Online Enhanced Semantic Hashing: Towards Effective and Efficient Retrieval for Streaming Multi-Modal Data

Xiao-Ming Wu, Xin Luo*, Yu-Wei Zhan, Chen-Lu Ding, Zhen-Duo Chen, Xin-Shun Xu

School of Software, Shandong University
{wuxiaoming.alg, luoixin.lxin, zhanyuweiliif, dingchenlu200103, chenzd.sdu}\@gmail.com, xuxinshun@sdu.edu.cn

Abstract

With the vigorous development of multimedia equipments and applications, efficient retrieval of large-scale multi-modal data has become a trendy research topic. Thereinto, hashing has become a prevalent choice due to its retrieval efficiency and low storage cost. Although multi-modal hashing has drawn lots of attention in recent years, there still remain some problems. The first point is that existing methods are mainly designed in batch mode and not able to efficiently handle streaming multi-modal data. The second point is that all existing online multi-modal hashing methods fail to effectively handle unseen new classes which come continuously with streaming data chunks. In this paper, we propose a new model, termed Online enhAnced Semantic haShing (OASIS). We design novel semantic-enhanced representation for data, which can help handle the new coming classes, and thereby construct the enhanced semantic objective function. An efficient and effective discrete online optimization algorithm is further proposed for OASIS. Extensive experiments show that our method can exceed the state-of-the-art models. For good reproducibility and benefiting the community, our code and data are already publicly available.

Introduction

With the rapid development of electronic devices and real-world applications, the past decades have witnessed a dramatic increase in the number of multimedia data (Kang, Li, and Zhou 2016; Li and Tang 2016; Xu et al. 2020; Li, Tang, and Mei 2018; Zhu et al. 2020b). Therefore, efficient retrieval of multimedia data has become a hot research topic. Learning to hash has arisen to be a promising choice because of its fast retrieval speed and low storage consumption (Li et al. 2020; Chen et al. 2021b; Weng and Zhu 2021; Liu et al. 2019b). Roughly speaking, we could divide existing methods into uni-modal hashing (Shi et al. 2022; Wang et al. 2018a; Liu et al. 2019a; Wang et al. 2019; Luo et al. 2019), cross-modal hashing (Liu et al. 2019c; Xie et al. 2020; Jin, Li, and Tang 2020; Nie et al. 2020; Hu et al. 2021), and multi-modal hashing (Liu et al. 2012; Shen et al. 2015; Zhu et al. 2020a). Thereinto, multi-modal hashing requires that both database and query samples provide heterogeneous multi-modal features. In this paper, we focus on multi-modal hashing which draws a significant need in real-world multimedia retrieval tasks while is relatively unexplored.

Despite the promising performance of hashing methods, failing to efficiently learn from streaming data may be an obstacle to applying them to real-world applications. Those batch-based hashing methods assume all training data are available in advance for training and such batch mode needs large space cost. To overcome the limitation, many efforts have been devoted to online hashing. Similarly, we could roughly classify the literature into online uni-modal hashing (Huang, Yang, and Zheng 2013; Cakir and Sclaroff 2015; Cakir et al. 2017; Chen, King, and Lyu 2017; Weng and Zhu 2020; Tian, Ng, and Wang 2019; Chen et al. 2021a), online cross-modal hashing (Xie, Shen, and Zhu 2016; Qi, Wang, and Li 2017; Wang, Luo, and Xu 2020; Yi et al. 2021; Zhan et al. 2022), and online multi-modal hashing (Xie et al. 2017; Lu et al. 2019a).

In online hashing literature, one of the most important problems is the class incremental problem, which means new (unknown) categories may appear along with new data chunks. However, most existing works fail to solve it and only a few solutions have been proposed. Some works rise to the challenge by means of offline coding strategies, such as Error Correcting Output Codes (Cakir, Bargal, and Sclaroff 2017) and Hadamard matrix (Lin et al. 2018, 2020). Some methods try to increase the number of hash bits for better representation (Mandal, Annadani, and Biswas 2018). Some efforts are made through training multiple complementary hash tables incrementally (Tian et al. 2020). Some methods design an end-to-end model to figure out this problem (Wu et al. 2019; Chen et al. 2019). However, these solutions are designed for uni-modal and cross-modal hashing while the class incremental problem in online multi-modal hashing is still an open problem without investigations. To the best of our knowledge, the only two online multi-modal hashing methods are Online Dynamic Multi-View Hashing (ODMVH) (Xie et al. 2017) and Flexible Online Multi-modal Hashing (FOMH) (Lu et al. 2019a). However, ODMVH is an unsupervised method and FOMH implicitly assumes that no new categories come with streaming data. In other words, no existing online multi-modal hashing can handle the scenario where new (unknown) categories continually appear along with new data chunks. Besides,
we argue that the problem settings of some existing online hashing, which could tackle the class incremental problem, may be impractical: 1) some may relearn the hash codes of old data; 2) some may reuse the original features of old data. When facing large-scale applications, those settings will become inefficient. Hence, in this paper, we formalize the problem by giving some constraints. 1) Multimedia data come in a streaming fashion. 2) The hash codes keep unchanged once learnt and the hash code length is fixed. 3) When updating the hash function, features of new data can be utilized but features of old data chunks are unusable. 4) New (unknown) categories may continually appear along with new data chunks.

