Integrating Semantics and Neighborhood Information with Graph-Driven Generative Models for Document Retrieval

Anonymous ACL-IJCNLP submission

Abstract

With the need of fast retrieval speed and small memory footprint, document hashing has been playing a crucial role in large-scale information retrieval. To generate high-quality hashing code, both semantics and neighborhood information are crucial. However, most existing methods leverage only one of them or simply combine them via some intuitive criteria, lacking a theoretical principle to guide the integration process. In this paper, we encode the neighborhood information with a graph-induced Gaussian distribution, and propose to integrate the two types of information with a graph-driven generative model. To deal with the complicated correlations among documents, we further propose a tree-structured approximation method for learning. Under the approximation, we prove that the training objective can be decomposed into terms involving only singleton or pairwise documents, enabling the model to be trained as efficiently as uncorrelated ones. Extensive experimental results on three benchmark datasets show that our method achieves superior performance over state-of-the-art methods, demonstrating the effectiveness of the proposed model for simultaneously preserving semantic and neighborhood information.

1 Introduction

Similarity search plays a pivotal role in a variety of tasks, such as image retrieval (Jing and Baluja, 2008; Zhang et al., 2018), plagiarism detection (Stein et al., 2007) and recommendation systems (Koren, 2008). If the search is carried out in the original continuous feature space directly, the requirements of computation and storage would be extremely high, especially for large-scale applications. Semantic hashing (Salakhutdinov and Hinton, 2009) sidesteps this problem by learning a compact binary code for every item such that similar items can be efficiently found according to the Hamming distance of binary codes.

Unsupervised semantic hashing aims to learn for each item a binary code that can preserve the semantic similarity information of original items, without the supervision of any labels. Motivated by the success of deep generative models (Kingma and Welling, 2013; Rezende et al., 2014) in unsupervised representation learning, many recent methods approach this problem from the perspective of deep generative models, leading to state-of-the-art performance on benchmark datasets. Specifically, these methods train a deep generative model to model the underlying documents and then use the trained generative model to extract continuous or binary representations from the original documents (Chaidaroon and Fang, 2017; Shen et al., 2018; Dong et al., 2019; Zheng et al., 2020). The basic principle behind these generative hashing methods is to have the hash codes retaining as much semantics information of original documents as possible so that semantically similar documents are more likely to yield similar codes.

In addition to semantics information, it is widely observed that neighborhood information among the documents is also useful to generate high-quality hash codes. By constructing an adjacency matrix from the raw features of documents, neighbor-based methods seek to preserve the information in the constructed adjacency matrix, such as the locality-preserving hashing (He et al., 2004; Zhao et al., 2014), spectral hashing (Weiss et al., 2009; Li et al., 2012), and etc. However, since the ground-truth neighborhood information is not available and the constructed one is neither accurate nor complete, neighbor-based methods alone do not perform as well as the semantics-based ones. Despite both semantics and neighborhood information are derived from the original documents, different aspects are emphasized in them. Thus, to obtain

higher-quality hash codes, it has been proposed to incorporate the constructed neighborhood information into semantics-based methods. For examples, Chaidaroon et al. (2018) and Hansen et al. (2020) require the hash codes can reconstruct neighboring documents, in addition to the original input. Other works (Shen et al., 2019; Hansen et al., 2019) use an extra loss term, derived from the approximate neighborhood information, to encourage similar documents to produce similar codes. However, all of the aforementioned methods exploit the neighborhood information by using it to design different kinds of regularizers to the original semantics-based models, lacking a basic principle to unify and leverage them under one framework.

To fully exploit the two types of information, in this paper, we propose a hashing method that unifies the semantics and neighborhood information with the *graph-driven generative models*. Specifically, we first encode the neighborhood information with a multivariate Gaussian distribution. With this Gaussian distribution as a prior in a generative model, the neighborhood information can be naturally incorporated into the semantics-based hashing model. Despite the simplicity of the modeling, the correlation introduced by the neighbor-encoded prior poses a significant challenge to the training since it invalidates the widely used identical-and-independent-distributed (*i.i.d.*) assumption, making all documents correlated. To address this issue, we propose to use a tree-structured distribution to capture as much as possible the neighborhood information. We prove that under the tree approximation, the evidence lower bound (ELBO) can be decomposed into terms involving only singleton and pairwise documents, enabling the model to be trained as efficiently as the models without considering the document correlations. To capture more neighborhood information, a more accurate approximation by using multiple trees is also developed. Extensive experimental results on three public datasets demonstrate that the proposed method can outperform state-of-the-art methods, indicating the effectiveness of the proposed framework in unifying the semantic and neighborhood information for document hashing.

2 Preliminaries

Semantics-Based Hashing Due to the similarities among the underlying ideas of these methods, we take the variational deep semantic hashing (VDSH) (Chaidaroon and Fang, 2017) as an example to illustrate their working flow. Given a document $\boldsymbol{x} \triangleq \{w_j\}_{j=1}^{|\boldsymbol{x}|}$, VDSH proposes to model a document by a generative model as

$$p(\boldsymbol{x}, \boldsymbol{z}) = p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}), \tag{1}$$

where $p(\boldsymbol{z})$ is the prior distribution and is chosen to be the standard Gaussian distribution $\mathcal{N}(\boldsymbol{z}; \mathbf{0}, \boldsymbol{I}_d)$, with $\boldsymbol{I}_d$ denoting the d-dimensional identity matrix; and $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is defined to be

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) = \prod_{w_i \in \boldsymbol{x}} p_{\boldsymbol{\theta}}(w_i|\boldsymbol{z}) \tag{2}$$

with

$$p_{\boldsymbol{\theta}}(w_i|\boldsymbol{z}) \triangleq \frac{\exp(\boldsymbol{z}^T E w_i + b_i)}{\sum_{j=1}^{|V|} \exp(\boldsymbol{z}^T E w_j + b_j)}, \tag{3}$$

in which w_j denotes the $|V|$-dimensional one-hot representation of the j-th word, with $|\boldsymbol{x}|$ and $|V|$ denoting the document and vocabulary size, respectively; and $E \in \mathbb{R}^{d \times |V|}$ represents the learnable embedding matrix. For a corpus containing N documents $\boldsymbol{X} = \{\boldsymbol{x_1}, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N\}$, due to the *i.i.d.* assumption for documents, it is modelled by simply multiplying individual document models as

$$p(\boldsymbol{X}, \boldsymbol{Z}) = \prod_{k=1}^{N} p_{\boldsymbol{\theta}}(\boldsymbol{x}_k|\boldsymbol{z}_k)p(\boldsymbol{z}_k), \tag{4}$$

where $\boldsymbol{Z} \triangleq [\boldsymbol{z}_1; \boldsymbol{z}_2; \cdots; \boldsymbol{z}_N]$ denotes a long vector obtained by concatenating the individual vectors $\boldsymbol{z}_i$. The model is trained by optimizing the evidence lower bound (ELBO) of the log-likelihood function $\log p(\boldsymbol{X})$. After training, outputs from the trained encoder are used as documents' representations, from which binary hash codes can be obtained by thresholding the real-valued representations.

Neighborhood Information The ground-truth semantic similarity information is not available for the unsupervised hashing task in practice. To leverage this information, an affinity $N \times N$ matrix $\boldsymbol{A}$ is generally constructed from the raw features (*e.g.*, the TFIDF) of original documents. For instances, we can construct the matrix as

$$a_{ij} = \begin{cases} e^{-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{\sigma}}, & \boldsymbol{x}_i \in \mathcal{N}_k(\boldsymbol{x}_j) \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where a_{ij} denotes the (i, j)-th element of $\boldsymbol{A}$; and $\mathcal{N}_k(\boldsymbol{x})$ denotes the k-nearest neighbors of document $\boldsymbol{x}$. Given the affinity matrix $\boldsymbol{A}$, some methods

have been proposed to incorporate the neighborhood information into the semantics-based hashing models. However, as discussed above, these methods generally leverage the information based on some intuitive criteria, lacking theoretical supports behind them.

3 A Hashing Framework with Unified Semantics-Neighborhood Information

In this section, we present a more effective framework to unify the semantic and neighborhood information for the task of document hashing.

