Unsupervised Semantic Hashing with Pairwise Reconstruction

Casper Hansen*
University of Copenhagen
c.hansen@di.ku.dk

Christian Hansen*
University of Copenhagen
chrh@di.ku.dk

Jakob Grue Simonsen
University of Copenhagen
simonsen@di.ku.dk

Stephen Alstrup
University of Copenhagen
s.alstrup@di.ku.dk

Christina Lioma
University of Copenhagen
c.lioma@di.ku.dk

ABSTRACT

Semantic Hashing is a popular family of methods for efficient similarity search in large-scale datasets. In Semantic Hashing, documents are encoded as short binary vectors (i.e., hash codes), such that semantic similarity can be efficiently computed using the Hamming distance. Recent state-of-the-art approaches have utilized weak supervision to train better performing hashing models. Inspired by this, we present Semantic Hashing with Pairwise Reconstruction (PairRec), which is a discrete variational autoencoder based hashing model. PairRec first encodes weakly supervised training pairs (a query document and a semantically similar document) into two hash codes, and then learns to reconstruct the same query document from both of these hash codes (i.e., pairwise reconstruction). This pairwise reconstruction enables our model to encode local neighbourhood structures within the hash code directly through the decoder. We experimentally compare PairRec to traditional and state-of-the-art approaches, and obtain significant performance improvements in the task of document similarity search.

KEYWORDS

Semantic Hashing; Variational; Pairwise Reconstruction

ACM Reference Format:
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 (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3397271.3401220

1 INTRODUCTION

Document similarity search is a core information retrieval task, where semantically similar documents are retrieved based on a query document. Large-scale retrieval requires methods that are both effective and efficient, and that can–ideally–be trained in an unsupervised fashion due to the high cost associated with labeling massive data collections. To this end, Semantic Hashing [11] methods learn to transform objects (e.g., text documents) into short binary vector representations, which are called *hash codes*. The semantic similarity between two documents can then be computed using the Hamming distance, i.e., the sum of differing bits between two hash codes, which can be implemented highly efficiently on hardware due to operating on fixed-length bit strings (real-time retrieval among a billion hash codes [12]). Hash codes are typically the same length as a machine word (32 or 64 bits), thus the storage cost for large document collections is relatively low.

The state-of-the-art on unsupervised semantic hashing uses *weak supervision* in different ways to learn hash codes that better encode the structure of local neighbourhoods around each document. NbrReg [2] used BM25 to associate each document with an aggregation of the most similar neighbourhood documents, where two different decoders are trained to reconstruct the document hash code to both the original *and* aggregated neighbourhood document. However, we argue that using multiple different decoders on a single hash code is ineffective, since each decoder will attempt to enforce (potentially) different semantics, which may harm generalization of the hash code. Additionally, an aggregated neighbourhood document is not a *real* document encountered during retrieval, which means that learning from it can introduce further semantic shift. Recently, RBSH [7] proposed to use weak supervision for incorporating a ranking objective in the model, with the aim of improving the hash codes performance in document ranking tasks. However, RBSH uses two weakly (positively and negatively) labeled documents to generate a ranking triplet, each of which is obtained from a noisy relevance estimate, which may lead to larger inaccuracies when combined.

To address the above problems, we propose to use weak supervision to extract the top-K most similar documents to a given query document, which are split into K pairs, each consisting of the query document and a top-K document. Using an end-to-end discrete variational autoencoder architecture, each document within a pair is encoded to a hash code, and through a single decoder they are both trained in an unsupervised fashion to be able to reconstruct the query document (i.e., they are *pairwise* reconstructed to obtain the query document). In contrast to NbrReg [2], our PairRec aims at learning a more generalizable decoding through a single decoder used on pairs of (non-aggregated) documents, as opposed to using different decoders as done in NbrReg. In contrast to RBSH [7], our PairRec is only based on a single weakly labeled document per sample, thus aiming at reducing the inaccuracies originating from comparing noisy relevance estimates for ranking in RBSH.

*Both authors share the first authorship

SIGIR '20, July 25–30, 2020, Virtual Event, China

ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401220

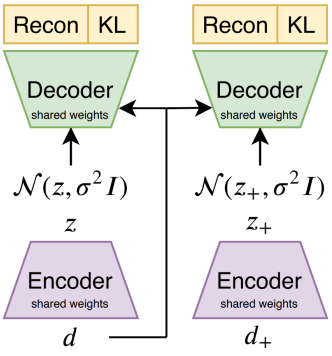

Figure 1: PairRec model overview.

