CLIP4Hashing: Unsupervised Deep Hashing for Cross-Modal Video-Text Retrieval

Yaoxin Zhuo*
Arizona State University
yzhuo6@asu.edu

Yikang Li
OPPO US Research Center
yikang.li1@oppo.com

Jenhao Hsiao
OPPO US Research Center
mark@oppo.com

Chiuman Ho
OPPO US Research Center
chiuman@oppo.com

Baoxin Li
Arizona State University
baoxin.li@asu.edu

ABSTRACT

With the ever-increasing multimedia data on the Web, cross-modal video-text retrieval has received a lot of attention in recent years. Deep cross-modal hashing approaches utilize the Hamming space for achieving fast retrieval. However, most existing algorithms have difficulties in seeking or constructing a well-defined joint semantic space. In this paper, an unsupervised deep cross-modal video-text hashing approach (**CLIP4Hashing**) is proposed, which mitigates the difficulties in bridging between different modalities in the Hamming space through building a single hashing net by employing the pre-trained CLIP model [24]. The approach is enhanced by two novel techniques, the dynamic weighting strategy and the design of the min-max hashing layer, which are found to be the main sources of the performance gain. Compared with conventional deep cross-modal hashing algorithms, **CLIP4Hashing** does not require data-specific hyper-parameters. With evaluation using three challenging video-text benchmark datasets, we demonstrate that **CLIP4Hashing** is able to significantly outperform existing state-of-the-art hashing algorithms. Additionally, with larger bit sizes (*e.g.*, 2048 bits), **CLIP4Hashing** can even deliver competitive performance compared with the results based on non-hashing features.

CCS CONCEPTS

• **Information systems** → **Multimedia and multimodal retrieval**; **Video search**.

KEYWORDS

Cross-modal retrieval, Video-text retrieval, Hashing, Deep learning

ACM Reference Format:
Yaoxin Zhuo, Yikang Li, Jenhao Hsiao, Chiuman Ho, and Baoxin Li. 2022. CLIP4Hashing: Unsupervised Deep Hashing for Cross-Modal Video-Text Retrieval. In *Proceedings of the 2022 International Conference on Multimedia Retrieval (ICMR '22), June 27–30, 2022, Newark, NJ, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3512527.3531381

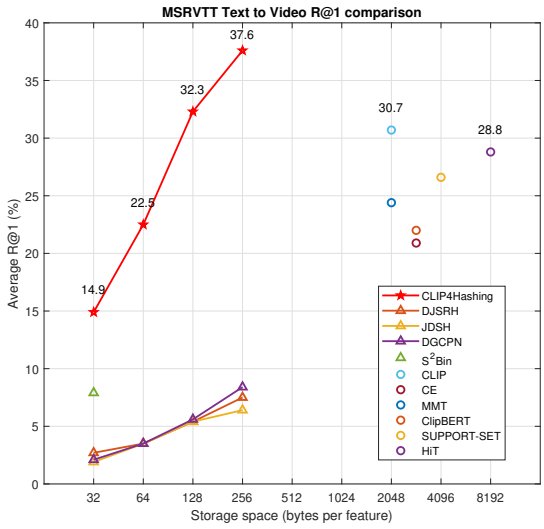

Figure 1: MSRVTT text to video R@1 comparison (Storage space vs. Average R@1). The triangles represent hashing methods (trained with the same CLIP feature). The circles represent non-hashing methods. We converted bit size and float number to bytes. The CLIP4Hashing significantly outperforms other hashing methods, and has competitive performance compared with the non-hashing methods.

1 INTRODUCTION

With explosively increasing amount of data on social media platforms like Twitter and TikTok, fast and accurate cross-modal retrieval is gaining much attention. Compared with the traditional cross-modal retrieval algorithms that work in the continuous feature space, hashing-based algorithms can retrieve data faster and more efficiently. Conventional hashing-related visual tasks usually focused only on the visual modality, *i.e.*, the query and retrieval data are both images or videos. In addition, single-modality hashing methods are concerned more with seeking discriminative feature representations than finding the representations of relationships among the data.