3.1 Reformulating the VDSH

To introduce the neighborhood information into the semantics-based hashing models, we first rewrite the VDSH model into a compact form as

$$p(\boldsymbol{X}, \boldsymbol{Z}) = p_{\boldsymbol{\theta}}(\boldsymbol{X}|\boldsymbol{Z})p_{\mathcal{I}}(\boldsymbol{Z}), \tag{6}$$

where $p_{\boldsymbol{\theta}}(\boldsymbol{X}|\boldsymbol{Z}) = \prod_{k=1}^{N} p_{\boldsymbol{\theta}}(\boldsymbol{x}_k|\boldsymbol{z}_k)$; and the prior $p_{\mathcal{I}}(\boldsymbol{Z}) = \prod_{k=1}^{N} p(\boldsymbol{z}_k)$, which can be shown to be

$$p_{\mathcal{I}}(\boldsymbol{Z}) = \mathcal{N}\left(\boldsymbol{Z}; \boldsymbol{0}, \boldsymbol{I}_N \otimes \boldsymbol{I}_d\right). \tag{7}$$

Here, $\otimes$ denotes the Kronecker product and the subscript $\mathcal{I}$ indicates independence among $\boldsymbol{z}_k$. The ELBO of this model can be expressed as

$$\mathcal{L} = \underbrace{\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{X}|\boldsymbol{Z})]}_{\mathcal{L}_1} - \underbrace{KL(q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})||p_{\mathcal{I}}(\boldsymbol{Z}))}_{\mathcal{L}_2}$$

where $KL(\cdot)$ denotes the Kullback-Leibler (KL) divergence. By restricting the posterior to independent Gaussian form

$$q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X}) = \prod_{k=1}^{N} \underbrace{\mathcal{N}\left(\boldsymbol{z}_k; \boldsymbol{\mu}_k, diag(\boldsymbol{\sigma}_k^2)\right)}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_k|\boldsymbol{x}_k)}, \tag{8}$$

the $\mathcal{L}_1$ can be handled using the reparameterization trick. Thanks to the factorized forms assumed in $q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})$ and $p_{\mathcal{I}}(\boldsymbol{Z})$, the $\mathcal{L}_2$ term can also be expressed analytically and evaluated efficiently.

3.2 Injecting the Neighborhood Information

Given an affinity matrix $\boldsymbol{A}$, the covariance matrix $\boldsymbol{I}_N + \lambda\boldsymbol{A}$ can be used to reveal the neighborhood information of documents, where the hyperparameter $\lambda \in [0, 1)$ is used to control the overall correlation strength. If two documents are neighboring, then the corresponding correlation value in $\boldsymbol{I}_N + \lambda\boldsymbol{A}$ will be large; otherwise, the value will be zero. To have the neighborhood information reflected in document representations, we can require that the representations $\boldsymbol{z}_i$ are drawn from a Gaussian distribution of the form

$$p_{\mathcal{G}}(\boldsymbol{Z}) = \mathcal{N}\left(\boldsymbol{Z}; \boldsymbol{0}, (\boldsymbol{I}_N + \lambda\boldsymbol{A}) \otimes \boldsymbol{I}_d\right), \tag{9}$$

where the subscript $\mathcal{G}$ denotes that the distribution is constructed from a neighborhood graph. To see why the representations $\boldsymbol{Z} \sim p_{\mathcal{G}}(\boldsymbol{Z})$ have already reflected the neighborhood information, let us consider an example with three documents $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3\}$, in which $\boldsymbol{x}_1$ is connected to $\boldsymbol{x}_2$, $\boldsymbol{x}_2$ is connected to $\boldsymbol{x}_3$, and no connection exists between $\boldsymbol{x}_1$ and $\boldsymbol{x}_3$. Under the case that $\boldsymbol{z}_i$ is a two-dimensional vector $\boldsymbol{z}_i \in \mathbb{R}^2$, we have the concatenated representations $[\boldsymbol{z}_1; \boldsymbol{z}_2; \boldsymbol{z}_3]$ follow a Gaussian distribution with covariance matrix of

	$\boldsymbol{z}_1$		$\boldsymbol{z}_2$		$\boldsymbol{z}_3$	
$\boldsymbol{z}_1$	1	0	λa_{12}	0	0	0
	0	1	0	λa_{12}	0	0
$\boldsymbol{z}_2$	λa_{21}	0	1	0	λa_{23}	0
	0	λa_{21}	0	1	0	λa_{23}
$\boldsymbol{z}_3$	0	0	λa_{32}	0	1	0
	0	0	0	λa_{32}	0	1

From the property of Gaussian distribution, it can be known that $\boldsymbol{z}_1$ is strongly correlated with $\boldsymbol{z}_2$ on the corresponding elements, but not with $\boldsymbol{z}_3$. This suggests that $\boldsymbol{z}_1$ should be similar to $\boldsymbol{z}_2$, but different from $\boldsymbol{z}_3$, which is consistent with the neighborhood relation that $\boldsymbol{x}_1$ is a neighbor of $\boldsymbol{x}_2$, but not of $\boldsymbol{x}_3$.

Now that the neighborhood information can be modeled by requiring $\boldsymbol{Z}$ being drawn from $p_{\mathcal{G}}(\boldsymbol{Z})$, and the semantic information can be reflected in the likelihood function $p_{\boldsymbol{\theta}}(\boldsymbol{X}|\boldsymbol{Z})$. The two types of information can be taken into account simultaneously by modeling the corpus as

$$p(\boldsymbol{X}, \boldsymbol{Z}) = p_{\boldsymbol{\theta}}(\boldsymbol{X}|\boldsymbol{Z})p_{\mathcal{G}}(\boldsymbol{Z}). \tag{10}$$

Comparing to the VDSH model in (6), it can be seen that the only difference lies in the employed priors. Here, a neighborhood-preserving prior $p_{\mathcal{G}}(\boldsymbol{Z})$ is employed, while in VDSH, an independent prior $p_{\mathcal{I}}(\boldsymbol{Z})$ is used. Although only a modification to the prior is made from the perspective of modeling, significant challenges are posed for the training. Specifically, by replacing $p_{\mathcal{I}}(\boldsymbol{Z})$ with $p_{\mathcal{G}}(\boldsymbol{Z})$ in the $\mathcal{L}_2$ of $\mathcal{L}$, it can be shown that the expression of $\mathcal{L}_2$ involves the matrix $\left((\boldsymbol{I}_N + \lambda\boldsymbol{A}) \otimes \boldsymbol{I}_d\right)^{-1}$, with the exact expression

given in the Supplementary. Due to the introduced dependence among documents, for example, if the corpus contains over 100,000 documents and the representation dimension is set to 100, the $\mathcal{L}_2$ involves the *inverse* of matrices with dimension as high as 10^7, which is computationally prohibitive in practice.

4 Training with Tree Approximations

Although the prior $p_{\mathcal{G}}(\boldsymbol{Z})$ captures the full neighborhood information, its induced model is not practically trainable. In this section, to facilitate the training, we first propose to use a tree-structured prior to partially capture the neighborhood information, and then extend it to multiple-tree case for more accurate modeling.

4.1 Approximating the Prior $p_{\mathcal{G}}(\boldsymbol{Z})$ with a Tree-Structured Distribution

The matrix $\boldsymbol{A}$ represents a graph $\mathbb{G} \triangleq (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, 2, \cdots, N\}$ is the set of document indices; and $\mathcal{E} = \{(i,j)|a_{ij} \neq 0\}$ is the set of connections between documents. From the graph $\mathbb{G}$, a spanning tree $\mathbb{T} = (\mathcal{V}, \mathcal{E}_T)$ can be obtained easily, where $\mathcal{E}_T$ denotes the set of connections on the tree.[1] Based on the spanning tree, we construct a new distribution as

$$p_{\mathcal{T}}(\boldsymbol{Z}) = \prod_{i\in\mathcal{V}} p_{\mathcal{G}}(\boldsymbol{z}_i) \prod_{(i,j)\in\mathcal{E}_T} \frac{p_{\mathcal{G}}(\boldsymbol{z}_i, \boldsymbol{z}_j)}{p_{\mathcal{G}}(\boldsymbol{z}_i) p_{\mathcal{G}}(\boldsymbol{z}_j)}, \quad (11)$$

where $p_{\mathcal{G}}(\boldsymbol{z}_i)$ and $p_{\mathcal{G}}(\boldsymbol{z}_i, \boldsymbol{z}_j)$ represent one- and two-variable marginal distributions of $p_{\mathcal{G}}(\boldsymbol{Z})$, respectively. From the properties of Gaussian distribution, it is known that

$$\begin{aligned} p_{\mathcal{G}}(\boldsymbol{z}_i) &= \mathcal{N}(\boldsymbol{z}_i; \boldsymbol{0}, \boldsymbol{I}_d), \\ p_{\mathcal{G}}(\boldsymbol{z}_i, \boldsymbol{z}_j) &= \mathcal{N}([\boldsymbol{z}_i; \boldsymbol{z}_j]; \boldsymbol{0}, (\boldsymbol{I}_2 + \lambda \boldsymbol{A}_{ij}) \otimes \boldsymbol{I}_d), \end{aligned} \quad (12)$$

where $\boldsymbol{A}_{ij} \triangleq \begin{bmatrix} 0 & a_{ij} \\ a_{ji} & 0 \end{bmatrix}$. Because $p_{\mathcal{T}}(\boldsymbol{Z})$ is defined on a tree, as proved in (Wainwright and Jordan, 2008), it is guaranteed to be a valid probability distribution, and more importantly, it satisfies the following two relations: i) $p_{\mathcal{T}}(\boldsymbol{z}_i) = p_{\mathcal{G}}(\boldsymbol{z}_i)$; ii) $p_{\mathcal{T}}(\boldsymbol{z}_i, \boldsymbol{z}_j) = p_{\mathcal{G}}(\boldsymbol{z}_i, \boldsymbol{z}_j)$ for any $(i,j) \in \mathcal{E}_T$, where $p_{\mathcal{T}}(\boldsymbol{z}_i)$ and $p_{\mathcal{T}}(\boldsymbol{z}_i, \boldsymbol{z}_j)$ denote the marginal distributions of $p_{\mathcal{T}}(\boldsymbol{Z})$. That is, the tree-structured distribution $p_{\mathcal{T}}(\boldsymbol{Z})$ captures the neighborhood information reflected on the spanning tree $\mathbb{T}$. By using $p_{\mathcal{T}}(\boldsymbol{Z})$ to replace $p_{\mathcal{I}}(\boldsymbol{Z})$ of $\mathcal{L}_2$, it can be shown that $\mathcal{L}_2$ can be expressed as the summation of terms involving only one or two variables, which can be handled easily. Due to the limitation of space, the concrete expression for the lower bound is given in the Supplementary Material.

