ICIC Express Letters
Volume **14**, Number **4**, April **2020**

 ISSN 1881-803X
pp. **417–422**

CONDITIONAL TEXT HASHING UTILIZING PAIR-WISE MULTI CLASS LABELS

Richeng Xuan[1], Junho Shim[2,*] and Sang-Goo Lee[1]

[1]Department of Computer Science and Engineering
Seoul National University
1 Gwanak-ro, Gwanak-gu, Seoul 08826, Korea
{ xuanricheng; sglee }@europa.snu.ac.kr

[2]Department of Computer Science
Sookmyung Women's University
Cheongpa-ro 47-gil 100 (Cheongpa-dong 2ga), Yongsan-gu, Seoul 04310, Korea
*Corresponding author: jshim@sookmyung.ac.kr

Received October 2019; accepted January 2020

Abstract. *With the massive explosion of mosquito data, dealing with large amounts of text data has become problematic. Although more and more complex methods can understand and process semantics more accurately, they become less applicable to large-scale text data. Text hashing is one of the effective approaches, since it may reduce processing speed to a completely different scale extreme. Previous study has focused on single document classification, or the document pair binary classification problem. In this paper, we propose a supervised multi conditional semantic text hashing method. Experimental results on public datasets show that our method can generate multi conditional hash code.*
Keywords: Natural language processing, Semantic hashing, Machine learning, Similarity search

1. **Introduction.** In recent years, natural language processing has investigated a variety of studies. Because natural language data is produced in large quantities on the Internet and contains a lot of valuable information, understanding natural language has always been the goal of researchers. Many researchers have tried to capture the potential semantic relations hidden in natural language. However, due to the complexity of the model, many approaches can no longer be used in textual big data with large volumes, so some researchers have begun to reduce the complexity of the approaches, or compress the text data.

Text hashing is one of the efficient text compression methods. It improves both computational efficiency and search quality levels [1]. Recently, many text hashing methods have been developed. Most use class label information, and the classification accuracy is quite good [3]. The evaluation metric of these approaches is the percentage of documents from among 100 retrieved documents that have a label identical to the query document. This metric can suitably simulate the finding of identical same class data. Although there is no problem with this approach, sometimes we prefer that the hash result retains the relationship between the two texts, so that there is a hashing method that utilizes the relationship [4]. Utilizing pairwise semantic relation information is better matching for practical application. Unfortunately, this approach can only use binary labeled information, but the relationship between text pairs in the real world cannot be only binary.

In this paper, we want to learn the text hashing function that can utilize the multi-class relation information of text pairs. The contributions of this paper are as follows. First, we present two text hashing models that learn multi-class label information. Second, we test

DOI: 10.24507/icicel.14.04.417

the performance of the two models with experiments. The rest of this paper is organized as follows. Section 2 reviews previous related approaches. Section 3 presents our hashing model. Section 4 provides the experimental results. Finally, Section 5 concludes the paper.

2. **Related Work.** Due to the computational and storage efficiency levels of compact binary codes, hashing has been widely used for approximate nearest neighbor searches. Traditional data independent data hashing methods, such as locality-sensitive hashing (LSH) and spectral hashing have been widely used [5,6]. When supervised information, such as class labels or relative similarity, is available, a supervised data dependent hashing method is a better choice. With the revival of neural networks, deeper learning models are widely used for hashing [11]. These methods were first applied to high-dimensional data or image data. However, if directly used for text data, they usually fail to capture the semantic similarity with regard to the original text. Hence, many text hashing methods have been proposed.

At the very beginning, text semantic hashing used an autoencoder to learn hash functions [1]. These methods built multiple Boltzmann machines (RBMs) to learn the binary unit that is capable of modelling the input text word count data. After training, the binary hash code of any document is acquired by simply thresholding the output of the deepest layer. Furthermore, several studies have explored the power of convolutions neural networks for text hashing with the help of word embedding. For example, in [12], they cleverly changed the model in image processing to carry out text modeling.