*Work was partially done while doing an internship at OPPO US Research Center

ICMR '22, June 27–30, 2022, Newark, NJ, USA

ACM ISBN 978-1-4503-9238-9/22/06…$15.00
https://doi.org/10.1145/3512527.3531381

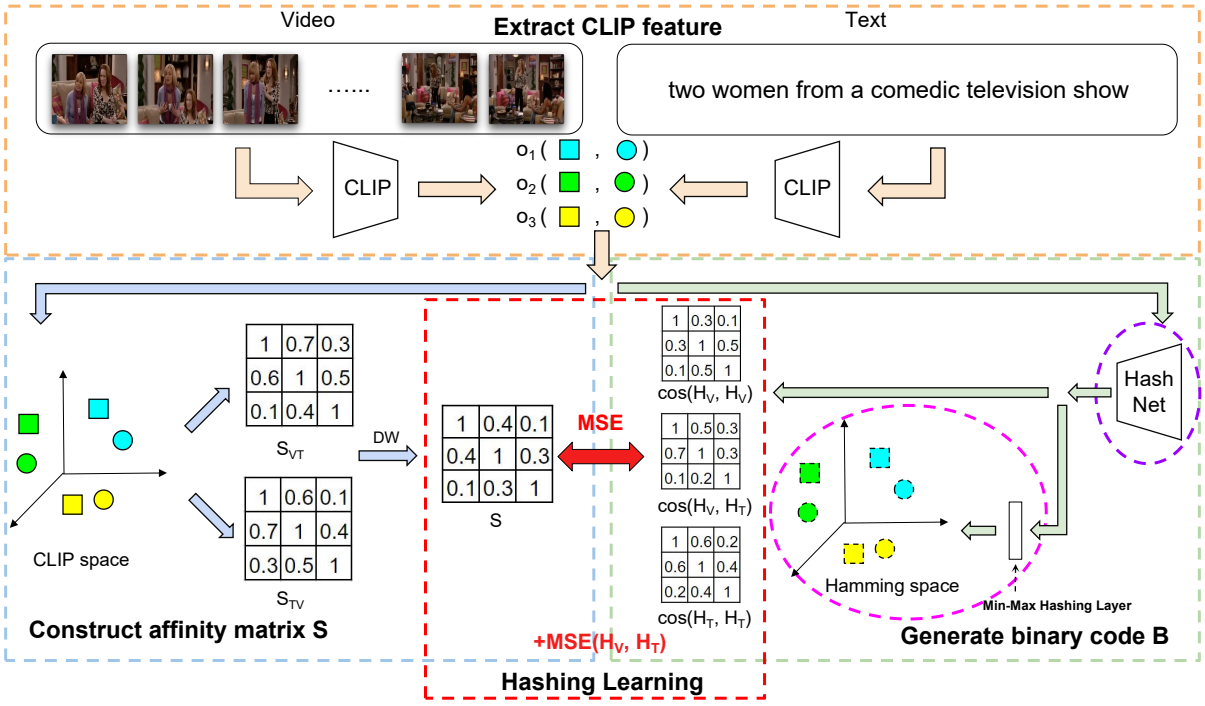

Figure 2: The pipeline of the proposed CLIP4Hashing (best viewed in color).

Some researchers [3, 18, 25] attempted to apply the single-modality methods to multi-modality tasks directly. The performance is often sub-optimal since cross-modality information is not explicitly considered. Besides, the common method of supervised hashing would often lead to a more classification-like task, which is not well supported in the Hamming space.

