Unsupervised Deep Generative Adversarial Hashing Network

Kamran Ghasedi Dizaji[1], Feng Zheng[1], Najmeh Sadoughi[1], Yanhua Yang[2], Cheng Deng[2], Heng Huang[1]

kag221@pitt.edu, feng.zheng@pitt.edu, najme.sadoughi@gmail.com, yanhyang@xidian.edu.cn, chdeng.xd@gmail.com, heng.huang@pitt.edu

[1]Electrical and Computer Engineering Department, University of Pittsburgh, PA, USA [2]Xidian University, Xi'an, Shanxi, China

CVPR 2018 Salt Lake City

Challenges

- Unsupervised hashing methods either utilize shallow models with hand-crafted features as inputs, or employ deep architectures for obtaining both discriminative features and binary hash codes.
- The shallow hash functions suffer from hand-crafted features and dimension reductions techniques, and may not capture the non-linear similarities between real-world images due to their low capacity.
- The unsupervised deep hash functions have not shown satisfactory improvements against their shallow alternatives due to overfitting problem in lack of any supervisory signals.

Contributions

- We propose a novel framework for unsupervised hashing model by coupling a deep hash function and a generative adversarial network.
- We introducing a new hashing objective resulting in minimum entropy, uniform frequency, consistent, and independent hash bits for real images, regularized by the adversarial and collaborative loss functions on synthesized images.
- Achieving state-of-the-art results compared to alternative models on information retrieval and clustering tasks.

HashGAN Objective Function

- General loss

$$\mathcal{L}_{total} = \mathcal{L}_{adv} + \mathcal{L}_{hash} + \mathcal{L}_{col}$$

- Adversarial loss

$$\max_{\mathcal{D}} \ \mathbb{E}_{\mathbf{x}\sim P(\mathbf{x})}\big[\log(\mathcal{D}(\mathbf{x}))\big] + \mathbb{E}_{\mathbf{z}\sim P(\mathbf{z})}\big[\log(1-\mathcal{D}(\mathcal{G}(\mathbf{z}))\big]$$

- Hashing loss

$$\min_{\mathcal{E}} \ \underbrace{-\sum_{i=1}^{N}\sum_{k=1}^{K} t_{ik}\log t_{ik} + (1-t_{ik})\log(1-t_{ik})}_{\text{minimum entropy bits}} + \underbrace{\sum_{i=1}^{N}\sum_{k=1}^{K}\|t_{ik}-\tilde{t}_{ik}\|_2^2}_{\text{consistent bits}}$$

$$+ \underbrace{\sum_{k=1}^{K} f_k\log f_k + (1-f_k)\log(1-f_k)}_{\text{uniform frequency bits}} + \underbrace{\|\mathbf{W}_{\mathcal{E}}^{L^\top}\mathbf{W}_{\mathcal{E}}^{L}-\mathbf{I}\|_2^2}_{\text{independent bits}}$$

- Collaborative loss

$$\min_{\mathcal{E}} \ \mathbb{E}_{\mathbf{z}\sim P(\mathbf{z})}\big[\|\mathcal{E}(\mathcal{G}(\mathbf{z}))-\mathbf{b}'\|_2^2\big]$$

