Image Retrieval using Hash Codings on the Kendall Tau Distance Between Embeddings

Guillermo García[1] and Javier Civera[2]

***Abstract*— The objective of any research work is to find methods with very high performance, as simple as possible and, flexible enough that they can be applied to many tasks with little or no modification at all.**

With this in mind as a clear objective, we propose a flexible and simple approach to binary representation learning, in which binary codes are generated in an unsupervised way. Additionally, we investigate the network architecture that best suits the hash model to achieve maximum performance, i.e. Siamese and Triplet Networks are used to learn real embeddings of images in continuous-space.

Our method relies on comparisons of the dimensions of real embeddings to obtain binary codes. No threshold function is used, so robustness concerning different tasks is achieved. As a preprocessing before the binarization step, PCA is performed in the training data to learn a mapping that is used both to reduce the dimension of the real embeddings and to order them in decreasing order according to the variance that each dimension represents. In this way, much more robust binary codes are learned.

The flexibility of the method allows applying it to any task of Object/Scene Recognition with arbitrarily network architecture. Experiments are conducted on conventional datasets such as MNIST and CIFAR-10. Moreover, we also evaluate our method in a scene recognition task of a real-word dataset, 'PARTITIONED NORDLAND', so that we can test our model in different and more complex tasks. Our model outperforms the state-of-the-art methods with a much simpler and flexible approach.

I. INTRODUCTION

Approximate Nearest Neighbour (ANN) search has attracted ever-increasing attention in the era of big data. Thanks to the extremely low costs for computing Hamming distances, binary coding/hashing has been appreciated as an efficient solution to ANN search. It aims at shrinking the embedding size of data and producing binary features to speedup the computation of distance-based pair-wise data relevance. Real applications go from optimal searches in big volumes of data, to place recognition in robotics. In this last, several challenges, which we tackle in this work, are present: Variability in the visual appearance of the places due to viewpoint and illumination changes, occlusions and scene dynamics, or retrieval time constraints because of the real-time operation of robots and the huge number of images in the database.

Some of the first addressed hash methods relied in binarization of SIFT descriptors extracted from images [36], [41]. Those were presumably slow and encouraged to tackle real-time applications, other proposed very simple and fast methods comparing even directly intensity values of pixels [3].

The introduction of deep learning has been a huge breakthrough in performance also in the production of hash codes. An indifferentiable sign activation function is typically used on the top of the encoding network so that binary codes are directly obtained. Various methods have been proposed to empower the encoder with the ability to properly locate data in the Hamming space [6], [34], [49] . In this models the optimization problem with binary constraints lead to NP-Hard problems which have not straightforward solution. In this work, this problem is tackled by proposing a decoupled architecture in which real embeddings are learned first and a hash method is performed next. Decoupled architectures were already proposed by [11], [42] as a solution to this problem.

Recent learning models for binary representation typically use complex graph models or even auxiliary generative models [9], [4], [40], [53]. By distinguishing synthesized data from real ones, the encoder implicitly acknowledges the respective data distribution.

However, in order to formulate realistic and efficient hashing models, one may wonder if these non-trivial components are really needed. In this paper, we answer this question by proposing a simple approach to binary representation learning. Our specific contributions over the state of the art are:

- Show that model complexity is not synonymous with high performance and that simple models with properly thought methods can out-performed them.
- Offer a flexible hash method that can be used with State-of-the-art results in binary Object/Scene recognition with arbitrarily network architecture. Our intuition tells us that it may have a high performance also in other interesting tasks e.g Hash codes for documents, songs, etc
- Set benchmark results for hash codes in weather-variance Scene recognition in order to face the trending problem of Tiny ML in mobile robots.

The rest of this paper is structured as follows. Section II analyzes the related work in hash methods. Section III explains how real embeddings are obtained in each dataset. Section IV introduces the novel hash method. Section V presents our results compared with widely recognized hash-

*This work was not supported by any organization

[1] is with School of Engineering and Architecture, University of Zaragoza, Spain guilleuniversidadjaca@gmail.com

[2]Javier Civera is with the RoPeRT - Robotics, Perception and Real Time Group, University of Zaragoza, Spain jcivera@unizar.es

ing methods. Finally, in section VI we summarize our conclusions.