Recent efforts [8, 11, 13, 31] have looked into hashing for cross-modal retrieval, which is more practical but challenging, as it is necessary to model the semantic relationships between different modalities. On this task, mainstream methods often utilize the joint semantic affinity matrices to guide the learning of the Hamming space [17, 26, 32, 40], where a key step is the creation of the similarity matrices. Constructing a good affinity matrix that properly captures cross-modal correlation for high-performance retrieval is a non-trivial task, and most existing methods rely on ad hoc procedures that require complicated hyper-parameter tuning (such as in [17, 32, 37]). Accordingly, the final retrieval performance is often limited by these ad hoc procedures.

To tackle the issues mentioned above, we propose a novel unsupervised deep cross-modal hashing approach named **CLIP4Hashing**. It utilizes the pre-trained CLIP model [24] to construct the affinity matrix and build one single hashing model, which helps to mitigate the difficulties in bridging between different modalities in the Hamming space. A dynamic weighting strategy is introduced for conditioning the affinity matrix, leading to better discriminative learning. Further, a min-max hashing layer is designed for improving binary code generation. Both modules do not require dataset-specific hyper-parameters and thus result in a more efficient yet flexible hashing scheme. **CLIP4Hashing** is learned in an unsupervised fashion, without requiring category or attribute information of the videos. As Figure 1 shows, **CLIP4Hashing** significantly outperforms existing state-of-the-art hashing methods.

The key contributions of this work include:

- **CLIP4Hashing** is the first approach employing the CLIP features for video-text hashing for retrieval. This allows us to introduce a hashing model that consists of only a single hashing network and thus naturally mitigates the difficulty in bridging different modalities.
- We propose dynamic weighting for conditioning the affinity matrix for better learning and a parameter-free min-max hashing layer for generating binary codes, both boosting the performance of the approach.
- Our approach significantly outperforms existing hashing methods on three video-text benchmarks, establishing new state-of-the-art results. Furthermore, it even has competitive performance compared with non-hashing methods when the bit size is large.

2 RELATED WORK

Cross-modal hashing methods often focus on projecting data from different modalities into a common Hamming space to learn binary codes. Moreover, the paired instances should be close to each other in the learned Hamming space compared with the unpaired ones.

Most of the current research focuses on image-text retrieval tasks where the tags or labels are often utilized in the text modality.

Table 1: MSRVTT results of different methods. Notes: * indicates the methods are non-hashing methods and all hashing methods are trained with same features from the pre-trained CLIP model.

Method name	bit size	Text-to-video retrieval				Video-to-text retrieval			
		R@1 ↑	R@5 ↑	R@10 ↑	MdR ↓	R@1↑	R@5 ↑	R@10↑	MdR ↓
CE (2019) [19]	*	20.9	48.8	62.4	6	20.6	50.3	64.0	5.3
MMT (2020) [7]	*	24.4	56.0	67.8	4	24.6	54.0	67.1	4
ClipBERT 8×2 *(2021)* [12]	*	22.0	46.8	59.9	6	-	-	-	-
SUPPORT-SET (2021) [22]	*	26.6	55.1	67.5	3	27.4	56.3	67.7	3
HiT (2021) [16]	*	28.8	60.3	72.3	3	27.7	59.2	72.0	3
Pre-trained CLIP model (2021) [24]	*	30.7	53.1	62.6	5	26.0	51.7	62.4	5
S^2Bin (with attributes) [23]	256	7.9	22.5	32.3	**31**	-	-	-	-
DJSRH (2019) [32]	256	2.7	9.5	14.8	96	2.6	9.0	14.2	81
	512	3.5	11.3	17.1	98	3.3	10.5	16.7	98
	1024	5.4	16.2	23.0	57	5.4	16.8	23.9	57
	2048	7.5	20.0	28.2	46	7.0	19.8	28.9	46
JDSH (2020) [17]	256	1.9	8.7	17.1	49	2.1	8.0	14.8	54
	512	3.5	13.3	23.1	37	3.2	12.1	20.7	42
	1024	5.4	16.0	26.6	30	4.1	14.6	23.8	33
	2048	6.4	19.6	30.4	25	6.2	17.8	26.6	31
DGCPN (2021) [37]	256	2.1	8.5	15.9	71	2.1	8.2	14.1	79
	512	3.5	11.7	18.4	100	4.1	13.1	20.0	78
	1024	5.6	19.3	29.1	38	6.1	18.2	26.9	41
	2048	8.4	23.2	34.7	27	7.6	22.9	32.5	28
CLIP4Hashing	256	**14.9**	**27.4**	**33.5**	34	**14.7**	**27.4**	**33.7**	**33**
	512	**22.5**	**39.7**	**47.8**	**13**	**22.7**	**37.4**	**47.3**	**13**
	1024	**32.3**	**52.1**	**61.3**	**5**	**32.9**	**51.5**	**59.7**	**5**
	2048	**37.6**	**56.9**	**66.3**	**3**	**37.0**	**58.3**	**66.0**	**3**

