over state-of-the-art baselines.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

Keywords

Locality Sensitive Hashing; Cross-Modal Retrieval

1. INTRODUCTION

Hashing-based approximate nearest neighbour (ANN) search has emerged as an effective technique for efficiently finding nearest neighbours in large multimedia data collections. Data-points are transformed into compact binary codes via projection [6] and quantisation [7] operations that ensure similar data-points are assigned hashcodes with low Hamming distance. Hashcodes of length K are generated by learning K hyperplanes within the data-space. Depending on which side of a hyperplane a data-point falls its hashcode is appended with a 1 or 0. Each subspace formed by the K hyperplanes constitutes the bucket of a hashtable. Similarity preserving hashcodes can therefore be used as the indices into the buckets of a hashtable for constant time search: a given query need only be compared to data-points falling within the same bucket vastly cutting down the search space.

Most previous hashing research has focused on generating binary codes for data-points within the same modality, for example, a text query executed against a database consisting of textual documents. However it is frequently the case that similar data-points exist in different modalities, for example a Wikipedia page discussing Einstein and an associated image of the scientist. An interesting research question is whether an effective hashing scheme can be constructed to learn hashcodes that are also similar *across* disparate modalities - in this case the Einstein Wikipedia article will ideally be assigned a similar hashcode to the relevant embedded image. Hashing methods that effectively bridge the cross-modal gap will enable the efficiency of ANN search to be expanded to cross-modal data.

In this paper we propose Regularised Cross-Modal Hashing (RCMH), an extension of the *unimodal* hashing model Graph Regularised Hashing (GRH) [6]. RCMH employs a three-step iterative scheme to learn a set of K hash functions: in the first step hashcodes are assigned to the training images using K learnt hyperplanes from the last iteration; in the second step a formulation of graph regularisation [2] refines the distribution of annotation binary bits so that annotations from neighbouring images are assigned similar bits. In the third step K binary classifiers are trained to predict the regularised bits with maximum margin. RCMH forms a cross-modal bridge by subsequently projecting visual descriptors into the learnt annotation Hamming space: this is achieved by learning K binary classifiers in visual space with the bits of the associated annotations as labels.

2. RELATED WORK

Cross-modal hashing research has received increased interest over the past several years due to the recent emergence of large freely available cross-modal datasets from sources such as Flickr. Existing cross-modal hashing schemes seek to jointly preserve the within-modality and between-modality similarities of related data-points in a shared Hamming space. This requirement is frequently solved by learning two sets of K hyperplanes that partition each space into buckets in a manner that yields similar hashcodes for similar data-points both within and across modalities.

Cross-Modal Semi-Supervised Hashing (CMSSH) [1] integrates eigendecomposition and boosting to learn a common multi-modal space for hashing. Cross-View Hashing (CVH)

SIGIR'15, August 09 - 13, 2015, Santiago, Chile.

ACM 978-1-4503-3621-5/15/08 ...$15.00.
DOI: http://dx.doi.org/10.1145/2766462.2767816.