In summary, we **contribute** a novel weakly supervised semantic hashing approach named PairRec, based on our concept of pairwise reconstruction for encoding local neighbourhood structures within the hash code. We experimentally evaluate the effectiveness of PairRec against traditional and state-of-the-art semantic hashing approaches, and show that PairRec obtains significant improvements in the task of document similarity search. In fact, PairRec hash codes generally perform similar or better than the state-of-the-art while using 2-4x fewer bits.

2 RELATED WORK

Early work on semantic hashing used techniques adopted from spectral clustering [14], encapsulating global similarity structures, and later local similarity structures between neighbours found using k-nearest neighbour [16]. Following the popularity of deep learning, VDSH [3] was proposed as a neural model enabling complex encoding of documents, that aimed to learn more descriptive hash codes. Inspired by the benefit of weak supervision in related domains [4, 6], NbrReg [2] was proposed for incorporating aggregated neighbourhood documents in the hash code decoder, for the purpose of incorporating local similarity structure. However, these methods do not learn the hash code in an end-to-end fashion, since they rely on a post-processing rounding stage. To this end, NASH [13] was proposed as an end-to-end trainable variational autoencoder, where bits were sampled according to a learned sample probability vector from a Bernoulli distribution. As a step towards more expressive document encoding, BMSH [5] utilized a Bernoulli mixture prior generative model, but was only able to outperform a simple version of the NASH model, and not consistently outperform the proposed full version. Lastly, RBSH [7] was the first semantic hashing approach that utilized a ranking objective in the model (through sampling semantically similar documents [9]), thus enabling the hash codes to combine both local and global structures for improved retrieval performance. RBSH was able to significantly outperform existing state-of-the-art semantic hashing approaches. Recently, semantic hashing has also been successfully applied to the task of cold-start collaborative filtering, where recent advances enabled a better semantic representation of the items [8].

3 PAIRWISE RECONSTRUCTION BASED HASHING

Pairwise reconstruction based hashing (PairRec) is a discrete variational autoencoder with a pairwise reconstruction loss. Given a document d, PairRec generates an m-bit hash code $z \in \{0,1\}^m$ for d, such that two semantically similar documents have low Hamming distance. Specifically, z is sampled by repeating m consecutive Bernoulli trials based on learned sampling probabilities. Given a similarity function, PairRec is trained on pairs of semantically similar documents, and learns to encode local document neighbourhood structures by training to reconstruct one of the documents from both hash codes (i.e., pairwise reconstruction). We first cover the model architecture and then the pairwise reconstruction loss function. Figure 1 shows a model overview.

To compute the hash code z, we let the document likelihood be conditioned on z and define the conditional document likelihood as a product over word probabilities:

$$p(d|z) = \prod_{j \in \mathcal{W}_d} p(w_j|z) \tag{1}$$

where $\mathcal{W}_d$ denotes the set of all unique words in document d. Based on this, the document log likelihood can be found as:

$$\log p(d) = \log \sum_{z \in \{0,1\}^m} p(d|z)p(z) \tag{2}$$

where $p(z)$ is the hash code prior of a Bernoulli distribution with equal probability of sampling 0 and 1. However, maximizing $\log p(d)$ is intractable in practice [10], so instead we maximise the variational lower bound:

$$\log p(d) \geq E_{Q(\cdot|d)}[\log p(d|z)] - \mathrm{KL}(Q(z|d)||p(z)) \tag{3}$$

where $Q(z|d)$ is a learned approximation of the posterior distribution, and KL is the Kullback-Leibler divergence, which has a closed form solution for Bernoulli distributions [13]. Next, we cover our model's encoder ($Q(z|d)$) and decoder ($p(d|z)$), and subsequently specify the pairwise reconstruction loss.

3.1 Encoder

The approximate posterior $Q(z|d)$ is computed using a feedforward network with two hidden layers with ReLU activations, and a final output layer using a sigmoid activation to get the sampling probability for each bit:

$$Q(z|d) = \mathrm{FF}_\sigma(\mathrm{FF}_{\mathrm{ReLU}}(\mathrm{FF}_{\mathrm{ReLU}}(d \odot e_{\mathrm{imp}}))) \tag{4}$$

where FF denotes a single feed forward layer, $\odot$ is elementwise multiplication, and e_{imp} is a learned word level importance [7]. During training, the bits are Bernoulli sampled according to their sampling probabilities, while the most probable bits are chosen greedily for evaluation. This enables exploration during training, and a deterministic evaluation output. As the sampling is non differentiable, the straight through estimator is used to do back propagation through the sampling [1].