Deng *et al.* [4] proposed the triplet-based hashing model that utilizes the triplet labels to capture the semantic correlations between image and text. Su *et al.* [32] proposed DJSRH, which constructs a joint-semantics similarity matrix to learn the Hamming space in order to reconstruct the joint-semantic relations. The authors of [17] proposed a strategy termed JDSH for cross-modal retrieval, which unfortunately needs global hyper-parameters based on the statistics of the dataset. Scalable Discrete Matrix Factorization Hashing (SCRATCH) [3] utilizes matrix factorization to learn the latent binary representation of labels and features. UCH [13] coupled the GAN to build two cycled networks for learning the hashing. The outer network learns common representations and the inner network generates the binary codes. The work in [6] proposed a method named MSFH that learns the hashing by preserving the topology of the original data with the help of matrix decomposition. Unsupervised Knowledge Distillation (UKD) was introduced in [9]. It uses the output from an unsupervised teacher-student optimization method to guide supervised hashing learning. Xie *et al.* [35] developed a deep hashing approach (CPAH) to maintain the semantic relationship between different modalities. It consists of a refined module and a multi-task adversarial learning module. The method DGCPN [37] is a deep graph-neighbor coherence preserving network, which exploits three types of data similarities. However, it requires the whole global information of dataset to build the affinity matrix based on the K-nearest neighbors. All these recent cross-modal hashing works are designed for image-text retrieval and cannot be readily adopted into the video-text domain.

Some research works focusing on the video-text domain have been published in recent years. Qi *et al.* designed a framework named S^2Bin [23], considering the spatial-temporal context of the video data and the semantic relationships among data in different modalities. Nevertheless, it requires video action proposals and the attributes for further guidance. Jin *et al.* [11] proposed a deep semantic multi-modal hashing network, which contains two sets of modality-specific hashing functions to preserve the semantic information of both inter-modalities and intra-modalities. However, it is trained in a supervised fashion. Supervised learning for hashing has the potential disadvantage of leading to a classification-like task [5, 15] where significant collision of hashing codes would occur. **CLIP4Hashing** avoids this issue by not requiring any category or label information.

3 APPROACH

We first introduce the notations. The m variable denotes the batch size and $O = \{o_1, o_2, ..., o_m\}$ is the m training instances in each batch. $\mathbf{V}$ and $\mathbf{T}$ represent the videos and texts respectively. The features of videos and texts are extracted by the CLIP model. They are denoted as $\mathbf{F}_V = [f_v^1, f_v^2, ..., f_v^m] \in \mathbb{R}^{m \times d}$ and $\mathbf{F}_T = [f_t^1, f_t^2, ..., f_t^m] \in \mathbb{R}^{m \times d}$,

Table 2: DeDiMo results of different methods. Notes: * indicates the methods are non-hashing methods and all hashing methods are trained with same features from the pre-trained CLIP model.