To overcome the aforementioned limitations and satisfy the requirements, in this paper, we propose a novel method termed Online enhanced Semantic hashing (OASIS), which could learn hash codes and functions when the multimedia instances arrive in streaming fashion. As far as we know, this is the first attempt to investigate the class incremental problem in the context of multi-modal hashing. The main contributions are as follows.

- **Task contribution.** We thoroughly investigate the online multi-modal hashing and define a problem setting to better organize this research topic. The proposed task is more challenging and more practical.

- **Technical contribution.** This paper conceives a new online multi-modal hashing method and proffers an efficient and effective discrete online optimization algorithm. We generate a novel semantic-enhanced representation for data and thereby construct a semantically enhanced objective function. Besides, with the designed representation, we could handle new categories well. Extensive experiments on two benchmark datasets show that our method can exceed the state-of-the-art models.

- **Community contribution.** Our codes are already publicly available\(^1\). We hope our work may become one of the enablers for the valuable but relatively unexplored topic and the learning to hash community.

**Method**

As shown in Figure 1, our proposed OASIS contains several modules, i.e., semantic-enhanced representation generation, enhanced semantic hashing, and hash function learning. Details of our OASIS are introduced in this section.

**Notations and Problem Definition**

In this paper, we assume that multi-modal data comes at a streaming manner. At the \(t\)-th round, suppose that we have \(N^{(t)}\) old multi-modal samples \(\tilde{X}^{(t)}\), which are observed before current round, and \(n^{(t)}\) new ones \(\tilde{X}^{(t)}\), where \(N^{(t)} = n^{(t)} + \cdots + n^{(t-1)}\). More specifically, as all samples contain features from \(M\) modalities, we denote the feature matrix of old data from the \(m\)-th modality as \(\tilde{X}_m^{(t)} \in R^{d_m \times N^{(t)}}\) and the new samples’ feature matrix of the \(m\)-th modality as \(\tilde{X}_m^{(t)} \in R^{d_m \times n^{(t)}}\), where \(m \in \{1, \ldots, M\}\) and \(d_m\) is the dimensionality. Meanwhile, we can acquire the labels \(\tilde{I}^{(t)} \in \{0, 1\}^{c_n^{(t)} \times n^{(t)}}\) of \(\tilde{X}^{(t)}\), where \(c_n^{(t)}\) is the number of old classes observed in former rounds, \(c_n^{(t)}\) is the number of new classes which first appear in the \(t\)-th round and \(c_n^{(t)} = c_n^{(1)} + \cdots + c_n^{(t-1)}\). Then, we learn the hash codes \(\tilde{B}^{(t)} \in \{-1, 1\}^{r \times n^{(t)}}\) for \(\tilde{X}^{(t)}\), where \(r\) is the code length. Similarly, \(\tilde{L}^{(t)}\) and \(\tilde{B}^{(t)} \in \{-1, 1\}^{r \times N^{(t)}} = [\tilde{B}^{(1)}, \ldots, \tilde{B}^{(t-1)}]\) are the label matrix and hash codes of \(\tilde{X}^{(t)}\), respectively.

We let \(I, 1, 0\) denote the identity matrix, an all-one matrix, and an all-zero matrix, respectively. Regarding the definition of the operator, we use \(\text{tr}(\cdot)\) to represent the trace operation of the matrix, \(\cdot\) to represent the Frobenius form of the matrix, and \(\text{sign}(\cdot)\) to represent the sign function.

In this paper, we focus on the crucial but understudied task, i.e., online multi-modal hashing. We hope our model

---

\(^1\)https://github.com/DravenALG/OASIS
satisfies the following requirements. 1) Multi-modal data come in a streaming fashion (\(\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \ldots\) and so on). 2) The hash codes keep unchanged once learnt and the hash code length \(r\) is fixed. 3) When updating the hash function, features of new data \(\mathbf{X}^{(t)}\) can be utilized but \(\mathbf{X}^{(t)}\) is unavailable. 4) New (unknown) categories may continually appear along with new data chunks (\(c^{(t)}_n \geq 0\)).