[1] We assume the graph is connected. For more general cases, results can be derived similarly.

4.2 Imposing Correlations on the Posterior

The posterior distribution $q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})$ in the previous section is assumed to be in independent form, as the form shown in (8). But since a prior $p_{\mathcal{T}}(\boldsymbol{Z})$ considering the correlations among documents is used, assuming an independent posterior is not appropriate. Hence, we follow the tree-structured prior and also construct a tree-structured posterior

$$q_{\mathcal{T}}(\boldsymbol{Z}|\boldsymbol{X}) = \prod_{i\in\mathcal{V}} q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i) \prod_{(i,j)\in\mathcal{E}_T} \frac{q_{\boldsymbol{\phi}}(\boldsymbol{z}_i, \boldsymbol{z}_j|\boldsymbol{x}_i, \boldsymbol{x}_j)}{q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)\, q_{\boldsymbol{\phi}}(\boldsymbol{z}_j|\boldsymbol{x}_j)},$$

where $q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)$ is the same as that in (8); and $q_{\boldsymbol{\phi}}(\boldsymbol{z}_i, \boldsymbol{z}_j|\boldsymbol{x}_i, \boldsymbol{x}_j)$ is also defined to be Gaussian, with its mean defined as $[\boldsymbol{\mu}_i; \boldsymbol{\mu}_j]$ and covariance matrix defined as

$$\begin{bmatrix} diag(\boldsymbol{\sigma}_i^2) & diag(\boldsymbol{\gamma}_{ij} \odot \boldsymbol{\sigma}_i \odot \boldsymbol{\sigma}_j) \\ diag(\boldsymbol{\gamma}_{ij} \odot \boldsymbol{\sigma}_i \odot \boldsymbol{\sigma}_j) & diag(\boldsymbol{\sigma}_j^2) \end{bmatrix}, \quad (13)$$

in which $\boldsymbol{\gamma}_{ij} \in \mathbb{R}^d$ controls the correlation strength between $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$, whose elements are restricted in $(-1, 1)$ and $\odot$ denotes the Hadamard product. By taking the correlated posterior $q_{\mathcal{T}}(\boldsymbol{Z}|\boldsymbol{X})$ into the ELBO, we obtain

$$\begin{aligned} \mathcal{L}_{\mathcal{T}} = & \sum_{i\in\mathcal{V}} \mathbb{E}_{q_{\boldsymbol{\phi}}}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_i|\boldsymbol{z}_i)] - KL(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i)||p_{\mathcal{G}}(\boldsymbol{z}_i)) \\ & - \sum_{(i,j)\in\mathcal{E}_T} \Big(KL\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i, \boldsymbol{z}_j|\boldsymbol{x}_i, \boldsymbol{x}_j)||p_{\mathcal{G}}(\boldsymbol{z}_i, \boldsymbol{z}_j)\right) \\ & - KL(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i)||p_{\mathcal{G}}(\boldsymbol{z}_i)) - KL(q_{\boldsymbol{\phi}}(\boldsymbol{z}_j)||p_{\mathcal{G}}(\boldsymbol{z}_j)) \Big), \end{aligned}$$

where we briefly denote the variational distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)$ as $q_{\boldsymbol{\phi}}(\boldsymbol{z}_i)$. Since $p_{\mathcal{G}}(\boldsymbol{z}_i)$, $p_{\mathcal{G}}(\boldsymbol{z}_i, \boldsymbol{z}_j)$, $q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)$ and $q_{\boldsymbol{\phi}}(\boldsymbol{z}_i, \boldsymbol{z}_j|\boldsymbol{x}_i, \boldsymbol{x}_j)$ are all Gaussian distributions, the KL-divergence terms above can be derived in closed-form. Moreover, it can be seen that $\mathcal{L}_{\mathcal{T}}$ involves only single or pairwise variables, thus optimizing it is as efficient as the models without considering document correlation.

With the trained model, hash codes can be obtained by binarizing the posterior mean $\boldsymbol{\mu}_i$ with a threshold, as done in (Chaidaroon and Fang, 2017). However, if without any constraint, the range of mean lies in $(-\infty, +\infty)$. Thus, if we binarize it directly, lots of information in the original representations will be lost. To alleviate this

problem, in our implementation, we parameterize the posterior mean $\boldsymbol{\mu}_i$ by a function of the form $\boldsymbol{\mu}_i = sigmoid(nn(\boldsymbol{x}_i)/\tau)$, where the outermost sigmoid function forces the mean to look like binary value and thus can effectively reduce the quantization loss, with $nn(\cdot)$ denoting a neural network function and τ controlling the slope of the sigmoid function.

4.3 Extending to Multiple Spanning Trees

Obviously, approximating the graph with a spanning tree may lose too much information. To alleviate this issue, we propose to capture the similarity information by a mixture of multiple distributions, with each built on a spanning tree. Specifically, we first construct a set of M spanning trees $\mathcal{T}_{\mathbb{G}} = \{\mathbb{T}_1, \mathbb{T}_2, \cdots, \mathbb{T}_M\}$ from the original graph $\mathbb{G}$. Based on the set of spanning trees, a mixture-distribution prior and posterior can be constructed as

$$p_{\mathcal{MT}}(\boldsymbol{Z}) = \frac{1}{M}\sum_{\mathcal{T}\in\mathcal{T}_{\mathbb{G}}} p_{\mathcal{T}}(\boldsymbol{Z}), \tag{14}$$

$$q_{\mathcal{MT}}(\boldsymbol{Z}|\boldsymbol{X}) = \frac{1}{M}\sum_{\mathcal{T}\in\mathcal{T}_{\mathbb{G}}} q_{\mathcal{T}}(\boldsymbol{Z}|\boldsymbol{X}), \tag{15}$$

where $p_{\mathcal{T}}(\boldsymbol{Z})$ and $q_{\mathcal{T}}(\boldsymbol{Z}|\boldsymbol{X})$ are the prior and posterior defined on the tree $\mathcal{T}$, as done in (11) and (13). By taking the mixture distributions above into the ELBO of $\mathcal{L}$ to replace the prior and posterior, we can obtain a new ELBO, denoted as $\mathcal{L}_{\mathcal{MT}}$. Obviously, it is impossible to obtain a closed-form expression for the bound $\mathcal{L}_{\mathcal{MT}}$. But as proved in (Tang et al., 2019), by using the log-sum inequality, $\mathcal{L}_{\mathcal{MT}}$ can be further lower bounded by

$$\widetilde{\mathcal{L}}_{\mathcal{MT}} = \frac{1}{M}\sum_{\mathcal{T}\in\mathcal{T}_{\mathbb{G}}} \mathcal{L}_{\mathcal{T}}. \tag{16}$$

Given the expression of $\mathcal{L}_{\mathcal{T}}$, the lower bound of $\widetilde{\mathcal{L}}_{\mathcal{MT}}$ can also be expressed in closed-form and optimized efficiently. For detailed derivations and concrete expressions, please refer to the Supplementary.

4.4 Details of Modeling

The parameters $\boldsymbol{\mu}_i, \boldsymbol{\mu}_j, \boldsymbol{\sigma}_i, \boldsymbol{\sigma}_j$ and $\boldsymbol{\gamma}_{ij}$ in the approximate posterior distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)$ of (8) and $q_{\boldsymbol{\phi}}(\boldsymbol{z}_i, \boldsymbol{z}_j|\boldsymbol{x}_i, \boldsymbol{x}_j)$ of (13) are all defined as the outputs of neural networks, with the parameters denoted as $\boldsymbol{\phi}$. Specifically, the entire model is mainly composed of three components:

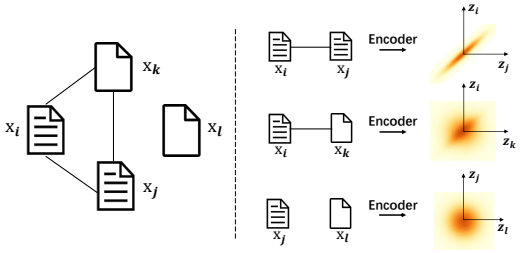


Figure 1: Illustration of how the proposed model preserves the semantic and similarity information in the representations, where the color and link represent semantic similarity and neighborhood, respectively.

i) The variational encoder $q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)$, which takes single document as input, and outputs the mean and variance of Gaussian distribution, *i.e.,* $[\boldsymbol{\mu}_i; \boldsymbol{\sigma}_i^2] = f_{\boldsymbol{\phi}}(\boldsymbol{x}_i)$;

ii) The correlated encoder, which takes pairwise documents as input, and outputs the correlation coefficient, *i.e.*, $\boldsymbol{\gamma}_{ij} = f_{\boldsymbol{\phi}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Note that the correlation encoder is required to be order-irrelevant, that is, $f_{\boldsymbol{\phi}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = f_{\boldsymbol{\phi}}(\boldsymbol{x}_j, \boldsymbol{x}_i)$, which is achieved in this paper as $f_{\boldsymbol{\phi}} = \frac{1}{2}\big(f_{\boldsymbol{\phi}}(\boldsymbol{x}_i, \boldsymbol{x}_j) + f_{\boldsymbol{\phi}}(\boldsymbol{x}_j, \boldsymbol{x}_i)\big)$;

iii) The generative decoder $p_{\boldsymbol{\theta}}(\boldsymbol{x}_i|\boldsymbol{z}_i)$, which takes the latent variable $\boldsymbol{z}_i$ as input and output the document $\boldsymbol{x}_i$. The decoder is modeled by a neural network parameterized by $\boldsymbol{\theta}$.