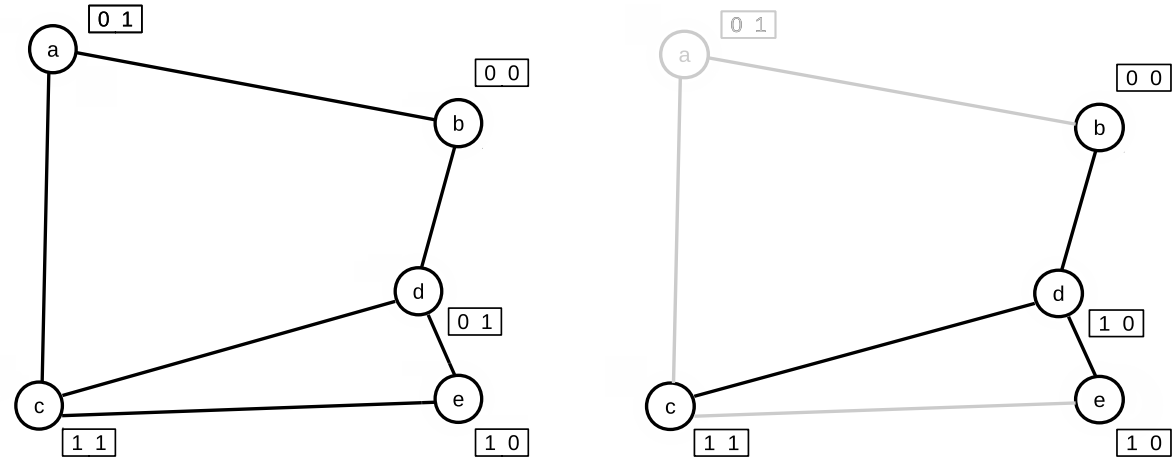

Figure 1: Regularisation step: the hashcode for node d is updated to be more similar with its neighbours (c,b,e)

[5] employs Canonical Correlation Analysis (CCA) [4] to learn a shared latent space from two modalities. The authors of [10] proposed Co-Regularised Hashing (CRH) that learns hash functions for each bit by solving DC (difference-of-convex function) programs, with multiple bits learnt via boosting. Inter-Media Hashing (IMH) [9] minimises a loss function consisting of graph Laplacian terms for intra-modal similarity, a trace minimisation term for inter-modal similarity and linear regression for out-of-sample prediction.

The closest related work to RCMH is Predictable View Hashing (PDH) [8]. PDH employs an iterative scheme for hashcode learning: the annotation bits are used to learn K hyperplanes for the image modality, while the image bits are used to learn K hyperplanes for the annotation modality. RCMH is different to PDH in several aspects, including our novel method of bridging the modalities, our integration of graph regularisation [2] and the lack of an eigendecomposition step. We show that RCMH is much more effective.

3. REGULARISED CROSS-MODAL HASHING (RCMH)

3.1 Problem definition

Let $\mathbf{D} = \{(\mathbf{a}_i, \mathbf{v}_i) : i = 1 \ldots N\}$ denote a collection of N annotated images. Each image is represented by two components: the annotation $\mathbf{a}_i$, and the visual descriptor $\mathbf{v}_i$. The annotation $\mathbf{a}_i$ is a vector over textual features. Visual descriptor $\mathbf{v}_i$ is a vector of real-valued visual features. Our goal is to learn a pair of hash functions F, G that map annotations and visual descriptors into binary hashcodes consisting of K bits. We impose two constraints on our hash functions: **(i)** the annotation hashcode $F(\mathbf{a}_i)$ should be similar to the visual hashcode $G(\mathbf{v}_i)$ of the same image; and **(ii)** the annotation hashcodes $F(\mathbf{a}_i)$ and $F(\mathbf{a}_j)$ should be similar whenever images i and j are considered *neighbours*. The neighbourhood structure for the collection is dictated by an affinity matrix $\mathbf{S}$, where $S_{ij} = 1$ indicates that i and j are neighbours, and $S_{ij} = 0$ indicates they are not.

3.2 The algorithm

Our approach is based on a unimodal method GRH [6]. GRH is restricted to a single modality, while we propose a method that learns a pair of hash functions across two separate modalities: text annotations $\mathbf{a}_i$ and visual descriptors $\mathbf{v}_i$. The hash functions F, G are based on K hyperplanes each: $\mathbf{f}_1 \ldots \mathbf{f}_K$ for the space of words and $\mathbf{g}_1 \ldots \mathbf{g}_K$ for the space of visual features. The hyperplane $\mathbf{f}_j$ is used to assign the j'th bit in the annotation hashcode, while $\mathbf{g}_j$ determines the j'th bit in the visual hashcode. We initialise all hyperplanes randomly, and iteratively perform the following steps: **(1) hashing**, where the hyperplanes $\mathbf{f}_1 \ldots \mathbf{f}_K$ are used to assign hashcodes $\mathbf{b}_1 \ldots \mathbf{b}_N$ to the training images, **(2) regularisation**, where the hashcodes $\mathbf{b}_1 \ldots \mathbf{b}_N$ are made more consistent with the affinity matrix S and **(3) partitioning**, where we adjust the hyperplanes $\mathbf{f}_j, \mathbf{g}_j$ to be consistent with the j'th bit of the hashcodes from step (2).