Formulation

Basic Hashing Formulation. We start with one of the commonest formulations in hashing (Kang, Li, and Zhou 2016), i.e., \(|r \mathbf{S} - \mathbf{B}^T \mathbf{B}|^2_F\). One of the greatest weaknesses of this formulation comes from the size of the pairwise similarity matrix \(\mathbf{S}\) which is square of the number of training samples. To provide a remedy to this dilemma, many efforts have been made and the frequently-used strategy in online hashing literature is redefining \(\mathbf{S}\) as,

\[
\mathbf{S} = 2 \mathbf{P}^T \mathbf{P} - 11^T, \tag{1}
\]

where \(\mathbf{P}\) is the 2-norm normalized label matrix defined as \(\mathbf{P}_j = \mathbf{L}_j / ||\mathbf{L}_j||_2\), \(||\cdot||\) represents the length of the vector, \(\mathbf{L}_j\) and \(\mathbf{P}_j\) represents the \(j\)-th column of \(\mathbf{L}\) and \(\mathbf{P}\), respectively. Then, the computational complexity can be reduced from square to linear. For the sake of accommodating to online streaming data, pairwise \(\mathbf{S}^{(t)}\), whose size is \((N^{(t)} + n^{(t)}) \times (N^{(t)} + n^{(t)})\), at the \(t\)-th round could be split into four parts,

\[
\begin{align*}
\mathbf{S}^{(t)}_{oo} &= 2 \mathbf{P}^{(t)}_o \mathbf{P}^{(t)}_o - 11^T, \quad \mathbf{S}^{(t)}_{no} = 2 \mathbf{P}^{(t)}_o \mathbf{P}^{(t)}_o - 11^T, \\
\mathbf{S}^{(t)}_{on} &= 2 \mathbf{P}^{(t)}_o \mathbf{P}^{(t)}_o - 11^T, \quad \mathbf{S}^{(t)}_{nn} = 2 \mathbf{P}^{(t)}_o \mathbf{P}^{(t)}_o - 11^T, 
\end{align*}
\tag{2}
\]

where \(\mathbf{S}^{(t)}_{oo}\) is the semantic similarity between old data, \(\mathbf{S}^{(t)}_{on}\) is the semantic similarity between new data and old data, \(\mathbf{S}^{(t)}_{no}\) is the semantic similarity between new data and old data, \(\mathbf{S}^{(t)}_{nn}\) is the semantic similarity between new data and new data. \(\mathbf{S}^{(t)}_{no}\) and \(\mathbf{S}^{(t)}_{nn}\) are the supervised online multi-modal hashing (Lu et al. 2019a) and \(\mathbf{S}^{(t)}_{nn}\) only and omits third strategy. Such strategy may lose the information of old data. The state-of-the-art online cross-modal hashing (Wang, Luo, and Xu 2020) learns from the above four parts and is endowed with impressive results. However, it is impossible to handle class incremental problem because the incorrect dimensions for matrix multiplication error will happen when performing \(\mathbf{P}^{(t)} \mathbf{P}^{(t)}_o\) operation if new classes come. In other words, \(\mathbf{S}^{(t)}_{no}\) and \(\mathbf{S}^{(t)}_{nn}\) cannot be calculated due to the different number of the classes of the new and old data.

We propose a novel solution to tackle the class incremental problem which is detailedly shown below.

Semantic-Enhanced Representation. Label names are often naturally well separated from each other and contain good expressions of category-specific semantics (Wang et al. 2018b). Thus, we introduce the embedding of label names into our framework to gain more powerful semantic information to guide the hash learning.

In online setting, as new labels may continually appear along with new data chunk, we use \(\mathbf{K}^{(t)}_o \in \mathbb{R}^{c^{(t)}_o \times f}\) to represent the old labels’ name embeddings and \(\mathbf{K}^{(t)}_n \in \mathbb{R}^{c^{(t)}_n \times f}\) to denote new labels’ name embeddings, where \(f\) is the dimensionality of word2vec vector. By concatenating them, the overall label name embeddings can be obtained \(\mathbf{K}^{(t)} \in \mathbb{R}^{(c^{(t)}_o + c^{(t)}_n) \times f}\). In our work, Google’s word2vec model (Mikolov et al. 2013) is leveraged and \(f = 300\).

Then, we propose a semantic-enhanced matrix \(\mathbf{G}^{(t)} \in \mathbb{R}^{f \times n^{(t)}}\) that comprises both label matrix and label name embeddings. The semantic-enhanced matrix is defined as,

\[
\mathbf{G}^{(t)} = \frac{\mathbf{K}^{(t)} \mathbf{L}^{(t)}_j}{||\mathbf{K}^{(t)} \mathbf{L}^{(t)}_j||}, \quad j = 1, 2, \ldots n^{(t)}, \tag{3}
\]

where \(\mathbf{L}^{(t)}_j\) represents the \(j\)-th column of \(\mathbf{L}^{(t)}\), \(\mathbf{G}^{(t)}\) represents the \(j\)-th column of \(\mathbf{G}^{(t)}\), and \(\mathbf{|| \cdot ||}\) represents the length of vectors. Furthermore, we turn each column of the matrix into a unit vector, which can easily express the cosine similarity when calculating \(\mathbf{S}^{(t)}\). The semantic-enhanced matrix offers instances semantic-rich representations and plays an important role in guiding the learning of OASIS.

