Modeling and Integrating Neighborhood Information for Generative Semantic Hashing

Anonymous ACL submission

Abstract

Semantic hashing is an effective technique for large-scale information retrieval because of its fast speed and small memory footprint. It is known that both document content and neighborhood information are instrumental in generating high-quality hash codes. To leverage these two types of information simultaneously, we propose to place the problem under the framework of generative models and treat the neighborhood information as another observation, in addition to document content, which makes the key problem reduced to how to model the neighborhood information well. To model it, we first view the neighborhood as a collection of independent edges and propose to model them with Bernoulli distributions. However, the independent edge assumption makes the model not able to capture the global neighborhood structure. To alleviate this issue, a vertex-based neighborhood model is further developed by decomposing a graph into a set of subgraphs, with the global neighborhood structure of each subgraph modeled simultaneously by a conditional distribution over all vertices. When the vertex-based neighborhood model is integrated with existing generative hashing models, significant performance gains are observed compared to current state-of-the-art models on three publicly available datasets.

1 Introduction

Similarity search aims to find the most similar items to the query from a large dataset and enjoys extensive applications such as plagiarism analysis (Stein et al., 2007), image retrieval (Jing and Baluja, 2008) and collaborative filtering (Koren, 2008) etc. However, since the original features in applications are mostly real-valued, it is computationally expensive to evaluate the similarity between huge amount of pairs of items, and not friendly to storage, neither. In the area of document retrieval, semantic hashing has been widely used to tackle this problem by transforming the real-valued feature representations of documents into compact and informative binary codes, which enables us to find out similar documents with high speed and small memory footprint.

To generate high-quality hash codes that preserve the semantic information of documents, extensive efforts have been made. To effectively leverage the ubiquitous unlabeled data, using deep generative models to achieve unsupervised semantic hashing has attracted considerable attentions in recent years (Chaidaroon and Fang, 2017; Shen et al., 2018; Hansen et al., 2020; Ou et al., 2021). By modeling the documents contents (features) with a variational autoencoder (VAE) (Kingma and Welling, 2013), the obtained inference network is used to produce high-quality hash codes that preserve the semantic information of documents successfully.

In addition to document content (features), neighborhood information that describes the adjacency of different documents is also available under some circumstances. In practice, the neighborhood information could be collected along with the documents, but more often it is estimated from the documents through post-processing, *e.g.*, estimated according to the cosine similarities of TFIDF features. Neighborhood information is known to contain rich document similarity information that is useful for high-quality hash code generation. For instances, the locality-preserving hashing (He et al., 2004; Zhao et al., 2014) and spectral hashing (Weiss et al., 2009; Li et al., 2012) proposed to generate hash codes by decomposing the adjacency matrix estimated from document features. In addition to the neighborhood information, the original document features are also instrumental for hash codes. To generate high-quality hash codes, the better way should be to leverage both types of information simultaneously. Based on this idea, (Chaidaroon et al., 2018; Hansen et al., 2020) proposed to encourage the hash code of a document to reconstruct all adjacent documents. However, since adjacent documents do not necessarily contain similar content, it is not a good way to exploit the neighborhood information by requiring all adjacent documents to be reconstructed from one hash code. Recently, (Ou et al., 2021) proposed to represent the neighborhood information by a Gaussian distribution and then use it as a prior distribution for the generative model of document content. The model achieved a principled integration of the two types of heterogeneous information under one model. However, considering that the representational ability of a Gaussian distribution is known to be very limited, using a Gaussian distribution to represent the complicated neighborhood may cause too much information to be lost.

In this paper, to simultaneously leverage the document content and neighborhood information, we propose to treat the neighborhood information as another kind of observation, just like the document content. To model the neighborhood information, we first view the neighborhood graph as a collection of independent edges and then propose to model the neighborhood by modeling the edges with a set of independent Bernoulli distributions. However, the independent-edge assumption makes the method only focus on modeling document pairs, while neglecting the overall global neighborhood structures. To alleviate this issue, we then propose to model the neighborhood information from the perspective of vertices. Specifically, we propose to decompose the full graph into a set of subgraphs and then model the global adjacent structure of a subgraph by a conditional distribution over vertices. Since every subgraph contains all documents and a part of edges, the vertex-based neighborhood model is certainly better at capturing the global neighborhood structure than the edge-based model, which only focuses on modeling document pairs. Experimental results on three public datasets demonstrate that when the proposed vertex-based neighborhood model is integrated with existing generative semantic hashing models, significant performance gains can be observed compared to the current state-of-the-art methods.

2 Preliminaries

To preserve the semantic information of documents in hash codes, a widely used approach is to model the documents by a generative model and then extract the hash codes from its latent representations. Specifically, given a corpus $\mathcal{X} = \{x_i\}_{i=1}^N$, generative hashing methods (Chaidaroon and Fang, 2017; Ou et al., 2021) seek to model a document x_i by a latent-variable model as

$$p(x_i, z_i) = p_\theta(x_i|z_i)p(z_i), \qquad (1)$$

where $p(z_i) = \mathcal{N}(z_i; 0, I_d)$ is the prior distribution of latent variable $z_i \in \mathbb{R}^d$; and $p_\theta(x_i|z_i)$ is the decoder. In semantic hashing, it is often defined as $p_\theta(x_i|z_i) = \prod_{j=1}^{|x_i|} p_\theta(w_{ij}|z_i)$, where

$$p_\theta(w_{ij}|z_i) \triangleq \frac{\exp(z_i^T E w_{ij} + b_j)}{\sum_{k=1}^{|V|} \exp(z_i^T E w_{ik} + b_k)}, \qquad (2)$$

and w_{ij} denoting a one-hot vector representing the j-th word of document x_i; $E \in \mathbb{R}^{d \times |V|}$ is a learnable embedding matrix connecting latent code z_i and one-hot representation of word w_{ij}; and b_j is the bias. By assuming independence among different documents, the corpus $\mathcal{X}$ is modeled as

$$p(\mathcal{X}, Z) = p_\theta(\mathcal{X}|Z)p(Z), \qquad (3)$$

where $p_\theta(\mathcal{X}|Z) = \prod_{i=1}^N p_\theta(x_i|z_i)$ and $p(Z) = \prod_{i=1}^N p(z_i)$ with $p(z_i) = \mathcal{N}(z_i; 0, I_d)$; and $Z \triangleq [z_1, z_2, ..., z_N]$. The model can be trained by maximizing the evidence lower bound (ELBO) of the log-likelihood $\log p(\mathcal{X})$. After training, the hash code of a document x can be obtained from its corresponding representation in the latent space.