The model is trained by optimizing the lower bound $\widetilde{\mathcal{L}}_{\mathcal{MT}}$ w.r.t. $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$. After training, hash codes are obtained by passing the documents through the variational encoder and binarizing the outputs on every dimension by a the threshold value, which is simply set as 0.5 in our experiments.

To intuitively understand the insight behind our model, an illustration is shown in Figure 1. We see that if the two documents are neighbors and semantically similar, the representations will be strongly correlated to each other. But if they are not semantically similar neighbors, the representations become less correlated. If they are neither neighbors nor semantically similar, the representations become not correlated at all. Since our model can simultaneously preserve semantics and neighborhood information, we name it as **S**emantics-**N**eighborhood **U**nified **H**ahing (SNUH).

5 Related Work

Deep generative models (Rezende et al., 2014) have attracted a lot of attention in semantics-based hashing, due to their successes in unsuper-

vised representation learning. VDSH (Chaidaroon and Fang, 2017) first employed variational auto-encoder (VAE) (Kingma and Welling, 2013) to learn continuous representations of documents and then casts them into binary codes. However, for the sake of information leaky problem during binarization step, such a two-stage strategy is prone to result in local optima and undermine the performance. NASH (Shen et al., 2018) tackled this issue by replacing the Gaussian prior with Bernoulli and adopted the straight-through technique (Bengio et al., 2013) to achieve end-to-end training. To further improve the model's capability, Dong et al. (2019) proposed to employ mixture distribution as a priori knowledge and Zheng et al. (2020) exploited Boltzmann posterior to introduce correlation among bits. Beyond generative frameworks, AMMI (Stratos and Wiseman, 2020) achieved superior performance by maximizing the mutual information between codes and documents. Nevertheless, the aforementioned semantic hashing methods are consistently under the *i.i.d.* assumption, which means they ignore the neighborhood information.

Spectral hashing (Weiss et al., 2009) and self-taught hashing (Zhang et al., 2010) are two typical methods of neighbor-based hashing models, which both formulated the hashing problem as graph partitioning (Hagen and Kahng, 1992) by preserving local similarities. But these algorithms generally ignore the rich semantic information associated with documents. Recently, some VAE-based models tried to concurrently take account of semantic and neighborhood information, such as NbrReg (Chaidaroon et al., 2018), RBSH (Hansen et al., 2019) and PairRec(Hansen et al., 2020). However, as mentioned before, all of them simply regarded the proximity as regularization, lacking theoretical principles to guide the incorporation process. Thanks to the virtue of graph-induced distribution, we effectively preserve the two types of information in a theoretical framework.

6 Experiments

6.1 Experiment Setup

Datasets We verify the proposed methods on three public datasets which published by VDSH[2]: i) Reuters25178, which contains 10,788 news documents with 90 different categories; ii) TMC, which is a collection of 21,519 air traffic reports with 22 different categories; iii) 20Newsgroups (NG20), which consists of 18,828 news posts from 20 different topics. Note that the category labels of each dataset are only used to compute the evaluation metrics, as we focus on unsupervised scenarios.

[2]https://github.com/unsuthee/VariationalDeepSemanticHashing/tree/master/dataset

Baselines We compare our method with the following models: SpH (Weiss et al., 2009), STH (Zhang et al., 2010), VDSH (Chaidaroon and Fang, 2017), NASH (Shen et al., 2018), GMSH(Dong et al., 2019), NbrReg (Chaidaroon et al., 2018), CorrSH (Zheng et al., 2020) and AMMI (Stratos and Wiseman, 2020). For all baselines, we take the reported performance from their original papers.

Training Details For fair comparisons, we follow the same network architecture used in VDSH, GMSH and CorrSH, using a one-layer feed-forward neural network as the variational and the correlated encoder. The graph $\mathbb{G}$ is constructed with the K-nearest neighbors (KNN) algorithm based on cosine similarity on the TFIDF features of documents. In our experiments, the number of trees M and neighbors K are both fixed as 10, and the correlation strength coefficient λ in (12) is fixed to 0.99. According to the performance observed on the validation set, we choose the learning rate from $\{0.0005, 0.001, 0.003\}$, batch size from $\{32, 64, 128\}$, and the temperature τ in sigmoid function from $\{0.1, 0.2, \cdots, 1\}$, with the best used for evaluation on the test set. The model is trained using the Adam optimizer (Kingma and Ba, 2014). More detailed experimental settings, along with the generating method of spanning trees, are given in the supplementary materials.

Evaluation Metrics The retrieval precision is used as our evaluation metric. For each query document, we retrieve 100 documents most similar to it based on the Hamming distance of hash codes. Then, the retrieval precision for a single sample is measured as the percentage of the retrieved documents with the same label as the query. Finally, the average precision over the whole test set is calculated as the performance of the evaluated method.

6.2 Performance and Analysis

Overall Performance The performances of all the models on the three public datasets are shown in Table 1. We see that our model performs favorably to the current state-of-the-art method, yielding best average performance across different datasets and settings. Compared with VDSH and NASH, which

Method	Reuters				TMC				20Newsgroups				Avg
	16bits	32bits	64bits	128bits	16bits	32bits	64bits	128bits	16bits	32bits	64bits	128bits	
SpH	0.6340	0.6513	0.6290	0.6045	0.6055	0.6281	0.6143	0.5891	0.3200	0.3709	0.3196	0.2716	0.5198
STH	0.7351	0.7554	0.7350	0.6986	0.3947	0.4105	0.4181	0.4123	0.5237	0.5860	0.5806	0.5443	0.5662
VDSH	0.7165	0.7753	0.7456	0.7318	0.6853	0.7108	0.4410	0.5847	0.3904	0.4327	0.1731	0.0522	0.5366
NbrReg	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	0.4120	0.4644	0.4768	0.4893	0.4249
NASH	0.7624	0.7993	0.7812	0.7559	0.6573	0.6921	0.6548	0.5998	0.5108	0.5671	0.5071	0.4664	0.6462
GMSH	0.7672	0.8183	0.8212	0.7846	0.6736	0.7024	0.7086	0.7237	0.4855	0.5381	0.5869	0.5583	0.6807
AMMI	0.8173	0.8446	0.8506	0.8602	0.7096	0.7416	0.7522	0.7627	0.5518	0.5956	0.6398	0.6618	0.7323
CorrSH	0.8212	0.8420	0.8465	0.8482	0.7243	0.7534	0.7606	0.7632	**0.5839**	0.6183	0.6279	0.6359	0.7355
SNUH	**0.8320**	**0.8466**	**0.8560**	**0.8624**	**0.7251**	**0.7543**	**0.7658**	**0.7726**	0.5775	**0.6387**	**0.6646**	**0.6731**	**0.7474**

Table 1: The precision on three datasets with different numbers of bits in unsupervised document hashing.

Ablation Study		16bits	32bits	64bits	128bits
Reuters	SNUH_{ind}	0.7823	0.8094	0.8180	0.8385
	$\text{SNUH}_{\text{prior}}$	0.8043	0.8295	0.8431	0.8460
	SNUH	**0.8320**	**0.8466**	**0.8560**	**0.8624**
TMC	SNUH_{ind}	0.6978	0.7307	0.7421	0.7526
	$\text{SNUH}_{\text{prior}}$	0.7177	0.7408	0.7518	0.7528
	SNUH	**0.7251**	**0.7543**	**0.7658**	**0.7726**
NG20	SNUH_{ind}	0.4806	0.5503	0.6017	0.6060
	$\text{SNUH}_{\text{prior}}$	0.5443	0.6071	0.6212	0.6014
	SNUH	**0.5775**	**0.6387**	**0.6646**	**0.6731**

Table 2: The performance of variant models. SNUH_{ind} and $\text{SNUH}_{\text{prior}}$ indicate the model without considering any document correlations (independent) and only considering correlations in the prior, respectively.

