Self-Supervised Bernoulli Autoencoders for Semi-Supervised Hashing

R. Ñanculef[1], F. Mena[1], **A. Macaluso**[2], S. Lodi[2] and C. Sartori[2]

[1] Department of Informatics, Federico Santa María Technical University, Chile
[2] Department of Computer Science and Engineering, University of Bologna, Italy.

CIARP 2021, Porto.
May 10-13 2021.

Background & State of the Art

Similarity Search

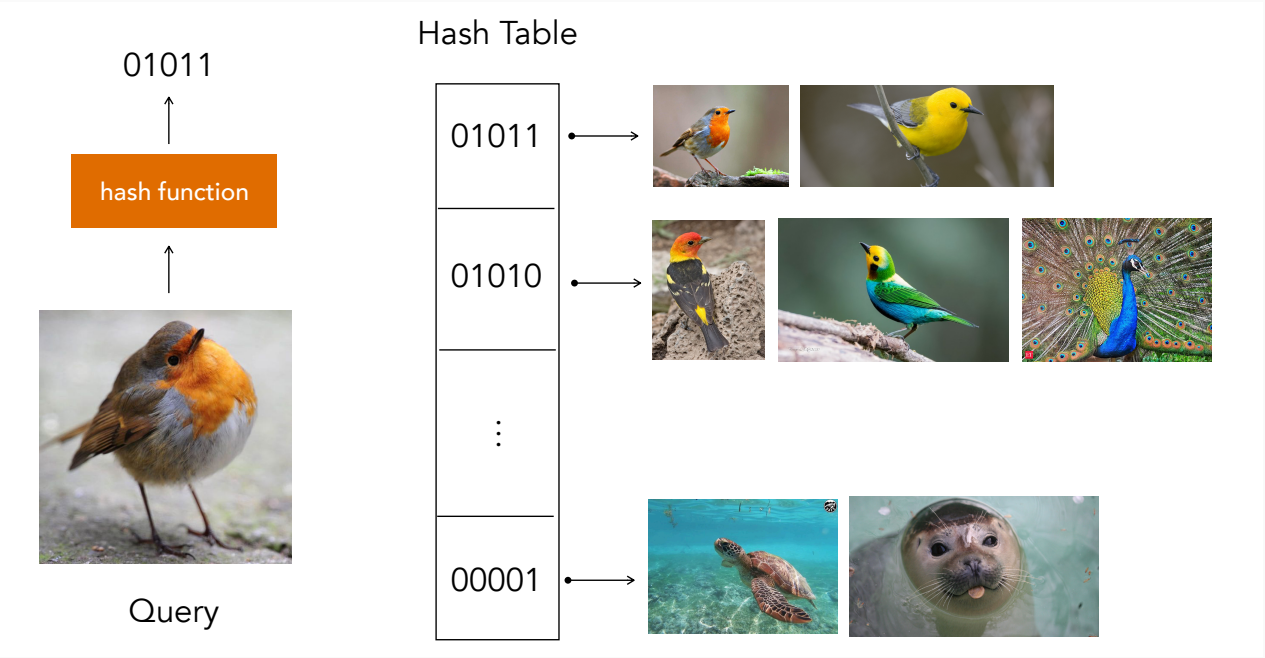
- Given a dataset $D = \{x^{(1)}, x^{(2)}, \ldots, x^{(N)}\}$, with $x^{(\ell)} \in \mathbb{X}\ \forall \ell \in [N]$, *similarity search* is the problem of finding the elements of D that are *similar* to a sample object $q \in \mathbb{X}$

Big Data Challenges

- Traditional methods for similarity search (e.g KD-trees, and R-trees) degrade quickly as the dimensionality of the data descriptors increases
- Our capacity to collect and store high-dimensional data such as images, audio, and text has growth explosively