Existing Neighborhood Information Construction and Integration Methods Under some circumstances, in addition to the textual data $\{x_i\}_{i=1}^N$, it is also possible to have the neighborhood data available during the training. In practice, the neighborhood data can be collected together with the documents. But more often, it is constructed from the collected documents through some post-processing methods. For instances, in (Hansen et al., 2020; Ou et al., 2021), a connection graph $G(\mathcal{V}, \mathcal{E})$ is constructed for documents based on the cosine similarities of their BM or TFIDF features, where $\mathcal{V} = \{1, 2, \cdots, N\}$ and $\mathcal{E}$ denote the vertices (documents) and edges, respectively. Specifically, in (Ou et al., 2021), each document is considered to be connected to its top K most similar documents, that is,

$$\mathcal{E} \triangleq \{(i,j)|j \in \text{top } K \text{ similar docs of doc } i\}. \quad (4)$$

In this paper, we follow to utilize the same neighborhood graph construction method.

Due to the richness of semantic-similarity information contained in the neighborhood data G, it has been proposed to integrate it with the textual data to produce higher-quality hash codes. To this end, PairRec (Hansen et al., 2020) proposed to enforce the hash code of a document to reconstruct its neighboring documents, too, in addition to the original document itself. But for the task of hashing, enforcing a hash code to reconstruct all neighboring documents is not reasonable since adjacent documents do not necessarily contain similar content, but only imply the same topic or category is shared. To better utilize the neighborhood information, SNUH (Ou et al., 2021) proposed to replace the independent prior $p(Z) = \prod_{i=1}^{N} \mathcal{N}(z_i; 0, I_d)$ with a neighborhood-informed Gaussian prior distribution $p(Z) = \mathcal{N}(Z; 0, \Sigma_G)$, where Σ_G denotes a $Nd \times Nd$ covariance matrix that specifies how different documents are correlated, and is derived from the neighborhood graph G. Although SNUH can unify the textual and neighborhood information under one model, representing the whole neighborhood information solely by a Gaussian prior is still too restrictive, especially in consideration of the limited representational ability of Gaussian distributions.

3 Neighborhood Information Modeling Methods

To leverage the neighborhood information, instead of representing it as a prior distribution as in SNUH (Ou et al., 2021), we view it as another type of observed data, just like the textual data $\mathcal{X} = \{x_i\}_{i=1}^{N}$. Specifically, we simultaneously model the corpus $\mathcal{X}$ and neighborhood data G by the following joint model

$$p(\mathcal{X}, G, Z) = p_\theta(\mathcal{X}|Z)p(G|Z)p(Z), \qquad (5)$$

where $p_\theta(\mathcal{X}|Z) = \prod_{i=1}^{N} p_\theta(x_i|z_i)$ and $p(Z) = \prod_{i=1}^{N} p(z_i) = \prod_{i=1}^{N} \mathcal{N}(z_i; 0, I_d)$ are the decoder of textual data and prior distribution, respectively, which are the same as previous models in (3); and $p(G|Z)$ denotes the decoder of neighborhood data, which will be elaborated detailedly in subsequent sections. Obviously, by viewing the neighborhood data as another observation, in addition to unifying the two types of information under a model, we can also resort to flexible decoders to capture the complex neighborhood information among different documents.

3.1 Modeling Neighborhood from the Perspective of Edges

The simplest way to model the neighborhood information is to view it as a collection of independent connections (edges) and disconnections (no edge). Under this perspective, the neighborhood information can be simply modeled as

$$p(G|Z) = \prod_{(i,j)\in\mathcal{E}} p(e_{ij} = 1|z_i, z_j) \times \prod_{(i,j)\in\bar{\mathcal{E}}} p(e_{ij} = 0|z_i, z_j), \qquad (6)$$

where $\bar{\mathcal{E}}$ is the complement set of $\mathcal{E}$, *i.e.*, the set containing all pairs of unconnected vertices in graph G; and $p(e_{ij}|z_i, z_j)$ is a Bernoulli distribution, which is used to indicate whether vertex i and j are connected. In this paper, this Bernoulli distribution is defined in the form

$$p(e_{ij} = 1|z_i, z_j) \triangleq \sigma\left((z_i^T z_j + b)/\tau\right), \qquad (7)$$

where τ is a scaling factor, b is the bias term; $\sigma(\cdot)$ is the sigmoid function.

From (7), we can see that if there is an edge between document i and j (that is, $e_{ij} = 1$), the neighborhood model will enforce their latent representations z_i and z_j to be similar. Otherwise, z_i and z_j will be pushed away from each other. In this way, the neighborhood information in G, along with the textual content $\mathcal{X}$, can be incorporated into the latent representations Z. However, when we model the neighborhood information using (6), it is implicitly assumed that for any two documents $(i, j) \in \bar{\mathcal{E}}$, they must be dissimilar. But as we construct the graph G like (4), to ensure the accuracy of added edges, each document is only connected to its top K most similar documents, while having the remaining ones unconnected. Obviously, it is unwise to require all document pairs $(i, j) \in \bar{\mathcal{E}}$ to output dissimilar latent representations because many of them may share similar semantic information. To address this issue, we propose to require only a portion of document pairs in $\bar{\mathcal{E}}$ to output dissimilar latent representations. To this end, we define a set $\mathcal{E}_0$ that contains the most dissimilar documents pairs, that is,

$$\mathcal{E}_0 \triangleq \{(i,j)|j \in \text{bottom } K' \text{ similar docs of } i\}, \quad (8)$$

where the similarity is evaluated according to the documents' TFIDF features. Then, only the document pairs from $\mathcal{E}_0$ are encouraged to output dissimilar latent representation, that is, the neighborhood