Method name	bit size	Text-to-video retrieval				Video-to-text retrieval			
		R@1 ↑	R@5 ↑	R@10 ↑	MdR ↓	R@1↑	R@5 ↑	R@10↑	MdR ↓
FSE (2018) [39]	*	13.9	36.0	-	11	-	-	-	-
CE (2019) [19]	*	16.1	41.1	-	8.3	15.6	40.9	-	8.2
ClipBERT 8 × 2 (2021) [12]	*	20.4	48.0	60.8	6	-	-	-	-
Pre-trained CLIP model (2021) [24]	*	29.4	53.3	64.7	4	20.1	47.7	58.1	6
DJSRH (2019) [32]	256	4.3	12.2	19.2	46	3.2	10.6	17.1	67
	512	7.4	18.7	27.2	44	6.4	17.9	26.8	40
	1024	9.1	22.2	31.7	30	8.2	21.7	30.1	31.5
	2048	12.8	29.4	40.8	19	9.9	25.5	34.7	24
JDSH (2020) [17]	256	1.9	10.1	17.9	37	2.1	9.4	17.2	44
	512	4.8	16.2	25.7	30	3.8	15.1	25.5	34.5
	1024	6.7	21.4	32.0	23	5.7	19.8	29.4	28
	2048	7.4	22.0	33.1	22	7.1	20.8	31.6	26
DGCPN (2021) [37]	256	2.5	10.0	16.9	97	3.0	10.5	17.2	89
	512	4.4	13.2	21.7	74	5.2	14.3	20.7	85
	1024	7.0	19.4	29.4	34	7.7	20.9	29.8	33.5
	2048	8.9	24.2	33.5	28	8.7	23.9	35.8	26
CLIP4Hashing	256	**14.2**	**26.4**	**33.8**	**37**	**14.0**	**27.4**	**35.5**	**38**
	512	**23.6**	**39.6**	**49.0**	**11**	**23.6**	**39.8**	**48.0**	**12**
	1024	**32.7**	**51.6**	**60.6**	**5**	**30.3**	**50.4**	**56.2**	**5**
	2048	**37.1**	**58.8**	**67.4**	**3**	**36.8**	**57.1**	**67.7**	**3**

where d (512 in our paper) is the dimension of the extracted feature from the pre-trained CLIP model. A HashNet, a multi-layer perceptron (MLP), is employed to produce the latent feature. The outputs of the HashNet are denoted as $\mathbf{H}_V = [h_v^1, h_v^2, ..., h_v^m] \in \mathbb{R}^{m \times Z}$ and $\mathbf{H}_T = [h_t^1, h_t^2, ..., h_t^m] \in \mathbb{R}^{m \times Z}$, corresponding to the input $\mathbf{F}_V$ and $\mathbf{F}_T$, where the Z is the target encoding bit size. The binary codes can be obtained as $\mathbf{B}_V = [b_v^1, b_v^2, ..., b_v^m] \in \{-1, +1\}^{m \times Z}$ and $\mathbf{B}_T = [b_t^1, b_t^2, ..., b_t^m] \in \{-1, +1\}^{m \times Z}$ from the $\mathbf{H}_V$ and $\mathbf{H}_T$ respectively by our designed min-max hashing layer, which will be introduced in section 3.4.

The overall framework of **CLIP4Hashing** is shown in Figure 2. As illustrated in the figure, the pre-trained CLIP model is first used to extract the video and text features. One branch of processing uses the extracted features to construct the affinity matrix $\mathbf{S}$. Another branch feeds the extracted features to the HashNet to generate latent features that are then associated with the affinity matrix in loss computation, which in turn guides HashNet learning. The min-max hashing layer generates the binary codes from the outputs of the HashNet. Details of these modules are elaborated in the following subsections.