Semantic Hashing

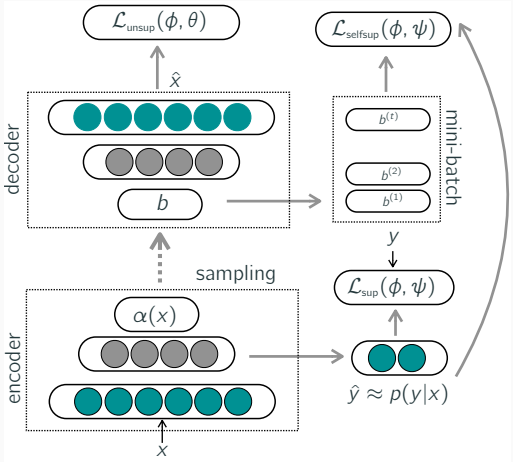
- Semantic hashing indexes use similarity-preserving binary codes $h(x) \in \{0,1\}^B$: similar objects can be accessed by looking at nearby positions in a hash table



- The index can easily fit in memory
- Constant time algorithms have been devised, but search is efficient even using a linear/exhaustive scan [4]

Variational Autoencoders

- Goal: learn a hash function using a specific architecture of neural networks (Autoencoders)



- The map $x \to h(x)$ is stochastic (non-deterministic): this favours generalization and enhance effective collisions (similar codes should produce similar data)

Bernoulli Variational Autoencoders

- The first method in this family used a Gaussian latent variables [1]
- CIARP's 2019 paper [3] showed that Bernoulli latent variables were more effective for hashing. The method was fully unsupervised

- **Theoretical Contribution:** We further extend the autoencoder's objective function using self-supervision. This makes the model more robust to the lack of annotations
- **Practical Contribution:** We are the first work studying the effect of supervision (and self-supervision) in Bernoulli autoencoders under varying levels of supervision

Method

Proposed Training Function

- We define as $q_\phi(b|x)$ the distribution/map implemented by the *encoder* (a neural net) and as $p_\theta(x|b)$ the distribution/map implemented by the *decoder* (another neural net)
- If $S = \{x^{(1)}, \ldots, x^{(n)}\} \subset D$ denotes the input data available for training, we assume the existence of annotations (sets of tags) $y^{(\ell)} \subset \mathbb{Y} = \{t_1, \ldots, t_K\}$ for the first $s \ll n$ objects
- Our method trains the autoencoder using an objective function

$$\mathcal{L} = \sum_{\ell=1}^{n} \mathcal{L}_{\text{unsup}}^{(\ell)} + \beta \sum_{\ell=1}^{s} \mathcal{L}_{\text{sup}}^{(\ell)} + \alpha \sum_{\ell,\ell'=1}^{n} \mathcal{L}_{\text{selfsup}}^{(\ell,\ell')}, \qquad (1)$$

with three different loss functions $\mathcal{L}_{\text{unsup}}^{(\ell)}, \mathcal{L}_{\text{sup}}^{(\ell)}$ and $\mathcal{L}_{\text{selfsup}}^{(\ell,\ell')}$, combined using hyper-parameters $\alpha, \beta \in \mathbb{R}$

Unsupervised Training

- As in any Bernoulli autoencoder [3], the unsupervised loss $\mathcal{L}^{(\ell)}_{\text{unsup}}$ takes the form:

$$\begin{aligned}\mathcal{L}^{(\ell)}_{\text{unsup}} &= \mathbb{E}_{q_\phi(b|x^{(\ell)})}\left[-\log p_\theta(x^{(\ell)}, b) + \log q_\phi(b|x^{(\ell)})\right] \\ &= \underbrace{\mathbb{E}_{q_\phi(b|x^{(\ell)})}\left[-\log p_\theta(x^{(\ell)}|b)\right]}_{\mathcal{L}_1} + \lambda \underbrace{D_{\text{KL}}\left(q_\phi(b|x^{(\ell)})||p_\theta(b)\right)}_{\mathcal{L}_2}.\end{aligned} \tag{2}$$

- The term $\mathcal{L}_1$ measures the expected error in the reconstruction of x from the hash code b. The term $\mathcal{L}_2$ measures the divergence between the encoder's distribution and a non-informative prior $\text{Ber}(0.5)\ \forall i \in [B]$ which favours balanced hash tables
- This objective function does not require/leverage labels/annotations