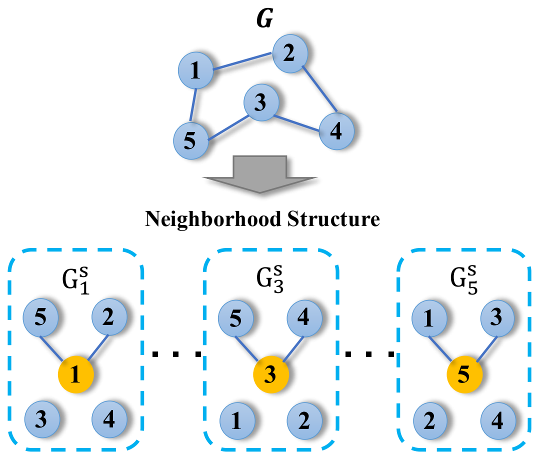


Figure 1: Example to demonstrate graph decomposition. The yellow vertices are the center vertices of subgraphs.

model is written as $p(G|Z) = \prod_{(i,j)\in\mathcal{E}} p(e_{ij} = 1|z_i, z_j) \times \prod_{(i,j)\in\mathcal{E}_0} p(e_{ij} = 0|z_i, z_j)$, or equivalently

$$p(G|Z) = \prod_{(i,j)\in\mathcal{E}} \sigma\left((z_i^T z_j + b)/\tau\right) \times \prod_{(i,j)\in\mathcal{E}_0} \left(1-\sigma\left((z_i^T z_j + b)/\tau\right)\right). \quad (9)$$

Moreover, under the most extreme case, we can set $\mathcal{E}_0 = \emptyset$, which means not to consider any unconnected document pairs.

3.2 Modeling Neighborhood from the Perspective of Vertices

In this section, we propose another way to model the neighborhood information from the perspective of vertices. Specifically, for a neighborhood graph $G(\mathcal{V}, \mathcal{E})$, we decompose it into into $|\mathcal{V}|$ subgraphs $G^s_i(\mathcal{V}, \mathcal{E}_i)$ for $i = 1, 2, \cdots, |\mathcal{V}|$, with the subgraph G^s_i describing the connection structure of vertex i to the rest vertices $\mathcal{V}\backslash i$. An example of graph decomposition is illustrated in Fig. 1, in which the neighborhood graph G is decomposed into five subgraphs. With the decomposition, the neighborhood information in G is equivalently represented in the set of subgraphs $G^s_1, G^s_2, \cdots, G^s_{|\mathcal{V}|}$. Therefore, to model the neighborhood information G given Z, we can model its subgraphs instead

$$p(G|Z) \triangleq \prod_{i\in\mathcal{V}} p(G^s_i|Z), \quad (10)$$

where $p(G^s_i|Z)$ is the model of neighborhood information of vertex i. For the subgraph G^s_i, we do not model it as a collection of independent distributions over edges, as done in Section 3.1. Instead, we model it as a set of conditional independent distributions over vertices, that is,

$$p(G^s_i|Z) = \prod_{j\in\mathcal{N}_i} p(u = j|Z, v = i), \quad (11)$$

where $\mathcal{N}_i \triangleq \{j|(i,j) \in \mathcal{E}\}$ denotes the neighbors of vertex i; and the conditional distribution $p(u|Z, v)$ is defined as

$$p(u|Z, v) = \frac{\exp(z_v^T z_u/\tau)}{\sum_{k\in\mathcal{V}\backslash v} \exp(z_v^T z_k/\tau)}, \quad (12)$$

where $u \in \mathcal{V}\backslash v$. It can be easily seen that $p(u|Z, v)$ describes the probability of vertex u being a neighbor of vertex v as vertex v is given. By substituting (11) and (12) into (10), we obtain the final model of neighborhood information as

$$p(G|Z) = \prod_{(i,j)\in\mathcal{E}} \frac{\exp(z_i^T z_j/\tau)}{\sum_{k\in\mathcal{V}\backslash i} \exp(z_i^T z_k/\tau)}. \quad (13)$$

It can be seen from (13) that if document i and j are adjacent (*i.e.*, $(i,j) \in \mathcal{E}$), their latent representations z_i and z_j will be encouraged to be similar so as to increase the probability $\frac{\exp(z_i^T z_j/\tau)}{\sum_{k\in\mathcal{V}\backslash i} \exp(z_i^T z_k/\tau)}$ of observing this adjacency. Thus, by using this model, the neighborhood information can be incorporated into the latent representation learning process smoothly. Note that the vertex-based neighborhood model does not explicitly contain terms concerning unconnected document pairs $(i,j) \in \bar{\mathcal{E}}$ as in edge-based model (6). This is because in the vertex-based model, since the summation of probabilities $\sum_{j\in\mathcal{V}\backslash i} p(j|Z, i) = 1$ always holds, if we try to increase the probabilies for $j \in \mathcal{N}_i$, the summation of probabilities $\sum_{j\in\overline{\mathcal{N}}_i} p(j|Z, i)$ regarding unconnected pairs will be decreased automatically, where $\overline{\mathcal{N}}_i$ represents the set of unconnected vertices of vertex i. Moreover, since only the summation $\sum_{j\in\overline{\mathcal{N}}_i} p(j|Z, i)$ is encouraged to decrease, it is still possible to allow the probabilities of some unconnected pairs to be large, which partially alleviates the issue that not all unconnected documents have dissimilar semantic information.

Compared with the edge-based model, a unique characteristic of the vertex-based neighborhood model is that it is able to perceive the global neighborhood information when dealing with a pair of documents $(i,j) \in \mathcal{E}$. That is because its probability term corresponding to a pair of documents depends on the latent representations of all documents due to the existence of denominator $\sum_{k\in\mathcal{V}\backslash i}\exp(z_i^T z_k/\tau)$. But in the edge-based model, we can see that the probability term corresponding to a pair of documents is only determined by latent representations of the two relevant documents. The global awareness of the vertex-based model makes it more advantageous in capturing the global neighborhood structure, as corroborated by later empirical experimental results.