3.1 Hashing Using a Single Unified Model

Most prior efforts [17, 32, 37] adopted a dual-encoder architecture for processing visual and textual inputs separately. Recent studies [28, 31, 33] have demonstrated that retrieval in a single domain has advantages and improved performance. Some cross-modal retrieval works [10, 20, 24, 29] focus on building one common semantic space for different modalities. Hence, in this work, we seek a well-defined semantic space onto which both visual and textual features can be projected and unified in the same domain. We choose the CLIP model [24] as the building block, which provides a semantic space constructed by training with over 400 million image-text pairs. Under this model, the extracted video and text features can be viewed as semantic vectors in the same domain, and thus we treat the cross-modal hashing problem similar to single-modality hashing, relying on the semantic relations of the feature vectors to capture the cross-modal information.

Similar to previous work [26, 32, 34, 40], we use the pre-trained features as inputs to our model. Following [32, 34], a three-layer MLP is used as the HashNet to obtain $\mathbf{H}_V$ and $\mathbf{H}_T$, which are used in conjunction with the affinity matrix in generating losses for guiding the training of the HashNet (see section 3.3). $\mathbf{H}_V$ and $\mathbf{H}_T$ are fed to the min-max hashing layer (to be discussed in section 3.4) to produce the binary codes $\mathbf{B}_V$ and $\mathbf{B}_T$. This single hashing module is termed as **SHM** in further discussion, which is shown as the purple dashed circle in Figure 2.

To illustrate the benefits of utilizing a single hashing network over the dual-network structure, the performance of these two structures on the MSRVTT dataset with 1024 bit size are shown in the first two rows of Table 4. The activation function of the **SHM** module is replaced by **Tanh** for a fair comparison with the dual network structure like [31]. More details can be found in section 4.5.

Table 3: MSVD results of different methods. Notes: * indicates the methods are non-hashing methods and all hashing methods are trained with same features from the pre-trained CLIP model.

Method name	bit size	Text-to-video retrieval				Video-to-text retrieval			
		R@1 ↑	R@5 ↑	R@10 ↑	MdR ↓	R@1↑	R@5 ↑	R@10↑	MdR ↓
CE (2019) [19]	*	19.8	49.0	63.8	6	-	-	-	-
SUPPORT-SET (2021) [22]	*	23.0	52.8	65.8	5	27.3	50.7	60.8	5
Pre-trained CLIP model (2021) [24]	*	43.9	72.1	81.3	2	43.5	75.4	85	2
DJSRH (2019) [32]	256	5.3	15.2	22.8	47	5.3	14.5	21.2	49
	512	9.2	23.9	34.7	22	6.9	20.2	28.7	29
	1024	12.6	31.3	44.4	15	9.8	24.6	34.3	25
	2048	18.2	40.3	52.4	9	9.6	28.8	41.5	16.5
JDSH (2020) [17]	256	6.5	20.3	31.4	25	6.2	21.5	32.7	25
	512	8.9	25.4	37.7	18	9.2	27.0	35.5	20
	1024	11.4	32.0	43.9	15	10.4	28.6	39.6	18
	2048	11.7	31.5	43.5	15	9.4	30.0	39.9	17
DGCPN (2021) [37]	256	8.8	25.5	35.5	22	9.2	24.8	35.8	21
	512	13.2	34.0	46	13	14.5	36.6	48.7	11
	1024	16.7	39.6	51	10	17.5	41.5	53.5	8
	2048	19.2	44.1	54.9	8	20.7	43.7	57.3	7
CLIP4Hashing	256	**34.4**	**52.4**	**61.9**	**5**	**35.2**	**53.9**	**62.9**	**5**
	512	**45.5**	**66.2**	**77.0**	**2**	**49.5**	**73.4**	**82.0**	**2**
	1024	**52.4**	**73.5**	**82.9**	**1**	**52.7**	**77.2**	**83.9**	**1**
	2048	**54.5**	**78.2**	**87.3**	**1**	**54.5**	**78.8**	**87.1**	**1**

3.2 Constructing Cross-Modal Affinity Matrix with Dynamic Weighting

For each instance pair, the visual and textual features ($\mathbf{F_V}$ and $\mathbf{F_T}$) are extracted by CLIP dual-encoders. Besides, for each video, mean-pooling frame fusion is used to combine per-frame features into a single feature vector f_v.

