

NASH: Toward End-to-End Neural Architecture for Generative Semantic Hashing

Presenter: Dinghan Shen*

Joint work with: Qinliang Su*, Paidamoyo Chapfuwa, Wenlin Wang, Guoyin Wang, Lawrence Carin, Ricardo Henao

Duke University & Sun Yat-sen University

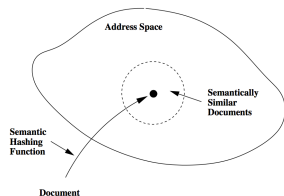
July 17, 2018

*Equal contribution

Background

Semantic Hashing

- **Fast and accurate similarity search** (i.e., *finding documents from a large corpus that are most similar to a query of interest*) is at the core of many information retrieval applications;
- One strategy is to represent each document as a **continuous vector**: such as Paragraph Vector [Le and Mikolov, 2014], Skip-thought vectors [Kiros et al., 2015], Inferred [Conneau et al., 2017], etc. **Cosine similarity** is typically employed to measure relatedness;
- **Semantic hashing** is an effective approach: the similarity between two documents can be evaluated by simply calculating **pairwise Hamming distances between hashing (binary) codes**;



Motivation & contributions

● Motivation:

- Existing semantic hashing approaches typically require **two-stage training procedures** (e.g. continuous representations are **crudely binarized** after training);
- Vast amount of **unlabeled data** is not fully leveraged for learning binary document representations.

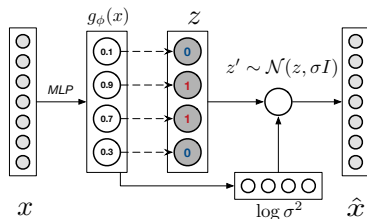
● Contributions:

- we propose a simple and generic neural architecture for text hashing that learns binary latent codes for documents, which be trained **an end-to-end manner**;
- We leverage a **Neural Variational Inference (NVI) framework**, which introduces **data-dependent noises** during training and makes effective use of unlabeled information.

Framework components

Hashing under the NVI Framework

- Notations: let x and z denote the input **document** and its corresponding **binary hash code**, respectively;
- We define a generative model that simultaneously accounts for both the **encoding distribution**, $p(z|x)$, and **decoding distribution**, $p(x|z)$,



- We define approximations $q_\phi(z|x)$ and $q_\theta(x|z)$ via inference and generative networks, parameterized by ϕ and θ , respectively.

Framework components

Training with Binary Latent Variables

- The generative term provides a natural training objective for semantic hashing: with the decoder network modeling $p(x|z)$, the key semantic information from x is naturally encapsulated.
- To tailor the NVI framework for semantic hashing, we cast z as a binary latent variable and assume a multivariate Bernoulli prior on z :

$$z : p(z) \sim \text{Bernoulli}(\gamma) = \prod_{i=1}^I \gamma_i^{z_i} (1 - \gamma_i)^{1-z_i}; \quad (1)$$

- The encoding (approximate posterior) distribution $q_\phi(z|x)$ is restricted to take the form $q_\phi(z|x) = \text{Bernoulli}(h)$, where h is inferred from x with the encoder network.

Framework components

Training with Binary Latent Variables

- We can obtain samples from the *Bernoulli posterior* either deterministically or stochastically:
- Suppose z is a l -bit hash code, the **deterministic binarization** is defined as (for $i = 1, 2, \dots, l$):

$$z_i = \mathbf{1}_{\sigma(g_\phi^i(x)) > 0.5} = \frac{\text{sign}(\sigma(g_\phi^i(x)) - 0.5) + 1}{2} \quad (2)$$

- **stochastic binarization** (where $\mu_i \sim \text{Uniform}(0, 1)$):

$$z_i = \mathbf{1}_{\sigma(g_\phi^i(x)) > \mu_i} = \frac{\text{sign}(\sigma(g_\phi^i(x)) - \mu_i) + 1}{2}, \quad (3)$$