4 Training

To train the neighborhood information integrated hashing model (5), the objective is to maximize the log-likelihood $\log p(\mathcal{X}, G) = \log\int p_\theta(\mathcal{X}|Z)p(G|Z)p(Z)dZ$. But due to the intractability of computing exact log-likelihood, we instead maximize its lower bound (ELBO) under the framework of variational inference

$$\mathcal{L} = \underbrace{\mathbb{E}_{q_\phi(Z|\mathcal{X})}[\log p_\theta(\mathcal{X}|Z)]}_{\mathcal{L}_X} - \underbrace{KL(q_\phi(Z|\mathcal{X})\|p(Z))}_{\mathcal{L}_{KL}} + \underbrace{\mathbb{E}_{q_\phi(Z|\mathcal{X})}[\log p(G|Z)]}_{\mathcal{L}_G}, \quad (14)$$

where $q_\phi(Z|\mathcal{X})$ is the variational posterior. In this paper, we assume $q_\phi(Z|\mathcal{X})$ to maintain a factorized Gaussian distribution form, that is, $q_\phi(Z|\mathcal{X}) \triangleq \prod_{i=1}^N q_\phi(z_i|x_i)$ with

$$q_\phi(z_i|x_i) = \mathcal{N}(z_i; \mu_i, diag(\beta_i^2)), \quad (15)$$

where $\mu_i \in \mathbb{R}^d$ and $\beta_i^2 \in \mathbb{R}^d$ denote the mean and variance vectors, respectively; and $diag(\cdot)$ means the diagonalization function. In our experiments, μ_i and β_i^2 are the outputs of two neural networks that take x_i as input and are parameterized by ϕ. To make the latent representation z_i more compatible with binary hash codes, we confine the range of μ_i to the interval $(0, 1)$ by using a sigmoid function at the end of the neural network.

By noticing $p_\theta(\mathcal{X}|Z) = \prod_{i=1}^N p_\theta(x_i|z_i)$ and $p(Z) = \prod_{i=1}^N \mathcal{N}(z_i; 0, I_d)$, combining with the factorized Gaussian assumption on $q_\phi(Z|X)$, the expressions for $\mathcal{L}_X$ and $\mathcal{L}_{KL}$ terms in (14) are exactly the same as those in previous generative semantic models, which can be found in Appendix A.1. As for the neighborhood-relevant term $\mathcal{L}_G$, its expressions w.r.t. the edge-based and vertex-based neighborhood models are

$$\mathcal{L}_G^{edge} = \sum_{(i,j)\in\mathcal{E}} \log\sigma\left((\tilde{z}_i^T\tilde{z}_j + b)/\tau\right) + \sum_{(i,j)\in\mathcal{E}_0} \log\left(1 - \sigma\left((\tilde{z}_i^T\tilde{z}_j + b)/\tau\right)\right), \quad (16)$$

$$\mathcal{L}_G^{vert} = \sum_{(i,j)\in\mathcal{E}} \left(\frac{\tilde{z}_i^T\tilde{z}_j}{\tau} - \log\left(\sum_{k\in\mathcal{V}\backslash i}\exp(\frac{\tilde{z}_i^T\tilde{z}_k}{\tau})\right)\right) \quad (17)$$

where $\tilde{z}_i = \mu_i + \epsilon\cdot\beta_i$ with ϵ being d-dimensional standard Gaussian noise, which is the reparameterization trick (Kingma and Welling, 2013), a well-known technique that is widely used for expectation estimation. By replacing $\mathcal{L}_G$ in (14) with either $\mathcal{L}_G^{edge}$ or $\mathcal{L}_G^{vert}$, the ELBO $\mathcal{L}$ can be optimized with SGD algorithms. Note that at each iteration, we do not need to consider all document pairs $(i, j) \in \mathcal{E}$ simultaneously, but only need to use a minibatch of them to reduce the computation cost, thanks to the factorized form of $\mathcal{L}_G^{edge}$ and $\mathcal{L}_G^{vert}$ over the document pairs $(i, j) \in \mathcal{E}$. After training, the hash code of a document x_i can be obtained by binarizing its posterior mean μ_i with a threshold, *e.g.*, 0.5 (Chaidaroon and Fang, 2017).

Efficient Training for Vertex-Based Neighborhood Model When we optimize the ELBO with $\mathcal{L}_G^{vert}$, we can use minibatchs from $\mathcal{E}$ to replace the full batch $\mathcal{E}$ to reduce the computational cost at every iteration. However, from (17), it can be seen that there is another summation over individual documents $\mathcal{V}\backslash i$ inside the $\log(\cdot)$ function. This means that if we want to compute the gradient of $\mathcal{L}_G^{vert}$ w.r.t. a pair of documents $(i, j) \in \mathcal{E}$, we have to take all documents in the training dataset into account. This makes using minibatchs from $\mathcal{E}$ to reduce complexity meaningless because we still need to consider all documents for every iteration. In order to effectively reduce the training complexity, we can further approximate $\mathcal{L}_G^{vert}$ by $\widetilde{\mathcal{L}}_G^{vert}$, where

$$\widetilde{\mathcal{L}}_G^{vert} = \sum_{i\in\mathcal{V}} \underbrace{\sum_{j\in\mathcal{N}_i}\left(\frac{\tilde{z}_i^T\tilde{z}_j}{\tau} - \log\left(\sum_{k\in\{j\}\cup\mathcal{S}_i}\exp(\frac{\tilde{z}_i^T\tilde{z}_k}{\tau})\right)\right)}_{\widetilde{\mathcal{L}}_G^{vert}(i)},$$

and $\mathcal{S}_i$ is a subset randomly drawn from $\mathcal{V}\backslash i$. Essentially, this method is to use summation over a small subset $\mathcal{S}$ to replace the summation over the set $\mathcal{V}\backslash i$. When the denominator of a softmax function contains a huge amount of terms, using a small proportion of them to approximate it during the training is a technique called negative sampling (NS), which is widely used in the training of word2vec (Mikolov et al., 2013), network embedding, etc., and has been found great success. Inspired by the theories of noise contrastive estimation (NCE) (Gutmann and Hyvärinen, 2010) and InfoNCE (van den Oord et al., 2019), we justify the negative-sampling approximation in a more rigorous way below.