II. RELATED WORK

Generally, hash functions can be divided into supervised [31], [47], [14], [27], [23], [7], [51], [48] and unsupervised methods [16], [46], [44], [18]. Our hash method itself belongs to the unsupervised class, since no labels are needed once that real embeddings are obtained. Real embeddings can be obtained either in a supervised or an unsupervised way. Neural architectures can be trained in a pure unsupervised way as [4] did with triplet neural networks, by minimizing the distance of an anchor image and its rotated version (i.e. positive pair) while maximizing the distance of the anchor image with a random image (i.e. negative pair). However, in this work, pairwise labels were used to obtained real embeddings as first proposed [27], so our work in its completeness belongs to supervised hash methods and for fairness it will be compared with them.

Our model differs greatly from [9], [4], [40], [53] in that we departing from the current trend of increasingly complex models with complicated graphs and/or auxiliary generative models, try to demonstrate that simple models that allow flexibility can also be associated with high performance.

The great majority of the deep learning hash methods[29], [28], [8], [45], [52] are trained in an end-to-end way, since they affirm that otherwise, the problem cannot be solved in an optimal way. In this way, end-to-end methods lose the flexibility of not being able to obtain hash codes of images without first training the entire neural network. Our method however allows us to obtain hash codes quickly, easily, and with the flexibility to obtain hash codes directly from real embeddings. In our work, we demonstrate the erroneous conception of the belief of these years that, this technique is sub-optimal, and that, therefore very good results cannot be obtained with it. End-to-end methods state that since the error in hamming space is not reduced explicitly, the problem cannot be optimized properly. In this work it is shown that flexibility and high-performance is not a trade-off.

III. LEARNING REAL EMBEDDINGS

The purpose of this section is to explain the different network architectures and training processes that are adopted for each dataset. The objective is to obtain real embeddings of images such that they can be discriminated based on their Euclidean distance. The architecture as well as the training method are essential to achieve good results in followings steps with binary embeddings, therefore, different possibilities are evaluated.

A. MNIST

After evaluating different network architectures we came to the conclusion that the best option is a Siamese network architecture (figure 2) with a *contrastive* loss function [15]. We use the network in figure 1 as backbone. This network is the used in [25], but the last layer is suppressed to have embeddings of dimension 128. Note that a triplet architecture yield very similar results, but with the same results we prefer the lightweight architecture.

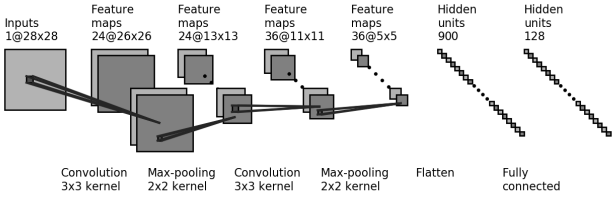

Fig. 1. Neural network used for training the MNIST dataset. [25]

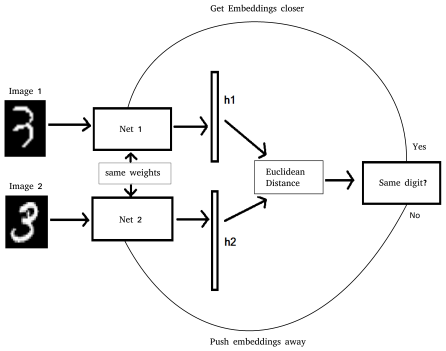

Fig. 2. Example of a Siamese network and its operation. h1 and h2 represent the embeddings of each image at the end of the network.

B. CIFAR-10

Different architectures are also evaluated for CIFAR-10 dataset. At the end we decide to use a feed-forward network with ResNetv2 50 [17] as a backbone. The whole Network with the last softmax layer is used for trainning with a Cateforical cross entropy loss. The trainable parameters are taken from [1]. For testing, we take embeddings from the last layer before the classification layer. The dimension of the extracted embeddings is 32,768.

C. PARTITION NORDLAND

For this dataset and following the work of [35], we adopt a triplet architecture (figure 3) with a *Wohlhart-Lepetit* loss. Note that Scene Recognition problem with drastic change in apareance is a much harder task and therefore a triplet arquitecture is needed to learn discriminative embeddings. We use the network VGG-16 [39] as backbone. The dimension of the embeddings extracted is 128.

IV. HASH CODINGS ON KENDALL'S TAU DISTANCE

Our goal is to learn a mapping function $\mathcal{F} : \mathcal{I} \rightarrow \mathbf{b}$, that transforms a RGB image $\mathcal{I} \in \{0, \ldots, 255\}^{w \times h \times 3}$ into a $q-$dimensonal binary vector $\mathbf{b} \in \mathbb{B}^q$. We will divide such function into two parts $\mathcal{F} = \mathcal{F}_2\left(\mathcal{F}_1\left(\mathcal{I}\right)\right)$.