Framework components

Training with Binary Latent Variables

- To estimate the parameters of the encoder and decoder networks, we maximize a **variational lower bound**:

$$\begin{aligned}\mathcal{L}_{\text{vae}} &= \mathbb{E}_{q_{\phi}(z|x)} \left[\log \frac{q_{\theta}(x|z)p(z)}{q_{\phi}(z|x)} \right], \\ &= \mathbb{E}_{q_{\phi}(z|x)} [\log q_{\theta}(x|z)] - D_{KL}(q_{\phi}(z|x) || p(z)),\end{aligned}\quad (4)$$

- The KL-divergence $D_{KL}(q_{\phi}(z|x) || p(z))$ encourages the **approximate posterior** $q_{\phi}(z|x)$ to be close to the **multivariate Bernoulli prior** $p(z)$;
- $D_{KL}(q_{\phi}(z|x) || p(z))$ can be written in closed-form.

Framework components

Training with Binary Latent Variables

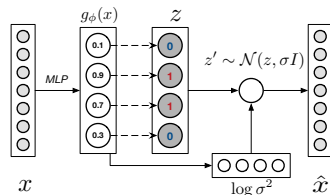
- It is challenging to **backpropagate gradients through the discrete (binary) latent variable**, since the derivative of the *sign* function is zero for almost all input values;
- Instead, we utilize the **straight-through (ST) estimator**, which was first introduced by [Hinton (2012)]. It simply backpropagates through the hard threshold by approximating the gradient $\partial z / \partial \sigma(g_\phi^i(x))$ as 1:

$$\begin{aligned} & \frac{d\mathbb{E}_{q_\phi(z|x)}[\log q_\theta(x|z)]}{\partial\phi} \\ &= \frac{d\mathbb{E}_{q_\phi(z|x)}[\log q_\theta(x|z)]}{dz} \frac{dz}{d\sigma(g_\phi^i(x))} \frac{d\sigma(g_\phi^i(x))}{d\phi} \\ &\approx \frac{d\mathbb{E}_{q_\phi(z|x)}[\log q_\theta(x|z)]}{dz} \frac{d\sigma(g_\phi^i(x))}{d\phi} \end{aligned} \tag{5}$$

Framework components

Injecting Data-dependent Noise to z

- We found that **injecting random Gaussian noise into z** makes the decoder a more favorable regularizer for the binary codes;



- The objective function in (4) can be written in a form similar to **the rate-distortion tradeoff**:

$$\min \mathbb{E}_{q_\phi(z|x)} \left[\underbrace{-\log q_\phi(z|x)}_{\text{Rate}} + \underbrace{\frac{1}{2\sigma^2} \|x - Ez\|_2^2}_{\beta \text{ Distortion}} + C \right], \quad (6)$$

Framework components

Extension to Supervised Hashing

- While labeled data are available, we can **explicitly learn a mapping** from latent variable z to labels y , here parametrized by a two-layer MLP followed by a fully-connected softmax layer.
- As a result, the loss function is a combination of variational lower bound and discriminative (cross-entropy) loss:

$$\mathcal{L} = -\mathcal{L}_{\text{vae}}(\theta, \phi; \mathbf{x}) + \alpha \mathcal{L}_{\text{dis}}(\eta; \mathbf{z}, \mathbf{y}). \quad (7)$$

Experiments

Datasets & Experimental Setup

- **Datasets:** we evaluate the proposed method on three benchmarks: *Reuters21578*, *20Newsgroups*, *TMC (SIAM text mining competition)*;
- **TFIDF features** are utilized as the input x for documents;
- we set the dimension of z , *i.e.*, **the number of bits** within the hashing code, as 8, 16, 32, 64, or 128;
- We employed **precision** as the evaluation metric: the percentage of documents among the top 100 retrieved ones that belong to the same label (topic) with the query document.