Proposition 4.1. *Define a function* $\mathcal{L}_S^{vert}(i) \triangleq \sum_{j\in\mathcal{N}_i}\left(\frac{s(z_i,z_j)}{\tau} - \log\left(\sum_{k\in\mathcal{V}\backslash i}\exp(\frac{s(z_i,z_k)}{\tau})\right)\right)$ *and its NS approximation* $\widetilde{\mathcal{L}}_S^{vert}(i) \triangleq \sum_{j\in\mathcal{N}_i}\left(\frac{s(z_i,z_j)}{\tau} - \log\left(\sum_{k\in\{j\}\cup\mathcal{S}_i}\exp(\frac{s(z_i,z_k)}{\tau})\right)\right)$. *If the score function* $s(z_i,z_j)$ *is sufficiently expressive, maximizing* $\widetilde{\mathcal{L}}_S^{vert}(i)$ *is equivalent to maximize* $\mathcal{L}_S^{vert}(i)$ *in the sense that at their optimal points, both of them can have the distribution* $p_S(u|Z,v) = \frac{\exp(s(z_v,z_u)/\tau)}{\sum_{k\in\mathcal{V}\backslash v}\exp(s(z_v,z_k)/\tau)}$ *equal to the same distribution* $\mathbb{P}(u|v)$, *where* $\mathbb{P}(u|v) = \frac{1}{|\mathcal{N}_v|}$ *for* $u \in \mathcal{N}_v$ *and* 0 *otherwise.*

Proof. Please refer to the Appendix A.2. □

Obviously, if we set $s(z_i,z_j) = z_i^T z_j$, then $\mathcal{L}_S^{vert}(i)$ and $\widetilde{\mathcal{L}}_S^{vert}(i)$ are reduced to $\mathcal{L}^{vert}(i) \triangleq \sum_{j\in\mathcal{N}_i}\left(\frac{z_i^T z_j}{\tau} - \log\left(\sum_{k\in\mathcal{V}\backslash i}\exp(\frac{z_i^T z_k}{\tau})\right)\right)$ and $\widetilde{\mathcal{L}}^{vert}(i)$, respectively. Thus, maximizing the NS surrogate $\widetilde{\mathcal{L}}_G^{vert}$ can be approximately viewed as maximizing the original $\mathcal{L}_G^{vert}$.

To facilitate discussion, we term the edge-based and vertex-based models as **N**eigh**b**o**r**hood **S**emantic **H**ashing from **E**dges (NbrSH$_E$) and **N**eigh**b**o**r**hood **S**emantic **H**ashing from **V**ertices (NbrSH$_V$) respectively. Additionally, NbrSH$_V$ without NS approximation is termed as NbrSH$_V^{Full}$.

5 Related Works

To generate high-quality hash codes with unsupervised hashing, extensive efforts have been made. VDSH (Chaidaroon and Fang, 2017) firstly introduced the variational autoencoder (VAE) (Kingma and Welling, 2013) into semantic hashing. To tackle the drawbacks brought by the two-stage training, NASH (Shen et al., 2018) replaced the Gaussian prior with Bernoulli prior and utilized the straight-through technique (Bengio et al., 2013) to achieve end-to-end training. Differing from modeling the documents contents with a generative model, AMMI (Stratos and Wiseman, 2020) sought to generate high-quality hash codes by maximizing the mutual information between documents and hash codes. Apart from the aforementioned semantic-based models, locality-preserving hashing (He et al., 2004; Zhao et al., 2014) and spectral hashing (Weiss et al., 2009; Li et al., 2012) are the neighborhood-based models that proposed to generate hash codes by decomposing the adjacency matrix estimated from document features.

Since different aspects of information are emphasized in documents content (features) and neighborhood among documents, many works have been done to take both semantic and neighborhood information in generating high-quality hash codes into account recently. For instances, RBSH (Hansen et al., 2019) imposed a ranking component into the loss function to model the similarity between documents, NbrReg (Chaidaroon et al., 2018) and PairRec (Hansen et al., 2020) required the hash code of a document to reconstruct its neighbors, and SNUH (Ou et al., 2021) integrated semantic and neighborhood information in a unified framework by representing the neighborhood information with a Gaussian distribution and using it as the prior distribution. However, as mentioned before, the unreasonable requirements in PairRec and the restrictive Gaussian prior in SNUH were limiting the utilization of neighborhood information. By modeling the whole subgraph simultaneously with a flexible framework, we effectively improve the performance of hash codes.

6 Experiments

6.1 Experiments Setup

Datasets Following previous works, three public datasets published by VDSH are utilized to verify our proposed model: 1) Reuters21578, which consists of 10,788 documents with 90 categories; 2) 20Newsgroups, which is a collection of 18,828 newsgroup posts with 20 different categories; 3) TMC, which is the collection of air traffic reports provided by NASA and contains 21,519 documents with 22 categories.

Baselines For unsupervised semantic hashing, we compare our proposed model with the following models: SpH (Weiss et al., 2009), STH (Zhang et al., 2010), VDSH (Chaidaroon and Fang, 2017), NbrReg (Chaidaroon et al., 2018), NASH (Shen et al., 2018), RBSH (Hansen et al., 2019), AMMI

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
RBSH	0.7911	0.8206	0.8371	0.8470	0.6901	0.7203	0.7400	0.7494	0.4878	0.5408	0.5758	0.5985	0.6999
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
PairRec	0.8244	0.8374	0.8543	0.8544	0.7210	0.7470	0.7609	0.7628	0.5637	0.6223	0.6413	0.6578	0.7373
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
NbrSH$_E$	0.8283	0.8522	0.8538	0.8606	0.7240	0.7526	0.7618	0.7668	0.5395	0.6050	0.6243	0.6143	0.7319
NbrSH$_V$	**0.8402**	**0.8547**	**0.8771**	**0.8804**	**0.7365**	**0.7621**	**0.7724**	**0.7779**	**0.6074**	**0.6576**	**0.6741**	**0.6785**	**0.7599**

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

(Stratos and Wiseman, 2020), PairRec (Hansen et al., 2020) and SNUH (Ou et al., 2021). For all baselines, the reported performances from original papers are taken except RBSH and PairRec since they employed a different preprocessing method on the datasets.