Real-valued embedding. We will learn a function $\mathcal{F}_1 : \mathcal{I} \rightarrow \mathbf{d}$, being $\mathbf{d}$ a $p-$dimensonal vector $\mathbf{d} \in \mathbb{R}^p$. For this, we will use siamese and triplet losses.

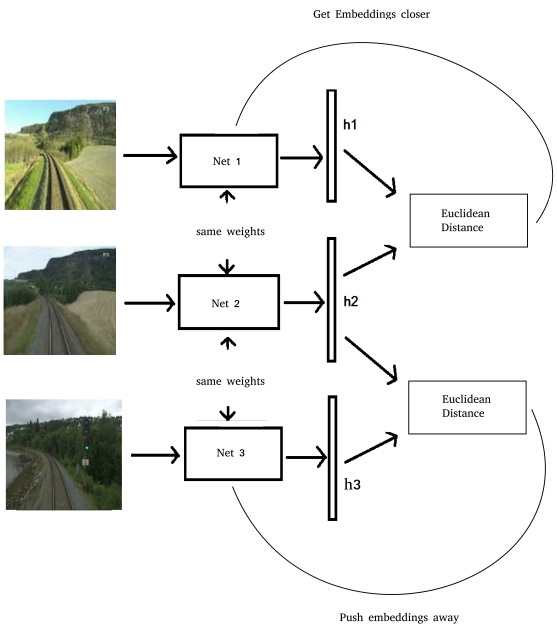

Fig. 3. Example of a triplet network and its operation. $h1$, $h2$ and $h3$ represent the embeddings of each image at the end of the network.

Binary comparisons. Here it lies our main contribution. $\mathcal{F}_2 : \mathbf{d} \rightarrow \mathbf{b}$ is a function that selects q different pairs $\{(x_1, y_1), \ldots, (x_i, y_i), \ldots, (x_q, y_q)\} : x_i, y_i \in \{1, \ldots, p\} \times \{1, \ldots, p\}, x_i \neq y_i$ of dimensions from the real-valued embedding $\mathbf{d}$ and outputs $\mathbf{b} = \begin{pmatrix} \tau_1 & \ldots & \tau_i & \ldots & \tau_q \end{pmatrix}^\top$, where τ_i is the result of the following binary test.

$$\tau_i = \tau(\mathbf{d}; x_i, y_i) = \begin{cases} 0 & \mathbf{d}(x_i) \leq \mathbf{d}(y_i) \\ 1 & \text{otherwise} \end{cases} \quad (1)$$

where $\mathbf{d}(x_i) = \mathbf{d}^\top \mathbf{e}_i$; $\mathbf{e}_i = \begin{pmatrix} 0 & \ldots & 0 & 1 & 0 & \ldots & 0 \end{pmatrix}^\top$ being the unit vector with the 1 in its entry x_i.

As such binary tests are non-differentiable, we will not learn in this work any of their parameters and the pairs (x_i, y_i) will be defined randomly. Extending the derivations done for image patches in [54] to image embeddings, our random tests $\tau(\mathbf{d}; x_i, y_i)$ approximate the Kendall tau distance between two embeddings. Let consider a binary descriptor composed of all $\binom{p}{2}$ binary comparisons between two real-valued descriptors $\mathbf{d}_1$ and $\mathbf{d}_2$.

Binarization by comparison of random pairs performed directly on the real embeddings, without performing PCA first, has been found not to offer good results. The reason is that the variance that explains the values of each dimension is evenly distributed among all dimensions, and therefore comparing individual dimensions is not very useful, since, in this case, the embedding represents the image correctly when it is considered completely. In the pairwise comparison, most yield wrong results, which undermines the results of the method.

A. Why does this work?

PCA is able to order the dimensions of the real embeddings according to the variance that each one represents, thus, when comparing different images, the variation of the dimensions of the real embedding in the first dimensions will differ significantly, and therefore, when comparing the dimensions to each other will result in binary codes with a large distance. On the contrary, if the images belong to the same class they will have similar real embeddings and comparisons will yield similar results after the comparison of the dimensions of the real embeddings of two images. In the figure 4 we represent the intuition of the hash model.It can be seen how the images that are the same but in different seasons have similar real embeddings and therefore a small Euclidean distance. When the embedding is binarized, the distance is even smaller. On the contrary, the distance between different images is also greater using binary codes than continuous ones. Therefore, the loss of information is minimal because we take advantage of the comparison characteristics of the image retrieval problem.