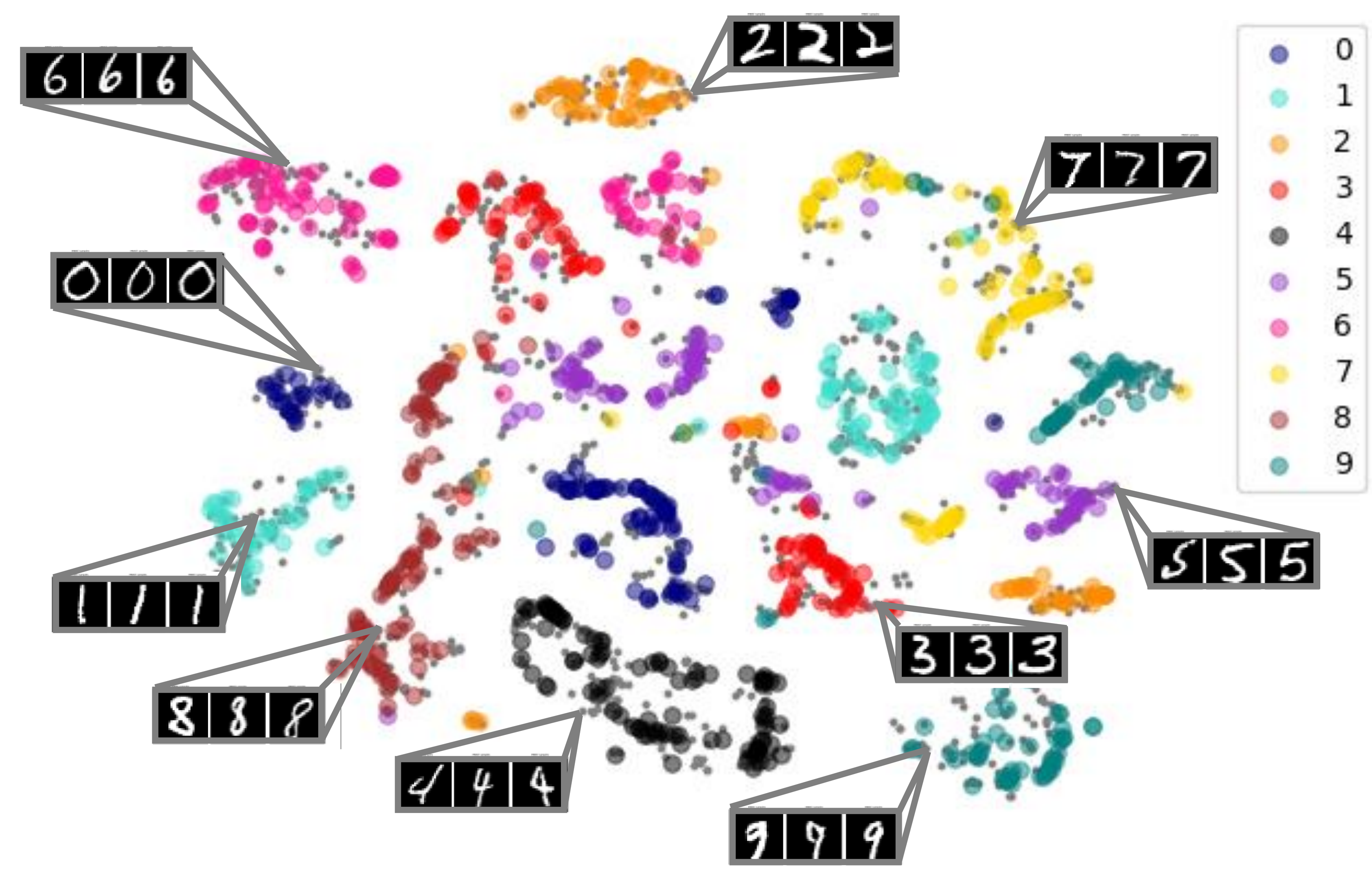


Figure 1: Visualization of *HashGAN* discriminative representations for a query set on *MNIST* using *TSNE* projection. The real and synthesized data are indicated by colored and gray circles respectively. Some of the synthesized images are randomly shown from different parts of space.

	Dataset	*CIFAR-10*						*MNIST*						Super. Pretrain
		mAP (%)			mAP@1000 (%)			mAP (%)			mAP@1000 (%)			
	Model	16	32	64	16	32	64	16	32	64	16	32	64	
Shallow	*KMH*	13.59	13.93	14.46	24.08*	23.56*	25.19*	32.12	33.29	35.78	59.12*	70.32*	67.62*	✗
	SphH	13.98	14.58	15.38	24.52*	24.16*	26.09*	25.81	30.77	34.75	52.97*	65.45*	65.45*	✗
	SpeH	12.55	12.42	12.56	22.10*	21.79*	21.97*	26.64	25.72	24.10	59.72*	64.37*	67.60*	✗
	PCAH	12.91	12.60	12.10	21.52*	21.62*	20.54*	27.33	24.85	21.47	60.98*	64.47*	63.31*	✗
	LSH	12.55	13.76	15.07	12.63*	16.31*	18.00*	20.88	25.83	31.71	42.10*	50.45*	66.23*	✗
	ITQ	15.67	16.20	16.64	26.71*	27.41*	28.93*	41.18	43.82	45.37	70.06*	76.86*	80.23*	✗
Deep	*DH*	16.17	16.62	16.96	-	-	-	43.14	44.97	46.74	-	-	-	✗
	DAR	16.82	17.01	17.21	-	-	-	-	-	-	-	-	-	✗
	DeepBit	-	-	-	19.43	24.86	27.73	-	-	-	28.18	32.02	44.53	✓
	UTH	-	-	-	28.66	30.66	32.41	-	-	-	43.15	46.58	49.88	✓
	HashGAN	**29.94**	**31.47**	**32.53**	**44.65**	**46.34**	**48.12**	**91.13**	**92.70**	**93.93**	**94.31**	**95.48**	**96.37**	✗