Training Details For the encoder network, we follow to utilize the same architecture elaborated in previous works for fair comparisons, using one fully connected layer as the encoder. The graph G is constructed with the k-nearest algorithm based on the cosine similarity of TFIDF feature. In our experiments, learning rate is fixed to 0.001, batch size is fixed to 64, the scaling factor in NbrSH$_E$ is fixed to d while it is fixed to $\frac{d}{16}$ in NbrSH$_V$. Additionally, the negative samples size in NbrSH$_V$ is simply set as 20 in all cases. As for the numbers of nearest-neighbors in NbrSH$_V$, we set 100 for Reuters, 20 for 20Newsgroups, and 50 for TMC, respectively. According to the precision of validation set, we select the numbers of nearest-neighbors in NbrSH$_E$ from $\{10, 20, ..., 100\}$. The Adam optimizer (Kingma and Ba, 2014) with default setting except learning rate is utilized to train the model.

Evaluation Metrics To evaluate the performance of our model, retrieval precision is utilized. For each query, we retrieve the top 100 similar documents based on the Hamming distance between hash codes. And the retrieval precision is the ratio of retrieved documents that share the same label with the query. Lastly, we measure the performance of models with the average precision across all queries in the test set.

6.2 Performance of Hash Codes

Extensive experiments on the three public datasets are conducted to verify the performance of NbrSH$_E$ and NbrSH$_V$. The testing retrieval precision is demonstrated in Table 1. We see that NbrSH$_V$ consistently outperforms all the baseline models by a substantial margin, yielding the best performance in all cases and the performance of NbrSH$_E$ is close to the state-of-the-art methods. Compared to the models that only utilized semantic information, such as VDSH, NASH, AMMI, etc, NbrSH$_V$ achieves superior performance by integrating the neighborhood information from the perspective of vertices. When it comes to the current SOTA method of SNUH, NbrSH$_V$ successfully enhances the average precision with more than 1.2% by utilizing a more flexible and expressive framework to integrate the neighborhood information. Since RBSH and PairRec employed a different prepossessing method on the datasets, we retrain them on our datasets and the results show that modeling the subgraphs of the neighboring graph instead of independent edges is a superior method to model the neighborhood information. Additionally, comparing NbrSH$_V$ with NbrSH$_E$, the improvement of performance meets our understanding of the informative global structure of neighborhood information. Moreover, by dividing a scaling factor in inner product computation, we consistently improve the performance with larger bits.

6.3 Impact of Number of Neighbors

To understand the impact of the number of selected neighbors, we train NbrSH$_V$ with $\{0, 10, ..., 100\}$ neighbors on three public datasets. We demonstrate the results with line plot in Fig. 2. Firstly, We observe that, compared to not considering any neighborhood information, ten neighbors can bring significant performance gains in most cases. Secondly, for 20Newsgroups and TMC, the model tends to achieve better performance with lesser neighbors

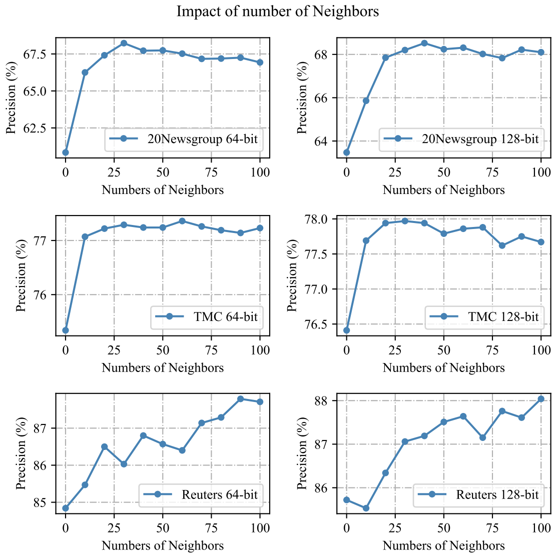


Figure 2: The retrieval precision of 64-bit and 128-bit hash codes with varying the number of selected neighbors on the three public datasets.

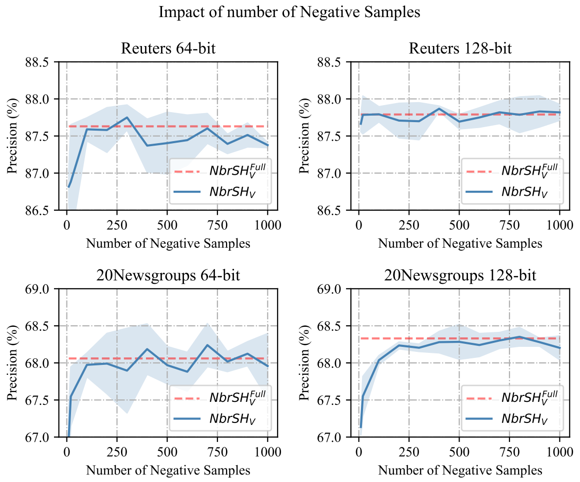
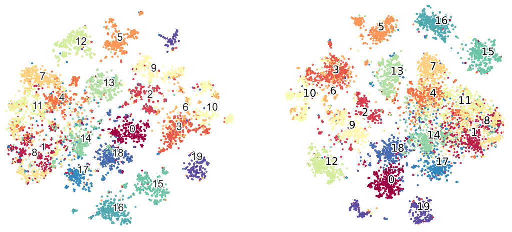


Figure 3: The retrieval precision of 64-bit and 128-bit hash codes with varying negative samples size on Reuters and 20Newsgroups. The red line is the result of NbrSH$_V^{Full}$ and the blue line is the mean result of NbrSH$_V$ with three different random seeds.