Supervised Training

- We assume that any available annotation $y^{(\ell)}$ has been encoded as a one-hot vector such that $y_j^{(\ell)} = 1$ if $x^{(\ell)} \in S$ has the label/tag t_j and $y_j^{(\ell)} = 0$ otherwise. This can accommodate multi-label scenarios
- The second component in our objective function asks the model to reconstruct these vectors using an auxiliary decoder $\hat{y}^{(\ell)} = p_\psi(y|x^{(\ell)})$

$$\mathcal{L}_{\text{sup}}^{(\ell)} = -\mathbb{E}\left[y^{(\ell)} \log p_\psi(y|x^{(\ell)})\right] = -\sum\nolimits_k y_k^{(\ell)} \log \hat{y}_k^{(\ell)}. \qquad (3)$$

- This objective function has been used in [1] and [2] to improve the performance of the hash function

Pairwise Supervision

- Another way to leverage annotations in the training process is asking the model to assign similar hash codes $b^{(\ell)}$ and $b^{(\ell')}$ to pairs of examples $x^{(\ell)}, x^{(\ell')}$ with similar annotations $y^{(\ell)}, y^{(\ell')}$, e.g. considering the loss function

$$\mathcal{L}_{\text{pair}}^{(\ell,\ell')} = \; y^{(\ell)T} y^{(\ell')} D_{\ell,\ell'}^{+} - (1 - y^{(\ell)T} y^{(\ell')}) D_{\ell,\ell'}^{-} \; , \tag{4}$$

  where $D^{\pm}$ denote distances in the Hamming space.

- This loss has been used in [2] and many previous works to improve hashing. Unfortunately, it suffers strongly the lack of annotations: if the fraction of labelled points is $\rho < 1$, the fraction of pairs that can be evaluated in (4) reduces to $\rho^2 \ll 1$

Self-Supervised Training

- We propose to replace the pairwise loss by a self-supervised learning mechanism in which the ground-truth labels are substituted by the model's predictions themselves $\hat{y}(z;\psi)$, i.e. using the loss

$$\mathcal{L}_{\text{selfsup}}^{(\ell,\ell')} = \hat{y}^{(\ell)T}\hat{y}^{(\ell')}D_{\ell,\ell'}^{+} - (1-\hat{y}^{(\ell)T}\hat{y}^{(\ell')})D_{\ell,\ell'}^{-} \tag{5}$$

- It is not hard to see that this loss function penalizes inconsistent predictions (the model predicts the same annotations but assigns different hash codes) and rewards consistent predictions (the model predicts the same annotations and assigns similar hash codes)
- As this loss does not require ground-truth annotations, it makes the model more robust to the lack of annotations

An Implementation Detail

Input: A set $S = \{x^{(1)}, \ldots, x^{(n)}\}$ and semantic labels $y^{(\ell)}$ for the first s.
Output: Trained parameters ϕ, θ, ψ.

Initialize ϕ, θ, ψ;
while *not converged* **do**
  Randomly split S into n/M batches of size M;
  foreach *mini-batch* B_j **do**
    Predict $\hat{y}^{(\ell)}$ for any $x^{(\ell)} \in B_j$;
    Average the gradients of (2) and (5) w.r.t. ϕ, θ, ψ among all the examples in B_j;
    Average the gradient of (3) w.r.t. ψ among the labelled examples in B_j;
    Perform backpropagation updates for ϕ, θ, ψ;
  end foreach
end while

For efficiency concerns, we can form the pairs required for the self-supervised loss at a mini-batch level. In this way, the computational cost of the algorithm reduces from quadratic to linear in n

Results

Dataset and Methods

- We evaluate our method on text retrieval tasks:
  - *20 Newsgroups*: 18000 posts, 20 topics.
  - *TMC*: 28000 air traffic reports, 22 tags.
  - *Google Search Snippets*: 12000 short documents, 8 classes.
  - (Image Retrieval) *CIFAR-10*: 60000 RGB images (32×32), 10 classes.
- Variational autoencoder methods:
  - **VDHS-S** [1]: Gaussian latent variables, combines typical unsupervised learning approach and pointwise supervision.
  - **PHS-GS** [2]: Bernoulli latent variables, combines unsupervised learning, and both pointwise and pairwise supervision.
  - **SSB-VAE** - our proposed method: Bernoulli latent variable, unsupervised learning, pointwise and self-supervision.