Although the comparisons are random, they are the same for all images. Therefore, although the embedding loses the ability to explain image information in the first instance due to the transition from continuous to binary, the precision is maintained when comparing the distance of the embedding of each image with the others.

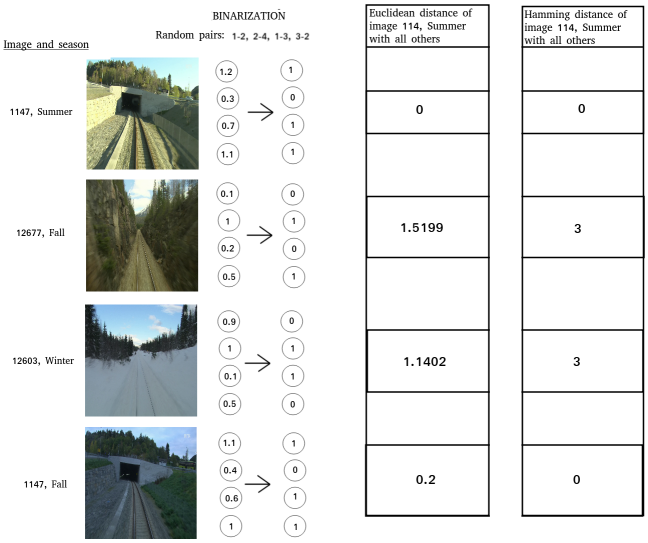

Fig. 4. Intuition of the binarization function. Calculation of Euclidean and Hamming distances .

B. Design of binarization method

1) PCA: In this section the mean average precision is compared by applying PCA to reduce the dimension of the real embedding, and without applying PCA, in MNIST and CIFAR-10 database.

It can be seen in the case of CIFAR-10 that the results without PCA take values close to 0.1, which is the result that a random algorithm would obtain in this 10-category

TABLE I

PERFORMANCE COMPARISON (MAP @ 1000, %) OF THE BINARIZATION ALGORITHMS IN THE MNIST AND CIFAR-10 DATA SET. THIS TABLE SHOWS THE MEAN MEAN PRECISION (MAP) OF THE TOP 1,000 RETURNED IMAGES WITH RESPECT TO DIFFERENT NUMBERS OF HASH BITS AND DEPENDING ON WHETHER OR NOT PCA IS APPLIED BEFORE THE BINARIZATION FUNCTION.

Dataset	PCA	16 bit	32 bit	64 bit
MNIST	No	0,512	0,536	0,484
MNIST	Yes	0.9916	0,992	0,9908
CIFAR-10	No	0,11	0,13	0,11
CIFAR-10	Yes	0,8455	0,9097	0,8992

problem. That is, the embeddings without applying PCA do not provide any further information for the task.

When applying PCA the dimensions are ordered in such a way that the former explain more variance than the latter, in this way each dimension contains relevant information from the image and the comparison of dimensions offers discriminating results to differentiate whether two images belong to the same class or not.

2) Dimension of real embedding and way of comparison: Once the method has been shown to work better by performing PCA before the comparison step, the next question is, how much should we reduce the dimension of the real embedding before doing the comparison step? It must be taken into account that there is a minimum dimension to which it can be reduced according to the dimension of the binary code that we want. Thus, if we want, for example, a 32-bit binary code, the minimum dimension to which we can reduce the real embedding is 9, with which we can make up to 36 comparisons without repeating any.

Comparisons can be random, or assigning probabilities to the dimensions according to the importance we assign to it.

The first method that is proposed consists in performing comparisons with an exponential probability distribution, thus we assign greater probability to take the first dimensions. This makes sense because PCA orders the dimensions in a way that the former explain more variance.

The second method proposed in making the comparisons in a random way.

In the following the results will be shown the binarization by random comparison and with exponential function with different dimensions of the real embedding.

TABLE II

PERFORMANCE COMPARISON (MAP @ 1000, %) OF THE BINARIZATION ALGORITHMS IN THE CIFAR-10 DATA SET FOR A 64-BIT BINARY EMBEDDING. THIS TABLE SHOWS THE MEAN MEAN PRECISION (MAP) OF THE TOP 1,000 RETURNED IMAGES WITH RESPECT TO DIFFERENT DIMENSIONS OF THE ACTUAL EMBEDDING AND COMPARISON METHOD.