(a) NbrSH$_V$ (b) SNUH

Figure 4: Visualization of the 64-dimensional hash codes generated by our proposed models for the 20Newsgroups dataset with the t-SNE technique.

while Reuters prefer to take more neighbors into account. In a word, it is a trade-off between the increasing information provided by more neighbors and the decreasing accuracy of selected neighbors that indeed share the same label with the starting vertex. Finally, for the same dataset, the best number of neighbors tends to be similar across different lengths of hash codes.

6.4 Impact of Negative Samples Size in Efficient Training

To understand the impact of negative samples size in efficient training for vertex-based model, we train NbrSH$_V$ with $\{10, 20, 100, 200, ..., 1000\}$ negative samples. And in each case, we train the model with three different random seeds to measure if it is sensitive to different initialization states. Then we compare the average retrieval precision of each negative sample size with the result of NbrSH$_V^{Full}$ in Fig. 3. We can observe that the results of NbrSH$_V$ with different negative samples sizes are scattered around the result of NbrSH$_V^{Full}$, demonstrating the feasibility of training the model by maximizing $\widetilde{\mathcal{L}}_G^{vert}$ instead of $\mathcal{L}_G^{vert}$ and the insensibility of negative samples size. Moreover, the performance of efficient training is stable in most cases with different random seeds.

6.5 Visualization of Hash Codes

To intuitively evaluate the quality of generated hash codes of our proposed model, we utilize the t-SNE (van der Maaten and Hinton, 2008) technique to transform the 64-dimensional hash codes into 2-dimensional vectors. By comparing with the visualization result of SNUH, shown in Fig. 4, the hash codes generated by NbrSH$_V$ are more separable, demonstrating the superiority of our proposed model.

7 Conclusion

We have proposed an effective and efficient hashing method to leverage both the semantics and neighborhood information among documents. In particular, we viewed the neighborhood information as another kind of observation and utilized a vertex-based model to model the global adjacent structure of each subgraph of the neighborhood. By integrating the vertex-based neighborhood model with existing generative hashing models, significant performance gains were observed compared to current state-of-the-art methods on three public datasets.

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.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 297–304, Chia Laguna Resort, Sardinia, Italy. PMLR.

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.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119. Curran Associates, Inc.

Zijing Ou, Qinliang Su, Jianxing Yu, Bang Liu, Jingwen Wang, Ruihui Zhao, Changyou Chen, and Yefeng Zheng. 2021. Integrating semantics and neighborhood information with graph-driven generative models for document retrieval. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2238–2249, Online. Association for Computational Linguistics.

Dinghan Shen, Qinliang Su, Paidamoyo Chapfuwa, Wenlin Wang, Guoyin Wang, Ricardo Henao, and Lawrence Carin. 2018. NASH: Toward end-to-end neural architecture for generative semantic hashing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2041–2050.

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. In *Proceedings of the 37th International Conference on Machine Learning*, pages 9144–9154.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2019. Representation learning with contrastive predictive coding.

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

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.

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

A Appendices

A.1 Expressing $\mathcal{L}_X$ and $\mathcal{L}_{KL}$ in Analytical Form

According to the definition in section 2, we have

$$p_\theta(\mathcal{X}|Z) = \prod_{i=1}^{N} p_\theta(x_i|z_i) = \prod_{i=1}^{N}\prod_{j=1}^{|x_i|} p_\theta(w_{ij}|z_i)$$

with

$$p_\theta(w_{ij}|z_i) \triangleq \frac{\exp(z_i^T E w_{ij} + b_j)}{\sum_{k=1}^{|V|}\exp(z_i^T E w_{ik} + b_k)}.$$

Therefore, by utilizing Monte Carlo Sampling and reparametrization trick, $\mathcal{L}_X$ can be expressed in an analytical form

$$\mathcal{L}_X = \sum_{i=1}^{N}\sum_{j=1}^{|x_i|} \log \frac{\exp(\tilde{z}_i^T E w_{ij} + b_j)}{\sum_{k=1}^{|V|}\exp(\tilde{z}_i^T E w_{ik} + b_k)},$$

where $\tilde{z}_i = \mu_i + \epsilon \cdot \beta_i$ with ϵ being d-dimensional standard Gaussian noise.

As for $\mathcal{L}_{KL}$, since $q_\phi(Z|\mathcal{X}) = \prod_{i=1}^{N} q_\phi(z_i|x_i)$ and $p(Z) = \prod_{i=1}^{N} p(z_i)$, it can be decomposed into the summation of N terms

$$\mathcal{L}_{KL} = \sum_{i=1}^{N} KL(q_\phi(z_i|x_i) \| p(z_i))\,.$$

Because $q_\phi(z_i|x_i) = \mathcal{N}(z_i; \mu_i, diag(\beta_i^2))$ and $p(z_i) = \mathcal{N}(0, I_d)$, $KL(q_\phi(z_i|x_i)\|p(z_i))$ can be derived as

$$\begin{aligned}
&KL(q_\phi(z_i|x_i)\|p(z_i)) \\
&= \mathbb{E}_{q_\phi}\left[\log q_\phi(z_i|x_i) - \log p(z_i)\right] \\
&= \mathbb{E}_{q_\phi}\Bigg[\log \prod_{n=1}^{d} \frac{1}{\sqrt{2\pi\beta_{in}^2}} \exp(-\frac{(z_{in}-\mu_{in})^2}{2\beta_{in}^2}) \\
&\qquad\qquad - \log \prod_{n=1}^{d} \frac{1}{\sqrt{2\pi}} \exp(-\frac{z_{in}^2}{2})\Bigg] \\
&= \mathbb{E}_{p(\epsilon_i)}\Bigg[\sum_{n=1}^{d} \frac{\mu_{in}^2 + 2\mu_{in}\beta_{in}\epsilon_{in} + \beta_{in}^2\epsilon_{in}^2}{2} \\
&\qquad\qquad - \frac{2\log\beta_{in} + \epsilon_{in}^2}{2}\Bigg] \\
&= \frac{1}{2}\sum_{n=1}^{d}(\mu_{in}^2 + \beta_{in}^2 - 2\log\beta_{in} - 1)
\end{aligned}$$

where we utilize the reparametrization trick to transform z_i into $\mu_i + \epsilon_i \cdot \beta_i$. Therefore, $\mathcal{L}_{KL}$ can be expressed in an analytical form

$$\mathcal{L}_{KL} = \frac{1}{2}\sum_{i=1}^{N}\sum_{n=1}^{d}(\mu_{in}^2 + \beta_{in}^2 - 2\log\beta_{in} - 1)$$