3.2 Decoder

The decoder should reconstruct the original document d. Previous work has shown a single linear projection works well [7, 13] because the hash codes are used for (linear) Hamming distance computations. We compute the word probabilities by a softmax, where the logit for a single word is given by:

$$\mathrm{logit}(w|z) = f(z)^T (E_{\mathrm{word}}(I(w) \odot e_{\mathrm{imp}})) + b_w \tag{5}$$

Table 1: Dataset statistics

	documents	multi-class	classes	unique words
TMC	28,596	Yes	22	18,196
reuters	9,848	Yes	90	16,631
agnews	127,598	No	4	32,154

where $f(z)$ is a noise infused hash code, E_{word} is a word embedding learned during training, $I(w)$ is a one-hot encoding of word w, e_{imp} is the word level importance also used in the encoder, and b_w is a word level bias term. The noise infusion is done by adding Gaussian noise with zero mean and variance σ^2 to the hash code, resulting in lower variance for the gradient estimates [10]. We apply variance annealing to reduce the variance over time while training the model. Thus, the conditional document log likelihood is given by:

$$\log p(d|z) = \sum_{j \in \mathcal{W}_d} \log \frac{e^{\text{logit}(w_j|z)}}{e^{\sum_{i \in \mathcal{W}_{\text{all}}} \text{logit}(w_i|z)}} \tag{6}$$

where $\mathcal{W}_{\text{all}}$ is the set of unique words over all documents.

3.3 Pairwise Reconstruction

PairRec assumes access to some similarity function, which given a document d can be used to obtain a set of the K most similar documents $\mathcal{D}_d^K$. A training pair (d, d_+) is constructed from the document d and a single document sampled from the set, i.e., $d_+ \in \mathcal{D}_d^k$. Using the variational lower bound from Eq. 3, the pairwise reconstruction loss for the pair is defined as:

$$\begin{aligned}\mathcal{L}_{\text{PairRec}} = &- E_{Q(\cdot|d)}[\log p(d|z)] + \beta \text{KL}(Q(z|d)||p(z)) \\ &- E_{Q(\cdot|d_+)}[\log p(d|z_+)] + \beta \text{KL}(Q(z_+|d_+)||p(z_+))\end{aligned} \tag{7}$$

Note that this is a negation of the variational lower bound because the loss needs to be minimized. The loss consists of two parts: (i) the first part is an ordinary variational lower bound for document d; (ii) in the second variational lower bound, document d_+ is used in the encoding, while the decoding is of document d. This transfers local neighbourhood structure from the document space into the Hamming space, since z_+ needs to be able to reconstruct the original d. Lastly, the KL divergence is weighed by a tuneable parameter.

4 EXPERIMENTAL EVALUATION

We use 3 publicly available datasets commonly used in related work [3, 7, 13] consisting of TMC, reuters, and agnews (see Table 1). *TMC*[1] is a multi-class dataset of air trafic reports. *reuters*[2] is a multi-class dataset of news, and filtered such that a document is removed if none of its labels are among the 20 most frequent labels (similarly done by [3, 7, 13]). Lastly, *agnews* [17] is a single-class dataset of news.

We use the preprocessed data provided in [7], where TF-IDF is used as the document representation and words occurring only once are removed, as well as words occurring in more than 90% of the documents. The datasets were split into training, validation, and testing (80%/10%/10%). We use the validation loss to determine when to stop training a model (using early stopping with a patience of 5 epochs).

[1]https://catalog.data.gov/dataset/siam-2007-text-mining-competition-dataset
[2]http://www.nltk.org/book/ch02.html

4.1 Baselines and Tuning

We compare our PairRec against traditional post-processing rounding approaches (SpH [14], STH [16], and LCH [15]), neural post-processing rounding approaches (VDSH [3] and NbrReg [2]), and neural end-to-end approaches (NASH [13] and RBSH [7]). NbrReg and RBSH both make use of weak supervision as discussed in Section 1. The baselines are tuned as described in their original papers.

In PairRec[3], we tune the number of hidden units in each encoder layers across {500, 1000}, and the number of top K reconstruction pairs across {1, 2, 5, 10, 25, 50, 100, 150, 200}. For obtaining the reconstruction pairs, we generate 64 bit STH [16] hash codes and retrieve the top K most semantically similar documents (STH was also used by RBSH [7]). For the KL divergence, we tune β from $\{0, 0.01, 0.1\}$. Note that when $\beta = 0$ is chosen, it corresponds to removing the regularizing KL divergence from the loss. For the variance annealing, we use an initial value of 1 and reduce it by 10^{-6} every iteration (as done in [7]). Lastly, we use the Adam optimizer with a learning rate of 0.0005.