Dimension	Random	Exponential
16	0,8101	0,8167
14	0,8569	0,8514
12	0,9097	0,8998

The best results are obtained for the lowest possible dimension (12 in the case of 64 bits of the binary embedding) and for a random dimension comparison. Intuitively, this is explained since the smaller the dimension of the real embedding, the more variance it will explain each dimension individually and on the other hand, the randomness causes greater diversity in the compared pairs.

V. EXPERIMENTAL RESULTS

We evaluate the performance of the proposed method on three different datasets in the category of object recognition and scene recognition, so that general conclusions about the method can be extracted. They are MNIST, CIFAR-10 and PARTITIONED NORDLAND.

A. Datasets and metrics

MNIST [26]: includes 70,000 28×28 scale images of hand written digits (0-9) across 10 classes. We follow the common setting [9] and evaluate our method with mAP@all. For that, we select 1,000 images (100 per class) as the query set. The remaining 69,000 images are regarded as the database.

CIFAR-10 [22]: contains 60,000 32×32 colored images balanced across 10 classes (i.e. airplane, automobile,bird, cat, deer, dog, frog, horse, ship and truck). We follow the common setting [38] and evaluate our method with mAP@all. For that, we select 1,000 images (100 per class) as the query set. The remaining 59,000 images are regarded as the database.

PARTITIONED NORDLAND [35]: The Nordland railway videos were used to build this database. The Norwegian Broadcasting Company (NRK) made a documentary about the Nordland Railway, a railway line between the cities of Trondheim and Bodø. They filmed the 729 km journey with a camera in front of the train in winter, spring, autumn and summer.

There is a test set with three different sequences of 1,150 images (a total of 3,450, in yellow in the figure 5). The rest of the images are training images (24,569, in red in the figure 5). By using multiple sections, you increase the variety of places and appearance changes contained in the test set. A separation of a few kilometers is also left between each test and train section, discarding some images to guarantee the difference between the test and train data.

Given the similarity between consecutive images, in this work, and following [35], it is considered that two images are from the same place if they are temporarily separated by 3 images or less. We apply a sliding window of 5 images over the entire data set to group the images taken from five consecutive seconds.

To evaluate the recognizer, images of places in one season of the year are used as a reference and in another season as input. The closest places are noted for each input image and the following metric is obtained:

Fraction of correct places (fc): It is the number of times that the place predicted by the system coincides with the input place, that is:

$$fc = \frac{\text{Number of places hit}}{\text{Number of places evaluated}} \tag{2}$$

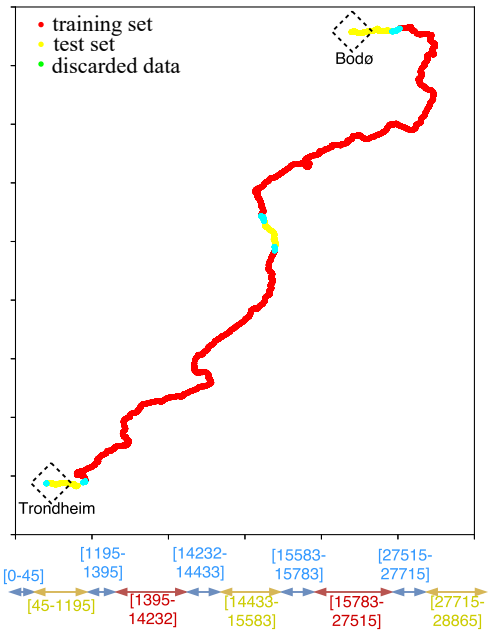

Fig. 5. Separation of training and test sets.[35]

B. Comparison with existing methods

Our method is compared with various widely recognized hashing baselines, including, DeepBit [28], UTH [20], HashGAN [9], HashGAN [9],JMLH [38],ITQ [12], PGDH [50], SDH [37], KSH [32], or GreedyHash [43].

TABLE III
PERFORMANCE COMPARISON (MAP, %) OF DIFFERENT BINARIZATION ALGORITHMS ON THE MNIST DATASET. THIS TABLE SHOWS THE MEAN AVERAGE PRECISION (MAP@ALL) WITH RESPECT TO DIFFERENT NUMBER OF HASH BITS.