Table 1: Image retrieval results (mAP and mAP@1000) of unsupervised hash functions on CIFAR-10 and MNIST datasets, when the number of hash bits are 16, 32 and 64. The usage of supervised pretraining is shown for each model using the tick sign.

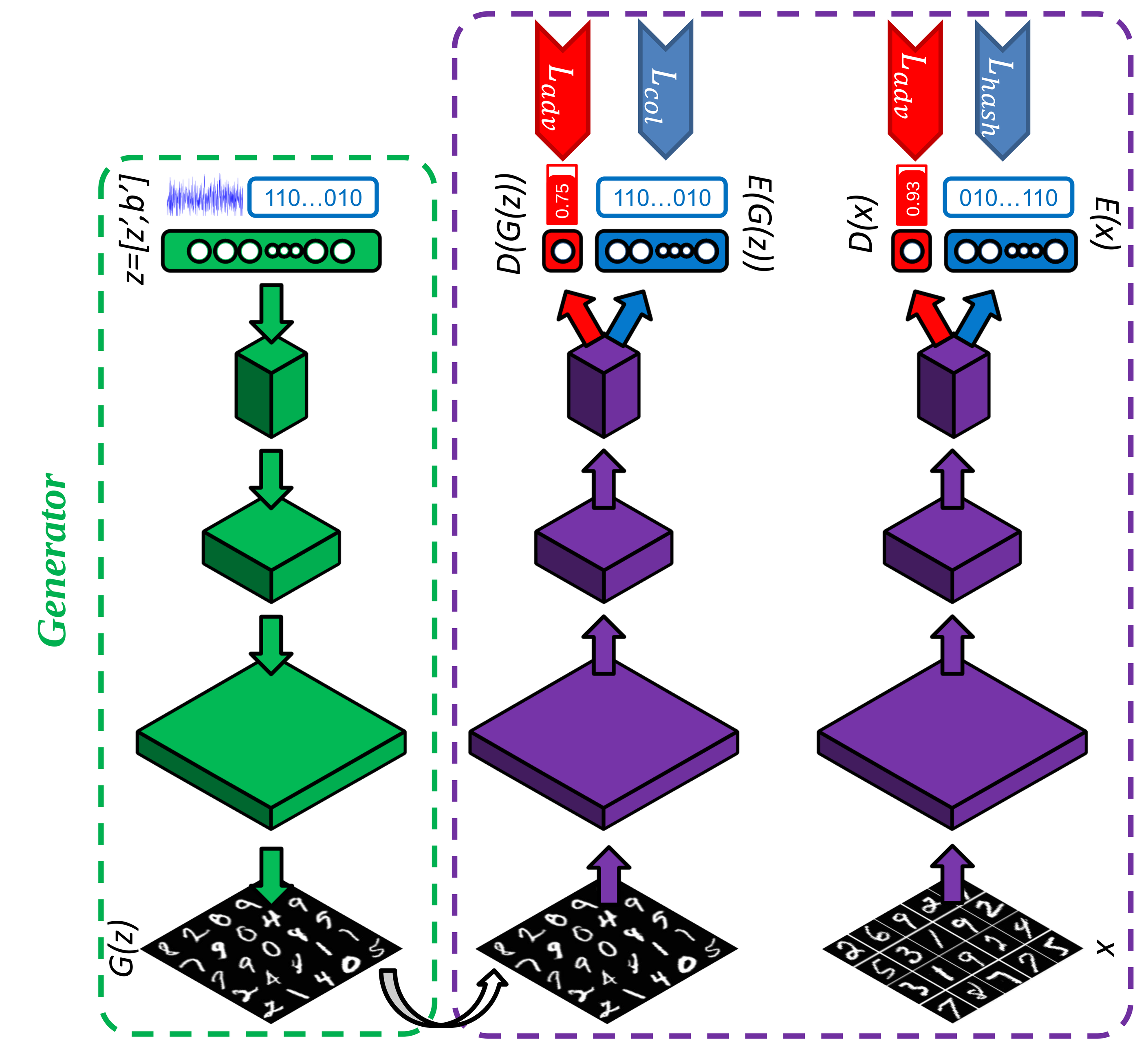
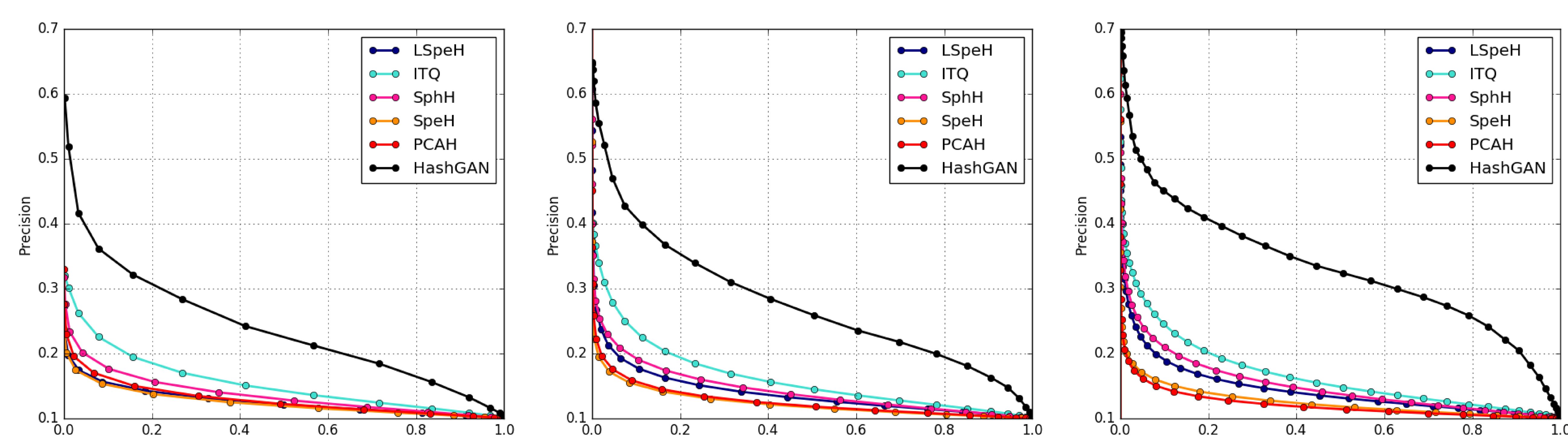