3.2.1 Step 1: Hashing

We start by assigning a K-bit binary hashcode $\mathbf{b}_i$ to each training image i. Each of the K bits in $\mathbf{b}_i$ is based on a dot product between the image annotation $\mathbf{a}_i$ and one of the hyperplanes $\mathbf{f}_1 \ldots \mathbf{f}_K$:

$$\mathbf{b}_i = F(\mathbf{a}_i) = H\left[\mathbf{a}_i^\intercal \mathbf{f}_1 \ldots \mathbf{a}_i^\intercal \mathbf{f}_K\right] \quad (1)$$

Here $H(x)$ is the Heaviside step function that returns 0 for negative x and 1 otherwise. At test time, hashcodes of visual features $G(\mathbf{v}_i)$ can be computed in the same manner, but using the visual hyperplanes $\mathbf{g}_1 \ldots \mathbf{g}_K$.

3.2.2 Step 2: Regularisation

The aim of this step is to make the hashcodes we obtained in step 1 more consistent with the affinity matrix $\mathbf{S}$. Specifically, whenever images i and j are neighbours, we would like the hashcodes $\mathbf{b}_i$ and $\mathbf{b}_j$ to be similar in terms of their Hamming distance. We achieve this by interpolating the hashcode of image i with the hashcodes of all *neighbouring* images j for which $S_{ij} = 1$. Our approach is similar to the *score regularisation* method of [2]. Formally, we regularise the hashcodes via the following equation:

$$\mathbf{B} \leftarrow H\left(\alpha\ \mathbf{S}\mathbf{D}^{-1}\mathbf{B} + (1-\alpha)\ \mathbf{B} - 0.5\right) \quad (2)$$

Here $\mathbf{S}$ is the affinity matrix and $\mathbf{D}$ is a diagonal matrix containing the number of neighbours for each image. The matrix $\mathbf{B} \in \{0,1\}^{N \times K}$ represents the hashcodes assigned to every image in step 1 of the algorithm and α is a free parameter that specifies how aggressively we regularise the