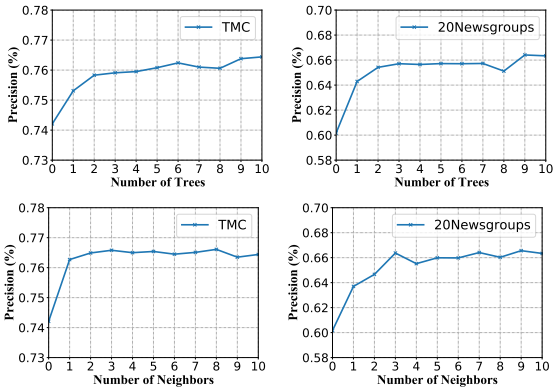


Figure 2: The precision of 64-bit hash codes on TMC and 20Newsgroups datasets with varying number of trees M and neighbors K.

simply employ isotropic Gaussian and Bernoulli prior, respectively, we can observe that our model, which leverages correlated prior and posterior distributions, achieves better results on all the three datasets. Although GMSH improves performance by exploiting a more expressive Gaussian mixture prior, our model still outperforms it by a substantial margin, indicating the superiority of incorporating document correlations. It is worth noting that, by unifying semantics and neighborhood information under the generative models, the two types of information can be preserved more effectively. This can be validated by that our model performs significantly better than NbrReg, which naively incorporates the neighborhood information by using a neighbor-reconstruction regularizer. The superiority of our unified method can be further corroborated in the comparisons with RBSH and PairRec, which are given in the Supplementary since they employed a different preprocessing method as the models reported here. Comparing to the current SOTA methods of AMMI and CorrSh, our method is still able to achieve better results by exploiting the correlation among documents. Moreover, thanks to the benefit of correlation regularization, remarkable gratuity can be acquired profitably in 64 and 128 bits.

Impact of Introducing Correlations in Prior and Posterior To understand the influences of the proposed document-correlated prior and posterior, we further experiment with two variants of our model: i) SNUH_{ind}: which does not consider document correlations in neither the prior nor the posterior distribution; ii) $\text{SNUH}_{\text{prior}}$: which only considers the correlations in the prior, but not in the posterior. Obviously, the proposed SNUH represents the method that leverage the correlations in both of the prior and posterior. As seen from Table 2, $\text{SNUH}_{\text{prior}}$ achieves better performance than SNUH_{ind}, demonstrating the benefit of considering the correlation information of documents only in the prior. By further taking the correlations into account in the posterior, improvements of SNUH can be further observed, which fully corroborates the superiority of considering document correlations in the prior and posterior. Another interesting observation is that the performance gap between SNUH_{ind} and $\text{SNUH}_{\text{prior}}$ becomes small as the length of bits increases. This may be attributed

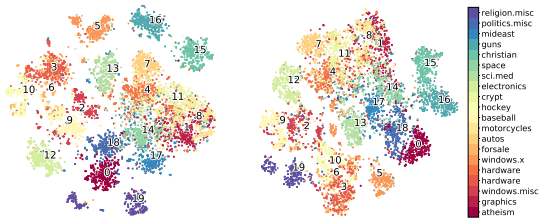
Distance	Category	Title/Subject
query	**hockey**	**NHL PLAYOFF RESULTS FOR GAMES PLAYED 4-21-93**
1	hockey	NHL PLAYOFF RESULTS FOR GAMES PLAYED 4-19-93
10	hockey	NHL Summary parse results for games played Thur, April 15, 1993
20	hockey	AHL playoff results (4/15)
50	forsale	RE: == MOVING SALE ===
70	hardware	Re: Quadra SCSI Problems?
90	politics.misc	Re: Employment (was Re: Why not concentrate on child molesters?

Table 3: Qualitative analysis of the learned 128-bit hash codes on the 20Newsgroups dataset. We present the documents with Hamming distance of 1, 10, 20, 50, 70 and 90 to the query.

to the fact that the increased generalization ability of models brought by large bits is inclined to alleviate the impact of priori knowledge. However, by additionally incorporating correlation constraints on posterior, significant performance gains would be obtained, especially in large bits scenarios.

Effect of Spanning Trees For more efficient training, spanning trees are utilized to approximate the whole graph by dropping out some edges. To understand its effects, we first investigate the *impact of the number of trees*. The first row of Figure 4 shows the performance of our method as a function of different numbers of spanning trees. We observe that, compared to not using any correlation, one tree alone can bring significant performance gains. As the tree number increases, the performance rises steadily at first and then converges into a certain level, demonstrating that the document correlations can be mostly captured by several spanning trees. Then, we further explore the *impact of the neighbor number* when constructing the graphs using the KNN method, as shown in the second row of Figure 4. It can be seen that more neighbors contributes to better performance. We hypothesize that this is partly due to the more diverse correlation information captured by the increasing number of neighbors. However, incorporating too many neighbors may lead to the problem of introducing noise and incorrect correlation information to the hash codes. That explains why no further improvement is observed after the number reaches a level.

Empirical Study of Computational Efficiency We also investigate the training complexity by comparing the training duration of our method and VDSH, on Tesla V100-SXM2-32GB. On the Reuters, TMC, 20Newsgroups datasets with 64-bit hash codes, our method finishes one epoch of training respectively in 3.791s, 5.238s, 1.343s and VDSH in 2.038s, 4.364s, 1.051s. It can be seen that our model, though with much stronger performance, can be trained almost as efficiently as vanilla VDSH due to the tree approximations.



(a) SNUH (b) AMMI

Figure 3: Visualization of the 64-dimensional latent semantic embeddings learned by the proposed models for the 20Newsgroups dataset.

Case Study In Table 3, we present a retrieval case of the given query document. It can be observed that as the Hamming distance increases, the semantic (topic) of the retrieved document gradually becomes more irrelevant, illustrating that the Hamming distance can effectively measure the document relevance.

Visualization of Hash Codes To evaluate the quality of generated hash code more intuitively, we project the latent representations into a 2-dimensional plane with the t-SNE (van der Maaten and Hinton, 2008) technique. As shown in Figure 3, the representations generated by our method are more separable than those of AMMI, demonstrating the superiority of our method.

7 Conclusion

We have proposed an effective and efficient semantic hashing method to preserve both the semantics and neighborhood information of documents. Specifically, we applied a graph-induced Gaussian prior to model the two types of information in a unified framework. To facilitate training, a tree-structure approximation was further developed to decompose the ELBO into terms involving only singleton or pairwise variables. Extensive evaluations demonstrated that our model significantly outperforms baseline methods by incorporating both the semantics and neighborhood information.

References

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.

Suthee Chaidaroon, Travis Ebesu, and Yi Fang. 2018. Deep semantic text hashing with weak supervision. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1109–1112.

Suthee Chaidaroon and Yi Fang. 2017. Variational deep semantic hashing for text documents. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 75–84.

Wei Dong, Qinliang Su, Dinghan Shen, and Changyou Chen. 2019. Document hashing with mixture-prior generative models. *arXiv preprint arXiv:1908.11078*.

Lars Hagen and Andrew B Kahng. 1992. New spectral methods for ratio cut partitioning and clustering. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 11(9):1074–1085.

Casper Hansen, Christian Hansen, Jakob Grue Simonsen, Stephen Alstrup, and Christina Lioma. 2019. Unsupervised neural generative semantic hashing. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Casper Hansen, Christian Hansen, Jakob Grue Simonsen, Stephen Alstrup, and Christina Lioma. 2020. Unsupervised semantic hashing with pairwise reconstruction. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Xiaofei He, Deng Cai, Haifeng Liu, and Wei-Ying Ma. 2004. Locality preserving indexing for document representation. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 96–103.

Yushi Jing and Shumeet Baluja. 2008. VisualRank: Applying PageRank to large-scale image search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1877–1890.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Yehuda Koren. 2008. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 426–434.

Peng Li, Meng Wang, Jian Cheng, Changsheng Xu, and Hanqing Lu. 2012. Spectral hashing with semantically consistent graph for image indexing. *IEEE Transactions on Multimedia*, 15(1):141–152.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*.

Ruslan Salakhutdinov and Geoffrey Hinton. 2009. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978.

Dinghan Shen, Qinliang Su, Paidamoyo Chapfuwa, Wenlin Wang, Guoyin Wang, Lawrence Carin, and Ricardo Henao. 2018. NASH: Toward end-to-end neural architecture for generative semantic hashing. *arXiv preprint arXiv:1805.05361*.

Yuming Shen, Li Liu, and Ling Shao. 2019. Unsupervised binary representation learning with deep variational networks. *International Journal of Computer Vision*, 127(11):1614–1628.

Benno Stein, Sven Meyer zu Eissen, and Martin Potthast. 2007. Strategies for retrieving plagiarized documents. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 825–826.

Karl Stratos and Sam Wiseman. 2020. Learning discrete structured representations by adversarially maximizing mutual information. *arXiv preprint arXiv:2004.03991*.

Da Tang, Dawen Liang, Tony Jebara, and Nicholas Ruozzi. 2019. Correlated variational auto-encoders. *arXiv preprint arXiv:1905.05335*.

Martin J Wainwright and Michael Irwin Jordan. 2008. *Graphical models, exponential families, and variational inference*. Now Publishers Inc.

Yair Weiss, Antonio Torralba, and Rob Fergus. 2009. Spectral hashing. In *Advances in Neural Information Processing Systems*, pages 1753–1760.

Dell Zhang, Jun Wang, Deng Cai, and Jinsong Lu. 2010. Self-taught hashing for fast similarity search. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 18–25.