In recent years, the probabilistic generative model has achieved great success in several domains. The variational auto-encoder (VAE) [7] is an appealing framework for generative modeling, as it is a hybrid of variational inference [8] and the deep neural network. VAEs acquire the advantages of both deep learning and probabilistic generative models. VAEs achieve state-of-the-art performance in many problems, especially with image data [9]. For text hashing, unsupervised and supervised variational deep semantic hashing (VDSH) [3] have been developed to preserve each content from a document during the text hashing process. These approaches use a VAE framework that is similar to the one presented here.

However, VDSH differs from our hashing scheme, in that it utilizes only categorical class label data to improve the accuracy of semantic text hashing. Pairwise label data have not been used in previous hashing methods. Variational pairwise supervised text hashing (VPSH) [4] utilizes pairwise label information. VPSH utilizes the pairwise information in the most direct and efficient way, and proves that in several text pair problems, text hashing can also get good returns.

3. **Proposed Method.** Let x denote the input text, and z denote the hash code of the given input text. z has a real value or binary code of n. We also refer to n as the number of bits. The encoding process is to infer z from document x, while the decoding process is to reconstruct x from the latent variable z. Intuitively, the latent variable learns from the corpus that captures the key semantic features from x. The hash function is the distribution of the encoding $p(z|x)$. All previous studies attempted to learn this distribution from the corpus. Let x be the bag-of-words representation of a document. x has a length of V, where V is the vocabulary of words that have appeared in the corpus. In this model, the approximation encoding distribution is $q_\phi(z|x)$, and the decoding distribution is $q_\theta(x|z)$, where ϕ and θ are the parameters of the encoder and decoder, respectively. Based on the VAE framework [7], we maximize the variational lower bound, instead of the marginal distribution. Finally, the objective function of text hashing is as follows:

$$L_{vae} = E_{q_\phi(z|x)}[\log q_\theta(x|z)] - D_{KL}(q_\phi(z|x)||p(z)) \tag{1}$$

Let x_1 and x_2 be an input texts pair, and z_1 and z_2 denote the hash code of the given input text pair. In order to use annotated data y to supervised learning, previous works add the label distribution of hash code $p(y|z)$. Previous works using categorical label information simply allow documents in the same category to have a similar hash code [3]. There are some studies that further allow models to learn pairwise distribution $p(y|z_1, z_2)$ [4]. On this basis, we have developed two different models to solve the problem of multiple conditional text hashing.

3.1. **Conditional model.** To utilize multi class information, we first extend to the conditional variational autoencoder (CVAE) [1]. While VAE essentially models the latent variables and data directly, CVAE models the latent variables, data, and both conditioned to some random variables. When CVAE is simply applied to text hashing, the objective function of the single text CVAE is as follows:

$$L_{cvae} = E_{q_\phi(z|x)}[\log q_\theta(x|z, y)] - D_{KL}(q_\phi(z|x, y)||p(z, y)) \tag{2}$$

Then extend Equation (2) into the form of a text pair:

$$\begin{aligned} L_{cvae} = {} & E_{q_\phi(z_1,z_2|x_1,x_2)}[\log q_\theta(x_1', x_2'|z_1, z_2, y)] - D_{KL}(q_\phi(z_1|x_1, y)||p(z_1, y)) \\ & - D_{KL}(q_\phi(z_2|x_2, y)||p(z_2, y)) \end{aligned} \tag{3}$$

In order to facilitate the expansion, the first term in Equation (2) is actually obtained by adding two regeneration losses of x_1 and x_2.