4.2 Evaluation Setup

Following related work [2, 3, 7, 13], we evaluate the semantic hashing approaches based on their top 100 retrieval performance using Prec@100 based on the Hamming distance. Given a query document, we define a retrieved document to be relevant if it shares at least one label with the query document (to ensure that we can accommodate the multi-class datasets, where each document may have one or more associated labels).

4.3 Results

We generate hash codes of {8, 16, 32, 64, 128} bits and report Prec@100 in Table 2. The best performing method for each dataset and bit size is highlighted in bold, and statistically significant improvements (0.05 level) using a two tailed paired t-test are indicated by ▲.

Our PairRec method consistently outperforms all the traditional and state-of-the-art approaches across all datasets on all bit sizes. RBSH, which also utilizes weak supervision for generating ranking triplets, consistently obtains the second best scores, indicating the benefit of weak supervision for semantic hashing. While NbrReg also makes use of weak supervision (for creating aggregated neighbourhood documents), it performs worse than both NASH and RBSH, but generally better than VDSH, to which its architecture is most similar to. The absolute Prec@100 increases depend on dataset and bit size, but overall PairRec improves state-of-the-art by 1-4%, which correspondingly enables PairRec hash codes to generally perform better or similar to state-of-the-art hash codes with 2-4x more bits.

4.4 Impact of Pairwise Reconstruction

The primary novelty of PairRec is the introduction of pairwise reconstruction. We study the impact of (i) the performance gain obtained by the pairwise reconstruction, and (ii) the performance variance across a varying number of document pairs.

[3]We make our code available at https://github.com/casperhansen/PairRec.

Table 2: Prec@100 with different bit sizes. Bold numbers highlights the highest scores, and ▲ represents statistically significant improvements over RBSH (the best baseline) at the 0.05 level using a two tailed paired t-test.

	Agnews					Reuters					TMC				
	8	16	32	64	128	8	16	32	64	128	8	16	32	64	128
SpH [14]	.3596	.5127	.5447	.5265	.5566	.4647	.5250	.6311	.5985	.5880	.5976	.6405	.6701	.6791	.6842
STH [16]	.6573	.7909	.8243	.8377	.8378	.6981	.7555	.8050	.7984	.7748	.6787	.7218	.7695	.7818	.7797
LCH [15]	.7353	.7584	.7654	.7800	.7879	.5619	.6235	.6587	.6610	.6586	.6546	.7028	.7498	.7817	.7948
VDSH [3]	.6418	.6754	.6845	.6802	.6714	.6371	.6686	.7063	.7095	.7129	.6989	.7300	.7416	.7310	.7289
NbrReg [2]	.4274	.7213	.7832	.7988	.7976	.5849	.6794	.6290	.7273	.7326	.7000	.7012	.6747	.7088	.7862
NASH [13]	.7207	.7839	.8049	.8089	.8142	.6202	.7068	.7644	.7798	.8041	.6846	.7323	.7652	.7935	.8078
RBSH [7]	.8066	.8288	.8363	.8393	.8381	.7409	.7740	.8149	.8120	.8088	.7620	.7959	.8138	.8224	.8193
PairRec (ours)	**.8119**▲	**.8354**▲	**.8452**▲	**.8492**▲	**.8498**▲	**.7502**▲	**.8028**▲	**.8268**▲	**.8329**▲	**.8468**▲	**.7656**▲	**.7991**▲	**.8239**▲	**.8280**▲	**.8303**▲

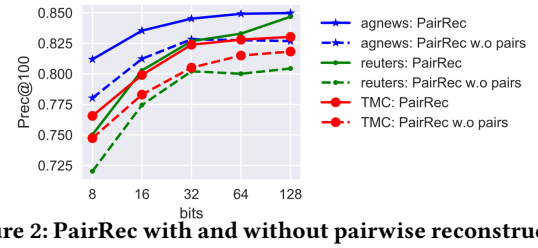

Figure 2: PairRec with and without pairwise reconstruction.

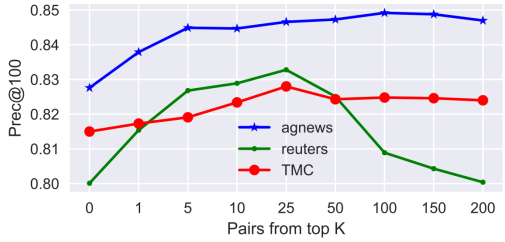

Figure 3: 64 bit PairRec while varying the top K.