Yanhao Zhang, Pan Pan, Yun Zheng, Kang Zhao, Yingya Zhang, Xiaofeng Ren, and Rong Jin. 2018. Visual search at Alibaba. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 993–1001.

Kang Zhao, Hongtao Lu, and Jincheng Mei. 2014. Locality preserving hashing. In *Twenty-eighth AAAI Conference on Artificial Intelligence*.

Lin Zheng, Qinliang Su, Dinghan Shen, and Changyou Chen. 2020. Generative semantic hashing enhanced via boltzmann machines. *arXiv preprint arXiv:2006.08858*.

A Derivation of Formulas

Derivation of $KL\left(q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})||p_{\mathcal{G}}(\boldsymbol{Z})\right)$ We provide detail to show how to get the exact expression of $KL\left(q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})||p_{\mathcal{G}}(\boldsymbol{Z})\right)$. In the main paper, we define $q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})$ as a diagonal Gaussian distribution and

$$p_{\mathcal{G}}(\boldsymbol{Z}) = \mathcal{N}\left(\boldsymbol{Z}; \boldsymbol{0}, (\boldsymbol{I}_N + \tau_{ij}\boldsymbol{A}) \otimes \boldsymbol{I}_d\right).$$

Therefore, we have

$$\begin{aligned}
&KL\left(q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})||p_{\mathcal{G}}(\boldsymbol{Z})\right) \\
&= -\sum_{k=1}^{N}\mathcal{I}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)\right) - \int q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})\log p_{\mathcal{G}}(\boldsymbol{Z})d\boldsymbol{Z} \\
&= -\sum_{k=1}^{N}\mathcal{I}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)\right) \\
&\quad -\int q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})\log\frac{1}{2\pi|\left(\boldsymbol{I}_N + \tau_{ij}\boldsymbol{A}\right)\otimes\boldsymbol{I}_d|^{1/2}}d\boldsymbol{Z} \\
&\quad -\int q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})\Big(-\frac{1}{2}\boldsymbol{Z}^T\big((\boldsymbol{I}_N+\tau_{ij}\boldsymbol{A})\otimes\boldsymbol{I}_d\big)^{-1}\boldsymbol{Z}\Big)d\boldsymbol{Z} \\
&= -\sum_{k=1}^{N}\mathcal{I}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)\right) + \frac{1}{2}\log|2\pi(\boldsymbol{I}_N+\tau_{ij}\boldsymbol{A})\otimes\boldsymbol{I}_d| \\
&\quad +\frac{1}{2}tr\Big(\big((\boldsymbol{I}_N+\tau_{ij}\boldsymbol{A})\otimes\boldsymbol{I}_d\big)^{-1}\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})}\big[\boldsymbol{Z}\boldsymbol{Z}^T\big]\Big),
\end{aligned}$$

where $\mathcal{I}(\cdot)$ denotes the entropy function; and $tr(\cdot)$ means the trace function.

Derivation of $KL\left(q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})||p_{\mathcal{T}}(\boldsymbol{Z})\right)$ In the main paper, we propose a tree-type distribution to introduce partial neighborhood information so that the $\mathcal{L}_2$ term can be expressed as the summation over terms involving only one or two variables. Here, we provide the detail derivation. Specifically, the $p_{\mathcal{T}}(\boldsymbol{Z})$ is defined as

$$p_{\mathcal{T}}(\boldsymbol{Z}) = \prod_{i\in\mathcal{V}} p_{\mathcal{G}}(\boldsymbol{z}_i) \prod_{(i,j)\in\mathcal{E}_T}\frac{p_{\mathcal{G}}(\boldsymbol{z}_i,\boldsymbol{z}_j)}{p_{\mathcal{G}}(\boldsymbol{z}_i)p_{\mathcal{G}}(\boldsymbol{z}_j)}.$$

Therefore, we have

$$\begin{aligned}
&KL\left(q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})||p_{\mathcal{T}}(\boldsymbol{Z})\right) \\
&= \int q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X})\log\frac{\prod_{i\in\mathcal{V}} q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)}{\prod_{i\in\mathcal{V}} p_{\mathcal{G}}(\boldsymbol{z}_i)\prod_{(i,j)\in\mathcal{E}_T}\frac{p_{\mathcal{G}}(\boldsymbol{z}_i,\boldsymbol{z}_j)}{p_{\mathcal{G}}(\boldsymbol{z}_i)p_{\mathcal{G}}(\boldsymbol{z}_j)}}d\boldsymbol{Z} \\
&= \sum_{i\in\mathcal{V}} KL\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)||p_{\boldsymbol{\theta}}(\boldsymbol{z}_i)\right) \\
&\quad - \sum_{(i,j)\in\mathcal{E}_T}\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_i,\boldsymbol{z}_j|\boldsymbol{x}_i,\boldsymbol{x}_j)}\left[\log\frac{p_{\mathcal{G}}(\boldsymbol{z}_i)p_{\mathcal{G}}(\boldsymbol{z}_j)}{p_{\mathcal{G}}(\boldsymbol{z}_i,\boldsymbol{z}_j)}\right].
\end{aligned}$$

Obviously, the KL divergence is decomposed into the terms involving singleton and pairwise variables, which can be calculated efficiently.

Expressing $\mathcal{L}_{\mathcal{T}}$ in Analytical Form Using the KL divergence for multivariate Gaussian: suppose both q and p are the probability density functions of multivariate normal distributions over $\mathbb{R}^d$ with mean $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2$ and $\boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2$, respectively. The KL divergence from q to p is

$$\frac{1}{2}\Big[\log\frac{|\boldsymbol{\Sigma}_2|}{|\boldsymbol{\Sigma}_1|} - d + tr\{\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1\} + (\boldsymbol{\mu}_2-\boldsymbol{\mu}_1)^T\boldsymbol{\Sigma}_2^{-1}(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1)\Big].$$

Through the direct application, we have

$$KL(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)||p_{\mathcal{G}}(\boldsymbol{z}_i)) = \frac{1}{2}\sum_{n=1}^{d}(\boldsymbol{\mu}_{in}^2 + \boldsymbol{\sigma}_{in}^2 - 1 - 2\log\boldsymbol{\sigma}_{in}).$$

For simplification, in the following, we use $\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1$ to represent the mean and variance matrix of $q_{\mathcal{T}}(\boldsymbol{z}_i, \boldsymbol{z}_j|\boldsymbol{x}_i, \boldsymbol{x}_j)$, respectively, and represent those of $p_{\mathcal{G}}(\boldsymbol{z}_i, \boldsymbol{z}_j)$ as $\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2$, respectively. Besides we denote λa_{ij} as τ_{ij} so we have $\tau_{ij} = \lambda a_{ij} = \lambda a_{ji}$.

Since it is a bit complicated to compute the KL divergence from $q_{\boldsymbol{\phi}}(\boldsymbol{z}_i, \boldsymbol{z}_j|\boldsymbol{x}_i, \boldsymbol{x}_j)$ to $p_{\mathcal{G}}(\boldsymbol{z}_i, \boldsymbol{z}_j)$, we deduce it step by step. First we apply the Cholesky decomposition on the covariance matrix of $p_{\mathcal{G}}(\boldsymbol{z}_i, \boldsymbol{z}_j)$

$$\boldsymbol{\Sigma}_2 = \begin{bmatrix} \boldsymbol{I}_d & 0 \\ \tau_{ij}\boldsymbol{I}_d & \sqrt{1-\tau_{ij}^2}\boldsymbol{I}_d \end{bmatrix}\begin{bmatrix} \boldsymbol{I}_d & \tau_{ij}\boldsymbol{I}_d \\ 0 & \sqrt{1-\tau_{ij}^2}\boldsymbol{I}_d \end{bmatrix},$$

Similarly, we also apply the Cholesky decomposition on the covariance matrix of $q_{\mathcal{T}}(\boldsymbol{z}_i, \boldsymbol{z}_j|\boldsymbol{x}_i, \boldsymbol{x}_j)$

$$\boldsymbol{\Sigma}_1 = \begin{bmatrix} \boldsymbol{\sigma}_i & \boldsymbol{0}_d \\ \boldsymbol{\gamma}_{ij}\boldsymbol{\sigma}_j & \sqrt{1-\boldsymbol{\gamma}_{ij}^2}\boldsymbol{\sigma}_j \end{bmatrix}\begin{bmatrix} \boldsymbol{\sigma}_i & \boldsymbol{\gamma}_{ij}\boldsymbol{\sigma}_j \\ \boldsymbol{0}_d & \sqrt{1-\boldsymbol{\gamma}_{ij}^2}\boldsymbol{\sigma}_j \end{bmatrix},$$

where we omit $diag(\cdot)$ for simplifying. Then, we have

$$\log\frac{|\boldsymbol{\Sigma}_2|}{|\boldsymbol{\Sigma}_1|}=\sum_{n=1}^{d}\log(1-\tau_{ij}^2)-\big(\sum_{n=1}^{d}\log\boldsymbol{\sigma}_{in}^2+\log\boldsymbol{\sigma}_{jn}^2+\log(1-\boldsymbol{\gamma}_{ijn}^2)\big)$$

$$tr\{\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1\}=\sum_{i=n}^{d}\frac{\boldsymbol{\sigma}_{in}^2+\boldsymbol{\sigma}_{jn}^2-2\tau_{ij}\gamma\boldsymbol{\sigma}_{in}\boldsymbol{\sigma}_{jn}}{1-\tau_{ij}^2}$$

$$(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1)^T\boldsymbol{\Sigma}_2^{-1}(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1)=\sum_{n=1}^{d}\frac{\boldsymbol{\mu}_{in}^2+\boldsymbol{\mu}_{jn}^2-2\tau_{ij}\boldsymbol{\mu}_{in}\boldsymbol{\mu}_{jn}}{1-\tau_{ij}^2}.$$