According to this formula, we can get the whole architecture, as shown in Figure 1. Assume the y condition has only three labels y_1, y_2, and y_3. z_1^1, z_1^2, and z_1^3 are the hidden variables generated by x_1 and three labels. The parameters ϕ and θ are shared for processing x_1 and x_2. So throughout the process, there is only one set of encoder parameters ϕ and decoder parameters θ. After training, all the text will generate three kinds of hash code according to the three conditions. If we want to get the label of a text pair, we must calculate the similarity of each condition, if the label with the highest similarity from s_1, s_2 and s_3 is the predicted label. In Figure 1, s_1 is computed by z_1^1 and z_2^1, s_2 is computed by z_1^2 and z_2^2, and so on.

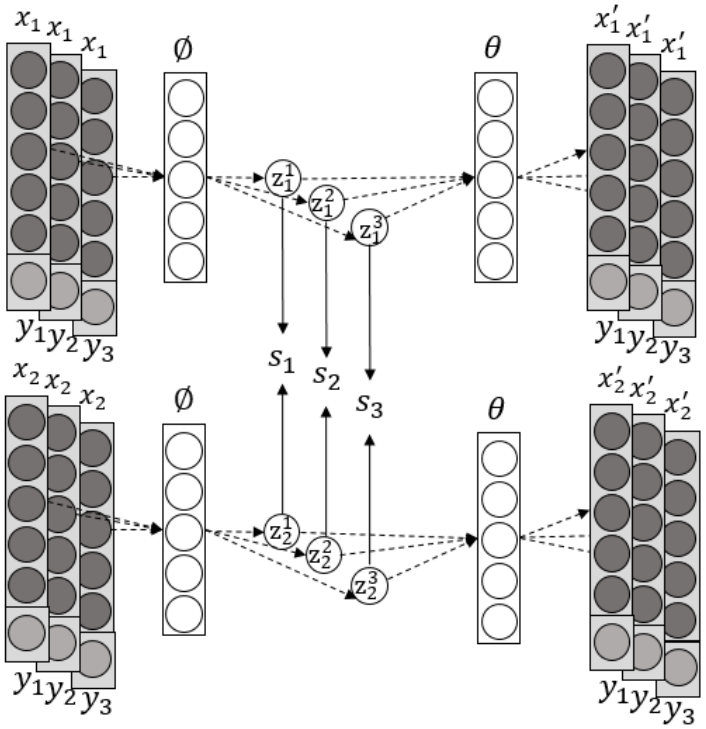

FIGURE 1. Architecture of conditional hashing

3.2. **Multi-model.** In the multiple VAE model, each model will train the corresponding class label. If there are three types of labels, there are three models. The first model, which has ϕ_1, only learns when the class is y_1. To utilize the pairwise label information in a direct way, the goal here is to address the distribution $p(y|z_1, z_2)$, where y is the label information of text x_1 and x_2. We assume that the label information is binary (y is 0 or 1). In order to make the model balance the variational lower bound and discriminate objective, the total objective is given as follows:

$$\begin{aligned} L_{cvae} = & E_{q_\phi(z_1,z_2|x_1,x_2)}[\log q_\theta(x_1', x_2'|z_1, z_2, y)] - D_{KL}(q_\phi(z_1|x_1, y)||p(z_1, y)) \\ & - D_{KL}(q_\phi(z_2|x_2, y)||p(z_2, y)) + W_t(y - \mathrm{sim}(z_1, z_2)) \end{aligned} \tag{4}$$

where, W_t is the label weight that controls the supervised influence. A high weight will cause the entire encoder to generate a more similar hash code for x_1 and x_2, in order to make the similarity higher.

According to this formula, we can get the whole architecture, as shown in Figure 2. Assume the y condition has only three labels y_1, y_2 and y_3, and that there are three kinds of encoders and decoders. The three parameters ϕ_1, ϕ_2, ϕ_3 are independent, and do not affect each other. ϕ_1 is only learned when the label information of x_1 and x_2 is y_1. Since the encoder with ϕ_1 is only trained when the label is equal to y_1, this encoder only learns the hash code with the label y_1.