Enhanced Semantic Hashing. Now, let’s revisit Eq.(2). With the semantic-enhanced representation of instances, we could redefine the pairwise semantic similarity as follows,

\[
\begin{align*}
\mathbf{S}^{(t)}_{oo} &= \mathbf{G}^{(t)} \mathbf{G}^{(t)}, \quad \mathbf{S}^{(t)}_{no} = \mathbf{G}^{(t)} \mathbf{G}^{(t)}, \\
\mathbf{S}^{(t)}_{on} &= \mathbf{G}^{(t)} \mathbf{G}^{(t)}, \quad \mathbf{S}^{(t)}_{nn} = \mathbf{G}^{(t)} \mathbf{G}^{(t)}, 
\end{align*}
\tag{4}
\]

where \(\mathbf{G}^{(t)} \in \mathbb{R}^{f \times n^{(t)}}\) denotes the semantic-enhanced representation of old samples and can be obtained before round \(t\). It is worth noting that Eq.(4) could naturally handle the class incremental problem. The size of \(\mathbf{G}^{(t)}\) is \(f \times n^{(t)}\) and the size of \(\mathbf{G}^{(t)}\) is \(f \times n^{(t)}\), the dimensions for matrix multiplication is correct. We elaborately bypass the obstacle mentioned above and could construct the connections between new and old classes. Then, we propose the loss guided by enhanced semantic pairwise similarity as,

\[
\begin{align*}
\min_{\mathbf{B}^{(t)} \in \{-1,1\}^{r \times n^{(t)}}} & \|r \mathbf{S}^{(t)}_{nn} - \mathbf{B}^{(t)T} \mathbf{B}^{(t)}\|^2_F \\
& + \|r \mathbf{S}^{(t)}_{on} - \mathbf{B}^{(t)T} \mathbf{B}^{(t)}\|^2_F + \|r \mathbf{S}^{(t)}_{no} - \mathbf{B}^{(t)T} \mathbf{B}^{(t)}\|^2_F, \tag{5}
\end{align*}
\]

where \(\mathbf{S}^{(t)}_{no}\) is omitted because to-be-learnt \(\mathbf{B}^{(t)}\) is irrelevant to it. In this equation, knowledge of both old and new classes can be embedded and may constructively guide the hash learning.

To further enhance the semantic information that guides the hash learning, we also directly construct the connection between hash codes and semantic-enhanced representation,

\[
\begin{align*}
\min_{\mathbf{B}^{(t)}, \mathbf{U}^{(t)}} & \|\mathbf{G}^{(t)} - \mathbf{U}^{(t)} \mathbf{B}^{(t)}\|^2_F + \|\mathbf{G}^{(t)} - \mathbf{U}^{(t)} \mathbf{B}^{(t)}\|^2_F \\
& + \theta \|\mathbf{U}^{(t)}\|^2_F, \quad s.t. \mathbf{B}^{(t)} \in \{-1,1\}^{r \times n^{(t)}}, \tag{6}
\end{align*}
\]

where \(\mathbf{U}^{(t)}\) is a mapping matrix and \(\theta\) is a hyperparameter controlling the regularization term. Here, we use the
same mapping matrix for both old and new data in order
that knowledge previously learnt could be compatible with
new knowledge. Note that, although this loss looks like the
widely-used term which uses hash codes for classification,
the idea behind Eq.(6) is totally different.

**Hash Function Learning.** Currently, linear hash func-
tions occupy the mainstream position in online hashing do-
main while neural network based methods are extremely
scarce. The possible reasons may be that: 1) Online hashing
focuses more on efficiency as almost all publications report
the training time comparison results; 2) Training of neural
network based hash functions is much more time-consuming
than the training of linear functions. Following the vast ma-
majority, we design our hash function learning module using
the efficient and straightforward linear mapping as follows,

\[
\begin{align*}
\min_{\mathbf{B}^{(t)} \in \{-1,1\}^{r \times n(t)}, \mathbf{W}^{(t)}_m, \mathbf{X}^{(t)}_m} & \sum_m \left( \left\| \mathbf{B}^{(t)} \mathbf{W}^{(t)}_m \mathbf{X}^{(t)}_m - \mathbf{B}^{(t)} - \mathbf{W}^{(t)}_m \mathbf{X}^{(t)}_m \right\|^2_F + \| \mathbf{B}^{(t)} - \mathbf{W}^{(t)}_m \mathbf{X}^{(t)}_m \|^2_F + \delta \| \mathbf{W}^{(t)}_m \|^2_F \right) \\
& \text{s.t.,} \mathbf{V}^{(t)} = n(t) \mathbf{I}_r, \mathbf{V}^{(t)} \mathbf{1}^T = 0,
\end{align*}
\]

(7)

where \(M\) represents the total number of modalities, \(\mathbf{W}^{(t)}_m\)
is the projection of the \(m\)-th modality, and \(\delta\) is a hyperpa-
parameter balancing the regularization term. Although Eq.(7)
is simple, it has several advantages beyond simplicity and
efficiency: 1) This equation reduces the quantization loss be-
tween the learnt hash code and the real-valued mapping re-
results; 2) The catastrophic forgetting problem could be allevi-
ated through simultaneously considering old and new data.
Although above loss contains \(\mathbf{X}^{(t)}_m\), our method still meets
the third requirement as can be seen in **Algorithm 1** whose
inputs do not need features of old data chunks. This is be-
cause of the proposed novel online optimization which is
presented below.