Hence, the KL divergence from $q_{\boldsymbol{\phi}}(\boldsymbol{z}_i,\boldsymbol{z}_j|\boldsymbol{x}_i,\boldsymbol{x}_j)$ to $p_{\mathcal{G}}(\boldsymbol{z}_i,\boldsymbol{z}_j)$ is

$$\begin{aligned}&KL\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i,\boldsymbol{z}_j|\boldsymbol{x}_i,\boldsymbol{x}_j)||p_0(\boldsymbol{z}_i,\boldsymbol{z}_j)\right)\\&=\frac{1}{2}\sum_{n=1}^{d}\Big\{\log(1-\tau_{ij}^2)\\&-\big(\log\boldsymbol{\sigma}_{in}^2+\log\boldsymbol{\sigma}_{jn}^2+\log(1-\boldsymbol{\gamma}_{ijn}^2)\big)-2\\&+\frac{\boldsymbol{\sigma}_{in}^2+\boldsymbol{\sigma}_{jn}^2-2\tau_{ij}\boldsymbol{\gamma}_{ijn}\boldsymbol{\sigma}_{in}\boldsymbol{\sigma}_{jn}+\boldsymbol{\mu}_{in}^2+\boldsymbol{\mu}_{jn}-2\tau_{ij}\boldsymbol{\mu}_{in}\boldsymbol{\mu}_{jn}}{1-\tau_{ij}^2}\Big\}.\end{aligned}$$

Then, we can express $\mathcal{L}_{\mathcal{T}}$ in an analytical form

$$\begin{aligned}\mathcal{L}_{\mathcal{T}}=&\sum_{i\in\mathcal{V}}\Big(\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_i|\boldsymbol{z}_i)]-\frac{1}{2}\sum_{n=1}^{d}(\boldsymbol{\mu}_{in}^2\\&+\boldsymbol{\sigma}_{in}^2-1-2\log\boldsymbol{\sigma}_{in})\Big)-\sum_{(i,j)\in\mathcal{E}_{\mathcal{T}}}\Big(\frac{1}{2}\sum_{n=1}^{d}\Big\{\log(1-\tau_{ij}^2)\\&-\big(\boldsymbol{\mu}_{in}^2+\boldsymbol{\mu}_{jn}^2+\boldsymbol{\sigma}_{in}^2+\boldsymbol{\sigma}_{jn}^2+\log(1-\boldsymbol{\gamma}_{ijn}^2)\big)\\&+\frac{\boldsymbol{\sigma}_{in}^2+\boldsymbol{\sigma}_{jn}^2-2\tau_{ij}\boldsymbol{\gamma}_{ijn}\boldsymbol{\sigma}_{in}\boldsymbol{\sigma}_{jn}+\boldsymbol{\mu}_{in}^2+\boldsymbol{\mu}_{jn}-2\tau_{ij}\boldsymbol{\mu}_{in}\boldsymbol{\mu}_{jn}}{1-\tau_{ij}^2}\Big\}\Big)\end{aligned}$$

Derivation of $\widetilde{\mathcal{L}}_{\mathcal{MT}}$ With $\mathcal{L}_{\mathcal{MT}}$, we extend the single-tree approximation to multi-tree approximation. Although the KL divergence between the mixture distributions does not have a closed-form solution, we can obtain its explicit upper bound by using the log-sum inequality as

$$\begin{aligned}\mathcal{L}_{\mathcal{MT}}&=\mathbb{E}_{q_{\mathcal{MT}}(\boldsymbol{Z}|\boldsymbol{X})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{X}|\boldsymbol{Z})]-KL\left(q_{\mathcal{MT}}(\boldsymbol{Z}|\boldsymbol{X})||p_{\mathcal{MT}}(\boldsymbol{Z})\right)\\&\geq\frac{1}{M}\sum_{\mathcal{T}\in\mathcal{T}_{\mathbb{G}}}\mathbb{E}_{q_{\mathcal{T}}(\boldsymbol{Z}|\boldsymbol{X})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{X}|\boldsymbol{Z})]-\frac{1}{M}\sum_{\mathcal{T}\in\mathcal{T}_{\mathbb{G}}}KL\left(q_{\mathcal{T}}(\boldsymbol{Z}|\boldsymbol{X})||p_{\mathcal{T}}(\boldsymbol{X})\right)\\&\triangleq\widetilde{\mathcal{L}}_{\mathcal{MT}}.\end{aligned}$$

We can further express $\widetilde{\mathcal{L}}_{\mathcal{MT}}$ in a more intuitive form as

$$\begin{aligned}&\sum_{i\in\mathcal{V}}\Big(\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_i|\boldsymbol{z}_i)]-KL(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)||p_{\mathcal{G}}(\boldsymbol{z}_i))\Big)\\&-\sum_{(i,j)\in\mathcal{E}_{\mathcal{T}}}w_{ij}\Big(KL(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i,\boldsymbol{z}_j|\boldsymbol{x}_i,\boldsymbol{x}_j)||p_{\mathcal{G}}(\boldsymbol{x}_i,\boldsymbol{x}_j))\\&-KL(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)||p_{\mathcal{G}}(\boldsymbol{z}_i))-KL(q_{\boldsymbol{\phi}}(\boldsymbol{z}_j|\boldsymbol{x}_j)||p_{\mathcal{G}}(\boldsymbol{z}_j))\Big),\end{aligned}$$

where $w_{ij}=\frac{|\{\mathcal{T}\in\mathcal{T}_{\mathbb{G}}|(i,j)\in\mathcal{E}_{\mathcal{T}}\}|}{M}$ denotes the proportion of times that the edge (i,j) appears. Using the analytical expression of $\mathcal{L}_{\mathcal{T}}$, the KL divergence of $\widetilde{\mathcal{L}}_{\mathcal{MT}}$ can be expressed in an analytical form.

To optimize this objective, we can construct an estimator of the ELBO of the full dataset, based on the minibatch

$$\begin{aligned}\widetilde{\mathcal{L}}_{\mathcal{MT}}&\simeq\widetilde{\mathcal{L}}_{\mathcal{MT}}^{M}\\&=\sum_{i\in\mathcal{V}^M}\mathcal{L}_{\mathcal{V}^M}(\boldsymbol{x}_i)-\sum_{(i,j)\in\mathcal{E}_T^M}w_{ij}\mathcal{L}_{\mathcal{E}_T^M}(\boldsymbol{x}_i,\boldsymbol{x}_j),\end{aligned}$$

where $\mathcal{V}^M$ is the subset of documents, $\mathcal{E}_T^M$ is the subset of edges and

$$\begin{aligned}\mathcal{L}_{\mathcal{V}^M}(\boldsymbol{x}_i)&\triangleq\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_i|\boldsymbol{z}_i)]-KL\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)||p_{\mathcal{G}}(\boldsymbol{z}_i)\right);\\\mathcal{L}_{\mathcal{E}_T^M}(\boldsymbol{x}_i,\boldsymbol{x}_j)&\triangleq KL(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i,\boldsymbol{z}_j|\boldsymbol{x}_i,\boldsymbol{x}_j)||p_{\mathcal{G}}(\boldsymbol{x}_i,\boldsymbol{x}_j))\\&-KL(q_{\boldsymbol{\phi}}(\boldsymbol{z}_i|\boldsymbol{x}_i)||p_{\mathcal{G}}(\boldsymbol{z}_i))-KL(q_{\boldsymbol{\phi}}(\boldsymbol{z}_j|\boldsymbol{x}_j)||p_{\mathcal{G}}(\boldsymbol{z}_j)).\end{aligned}$$

Then we can update the parameters by using the gradient $\nabla_{\boldsymbol{\phi},\boldsymbol{\theta}}\widetilde{\mathcal{L}}_{\mathcal{MT}}^{M}$. The training procedure is summarized in Algorithm 1.

B Tree Generation Algorithm

Algorithm 2 shows the spanning tree generation algorithm TreeGen(·) used in our graph-induced generative document hashing model. TreeGen(·) utilizes a depth-first search (DFS) algorithm to generate meaningful neighborhood information for each node. In this algorithm, $RC_{[\cdot]}$ means randomly choosing one index according to the indicator function; $ID_{[\cdot]}$ represents the set of node indexes satisfying the indicator condition and $\mathcal{N}(i)$ denotes the neighbors of node i. Due to the importance of edges precision, when choosing a neighbor (line 16 in Algorithm 2), instead of using uniform sampling, we exploit a temperature α to control the trade-off between the precision and diversity of

Algorithm 1 Model Training Algorithm

Input: Document representations $\boldsymbol{X}$; edges list of spanning trees $\boldsymbol{E}$; batch size b.
Output: Optimal parameters $(\boldsymbol{\theta}, \boldsymbol{\phi})$.