Different levels of supervision - Precision

A)	20-News			CIFAR			Snippets			TMC		
ρ	PSH	SSB-VAE	VDSH	PSH	SSB-VAE	VDSH	PSH	SSB-VAE	VDSH	PSH	SSB-VAE	VDSH
0.1	0.589	0.734	0.648	0.687	0.825	0.805	0.501	0.565	0.540	0.738	0.750	0.730
0.3	0.630	0.787	0.738	0.737	0.847	0.820	0.542	0.620	0.576	0.757	0.759	0.736
0.5	0.762	0.824	0.788	0.818	0.879	0.844	0.564	0.641	0.634	0.772	0.778	0.743
0.7	0.815	0.841	0.831	0.889	0.880	0.852	0.553	0.644	0.648	0.790	0.795	0.768
0.9	0.867	0.880	0.866	0.903	0.901	0.863	0.644	0.648	0.656	0.806	0.813	0.781
1.0	0.866	0.878	0.876	0.906	0.910	0.867	0.696	0.657	0.661	0.806	0.818	0.788

B)	20-News			CIFAR			Snippets			TMC		
ρ	PSH	SSB-VAE	VDSH	PSH	SSB-VAE	VDSH	PSH	SSB-VAE	VDSH	PSH	SSB-VAE	VDSH
0.1	0.595	0.711	0.582	0.635	0.816	0.781	0.482	0.621	0.522	0.723	0.725	0.705
0.3	0.678	0.762	0.705	0.718	0.849	0.789	0.569	0.612	0.580	0.740	0.751	0.719
0.5	0.752	0.770	0.744	0.820	0.870	0.811	0.598	0.614	0.623	0.750	0.763	0.715
0.7	0.802	0.829	0.794	0.877	0.884	0.818	0.551	0.634	0.627	0.764	0.782	0.753
0.9	0.826	0.870	0.831	0.904	0.906	0.832	0.635	0.647	0.647	0.768	0.803	0.770
1.0	0.872	0.873	0.846	0.906	0.909	0.836	0.666	0.641	0.649	0.759	0.808	0.777

Table 1: P@100 of the methods. ρ: level of supervision . a) 32 bits and b) 16 bits.