To construct the cross-modal affinity matrix, the cross-modal cosine similarity matrices are first calculated as:

$$\mathrm{S}_{VT} = d_{cos}(\mathbf{F}_V, \mathbf{F}_T) = \hat{\mathbf{F}}_V \hat{\mathbf{F}}_T^{\top} \tag{1}$$

$$\mathrm{S}_{TV} = d_{cos}(\mathbf{F}_T, \mathbf{F}_V) = \hat{\mathbf{F}}_T \hat{\mathbf{F}}_V^{\top} \tag{2}$$

where $\mathrm{S}_{VT} \in [0,1]^{m \times m}$ and $\mathrm{S}_{TV} \in [0,1]^{m \times m}$, and d_{cos} denotes the cosine similarity. $\hat{\mathbf{F}}$ represents the normalized features from the original $\mathbf{F}$.

Next, all diagonal elements of S_{VT} and S_{TV} are set to 1 because the paired video and text should be closest to each other. In the pre-defined semantic space, even though the diagonal values are more significant than other values in the original similarity matrices, they hardly reach 1 since the features from different modalities in the common semantic space are generally not the same.

Lastly, we use the average of the S_{VT} and S_{TV} to form the cross-modal affinity matrix:

$$\mathrm{S}_c = \frac{(\mathrm{S}_{VT} + \mathrm{S}_{TV})}{2} \tag{3}$$

This also makes S_c symmetric, similar to the case in the single modality. For simplicity, we refer to the model based on this affinity matrix S_c as **CLIPbase**.

For the above affinity matrix S_c, one observation is that the distances between unpaired instances are not well separated. One reason is that the features are learned by contrastive learning with enormous amount of data, which causes the learned distances of unpaired instances to stay in a small range. Previous hashing methods [17, 32, 37] only paid attention to the diagonal values in the affinity matrix, which means they are concerned more with the paired video-text relationships. But the unpaired video-text relations should not be ignored during hashing learning. Therefore, we design a remapping procedure to make the affinity matrix entries more distinctive by strengthening the off-diagonal values in the affinity matrix (which reflect relative relations between unpaired video-texts). This is inspired by the "histogram equalization" idea that effectively improves contrast by flattening the distribution of the entries. Specifically, starting with Eq. 3, a few more steps are taken to acquire the new weighted affinity matrix $\mathbf{S}$.

Firstly, we acquire the mean, min, and max values of S_c (denoted as $\hat{s}_{\text{mean}}$, $\hat{s}_{\text{min}}$ and $\hat{s}_{\text{max}}$) in each batch. Then each element $\hat{s}_{i,j}$ is determined to have either a "dissimilar pair" or a "similar pair" status, by comparing against $\hat{s}_{\text{mean}}$. The affinity matrix is weighted element-wise adaptively according to the status of the elements. The re-weighted $\hat{s}_{i,j}$ is represented as $s_{i,j}$ as following:

$$s_{i,j} = \begin{cases} W^{-}\hat{s}_{i,j}, \text{if} & \hat{s}_{i,j} \leq \hat{s}_{\text{mean}} \\ W^{+}\hat{s}_{i,j}, \text{if} & \hat{s}_{i,j} > \hat{s}_{\text{mean}} \end{cases} \tag{4}$$

where weights W^{-} and W^{+} are computed by:

$$W^{-} = \exp\left(-\frac{1}{2} \times \frac{\hat{s}_{\text{mean}} - \hat{s}_{i,j}}{\hat{s}_{\text{mean}} - \hat{s}_{\text{min}}} - \frac{1}{2}\right) \tag{5}$$

$$W^{+} = \exp(\frac{1}{2} \times \frac{\hat{s}_{i,j} - \hat{s}_{\text{mean}}}{\hat{s}_{\text{max}} - \hat{s}_{\text{mean}}} - \frac{1}{2}) \tag{6}$$

with W^{-} and W^{+} representing the weights for the "dissimilar pair" or "similar pair" respectively.