Figure 2: *HashGAN* architecture, including a generator (green), a discriminator (red) and an encoder (blue), where the last two share their parameters in several layers (red⊕blue=purple). The arrows on top represent the loss functions.

	Dataset	*MNIST*		*USPS*		*FRGC*		*STL-10*	
	Model	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC
Shallow	*K-means*	0.500	0.534	0.450	0.460	0.287	0.243	0.209*	0.284
	N-Cuts	0.411	0.327	0.675	0.314	0.285	0.235	-	-
	SC-LS	0.706	0.714	0.681	0.659	0.550	0.407	-	-
	AC-PIC	0.017	0.115	0.840	0.855	0.415	0.320	-	-
	SEC	0.779	0.804	0.511	0.544	-	-	0.245*	0.307
	LDMGI	0.802	0.842	0.563	0.580	-	-	0.260*	0.331
Deep	*NMF-D*	0.152	0.175	0.287	0.382	0.259	0.274	-	-
	DEC	0.816	0.844	0.586	0.619	0.505	0.378	0.284*	0.359
	JULE-RC	0.913	0.964	0.913	0.950	0.574	0.461	-	-
	DEPICT	**0.917**	**0.965**	**0.927**	0.964	**0.610**	**0.470**	0.303*	0.371*
	HashGAN	0.913	**0.965**	0.920	0.958	0.602	0.465	**0.316**	**0.394**

Table 2: Clustering performance of *HashGAN* and several other algorithms on four image datasets based on accuracy (ACC) and normalized mutual information (NMI).



(a) 16 bits (b) 32 bits (c) 64 bits

Figure 3: Precision-Recall curves on *CIFAR-10* database for *HashGAN* and five baselines with 16, 32, and 64 hash bits.