# Hashing with Binary Autoencoders

**Miguel Á. Carreira-Perpiñán**     **Ramin Raziperchikolaei**
EECS, University of California, Merced
http://eecs.ucmerced.edu

**Introduction.**   We consider the problem of binary hashing, where given a high-dimensional vector $\mathbf{x} \in \mathbb{R}^D$, we want to map it to an $L$-bit vector $\mathbf{z} = \mathbf{h}(\mathbf{x}) \in \{0,1\}^L$ using a hash function $\mathbf{h}$, while preserving the neighbors of $\mathbf{x}$ in the binary space. Binary hashing has emerged in recent years as an effective technique for fast search on image (and other) databases. While the search in the original space would cost $\mathcal{O}(ND)$ in both time and space, using floating point operations, the search in the binary space costs $\mathcal{O}(NL)$ where $L \ll D$ and the constant factor is much smaller. This is because the hardware can compute binary operations very efficiently and the entire dataset ($NL$ bits) can fit in the main memory of a workstation.

Many different hashing approaches have been proposed in the last few years. They formulate an objective function of the hash function $\mathbf{h}$ or of the binary codes that tries to capture some notion of neighborhood preservation. All these approaches have in common two things: $\mathbf{h}$ performs dimensionality reduction ($L < D$) and, as noted, it outputs binary codes ($\mathbf{h} \colon \mathbb{R}^D \to \{0,1\}^L$). The latter implies a step function or binarization applied to a real-valued function of the input $\mathbf{x}$. Optimizing this is difficult. In practice, most approaches follow a two-step procedure: first they learn a real hash function ignoring the binary constraints and then the output of the resulting hash function is binarized. This procedure can be seen as a "filter" approach [5] and is suboptimal. To obtain the optimal solution, we must optimize the objective function jointly over mappings and thresholds, respecting the binary constraints while learning $\mathbf{h}$; this is a "wrapper" approach. In this paper we show that this joint optimization can actually be carried out reasonably efficiently.

**Our hashing models.**   We consider a well-known model for continuous dimensionality reduction, the (continuous) autoencoder, defined in a broad sense as the composition of an encoder $\mathbf{h}(\mathbf{x})$ which maps a real vector $\mathbf{x} \in \mathbb{R}^D$ onto a real code vector $\mathbf{z} \in \mathbb{R}^L$ (with $L < D$), and a decoder $\mathbf{f}(\mathbf{z})$ which maps $\mathbf{z}$ back to $\mathbb{R}^D$ in an effort to reconstruct $\mathbf{x}$. For hashing, the encoder maps continuous inputs onto *binary* code vectors with $L$ bits, $\mathbf{z} \in \{0,1\}^L$, and we call it a *binary autoencoder (BA)*. Our desired hash function will be the encoder $\mathbf{h}$, and it should minimize the following function, given a dataset of high-dimensional patterns $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_N)$:

$$E_{\text{BA}}(\mathbf{h}, \mathbf{f}) = \sum_{n=1}^{N} \|\mathbf{x}_n - \mathbf{f}(\mathbf{h}(\mathbf{x}_n))\|^2$$

which is the usual least-squares error but where the code layer is binary. We will also consider a related model:

$$E_{\text{BFA}}(\mathbf{Z}, \mathbf{f}) = \sum_{n=1}^{N} \|\mathbf{x}_n - \mathbf{f}(\mathbf{z}_n)\|^2 \quad \text{s.t.} \quad \mathbf{z}_n \in \{0,1\}^L, \ n = 1, \dots, N$$

where $\mathbf{f}$ is linear and we optimize over the decoder $\mathbf{f}$ and the binary codes $\mathbf{Z} = (\mathbf{z}_1, \dots, \mathbf{z}_N)$ of each input pattern. We call this model (least-squares) *binary factor analysis (BFA)*. A hash function $\mathbf{h}$ can be obtained from BFA by fitting a binary classifier of the inputs to each of the $L$ code bits. It is a filter approach, since it first learns $\mathbf{Z}$ and then $\mathbf{h}$, while the BA is an optimal (wrapper) approach, since it optimizes jointly over $\mathbf{f}$ and $\mathbf{h}$.