$\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Initialize parameters
repeat
  $\mathcal{V}^M \leftarrow \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_b\} \sim \boldsymbol{X}$ ▷ Sample nodes
  $\mathcal{E}_T^M \leftarrow \{e_1, \cdots, e_b\} \sim \boldsymbol{E}$ ▷ Sample endges
  $\boldsymbol{g} \leftarrow \nabla_{\boldsymbol{\phi},\boldsymbol{\theta}} \widetilde{\mathcal{L}}_{\mathcal{MT}}^M(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathcal{V}^M, \mathcal{E}_T^M)$
  $\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Update parameters using gradients $\boldsymbol{g}$ (*e.g.*, Adam optimizer)
until convergence of parameters $(\boldsymbol{\theta}, \boldsymbol{\phi})$

Input	document pair $(\boldsymbol{x}_i; \boldsymbol{x}_j)$		
	Variational Enc		**Correlated Enc**
Encoder	Linear($\|V\|, d$) $\boldsymbol{\mu} = f(\cdot/\tau)$	Linear($\|V\|, d$) $\boldsymbol{\sigma} = g(\cdot)$	Linear($2\|V\|, d$) $\boldsymbol{\gamma} = 2 * f(\cdot) - 1$
Generator	Linear($d, \|V\|$)		

Table 4: The neural network architecture of the proposed model, in which $f(\cdot)$ and $g(\cdot)$ represent the sigmoid and softplus function, respectively.

edges. Specifically, the probability of sampling neighbor j of node i is

$$\frac{exp(cos(\boldsymbol{x}_j^T \boldsymbol{x}_i)/\alpha)}{\sum_{n \in \mathcal{N}(i)} exp(cos(\boldsymbol{x}_n^T \boldsymbol{x}_i)/\alpha)}.$$

We find the best configuration of α on the validation set with the values in $\{0.1, 0.2, \cdots, 1\}$.

C Experiment Details

For fair comparisons, we follow the experimental setting of VDSH. Specifically, the vocabulary size $|V|$ is 7164, 20000, and 10000 for Reuters, TMC and 20Newsgroups, respectively. The split of training, validation, and test set is as follows: 7752, 967, 964 for Reuters; 21286, 3498, 3498 for TMC and 11016, 3667, 3668 for 20Newsgroups, respectively. Moreover, the KL term in Eq. (18) of the main paper is weighted with a coefficient β to avoid posterior collapse. We find the best configuration of β on the validation set with the values in $\{0.01, 0.02, \cdots, 0.1\}$. To intuitively understand our model, we illustrate the whole architecture in Table 4.

D Additional Experiments

Effect of Spanning Trees Figure 4 shows the impact of using different numbers of spanning trees and neighbors on the Reuters dataset. We see that using one spanning tree alone can achieve significant performance gains, indicating the advantage of considering document correlations. Similar trend can be observed in using different number of neighbors while constructing the KNN graphs.

Algorithm 2 Spanning Tree Generation Algorithm

Input: Graph $\mathbb{G}$; number of trees n.
Output: Edges list of spanning trees $\boldsymbol{E}$.

procedure TREEGEN(n) ▷ Input: #tree n
  $\boldsymbol{E} = []$ ▷ Initial edges list
  for $k \leftarrow 0, \cdots, n-1$ **do**
    $\boldsymbol{V} = [False]^{|\mathcal{V}|}$ ▷ Visited node list
    while $False$ in $\boldsymbol{V}$ **do**
      $i \leftarrow RC_{[V==False]}$ ▷ Choose node
      $\boldsymbol{Q} = [i]$ ▷ Initial queue
      while $len(\boldsymbol{Q}) > 0$ **do**
        $i \leftarrow \boldsymbol{Q}[0]$
        $\boldsymbol{V}[i] \leftarrow True$
        $\boldsymbol{N} = ID_{[V[\mathcal{N}(i)]==False]}$
        if $len(\boldsymbol{N}) == 0$ **then**
          $POP(\boldsymbol{Q}, -1)$
          break
        end if
        $j \leftarrow RC_{[\boldsymbol{N}]}$ ▷ Choose neighbor
        $\boldsymbol{V}[j] \leftarrow True$
        $APPEND(\boldsymbol{Q}, j)$
        $APPEND(\boldsymbol{E}, [i, j])$
      end while
    end while
  end for
end procedure

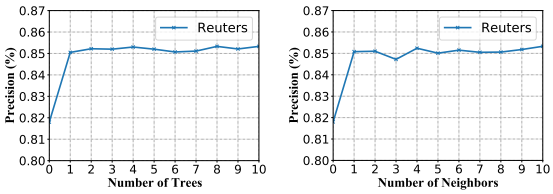


Figure 4: The precision of 64-bit hash codes on Reuters datasets with varying number of trees M and neighbors K.

Parameter Sensitivity To understand the robustness of our model, we conduct a parameter Sensitivity analysis of τ and β in Figure 5. Compared with $\beta = 0$ (without using neighborhood information), models with $\beta \neq 0$ improve performance significantly, but gradually performs steadily as β getting larger, which once again confirms the importance of simultaneously modeling semantic and neighborhood information. As for temperature coefficient τ used in variational encoder, our model performs steadily with various values of τ in the Reuters dataset. But in TMC and 20Newsgroups, increasing τ would deteriorate the model performance. Generally speaking, the model can achieve better performance with smaller τ (*i.e.*, steeper sigmoid function). As we utilize 0.5 as the threshold value, steeper sigmoid functions make it easier to

Word	**weapons**	**medical**	**companies**	**define**	**israel**	**book**
NASH	gun	treatment	company	definition	israeli	books
	guns	disease	market	defined	arabs	english
	weapon	drugs	afford	explained	arab	references
	armed	health	products	discussion	jewish	learning
	assault	medicine	money	knowledge	jews	reference
SNUH	weapon	medicine	inexpensive	defined	israeli	books
	armed	disease	expensive	definitions	arab	reference
	concealed	patients	cost	defines	arabs	chapter
	guns	physician	manufacturers	definition	palestinian	guide
	gun	treatment	design	arbitrary	gaza	origin

Table 5: The five nearest words in the semantic space learned by our model, compared with NASH.

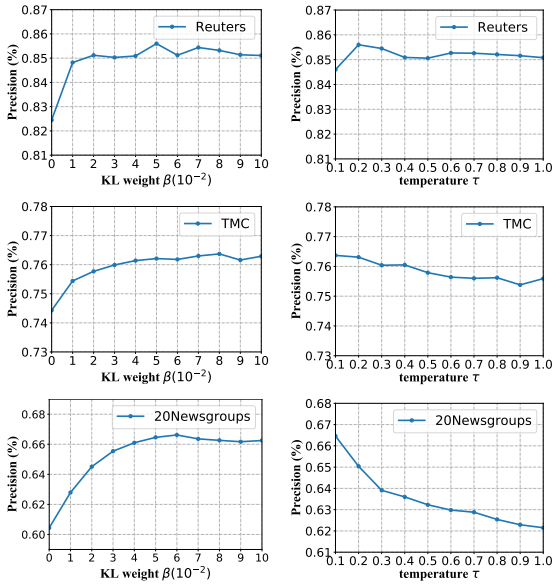


Figure 5: The precision of 64-bit hash codes on three datasets with varying temperature τ and KL weight β.

distinguish hash codes.

Comparing with RBSH and PairRec As mentioned before, the reason we do not directly compare our method with RBSH (Hansen et al., 2019) and PairRec (Hansen et al., 2020) is that their data processing methods are different from the mainstream methods (*e.g.*, VDSH, NASH, GMSH, NbrReg, AMMI and CorrSH). To further compare our method with them, we evaluate our model on three datasets that are published by RBSH[3]. The results are illustrated in Table 6. We observe that our method achieves the best performances in most experimental settings, which further confirms the superiority of simultaneously preserving the semantics and similarity information in a more principled framework.

Analysis of Semantic Information To illustrate the semantic information learned from the word

[3]https://github.com/casperhansen/RBSH

Datasets	Methods	16bits	32bits	64bits	128bits
Reuters	RBSH	0.7740	0.8149	0.8120	0.8088
	PairRec	0.8028	0.8268	0.8329	0.8468
	SNUH	**0.8063**	**0.8369**	**0.8483**	**0.8567**
TMC	RBSH	0.7959	0.8138	0.8224	0.8193
	PairRec	0.7991	0.8239	0.8280	0.8303
	SNUH	0.7901	0.8145	**0.8293**	**0.8329**
NG20	RBSH	0.6087	0.6385	0.6655	0.6668
	PairRec	n.a.	n.a.	n.a.	n.a.
	SNUH	0.5679	**0.6444**	**0.6806**	**0.7004**

Table 6: The precision of variant models on three datasets with different numbers of bits.

embedding matrix $E \in \mathbb{R}^{d\times|V|}$ in decoder network, we select the five nearest words based on the cosine similarity of their compact representation, with the result shown in Table 5. It can be seen that, compared with NASH, our model effectively groups semantically-similar words together in the learned vector space, demonstrating the preferable ability to capture semantic information of documents.