**Overall Objective Function.** In summary, by combining
all mentioned modules together and creating reasonable mod-
ifications, the total loss function of OASIS is shown below,

\[
\begin{align*}
\min_{\mathbf{B}^{(t)} \in \{-1,1\}^{r \times n(t)}, \mathbf{V}^{(t)} \in \{-1,1\}^{r \times r(t)}, \mathbf{U}^{(t)}, \mathbf{W}^{(t)}_m, \mathbf{X}^{(t)}_m} & \| \mathbf{B}^{(t)} - \mathbf{V}^{(t)} \|_F^2 + \\
& \alpha \| r \mathbf{S}^{(t)} (\mathbf{B}^{(t)} - \mathbf{V}^{(t)}) + \mathbf{S}^{(t)} \mathbf{B}^{(t)} \mathbf{V}^{(t)} \|_F^2 + \| r \mathbf{S}^{(t)} - \mathbf{B}^{(t)} \mathbf{V}^{(t)} \|_F^2 + \beta \| \mathbf{G}^{(t)} - \mathbf{U}^{(t)} \mathbf{V}^{(t)} \|_F^2 \\
& + \| \mathbf{G}^{(t)} - \mathbf{U}^{(t)} \mathbf{V}^{(t)} \|_F^2 + \| \mathbf{B}^{(t)} + \mathbf{U}^{(t)} \mathbf{V}^{(t)} \|_F^2 + \gamma \sum_m \| \mathbf{B}^{(t)} \|_F^2 \\
& - \mathbf{W}^{(t)}_m \mathbf{X}^{(t)}_m \|_F^2 + \| \mathbf{B}^{(t)} - \mathbf{W}^{(t)}_m \mathbf{X}^{(t)}_m \|_F^2 + \delta \| \mathbf{W}^{(t)}_m \|_F^2 \quad \text{s.t.,} \mathbf{V}^{(t)} \mathbf{V}^{(t)} \mathbf{1}^T = n(t) \mathbf{I}_r, \mathbf{V}^{(t)} \mathbf{1}^T = 0,
\end{align*}
\]

(8)

where \(\mathbf{V}^{(t)}\) is the real-valued approximation of hash code
with several constraints, \(\alpha, \beta,\) and \(\gamma\) are tradeoff parameters.
By using \(\mathbf{V}^{(t)}\) to approximate \(\mathbf{B}^{(t)}\), the hard optimization
problem of \(\mathbf{B}^{(t)}\) can be simplified. Besides, the constraint of
\(\mathbf{V}^{(t)} \mathbf{V}^{(t)} \mathbf{1}^T = n(t) \mathbf{I}_r\) can make each bit represent as much
information as possible, and the \(\mathbf{V}^{(t)} \mathbf{1}^T = 0\) constraint can make the hash code more discriminative.

**Online Optimization**

We propose a novel discrete online optimization for Eq.(8)
to learn hash codes and function at the \(t\)-th round. Specifi-
cally, we optimize the variables one by one and repeat the
procedure several times until convergence. At round \(t\), the
optimization steps are presented in the following.

**Step 1: Update \(\mathbf{V}^{(t)}\).** When other variables remain un-
changed, the optimization problem of \(\mathbf{V}^{(t)}\) can be simplified to
the following form.

\[
\begin{align*}
\max_{\mathbf{V}^{(t)}} & \text{tr}(\mathbf{Z} \mathbf{V}^{(t)} \mathbf{V}^{(t)} \mathbf{1}^T), \\
\text{s.t.} & \mathbf{V}^{(t)} \mathbf{V}^{(t)} \mathbf{1}^T = n(t) \mathbf{I}_r, \mathbf{V}^{(t)} \mathbf{1}^T = 0,
\end{align*}
\]

(9)

where \(\mathbf{Z} = \alpha \mathbf{D}^{(t)}_1 \mathbf{G}^{(t)} + \mathbf{B}^{(t)} + \gamma \mathbf{U}^{(t)} \mathbf{G}^{(t)} \) and \(\mathbf{D}^{(t)}_1 = \mathbf{B}^{(t)} \mathbf{G}^{(t)} \mathbf{G}^{(t)} \mathbf{T} + \mathbf{D}^{(t-1)}_1\). Notably, \(\mathbf{D}_1\) is an intermediate vari-
able. By temporarily storing \(\mathbf{D}^{(t-1)}_1\), which is calculated at
the previous \((t-1)\)-th round, and directly using it at the cur-
rent \(t\)-th round, the calculation of \(\mathbf{D}^{(t)}_1\) can be very efficient.