Experiments

Semantic Hashing Evaluation

Method	8 bits	16 bits	32 bits	64 bits	128 bits
LSH	0.2802	0.3215	0.3862	0.4667	0.5194
S-RBM	0.5113	0.5740	0.6154	0.6177	0.6452
SpH	0.6080	0.6340	0.6513	0.6290	0.6045
STH	0.6616	0.7351	0.7554	0.7350	0.6986
VDSH	0.6859	0.7165	0.7753	0.7456	0.7318
NASH	0.7113	0.7624	0.7993	0.7812	0.7559
NASH-N	0.7352	0.7904	0.8297	0.8086	0.7867
NASH-DN	0.7470	0.8013	0.8418	0.8297	0.7924

Table: Precision of the top 100 retrieved documents on *Reuters* dataset (*Unsupervised hashing*).

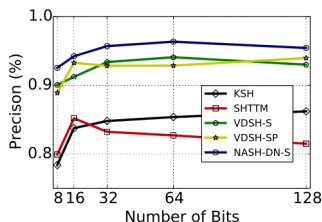


Figure: Precision of the top 100 retrieved documents on *Reuters* dataset (*Supervised hashing*).

● Fast similarity search:

- Consistently outperform several strong baseline methods;
- Enjoy the attractive property of end-to-end training;
- Same observations on other benchmarks.

Experiments

Ablation study

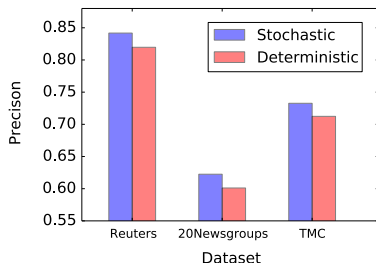


Figure: The precisions of the top 100 retrieved documents for NASH-DN with *stochastic* or *deterministic* binary latent variables.

Network	Encoder	Decoder
linear	0.5844	0.6225
one-layer MLP	0.6187	0.3559
two-layer MLP	0.6225	0.1047

Table: Ablation study with different encoder/decoder networks.

- Leveraging stochastically sampling during training generalizes better;
- Linear decoder networks gives rise to better empirical results.

Experiments

Qualitative Analysis

Category	Title/Subject	8-bit code	16-bit code
Baseball	<i>Dave Kingman for the hall of fame</i>	1 1 1 0 1 0 0 1	0 0 1 0 1 1 0 1 0 0 0 0 0 1 1 0
	<i>Time of game</i>	1 1 1 1 1 0 0 1	0 0 1 0 1 0 0 1 0 0 0 0 0 1 1 1
	<i>Game score report</i>	1 1 1 0 1 0 0 1	0 0 1 0 1 1 0 1 0 0 0 0 0 1 1 0
	<i>Why is Barry Bonds not batting 4th?</i>	1 1 1 0 1 1 0 1	0 0 1 1 1 1 0 1 0 0 0 0 0 1 1 0
Electronics	<i>Building a UV flashlight</i>	1 0 1 1 0 1 0 0	0 0 1 0 0 0 1 0 0 0 1 0 1 0 1 1
	<i>How to drive an array of LEDs</i>	1 0 1 1 0 1 0 1	0 0 1 0 0 0 1 0 0 0 1 0 1 0 0 1
	<i>2% silver solder</i>	1 1 0 1 0 1 0 1	0 0 1 0 0 0 1 0 0 0 1 0 1 0 1 1
	<i>Subliminal message flashing on TV</i>	1 0 1 1 0 1 0 0	0 0 1 0 0 1 1 0 0 0 1 0 1 0 0 1

Figure: Examples of learned compact hashing codes on 20Newsgroups dataset.





- NASH typically compresses documents with shared topics into very similar binary codes.

Conclusions

Take away

- This paper presents a first step towards end-to-end semantic hashing;
- A neural variational framework is introduced to optimize the hash function during training;
- The connections between the proposed method and rate-distortion theory are established.

Q&A

-  Distributed Representations of Sentences and Documents *ICML 2014*;
-  Skip-thought vectors *NIPS 2015*;
-  Supervised Learning of Universal Sentence Representations from Natural Language Inference Data *EMNLP 2017*;
-  Geoffrey Hinton. 2012. Neural networks for machine learning, coursera. URL: <http://coursera.org/course/neuralnets>;