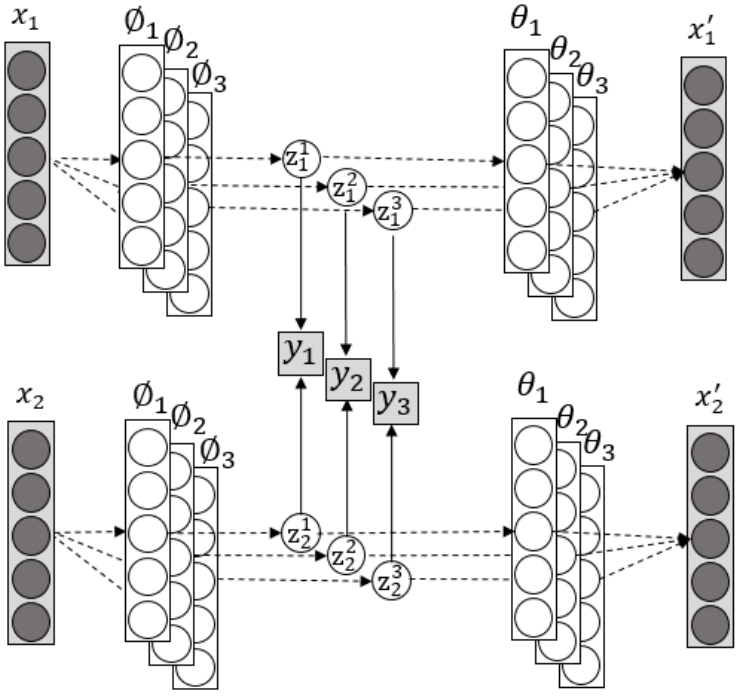

FIGURE 2. Architecture of multi-conditional hashing

After training, all the text will generate three kinds of hash code according to the three labels. If we want to get the label of the text pair, we must calculate the similarity of each label, if the label with the highest similarity is the predicted label. This is calculated in the same way as in the previous model.

4. **Experiments.**

4.1. **Dataset and set-ups.** In this paper, we use the Stanford Natural Language Inference (SNLI) Corpus dataset. It is a popular bench-mark dataset that contains 570k human-written English sentence pairs. Each text pair with human annotations indicates

whether each pair captures a contradiction/neutral/entailment relationship. We measure the three 3-way classification accuracy performance of our algorithm on this dataset.

The Adam optimizer is widely used in VAEs. We use the Adam optimizer [13] with a learning rate of 0.001, and use the learning rate exponential decay with a factor of 0.96 for every 10,000 steps. We also use the dropout technique with a value of 0.9 to alleviate over-fitting. VAE-based text hashing and several other models exploiting the same datasets use dropout at 0.9 in common. Therefore, we also use the same value. We set the starting label weight parameter in Equation (4) to 0.4. All experiments are conducted on a server with an Intel i7-6850K CPU, a NVIDIA GeForce GTX TITAN X GPU, and 16 GB of main memory.

4.2. **Accuracy.** We use the variational pairwise supervised text hashing (VPSH) as a baseline for comparisons. Because previous supervised methods do not utilize pairwise label information, we do not compare the performance with those methods. We use the easiest way to let VPSH train, and calculate the accuracy.

Table 1 shows the results of our model and the baseline. Our two models outperform VPSH with various numbers of bits. Although our two models both use the label information of the training set, the multi VAE model works significantly better than the others.

Table 1. Accuracy of 3-way classification of SNLI dataset in different bits

model	64-bit	128-bit	256-bit
VPSH	***44.2***	***45.2***	***45.3***
CVAE	***45.6***	**47.3**	**45.2**
MVAE	**57.7**	**59.7**	**59.4**

4.3. **Number of same similarity.** In order to clarify why the two proposed models show such different accuracy, we also made a special corresponding analysis. In our prediction phase, our two models will generate three similarities in SNLI, respectively. Then according to the similarity, we determine the prediction class. In fact, these three similarities are likely to be exactly the same; we must randomly choose the prediction class. This phenomenon is especially obvious when the number of bits is small.