Different levels of supervision - MAP

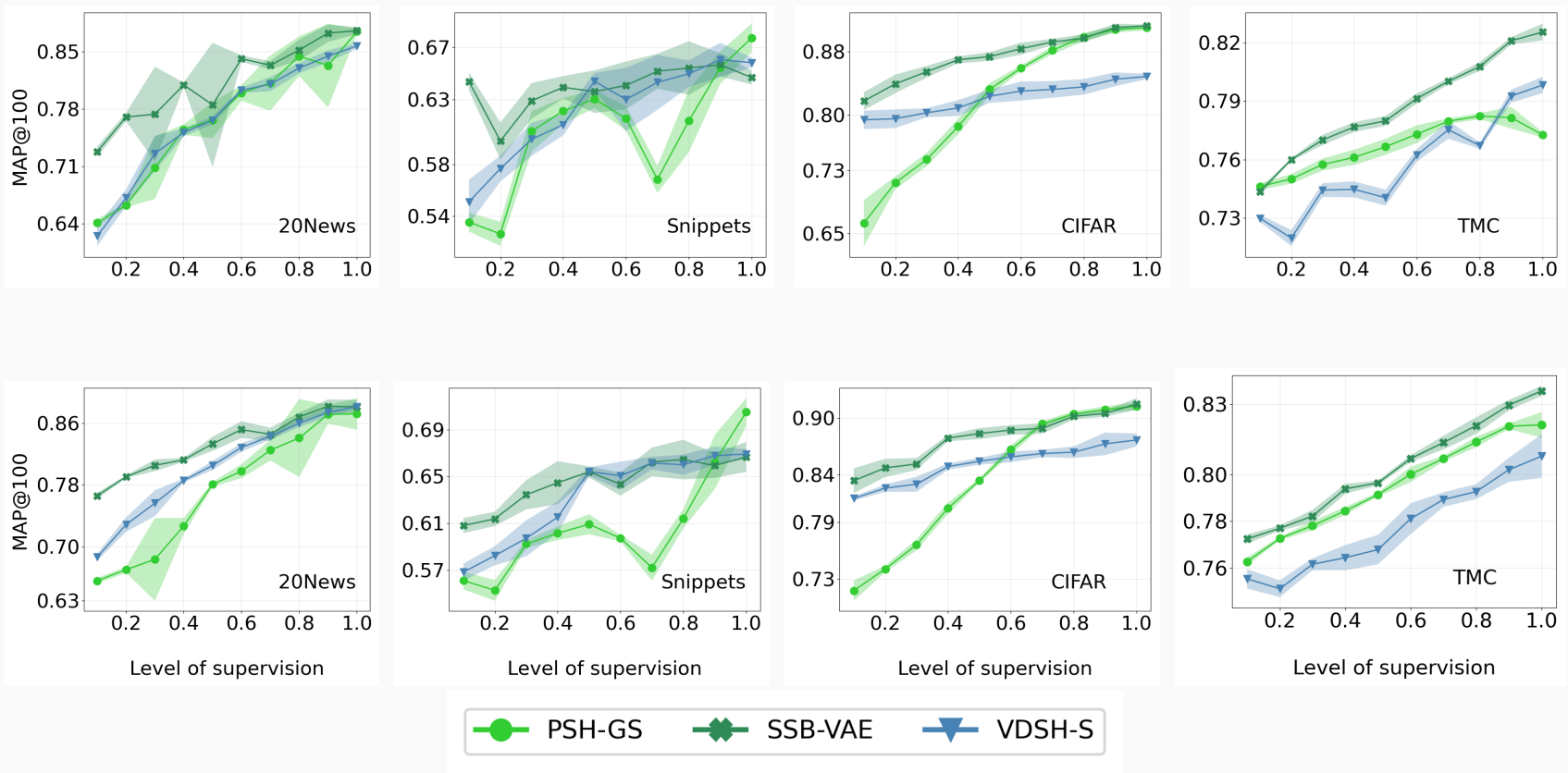


Figure 1: Mean average precision (MAP@100) of the different methods. 16 and 32 bits results are shown in the first row and second row respectively.

Robustness

	16 *bits*			32 *bits*		
	Friedman Test	PSH	VDSH	Friedman Test	PSH	VDSH
20-News	1.1×10^{-4}	6.3×10^{-5}	3.7×10^{-2}	5.0×10^{-4}	4.9×10^{-3}	1.0×10^{-3}
Snippets	7.4×10^{-3}	1.0×10^{-2}	8.9×10^{-1}	7.4×10^{-3}	1.0×10^{-2}	8.9×10^{-1}
TMC	4.5×10^{-5}	6.5×10^{-2}	2.3×10^{-5}	2.2×10^{-4}	1.9×10^{-2}	1.6×10^{-4}
CIFAR	2.0×10^{-2}	1.1×10^{-1}	1.2×10^{-2}	1.8×10^{-3}	1.9×10^{-2}	2.2×10^{-3}

Table 2: P-values of the Statistical Tests (in terms of p@100).

In all but two cases we obtain values below 5%

Conclusions

Conclusions

- We proposed a new type of semi-supervision for hashing based on variational autoencoders:
  - The model uses its own beliefs about the class distribution to enforce consistency between similarities in the code and label space.
- Experiments in text and image retrieval tasks confirmed that the proposed method degrades more gracefully when the models are stressed with scarcely annotated data.
  - very often it outperforms the baselines by a significant margin.
  - In scenarios of label abundance, the proposed method is competitive or even better than the best baseline.

In future, we plan to equip the method with adaptive loss weights and extend the experiments to cross-domain information retrieval.

Questions?

References i


S. Chaidaroon and Y. Fang.
Variational deep semantic hashing for text documents.
In *SIGIR*, pages 75–84, 2017.


S. Z. Dadaneh, S. Boluki, M. Yin, M. Zhou, and X. Qian.
Pairwise supervised hashing with bernoulli variational auto-encoder and self-control gradient estimator.
In *Proc. UAI*, 2020.


F. Mena and R. Ñanculef.
A binary variational autoencoder for hashing.
In *Proc. CIARP*, pages 131–141, 2019.


M. Norouzi, A. Punjani, and D. J. Fleet.
Fast exact search in Hamming space with multi-index hashing.
IEEE Pattern Anal. Mach. Intell., 36(6):1107–1119, 2014.