Let us return to the optimization of \(\mathbf{V}^{(t)}\). The simplified
optimization problem in Eq.(9) is similar to the form in (Liu
et al. 2014) and can be optimized as follows. First, we per-
form the eigenvalue decomposition of \(\mathbf{Z} \mathbf{J} \mathbf{Z}^T\) and the
formula is as follows,

\[
\mathbf{Z} \mathbf{J} \mathbf{Z}^T = (\mathbf{O} \mathbf{O} ^T \mathbf{S}^2 \mathbf{I}_r \mathbf{O} \mathbf{O} ^T)\mathbf{N} \mathbf{N} ^T.
\]

(10)

**Step 2: Update \(\mathbf{U}^{(t)}\).** Fixing other variables and setting
the derivative of Eq.(8) w.r.t. \(\mathbf{U}^{(t)}\) to zero, the update for-
ma can be obtained as follows,

\[
\mathbf{U}^{(t)} = \mathbf{D}^{(t)}_2 (\mathbf{D}^{(t)}_3 + \theta \mathbf{I})^{-1},
\]

(12)

where \(\mathbf{D}^{(t)}_2 = \mathbf{G}^{(t)} \mathbf{V}^{(t)} \mathbf{V}^{(t)} \mathbf{T} + \mathbf{D}^{(t-1)}_2\) and \(\mathbf{D}^{(t)}_3 = \mathbf{V}^{(t)} \mathbf{V}^{(t)} \mathbf{T} + \mathbf{D}^{(t-1)}_3\). Notably, \(\mathbf{D}_2\) and \(\mathbf{D}_3\) are intermediate variables.
By temporarily storing them at last round and directly using them at current round, the optimization can be extremely ef-

cient.

**Step 3: Update \(\mathbf{B}^{(t)}\).** When other variables are fixed, the
generation of hash codes can be simplified as,

\[
\max_{\mathbf{B}^{(t)}} \text{tr}(\mathbf{Q} \mathbf{B}^{(t)} \mathbf{V}^{(t)} \mathbf{V}^{(t)} \mathbf{1}^T), \quad \text{s.t.} \mathbf{B}^{(t)} \in \{-1,1\}^{r \times n(t)},
\]

(13)

where \(\mathbf{Q} = \alpha r \mathbf{D}^{(t)}_1 \mathbf{G}^{(t)} + \mathbf{V}^{(t)} + \gamma \sum_m \mathbf{W}^{(t)}_m \mathbf{X}^{(t)}_m\) and \(\mathbf{D}^{(t)}_4 = \mathbf{V}^{(t)} \mathbf{G}^{(t)} \mathbf{G}^{(t)} \mathbf{T} + \mathbf{D}^{(t-1)}_4\). Notably, similar with \(\mathbf{D}_1, \mathbf{D}_2,\) and \(\mathbf{D}_3, \mathbf{D}_4\) is called intermediate variable and is helpful for efficient optimization.
Because of $\text{tr}(Q \mathbf{B}(t)T) = \sum_{i,j} Q_{ij} \mathbf{B}_{ij}^T$ and $\mathbf{B}(t) \in \{-1, 1\}^{r \times n}$, we can let $\mathbf{B}_{ij} = 1$ when $Q_{ij}$ is positive and $\mathbf{B}_{ij} = -1$ when $Q_{ij}$ is negative to maximize the simplified optimization function, where $i$ and $j$ represent the $i$-th row and $j$-th column of the matrix. Therefore, we can get the following formula,

$$\mathbf{B}(t) = \text{sign}(Q).$$  

**Step 4: Update $W_m(t)$**. Similar to the update of $U(t)$, the update formula of $W_m(t)$ can be given as follows,

$$W_m(t) = D_5(t) (D_6(t) + \delta I)^{-1},$$  

where $D_5(t) = \mathbf{B}(t) \tilde{X}_m(t)T + D_5(t-1)$ and $D_6(t) = \tilde{X}_m(t) \tilde{X}_m(t)T + D_6(t-1)$. Here, we could temporarily store $D_5(t-1)$ and $D_6(t-1)$ at the last round and use them at $t$-th round to get $D_5(t)$ and $D_6(t)$ directly. In light of these intermediate variables, efficiency of the optimization can be ensured.

**Overall Algorithm.** For better understanding, we summarize the proposed online optimization in Algorithm 1. It’s worth noting that although we can find $\tilde{X}(t)$ in the objective function, the old feature is not needed in the real optimization. Our model satisfies all four requirements raised in the Problem Definition Section.

Besides, the sizes of $D_i(t-1)$ $(i = \{1, \cdots, 6\})$ are $r \times f$, $f \times r$, $r \times r$, $r \times f$, $r \times d_m$, and $d_m \times d_m$. Although these matrices contain rich information from old data, they are irrelevant with $N(t)$.

**Complexity Analysis.** The complexity of updating $\tilde{V}(t)$ is $O(10r + 3r^2 + 3rf + f)n(t) + 2rf + r^3$, the complexity of getting $U(t)$ is $O((2r + fr + f + r^2)n(t) + 2f^2 + r^3 + fr^2)$, the complexity of learning $\mathbf{B}(t)$ is $O((5r + 2fr + f + \sum_{m=1}^r rd_m)n(t) + rf)$, and the complexity of generating $W_m(t)$ is $O(M(d_m^2 + rd_m + d_m)n(t) + M(2d_m^2 + d_m^3 + rd_m^2 + rd_m))$. It is worth noting that variables $D_i(t-1)$ $(i = \{1, \cdots, 6\})$ can be obtained at the previous round and used directly at this round. Thus, there is no need to consider the time complexity of calculating them.