We count the proportion of the same similarity of the two models at 64-bit. Table 2 shows that only 45 percent of the similarity of the CVAE model is different, so many of the predictive classes are random, resulting in such low accuracy. In fact, we can see in Equation (3) that since all the encoders in the CVAE model are shared, when the label is changed only when the input is changed, the latent variable is likely to be similar, and the hash code is binarized according to this latent variable. In this case, the z_1^1 and z_1^2 are likely to be the same, and the same hash code calculates the same similarity.

Table 2. Probability of the same similarity in the three classes

model	Each one is different (%)	Two of the three are the same (%)	All are the same (%)
CVAE	***45***	**26**	***27***
MVAE	**98**	**1.3**	**0.04**

5. **Conclusions.** In this paper, we present a novel multi-conditional semantic text hashing method that exploits multi-label pairwise label information. More specifically, the method can learn text pairwise label information in a more direct manner. We also

experimented with comparing the two models and the baseline, and analyze the experimental results. If we want to map different relationships to the same text space, it is obviously not a sensible choice to use conditional generation with one shared encoder, because this method is likely to mix all the classes together, and finally generate hash code, in which there will be some problems.

Acknowledgment. This research was supported in part by the National Research Foundation of Korea (NRF), funded by the Ministry of Science, ICT and Future Planning, through the Basic Science Research Program under Grant 2017R1E1A1A03070004. It was also supported in part by NRF, through the BK21 Plus for Pioneers in Innovative Computing (Dept. of Computer Science and Engineering, SNU) under Grant 21A20151113068. The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

[1] R. Salakhutdinov and G. Hinton, Semantic hashing, *International Journal of Approximate Reasoning*, vol.50, no.7, pp.969-978, 2009.

[2] A. Alexandr and P. Indyk, Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions, *Communications of the ACM*, vol.51, no.1, p.117, 2008.

[3] S. Chaidaroon and Y. Fang, Variational deep semantic hashing for text documents, *Proc. of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2017.

[4] R. Xuan, J. Shim and S. Lee, Variational deep semantic text hashing with pairwise labels, *Proc. of the 13th International Conference on Ubiquitous Information Management and Communication (IMCOM 2019)*, 2019.

[5] M. Datar, N. Immorlica, P. Indyk and V. S. Mirrokni, Locality-sensitive hashing scheme based on p-stable distributions, *Proc. of the 20th Annual Symposium on Computational Geometry*, 2004.

[6] Y. Weiss, A. Torralba and R. Fergus, Spectral hashing, *Advances in Neural Information Processing Systems*, 2009.

[7] D. P. Kingma and M. Welling, Auto-encoding variational bayes, *arXiv Preprint*, arXiv:1312.6114, 2013.

[8] M. J. Wainwright and M. I. Jordan, *Graphical Models, Exponential Families, and Variational Inference*, Now Publishers, 2008.

[9] X. Yan, J. Yang, K. Sohn and H. Lee, Attribute2image: Conditional image generation from visual attributes, *European Conference on Computer Vision*, 2016.

[10] K. Sohn, H. Lee and X. Yan, Learning structured output representation using deep conditional generative models, *Advances in Neural Information Processing Systems*, 2015.

[11] H. Zhu, M. Long, J. Wang and Y. Cao, Deep hashing network for efficient similarity retrieval, *The 30th AAAI Conference on Artificial Intelligence*, 2016.

[12] Z. Yang, Z. Hu and R. Salakhutdinov, Improved variational autoencoders for text modeling using dilated convolutions, *Proc. of the 34th International Conference on Machine Learning*, pp.3881-3890, 2017.

[13] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, *arXiv Preprint*, arXiv:1412.6980, 2014.