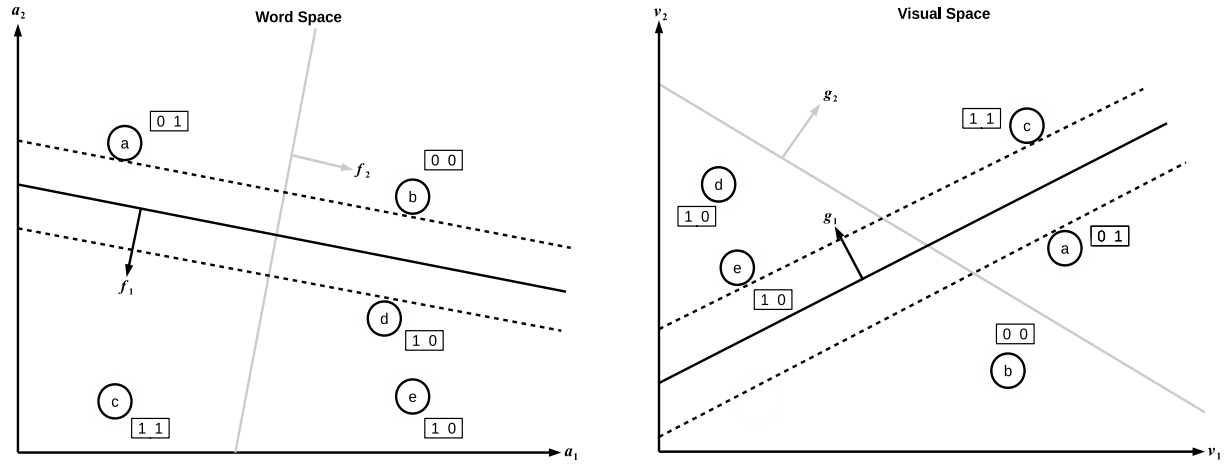

Figure 2: Partitioning step: hyperplanes are learnt in the annotation and visual space using annotation bits as labels.

bits. We show our approach intuitively in Figure 1. In the left side we show 5 images $a \ldots e$ with their initial hashcodes (K=2 bits for this example). The lines between images reflect the neighbourhood structure encoded in the affinity matrix $\mathbf{S}$. Image d has a hashcode 01, but its neighbours b, c, e have hashcodes 00, 11 and 10 respectively. The right side of Figure 1 shows the effect of equation (2) for image d: its hashcode changes to 10, which is more consistent with neighbouring hashcodes (on average).

3.2.3 Step 3: Partitioning

In the final step of the algorithm, we re-estimate the hyperplanes $\mathbf{f}_1 \ldots \mathbf{f}_K$ and $\mathbf{g}_1 \ldots \mathbf{g}_K$ to make them consistent with the regularised hashcodes from step 2 of the algorithm. For each bit $j = 1 \ldots K$, we treat the values $b_{1j} \ldots b_{Nj}$ as the training labels. Specifically, if $b_{ij} = 1$ then the annotation vector $\mathbf{a}_i$ constitutes a positive example for the hyperplane $\mathbf{f}_j$, and the visual vector $\mathbf{v}_i$ is a positive example for $\mathbf{g}_j$. If $b_{ij} = 0$ then $\mathbf{a}_i$ and $\mathbf{v}_i$ are negative examples for $\mathbf{f}_j$ and $\mathbf{g}_j$. Each hyperplane is learned using `liblinear` [3] to maximise the margin between positive and negative examples.

The approach is illustrated in Figure 2. We show five images $a \ldots e$ in two sets of coordinates: the word space on the left and the visual feature-space on the right. Each image is associated with a 2-bit hashcode, and each bit is used to learn a maximum-margin hyperplane that bisects the corresponding space. For example, the first bit has value 0 for images a, b and value 1 for images c, d, e, giving rise to hyperplanes $\mathbf{f}_1$ and $\mathbf{g}_1$, shown as dark lines on the left and the right parts of Figure 2. Note that $\mathbf{f}_1$ and $\mathbf{g}_1$ look very different, because they are defined over two completely different modalities: words on the left and visual features on the right. Similarly, the second bit results in the hyperplanes $\mathbf{f}_2$ and $\mathbf{g}_2$, shown in lighter colour.

3.3 Iteration and constraints

We repeat steps 1-3 above for a small number of iterations M. We briefly describe how the steps enforce the two constraints we imposed on our hash functions in section 3.1. Constraint (i) is enforced in step 3 of the algorithm, when we use the same bit values b_{ij} as targets for the word hyperplanes $\mathbf{f}_j$ and visual hyperplanes $\mathbf{g}_j$. Any image i will either be a positive example for both hyperplanes, or it will be negative for both, so at test time we can expect $\mathbf{a}_i^\intercal \mathbf{f}_j$ to yield the same bit value as $\mathbf{v}_i^\intercal \mathbf{g}_j$. Constraint (ii) is enforced in step 2 of our procedure, where the hashcode for image i is moved towards the centroid hashcode of its neighbours. The centroid (before it is binarised) is a point that minimizes aggregate Euclidean distance to the neighbours, so after step 2 hashcodes $\mathbf{b}_1 \ldots \mathbf{b}_N$ are expected to be more consistent with the neighbourhood structure $\mathbf{S}$.