Method	Reference	16 bit	32 bit	64 bit
DeepBit [28]	CVPR16	28.18	32.02	44.53
LSH [10]	VLDB00	42.10	50.45	66.23
UTH [21]	ACM17	43.15	46.58	49.88
SphH [19]	CVPR12	52.97	65.45	65.45
SpeH [46]	CVPR12	59.72	64.37	67.60
ITQ [12]	PAMI13	70.06	76.86	80.23
HashGAN [9]	CVPR18	94.31	95.48	96.37
Ours	Proposed	**99.16**	**99.20**	**99.08**

Our method achieves state-of-the-art results in both datasets i.e MNIST and CIFAR-10 . Interestingly, the performance (mAP) does not increase with a higher number of hash bits as our first intuition would suggest. Due to the characteristics of the method, when we compare real embeddings in high dimensions, comparisons do not tell us much more information because the dimensions of the embeddings are ordered according to the variance that they explain thanks to PCA. So, having more dimensions than the strictly needed shrinks a little bit the performance, although this is not very pronounced.

In the case of the PARTITION NORDLAND dataset and due to the lack of hashing method addressed with this dataset, we compare our results directly with the state-or-the-art with real embeddings [35].

TABLE IV
PERFORMANCE COMPARISON (MAP, %) OF DIFFERENT BINARIZATION ALGORITHMS ON THE CIFAR-10 DATASET. THIS TABLE SHOWS THE MEAN AVERAGE PRECISION (MAP@ALL) WITH RESPECT TO DIFFERENT NUMBER OF HASH BITS.

Method	Reference	16 bit	32 bit	64 bit
DGH [30]	NIPS14	19.9	20.0	21.2
ITQ [12]	PAMI13	20.1	20.7	23.5
AGH [33]	ICML11	21.7	20.5	18.2
KSH [32]	CVPR12	45.1	47.3	50.7
ITQ-CCA [13]	TPAMI12	46.3	49.8	50.5
CNNH [47]	AAAI14	45.3	50.9	53.7
SDH [37]	CVPR15	49.9	52.5	54.6
DNNH [24]	CVPR15	55.6	55.8	59.9
DHN [51]	AAAI16	56.4	60.3	62.6
HashNet [5]	TPAMI17	64.3	67.5	68.7
HashGAN [4]	CVPR18	66.8	73.1	74.9
PGDH [50]	ECCV18	74.1	74.7	76.2
MIHash [2]	ICCV17	76.0	77.6	76.1
GreedyHash [43]	NIPS18	78.6	81.0	83.3
JMLH [38]	IEEE19	80.5	84.1	83.7
Ours	Proposed	**84.55**	**90.97**	**89.92**

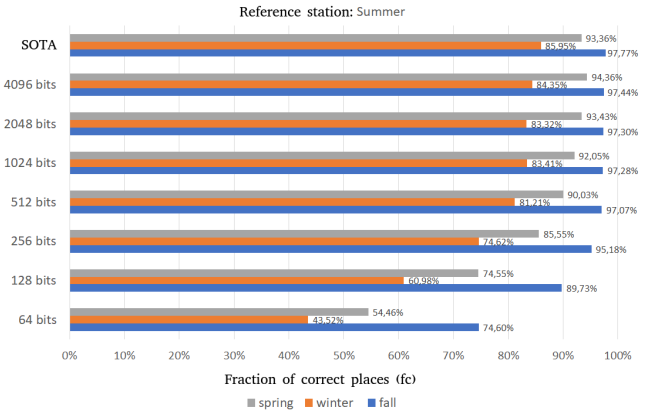

Fig. 6. Results with summer as the reference season for different dimensions of hash bits and, in the first row, the State-Of-the-Art (SOTA) results with real embeddings

In figures 6 and 7 We can observe that our results outperform in some cases the SOTA results with real embeddings in PARTITION NORDALAND dataset. That means that our method is not only able to learn binary embeddings for simple tasks such as object and handwritten recognition, but also for more complex scene. recognition tasks.

VI. CONCLUSIONS

In this work, we proposed a simple but very effective hash model with state-of-the-art results in MNIST and CIFAR-10 dataset. Our model relies in comparisons of the dimensions of real embeddings to obtain binary codes. The flexibility of the model allow us to apply the hash model to speed up any task in which real embeddings are already obtained and we want to binarize them. We showed that no complex models are needed to obtain high-quality binary codes.