From the above analysis, we can find that the online optimization is linearly related to the size of new data chunk $n(t)$ and is irrelevant to the old database $N(t)$. To conclude, our proposed optimization is efficient and scalable to large-scale online retrieval applications.

**Retrieval Precedure.** When query sample comes, we first learn its hash code $b_{query} = \text{sign}(\sum_{m=1}^M W_m(t)x_{query-m})$, where $W_m(t)$ is the up to date hashing projection and $x_{query-m}$ represents query’s feature of the $m$-th modality.

Then, we can calculate the Hamming distance of hash codes between query and database to measure the similarity. Those instances, which are considered to be similar, can be returned as retrieval results.

---

**Algorithm 1: The optimization of OASIS at the $t$-th round.**

**Input:** $\tilde{X}_m(t)$ $(m = 1, 2, \cdots, M)$, $\tilde{V}(t)$, $\tilde{X}(t)$, and $D_i(t-1)$ $(i = \{1, \cdots, 6\})$.

**Output:** $D_i(t)$ $(i = \{1, \cdots, 6\})$, $\mathbf{B}(t)$, and hash function.

**Main Algorithm:**

1. Randomly initialize $U(t)$, $\mathbf{B}(t)$, and $W_m(t)$.
2. **while** not converged or not reach the max iterations **do**
3. Update $\tilde{V}(t)$ with Eq.(11); save $D_i(t)$.
4. Update $U(t)$ with Eq.(12); save $D_2(t)$ and $D_3(t)$.
5. Update $\mathbf{B}(t)$ with Eq.(14); save $D_4(t)$.
6. Update $W_m(t)$ with Eq.(15); save $D_5(t)$ and $D_6(t)$.
7. **end while**

---

**Experiment**

**Experimental Settings.** In this paper, we chose two widely-used datasets, i.e., MIRFlickr (Huiskes and Lew 2008) and NUS-WIDE (Chua et al. 2009). MIRFlickr has 25,000 instances and 24 categories in total. Following (Jiang and Li 2019), we removed instances with rare tags that appear less than 20 times and finally had 20,015 instances left. Then, we passed images through pre-trained VGG network to obtain 4096-d deep features and expressed texts as 1386-d BOW features. NUS-WIDE has 269,648 instances and 81 categories. Following the setting of (Lu et al. 2019a), only the top 21 most common categories are used and 195,834 instances are finally left. Similar with MIRFlickr, we fed images into the VGG network to obtain 4096-d deep features and represented the text modality as 5018-d BOW features. Features used in this paper can be obtained from https://github.com/DravenALG/OASIS.

**Evaluation Protocols.** Mean Average Precision (MAP) is adopted as the evaluation criterion and larger value indicates better performance. As stated in above sections, during retrieval phase, we calculated the Hamming distance of hash codes between queries and database to measure the similarity. As both MIRFlickr and NUS-WIDE are multi-label datasets, we considered two samples to be similar if they share at least one common label.

Furthermore, we have designed three types of online experimental settings. The first online setting assumes that all categories are known before training. In other words, for $t$ from 2 to the maximum, $c_n(t)$ (the number of new classes which first appear at the $t$-th round) is always equal to 0. On the contrary, the second and third settings simulate the scenario where new (unknown) categories continually appear along with new data chunks ($c_n(t)>0$). For the second online setting, new data carries some old categories which first appear before current round and some new categories. For the third setting, all categories of new data are new (unknown).

1. **The first online setting:** For both datasets, we randomly selected 2,000 samples to form the test set and left the remaining samples as training set. That is, the size of training set of MIRFlickr is 18,015 and NUS-WIDE has
193,834 training points. The first 9 chunks of MIRFlickr have the same size of 2,000 while the 10th chunk includes the remaining 15 samples. For NUS-WIDE, size of the first 19 chunks is 10,000 while the remaining 3,834 samples constitute the 20th chunk. All data chunks are constructed by randomly selecting samples. (2) The second online setting: MIRFlickr is divided into 10 rounds and NUS-WIDE is divided into 20 rounds. Specifically, we assigned some labels for each round, ensuring that labels at the new round include the labels appear at previous rounds and at least one new category appear. Then, we selected eligible data according to the set of labels at each round. It is worth noting that the data at the new round would exclude the data at previous rounds to avoid duplication. Subsequently, the training data and test data of each round are randomly selected from the data of each round at a ratio of 9 : 1. (3) The third online setting: We stipulated that MIRFlickr has 4 rounds and NUS-WIDE has 8 rounds. We allocated some labels for each round. Different from the second setting, we set labels at the new round as totally different from the labels at the last rounds. Then, we selected the data based on the labels at each round. The training data and test data of each round are randomly selected from the data of each round at a ratio of 9 : 1. In all settings, we use all seen training data as the database and test data selected at each round as queries. More details can be found in our code.