4. EXPERIMENTS

4.1 Datasets

We evaluate RCMH on two publicly available benchmark datasets: Wiki[1] and NUS-WIDE[2]. Wiki is generated from 2,866 Wikipedia articles. Each article is described with text and an associated image. The visual modality is formed from 128-bit SIFT descriptors, while the annotation modality is represented as 10-dimensional probability distribution over LDA topics. NUS-WIDE is a web image dataset consisting of 269,648 images downloaded from Flickr. We keep the image-text pairs associated with the most frequent 10 classes and perform PCA on the tag co-occurrence features to form a 1,000-dimensional annotation feature set [10] . Each image is represented as a 500-D bag-of-words vector derived from SIFT descriptors. The ground truth nearest neighbours are based on the semantic labels supplied with the datasets, that is, if two images share a class in common they are regarded as true neighbours [10, 9]. Following previous work [10, 9] we randomly select 20% (Wiki) and 1% (NUS-WIDE) of the data-points as queries with the remainder forming the database over which our retrieval experiments are performed. We randomly sample 20% (Wiki) and 1% (NUS-WIDE) of the data-points from the database to form the training dataset (T) to learn the hash functions.

4.2 Baselines

CVH [5], CMSSH [1], CRH [10], IMH [9] and PDH [8].

4.3 Evaluation Protocol

We evaluate RCMH based on two cross-modal retrieval tasks: 1) *Image query vs. text database:* an image is used to

[1] `http://www.svcl.ucsd.edu/projects/crossmodal/`
[2] `http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm`

Task	Method	Code Length		
		K=24	K=48	K=64
Image Query vs. Text Database	CRH	0.1632	0.1752	0.1698
	CVH	0.1570	0.1519	0.1538
	CMSSH	0.1439	0.1501	0.1420
	IMH	0.1881	0.1892	0.1897
	PDH	0.2109	0.2186	0.2266
	RCMH	**0.2439**†	**0.2463**†	**0.2590**†
Text Query vs. Image Database	CRH	0.1266	0.1239	0.1267
	CVH	0.1284	0.1176	0.1185
	CMSSH	0.1119	0.1123	0.1124
	IMH	0.1507	0.1514	0.1491
	PDH	0.1790	0.1860	0.1902
	RCMH	**0.2066**†	**0.1918**†	**0.2201**†

Table 1: mAP scores for Wiki ($T = 574$). † indicates statistical significance vs. PDH (Wilcoxon: p-value < 0.01).

Task	Method	Code Length		
		K=24	K=48	K=64
Image Query vs. Text Database	CRH	0.3536	0.3539	0.3588
	CVH	0.3397	0.3436	0.3412
	CMSSH	0.3429	0.3386	0.3382
	IMH	0.4022	0.4019	0.4040
	PDH	0.4217	0.4245	0.4272
	RCMH	**0.4605**†	**0.4719**†	**0.4649**†
Text Query vs. Image Database	CRH	0.3495	0.3427	0.3481
	CVH	0.3394	0.3435	0.3410
	CMSSH	0.3429	0.3377	0.3492
	IMH	0.3926	0.3960	0.3997
	PDH	0.4053	0.4081	0.4096
	RCMH	**0.4325**†	**0.4380**†	**0.4350**†

Table 2: mAP scores for NUS-WIDE ($T = 1866$). † indicates statistical significance vs. PDH (Wilcoxon: p-value < 0.01).

retrieve the most related text in the text database; 2) *Text query vs. image database:* a text query is used to retrieve the most similar images from the image database. Retrieval accuracy is measured using *Hamming ranking* [10, 9]: binary codes are generated for both the query and the database items and the database items are then ranked in ascending order of the Hamming distance. We evaluate using mean average precision (mAP). Results are the average over 10 random query/database partitions.