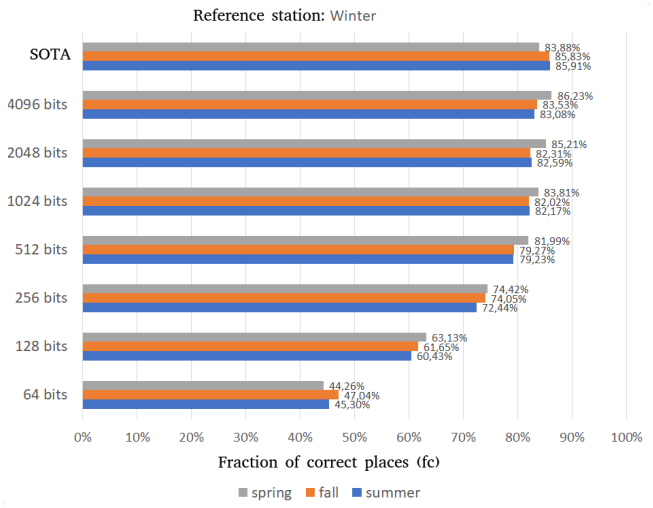

Fig. 7. Results with winter as the reference season for different dimensions of hash bits and, in the first row, the State-Of-the-Art (SOTA) results with real embeddings.

We also propose benchmark results for the non-addressed-yet problem of Scene recognition under weather-variance with binary codes. We obtained similar results to the state-of-the-art with real embeddings, so, robustness regarding difficult tasks is achieved.

REFERENCES

[1] MS Windows NT kernel description. `https://storage.googleapis.com/bit_models/`. Accessed: 2020-05-20.

[2] F. Cakir, K. He, S. A. Bargal, and S. Sclaroff. Mihash: Online hashing with mutual information. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 437–445, 2017.

[3] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua. Brief: Computing a local binary descriptor very fast. *IEEE transactions on pattern analysis and machine intelligence*, 34, 11 2011.

[4] Y. Cao, B. Liu, M. Long, and J. Wang. Hashgan: Deep learning to hash with pair conditional wasserstein gan. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1287–1296, 2018.

[5] Z. Cao, M. Long, J. Wang, and P. S. Yu. Hashnet: Deep learning to hash by continuation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5609–5618, 2017.

[6] T.-T. Do, A.-D. Doan, and N.-M. Cheung. Learning to hash with binary deep neural network. volume 9909, pages 219–234, 10 2016.

[7] A. Dosovitskiy and T. Brox. Generating images with perceptual similarity metrics based on deep networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 658–666. Curran Associates, Inc., 2016.

[8] Y. Duan, J. Lu, Z. Wang, J. Feng, and J. Zhou. Learning deep binary descriptor with multi-quantization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1183–1192, 2017.

[9] K. Ghasedi Dizaji, F. Zheng, N. Sadoughi, Y. Yang, C. Deng, and H. Huang. Unsupervised deep generative adversarial hashing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3664–3673, 2018.

[10] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. *Proceeding VLDB '99 Proceedings of the 25th International Conference on Very Large Data Bases*, 99, 05 2000.

[11] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *CVPR 2011*, pages 817–824, 2011.

[12] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2916–2929, 2013.

[13] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2916–2929, 2013.

[14] J. Guo, S. Zhang, and J. Li. Hash learning with convolutional neural networks for semantic based image retrieval. pages 227–238, 04 2016.

[15] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, CVPR '06, page 1735–1742, USA, 2006. IEEE Computer Society.

[16] K. He, F. Wen, and J. Sun. K-means hashing: An affinity-preserving quantization method for learning binary compact codes. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2938–2945, 2013.

[17] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[18] J. Heo, Y. Lee, J. He, S. Chang, and S. Yoon. Spherical hashing. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2957–2964, 2012.

[19] J. Heo, Y. Lee, J. He, S. Chang, and S. Yoon. Spherical hashing. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2957–2964, 2012.

[20] S. Huang, Y. Xiong, Y. Zhang, and J. Wang. Unsupervised triplet hashing for fast image retrieval. 02 2017.

[21] S. Huang, Y. Xiong, Y. Zhang, and J. Wang. Unsupervised triplet hashing for fast image retrieval. 02 2017.

[22] A. Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

[23] H. Lai, Y. Pan, Y. Liu, and S. Yan. Simultaneous feature learning and hash coding with deep neural networks. 04 2015.

[24] H. Lai, Y. Pan, Y. Liu, and S. Yan. Simultaneous feature learning and hash coding with deep neural networks. 04 2015.