Baselines. Three state-of-the-art methods, including FOMH (Lu et al. 2019a), OMH-DQ (Lu et al. 2019b), and SAPMH (Zheng et al. 2020), are adopted as baselines. To the best of our knowledge, FOMH is the latest and best online multi-modal hashing model. Both OMH-DQ and SAPMH are the most advanced batch-based multi-modal hashing and their hash functions and hash codes are retrained on all accumulated data (i.e., using $\mathbf{X}(t)$ and $\mathbf{X}(t)$) at each round. For all baselines, codes are publicly available and we set their parameters following provided instructions.

As stated in previous section, no existing multi-modal hashing methods try to handle the class incremental problem with streaming data, including our baselines. As the comparisons on the second and third settings still need to be conducted, we had to release the fourth requirement of the problem setting for baselines. That is, we assumed that baselines had already known all categories before training. It is worth noting that our OASIS always meets the four requirements formalized in the Problem Definition Section.

Implementation Details For OASIS, the parameter settings are: $\alpha = 10$, $\beta = 1$, $\gamma = 1$, $\theta = 1$, and $\delta = 1$. We set iteration number as 3. Our experiments are conducted on a Linux workstation with Intel XEON E5-2650 2.20GHz CPU, 128GB RAM.

Results and Comprehensive Analysis

MAP Results. The MAP values under all experimental settings on two datasets are shown in Table 1. We trained
The results of the convergence analysis show that the cost of a little more time is worthwhile to get much better accuracy. The excellent performance of our model, OASIS, is more practical because its parameters can be readily tuned and selected. We finally set \( \alpha = 10 \), \( \beta = 1 \), \( \gamma = 1 \), \( \theta = 1 \), and \( \delta = 1 \).

### Training Time Analysis

The training time consumptions of all methods with 16-bit hash code on MIRFlickr are shown in Table 2. Here, the first setting is adopted. Since there are only 15 training samples at the 10-th round, we only listed the training time of the first 9 rounds. As OMH-DQ and SAPMH are batch-based models, their hash functions and hash codes are retrained on all accumulated data at each round. Thus, we can find that their training time continuously increases. This phenomenon reflects that batch-based methods fail to efficiently handle streaming data and may be impractical when used in large-scale applications. As FOMH and OASIS are online models, we can find that their training time does not increase with data coming and is only linearly related to the size of new data chunk. Although the time consumption of FOMH is slightly lower, considering the excellent performance of our model, OASIS is more practical and it is worthwhile to get much better accuracy at the cost of a little more time.

### Convergence Analysis

The results of the convergence experiments under the first setting are shown in Figure 4. In this figure, we reported the MAP results versus number of iterations at the first five rounds on MIRFlickr in the case of 16-bit code length. It can be seen that our model can quickly achieve satisfactory performance after two iterations.

### Parameters Sensitivity Analysis

The experimental results versus different parameter values in the case of 16-bit code length are shown in Figure 5. We only conducted experiments on \( \alpha \), \( \beta \), and \( \gamma \) because \( \theta \) and \( \delta \) balance the regularization terms and can be empirically set. In addition, when performing sensitivity experiments on one parameter, we fixed other parameters. It can be seen from the experimental results that when \( \alpha \) is small, it has a massive impact on the experimental results so that the semantic pairwise similarity plays a vital role in our model. In addition, we can see that the model is not sensitive to the values of \( \beta \) and \( \gamma \). Although OASIS has five parameters in all, two of them can be empirically set and two of them are not sensitive. In other words, our method could be easily applied in practical scenarios because its parameters can be readily tuned and selected. We finally set \( \alpha = 10 \), \( \beta = 1 \), \( \gamma = 1 \), \( \theta = 1 \), and \( \delta = 1 \).

### Conclusion and Future Work

In this paper, we propose a novel hashing method for multimodal online retrieval, termed Online enHanced SemantIc hashing (OASIS). OASIS invents a semantic-enhanced representation to describe instances and designs a new objective function to fully learn from the rich semantic information. Besides, with the help of the semantic-enhanced representation, OASIS can handle new classes coming with streaming data well. Sufficient experiments have demonstrated that the performance of our model can surpass the start-of-the-art models. In addition, this paper tries to give explicit task definition and hopes to benefit the hashing community.

Class incremental problem in online multi-modal hashing domain is still challenging. In the future, we will further investigate this problem from the following aspects. 1) Trying to better preserve the knowledge from old classes, especially those that don’t appear for a long time. 2) Trying to transfer knowledge from old classes to new ones so as to better solve the class incremental problem.

### Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 61872428, 62172256, in part by Shandong Provincial Key Research and Development Program under Grant 2019JZSY010127, in part by Natural Science Foundation of Shandong Province under Grant ZR2019ZD06, ZR2020QF036, and in part by the Major Program of the National Natural Science Foundation of China under Grant 61991411.
References


Qi, M.; Wang, Y.; and Li, A. 2017. Online Cross-Modal Scene Retrieval by Binary Representation and Semantic