4.4 Parameter Optimisation

RCMH has three meta-parameters: the number of iterations M, the amount of regularisation α and the flexibility of margin C. We optimise all meta-parameters via grid search on the validation dataset. Holding the margin parameter constant at $C = 1$, we perform a grid search over $M \in \{1 \dots 5\}$ and $\alpha \in \{0.1, \dots, 0.9, 1.0\}$. We then hold M and α constant at their optimised values, and sweep $C \in \{0.01, 0.1, 1.0, 10, 100\}$. We equally weigh both classes (0 and 1). The IMH β and λ parameters are set via grid search over the range $\{10^{-6}, 10^{-3}, 1, 10^{3}, 10^{6}\}$. The CRH λ_x and λ_y parameters are tuned over the range $\{0.001, 0.01, 0.1, 1\}$ and $\{0.01, 0.1, 1.0, 10, 100, 1000\}$ for γ. We sweep C for PDH in an identical manner to RCMH.

4.5 Results

Our cross-modal retrieval results are presented in Tables 1-2. We observe that RCMH outperforms the baseline systems on both datasets and across all hashcode lengths. For example, for image-text retrieval, RCMH outperforms PDH by a substantial 16% relative mAP at 24 bits on the Wiki dataset and 9% on the NUS-WIDE dataset. We test the statistical significance of the gain in mAP vs. PDH using a Wilcoxon signed rank test on the mAP scores resulting from each random query/database partition: the difference is statistically significant for $p < 0.01$. This is an encouraging result: RCMH is devoid of a computationally expensive eigendecomposition step as most baselines [9, 8, 5], relying instead on graph regularisation to maintain the neighbourhood structure.

5. CONCLUSIONS

In this paper we introduced Regularised Cross-Modal Hashing (RCMH). RCMH employs an iterative three-step scheme to learn a shared multi-modal Hamming space: in the first step hashcodes are assigned to images based on learnt hyperplanes; in the second step hashcodes within the annotation space are refined by updating a node's hashcode to be the average of the hashcodes of its nearest neighbours. In the third step RCMH learns a set of hyperplanes to partition the annotation space into buckets using the bits from the previous step as labels. Visual descriptors are projected into the annotation Hamming space by learning a set of hyperplanes in the visual space using the associated annotation bits as labels. RCMH outperforms a set of strong cross-modal hashing baselines.

6. REFERENCES

[1] M. M. Bronstein, A. M. Bronstein, F. Michel, and N. Paragios. Data fusion through cross-modality metric learning using similarity-sensitive hashing. In *CVPR*, pages 3594–3601, 2010.

[2] F. Diaz. Regularizing query-based retrieval scores. In *IR*, pages 531–562, 2007.

[3] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. In *JLMR*, pages 1871–1874, 2008.

[4] D. R. Hardoon, S. R. Szedmak, and J. R. Shawe-taylor. Canonical correlation analysis: An overview with application to learning methods. In *NC*, pages 2639–2664, 2004.

[5] S. Kumar and R. Udupa. Learning hash functions for cross-view similarity search. In *IJCAI'11*, pages 1360–1365, 2011.

[6] S. Moran and V. Lavrenko. Graph regularised hashing. In *ECIR*, pages 135–146, 2015.

[7] S. Moran, V. Lavrenko, and M. Osborne. Neighbourhood preserving quantisation for LSH. In *SIGIR*, pages 1009–1012, 2013.

[8] M. Rastegari, J. Choi, S. Fakhraei, H. D. III, and L. S. Davis. Predictable dual-view hashing. In *ICML*, pages 1328–1336, 2013.

[9] J. Song, Y. Yang, Y. Yang, Z. Huang, and H. T. Shen. Inter-media hashing for large-scale retrieval from heterogeneous data sources. In *SIGMOD*, pages 785–796, 2013.

[10] Y. Zhen and D. yan Yeung. Co-regularized hashing for multimodal data. In *NIPS*, pages 1385–1393, 2012.