[25] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.

[26] Y. LeCun and C. Cortes. MNIST handwritten digit database. 2010.

[27] W.-J. Li, S. Wang, and W.-C. Kang. Feature learning based deep supervised hashing with pairwise labels. 11 2015.

[28] K. Lin, J. Lu, C.-S. Chen, and J. Zhou. Learning compact binary descriptors with unsupervised deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1183–1192, 2016.

[29] K. Lin, H.-F. Yang, J.-H. Hsiao, and C.-S. Chen. Deep learning of binary hash codes for fast image retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 27–35, 2015.

[30] W. Liu, C. Mu, S. Kumar, and S.-F. Chang. Discrete graph hashing. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3419–3427. Curran Associates, Inc., 2014.

[31] W. Liu, J. Wang, R. Ji, Y. Jiang, and S. Chang. Supervised hashing with kernels. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2074–2081, 2012.

[32] W. Liu, J. Wang, R. Ji, Y. Jiang, and S. Chang. Supervised hashing with kernels. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2074–2081, 2012.

[33] W. Liu, J. Wang, S. Kumar, and S. Chang. Hashing with graphs. pages 1–8, 01 2011.

[34] X. Liu, X. Nie, Q. Zhou, and Y. Yin. Supervised discrete hashing with mutual linear regression. In *Proceedings of the 27th ACM International Conference on Multimedia*, MM '19, page 1561–1568, New York, NY, USA, 2019. Association for Computing Machinery.

[35] D. Olid, J. M. Fácil, and J. Civera. Single-view place recognition under seasonal changes. In *PPNIV Workshop at IROS 2018*, 2018.

[36] G. Shakhnarovich. *Learning Task-Specific Similarity*. PhD thesis, USA, 2005.

[37] F. Shen, C. Shen, W. Liu, and H. T. Shen. Supervised discrete hashing. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 37–45, 2015.

[38] Y. Shen, J. Qin, J. Chen, L. Liu, F. Zhu, and Z. Shen. Embarrassingly simple binary representation learning. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[39] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[40] J. Song. Binary generative adversarial networks for image retrieval. 08 2017.

[41] C. Strecha, A. Bronstein, M. Bronstein, and P. Fua. Ldahash: Improved matching with smaller descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(1):66–78, 2012.

[42] C. Strecha, A. Bronstein, M. Bronstein, and P. Fua. Ldahash: Improved matching with smaller descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(1):66–78, 2012.

[43] S. Su, C. Zhang, K. Han, and Y. Tian. Greedy hash: Towards fast optimization for accurate hash coding in cnn. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 798–807. Curran Associates, Inc., 2018.

[44] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for scalable image retrieval. In *IEEE Conf on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[45] X. Wei, Y. Zhang, Y. Gong, and N. Zheng. Kernelized subspace pooling for deep local descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1867–1875, 2018.

[46] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1753–1760. Curran Associates, Inc., 2009.

[47] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan. Supervised hashing for image retrieval via image representation learning. *Proceedings of the National Conference on Artificial Intelligence*, 3:2156–2162, 01 2014.

[48] H. Yang, K. Lin, and C. Chen. Supervised learning of semantics-preserving hash via deep convolutional neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(2):437–451, 2018.

[49] Y. Yang, Y. Luo, W. Chen, F. Shen, J. Shao, and H. Shen. Zero-shot hashing via transferring supervised knowledge. pages 1286–1295, 06 2016.

[50] X. Yuan, L. Ren, J. Lu, and J. Zhou. Relaxation-free deep hashing via policy gradient. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[51] H. Zhu, M. Long, J. Wang, and Y. Cao. Deep hashing network for efficient similarity retrieval. In *AAAI*, 2016.

[52] M. Zieba, P. Semberecki, T. El-Gaaly, and T. Trzcinski. Bingan: Learning compact binary descriptors with a regularized gan. In *Advances in Neural Information Processing Systems*, pages 3608–3618, 2018.

[53] M. Zieba, P. Semberecki, T. El-Gaaly, and T. Trzciński. Bingan: Learning compact binary descriptors with a regularized gan. *ArXiv*, abs/1806.06778, 2018.

[54] A. Ziegler, E. Christiansen, D. Kriegman, and S. J. Belongie. Locally uniform comparison image descriptor. In *Advances in Neural Information Processing Systems*, pages 1–9, 2012.