A.2 Proof of Proposition 4.1

Proof. According to the definitions stated in Proposition 4.1, we have $\mathcal{L}_S^{vert}(i) \triangleq \sum_{j\in\mathcal{N}_i}\Big(\frac{s(z_i,z_j)}{\tau} - \log\Big(\sum_{k\in\mathcal{V}\setminus i}\exp(\frac{s(z_i,z_k)}{\tau})\Big)\Big)$ and its NS approximation $\widetilde{\mathcal{L}}_S^{vert}(i) \triangleq \sum_{j\in\mathcal{N}_i}\Big(\frac{s(z_i,z_j)}{\tau} - \log\Big(\sum_{k\in\{j\}\cup\mathcal{S}_i}\exp(\frac{s(z_i,z_k)}{\tau})\Big)\Big)$. The maximization of $\mathcal{L}_S^{vert}(i)$ and $\widetilde{\mathcal{L}}_S^{vert}(i)$ can be viewed as the maximization of distributions, that is

$$\max \mathcal{L}_S^{vert}(i) \Leftrightarrow \max \prod_{j\in\mathcal{N}_i} p(u{=}j|Z, v{=}i)$$

$$\max \widetilde{\mathcal{L}}_S^{vert}(i) \Leftrightarrow \max \prod_{j\in\mathcal{N}_i} \tilde{p}(u{=}j|Z, v{=}i)$$

where $p(u{=}j|Z, v{=}i)$ is defined in (12) and

$$\tilde{p}(u|Z, v) = \frac{\exp(s(z_v, z_u)/\tau)}{\sum_{k\in\{u\}\cup\mathcal{S}_v}\exp(s(z_v, z_k)/s)}.$$

By maximizing the log probability of $\tilde{p}(u{=}j|Z, v{=}i)$, we actually encouraging the model to distinguish which vertex in $\mathcal{S}_{ij} \triangleq \{j\}\cup\mathcal{S}_i$ is the neighbor of vertex i. In other word, the maximization encourages $\tilde{p}(u{=}j|Z, v{=}i)$ to approach its ground-truth distribution $\mathbb{P}(nbr{=}j|v{=}i)$, which is defined as

$$\begin{aligned}
&\mathbb{P}(nbr{=}j|v{=}i) \\
&= \frac{\mathbb{P}(u{=}j|v{=}i)\prod_{l\neq j} q(u{=}l)}{\sum_{k\in\mathcal{S}_{ij}}\mathbb{P}(u{=}k|v{=}i)\prod_{l\neq k} q(u{=}l)} \\
&= \frac{\frac{\mathbb{P}(u=j|v=i)}{q(u=i)}}{\sum_{k\in\mathcal{S}_{ij}}\frac{\mathbb{P}(u=k|v=i)}{q(u=k)}}
\end{aligned}$$

where $\mathbb{P}(u|v) = \frac{1}{|\mathcal{N}_v|}$ for $u \in \mathcal{N}_v$ and 0 otherwise, is the ground-truth distribution of $p(u|Z, v)$. If we further assume that the score function $s(z_i, z_j)$ is sufficiently expressive, we have

$$\begin{aligned}
p(u{=}j|Z, v{=}i) &= \mathbb{P}(u{=}j|v{=}i) \\
\tilde{p}(u{=}j|Z, v{=}i) &= \mathbb{P}(nbr{=}j|v{=}i)
\end{aligned}$$

Since $q(u)$ is a uniform distribution, we have

$$\exp(s(z_v, z_u)) \propto \mathbb{P}(u|v) = p(u|Z, v),$$

Distance	Title/Subject	Category
query	**Crypto is for hard-core hackers & spooks only**	**crypt**
5	RE: Once tapped, your code is no good any more	crypt
10	RE: Secret algorithm [Re: Clipper Chip and crypto key-escrow]	crypt
20	RE: Do we need the clipper for cheap security?	crypt
50	RE: AD conversion	mac.hardware
70	RE: Looking for MOVIES w/ BIKES	motorcycles
90	RE: Atlanta Hockey Hell!!	hockey

Table 2: The documents with Hamming distances of 5, 10, 20, 50, 70, and 90 to the query of the 128-bit hash codes on the 20Newsgroups dataset.

for all $i, j \in \mathcal{V}$. Then, the following relation holds

$$\max \widetilde{\mathcal{L}}_S^{vert}(i) \Leftrightarrow \max \mathcal{L}_S^{vert}(i),$$

in the sense that at their optimal points, both of them can have the distribution $p_S(u|Z, v)$ equal to the same distribution $\mathbb{P}(u|v)$. □

A.3 Model Architecture Details

Encoder Encoder consists of one fully connected layer to project the raw feature into the latent space. Specifically, given x_i, $\mu_i = sigmoid(F_1(x_i)/0.1)$ and $\sigma_i = softplus(F_2(x_i))$, where F_1 and F_2 are one-layer feed-forward neural networks and 0.1 is temperature for a faster convergence speed. Then, by utilizing reparameterization trick, $z_i = \mu_i + \epsilon \odot \sigma_i$, where $\epsilon \sim \mathcal{N}(0, I_d)$ and $\odot$ denotes element wise product.

Decoder of documents As indicated in (Shen et al., 2018), employing expressive nonlinear decoders likely destroy the distance-keeping property. Therefore, the decoder of documents simply consists of an embedding layer and $\hat{x}_i = softmax({z_i}^T E + b_{dec})$.

A.4 Case Study

To understand the document retrieval with Hamming distance intuitively, we present a retrieval case of a given query document, which is stated in Table 2. We can observe that the topic of the retrieved document becomes more irrelevant with the increase of the Hamming distance, demonstrating that the Hamming distance can effectively measure the relevance of documents.