Performance gain by pairwise reconstruction. We compute the Prec@100 with and without the pairwise reconstruction and plot the scores in Figure 2. The largest improvements occur for 64-128 bit on the reuters dataset, but across all datasets and bit sizes, pairwise reconstruction obtains consistent improvements. In comparison, the original RBSH paper [7] also did an ablation with and without weak supervision, but found their improvements to be primarily isolated to 8-16 bits. This further highlights the benefit of using a single weakly supervised document, rather than combining multiple sources for generating ranking triplets as done in RBSH.

Performance variance across number of pairs. We now investigate the impact of the choice of the number of pairs. We fix the bit size to 64 and plot the Prec@100 for all datasets using {0, 1, 5, 10, 25, 50, 100, 150, 200} pairs, where 0 corresponds to no pairwise reconstruction. The optimal values for agnews, reuters, and TMC are 100, 25, and 25, respectively. Interestingly, Prec@100 drops after 25 pairs on reuters, which most likely is due to a combination of its small dataset size and high number of classes, corresponding to pairs from top 50 and above no longer being sufficiently semantically similar to the original document. In contrast, for TMC and agnews, we observe no significant performance drop as the number of pairs is increased. In all cases, we note that the optimal value of pairs is also identified by the model parameter configuration with the minimum loss.

5 CONCLUSION

Inspired by recent advances in semantic hashing using weak supervision, we presented a novel semantic hashing approach with pairwise reconstruction (PairRec). PairRec is a discrete variational autoencoder trained on semantically similar document pairs (obtained through weak supervision), where the model is trained such that the hash codes from both pairwise documents reconstruct the same document. We denote this type of reconstruction as *pairwise reconstruction*; it enables PairRec to encode local neighbourhood structures within the hash code. In an experimental comparison, PairRec was shown to consistently outperform existing state-of-the-art semantic hashing approaches. These improvements generally enable PairRec hash codes to use 2-4x fewer bits than state-of-the-art hash codes while achieving the same or better retrieval performance.

REFERENCES

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

[2] Suthee Chaidaroon, Travis Ebesu, and Yi Fang. 2018. Deep Semantic Text Hashing with Weak Supervision. SIGIR, 1109–1112.

[3] Suthee Chaidaroon and Yi Fang. 2017. Variational deep semantic hashing for text documents. In *SIGIR*. 75–84.

[4] Mostafa Dehghani, Hamed Zamani, Aliaksei Severyn, Jaap Kamps, and W Bruce Croft. 2017. Neural ranking models with weak supervision. In *SIGIR*. ACM, 65–74.

[5] Wei Dong, Qinliang Su, Dinghan Shen, and Changyou Chen. 2019. Document Hashing with Mixture-Prior Generative Models. In *EMNLP*. 5226–5235.

[6] Casper Hansen, Christian Hansen, Stephen Alstrup, Jakob Grue Simonsen, and Christina Lioma. 2019. Neural Check-Worthiness Ranking with Weak Supervision: Finding Sentences for Fact-Checking. In *Companion Proceedings of WWW*. 994âĂŞ1000.

[7] Casper Hansen, Christian Hansen, Jakob Grue Simonsen, Stephen Alstrup, and Christina Lioma. 2019. Unsupervised Neural Generative Semantic Hashing. In *SIGIR*. 735âĂŞ744.

[8] Casper Hansen, Christian Hansen, Jakob Grue Simonsen, Stephen Alstrup, and Christina Lioma. 2020. Content-aware Neural Hashing for Cold-start Recommendation. In *SIGIR*. in press.

[9] Casper Hansen, Christian Hansen, Jakob Grue Simonsen, and Christina Lioma. 2019. Neural weakly supervised fact check-worthiness detection with contrastive sampling-based ranking loss. In *CLEF-2019 CheckThat! Lab*.

[10] Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *ICLR*.

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

[12] Ying Shan, Jian Jiao, Jie Zhu, and JC Mao. 2018. Recurrent binary embedding for gpu-enabled exhaustive retrieval from billion-scale semantic vectors. In *KDD*. 2170–2179.

[13] 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 *ACL*. 2041–2050.

[14] Yair Weiss, Antonio Torralba, and Rob Fergus. 2009. Spectral hashing. In *NeurIPS*. 1753–1760.

[15] Dell Zhang, Jun Wang, Deng Cai, and Jinsong Lu. 2010. Laplacian co-hashing of terms and documents. In *ECIR*. Springer, 577–580.

[16] Dell Zhang, Jun Wang, Deng Cai, and Jinsong Lu. 2010. Self-taught hashing for fast similarity search. In *SIGIR*. ACM, 18–25.

[17] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NeurIPS*. 649–657.