We use the recently proposed *method of auxiliary coordinates (MAC)* [1, 2] for optimization of BA and BFA. The idea is to break nested functional relationships judiciously by introducing variables as equality constraints, turning them into penalties and applying alternating optimization. We introduce
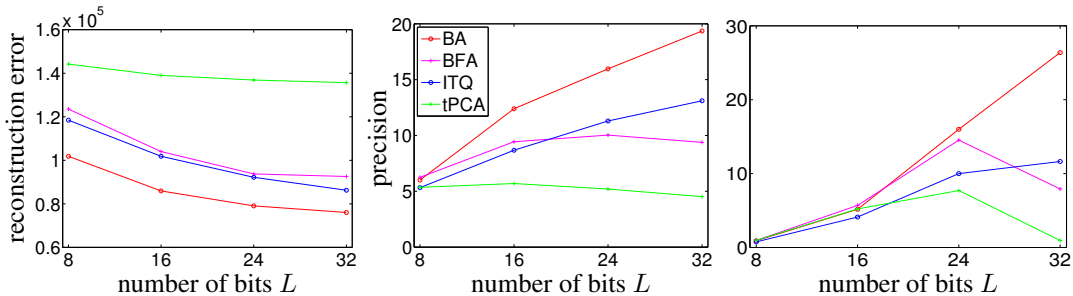
Figure 1: Wrapper vs filter optimization. *Left*: BA objective function. *Middle*: precision returning neighbors within Hamming distance $r = 2$. *Right*: precision returning $k = 50$ nearest neighbors.

as auxiliary coordinates the outputs of $\mathbf{h}$, i.e., the codes for each of the $N$ input patterns, and obtain the following equality-constrained problem:

$$\min_{\mathbf{h},\mathbf{f},\mathbf{Z}} \sum_{n=1}^{N} \|\mathbf{x}_n - \mathbf{f}(\mathbf{z}_n)\|^2 \quad \text{s.t.} \quad \mathbf{z}_n = \mathbf{h}(\mathbf{x}_n), \ \mathbf{z}_n \in \{0,1\}^L, \ n = 1, \ldots, N.$$

Note the codes are binary. We now apply the quadratic-penalty method and minimize the following objective function while progressively increasing $\mu$, so the constraints are eventually satisfied:

$$E_Q(\mathbf{h},\mathbf{f},\mathbf{Z};\mu) = \sum_{n=1}^{N} \left( \|\mathbf{x}_n - \mathbf{f}(\mathbf{z}_n)\|^2 + \mu \|\mathbf{z}_n - \mathbf{h}(\mathbf{x}_n)\|^2 \right) \text{ s.t. } \mathbf{z}_n \in \{0,1\}^L, \ n = 1, \ldots, N.$$

Now we apply alternating optimization over $\mathbf{Z}$ and $(\mathbf{h},\mathbf{f})$. This results in the following two steps:

- Over $\mathbf{Z}$ for fixed $(\mathbf{h},\mathbf{f})$, the problem separates for each of the $N$ codes. The optimal code vector for pattern $\mathbf{x}_n$ tries to be close to the prediction $\mathbf{h}(\mathbf{x}_n)$ while reconstructing $\mathbf{x}_n$ well. This binary optimization has the form of a binary proximal operators over few variables ($L$), so it can be solved exactly by enumeration, or approximately by alternating optimization.

- Over $(\mathbf{h},\mathbf{f})$ for fixed $\mathbf{Z}$, we obtain $L + 1$ independent problems for each of the $L$ single-bit hash functions and for $\mathbf{f}$.

We can now see the advantage of the auxiliary coordinates: the individual steps are (reasonably) easy to solve, and besides they exhibit significant parallelism. The resulting algorithm alternates a step over the encoder ($L$ classifications) and decoder (one regression) with a step over the codes ($N$ binary proximal operators).

**Experiments.** Fig. 1 shows the gain obtained by the MAC optimization, which respects the binary constraints, over the suboptimal, "filter" approach of relaxing the constraints (i.e., PCA) and then binarizing the result by thresholding at 0 (tPCA) or by optimal rotation (ITQ [4]). To compute the reconstruction error for tPCA and ITQ we find the optimal mapping $\mathbf{f}$ given their binary codes. We use the NUS-WIDE-LITE dataset [3], which contains $N = 27\,807$ images for training and $27\,808$ images for test. We use as ground truth the 50 true nearest neighbors and report results over a range of $L = 8$ to 16 bits. We can see that BA dominates all other methods in reconstruction error, as expected, and also in precision. This demonstrates that a better optimization of the objective significantly improves the hash function learned.

# References

[1] M. Á. Carreira-Perpiñán and W. Wang. Distributed optimization of deeply nested systems. Unpublished manuscript, arXiv:1212.5921, Dec. 24 2012.

[2] M. Á. Carreira-Perpiñán and W. Wang. Distributed optimization of deeply nested systems. *AISTATS*, 2014.

[3] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y.-T. Zheng. NUS-WIDE: A real-world web image database from National Univ. Singapore. In *Proc. ACM Conf. Image and Video Retrieval (CIVR)*, 2009.

[4] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin. Iterative quantization: A Procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Trans. PAMI*, 35(12):2916–2929, Dec. 2013.

[5] R. Kohavi and G. H. John. The wrapper approach. In H. Liu and H. Motoda, editors, *Feature Extraction, Construction and Selection. A Data Mining Perspective*. Springer-Verlag, 1998.