Such weights "stretch" the original elements non-linearly. The $-\frac{1}{2}$ term in Eq. 5 and 6 is for normalizing the re-weighted affinity matrix to stay within $[0, 1]$. This dynamic weighting step equalizes the distribution of the distances, making them more discriminative, which is helpful to hashing learning.

Finally, the new cross-modal affinity matrix can be formed as $\mathbf{S} = \{s_{i,j}\}_{i=1,j=1}^{m}$. For reference in the ablation study, the dynamic weighting step is termed as **DW** in the following sections. The entire process of constructing the affinity matrix with dynamic weighting is shown as the blue dashed rectangle in Figure 2.

3.3 Losses for Preserving Cross-Modal Similarity

To preserve the relationships between different modalities, three losses are proposed for the learning stage: the intra- and inter-modal similarity preserving losses and the consistency preserving loss. All the losses are computed based on the similarity matrix constructed by the output features before the min-max hashing layer. The intra-modal similarity is calculated as $\cos(\mathbf{H}_V, \mathbf{H}_V)$ and $\cos(\mathbf{H}_T, \mathbf{H}_T)$. The inter-modal similarity is acquired as $\cos(\mathbf{H}_V, \mathbf{H}_T)$. Therefore, the intra- and inter-modality similarity preserving losses can be calculated as:

$$\begin{aligned}\mathcal{L}_{intra} = \min_{\mathbf{H}_V, \mathbf{H}_T} &||\mathbf{S} - \cos(\mathbf{H}_V, \mathbf{H}_V)||_F^2 + \\ &||\mathbf{S} - \cos(\mathbf{H}_T, \mathbf{H}_T)||_F^2\end{aligned} \tag{7}$$

$$\begin{aligned}\mathcal{L}_{inter} = \min_{\mathbf{H}_V, \mathbf{H}_T} &||\mathbf{S} - \cos(\mathbf{H}_V, \mathbf{H}_T)||_F^2 + \\ &||\mathbf{S} - \cos(\mathbf{H}_T, \mathbf{H}_V)||_F^2\end{aligned} \tag{8}$$

where $||.||_F^2$ is the Frobenius norm. Moreover, depending on the assumption that the cross-modal task can be regarded as the single modal task, a consistency preserving loss is also proposed as follows:

$$\mathcal{L}_{con} = \min_{\mathbf{H}_V, \mathbf{H}_T} ||\mathbf{H}_V - \mathbf{H}_T||_F^2 \tag{9}$$

The model with this loss will be referred to as $\mathcal{L}_{con}$ in short in the following sections. Therefore, the designed model is trained to minimize the following loss function:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{intra} + \lambda_2 \mathcal{L}_{inter} + \lambda_3 \mathcal{L}_{con} \tag{10}$$

where λ_1, λ_2, and λ_3 control the trade-off among the intra-modal, inter-modal, and the consistency losses so that they all make comparable contributions. This hashing learning part is shown as red dashed rectangle in Figure 2.

3.4 Min-Max Hashing Layer for Quantization

Some intuitive binarization methods, like **Sign** function or **Tanh** function, are usually utilized for acquiring the binary codes. Additional loss terms based on binary codes are often added in training, which requires careful hyper-parameters tuning in the objective function. Therefore, these methods are sensitive to the choices of hyper-parameters. Further, they typically cannot explicitly consider the utilization of the Hamming space. A bi-half hashing layer was proposed in [14] to tackle the above issues by maximizing the bit entropy in the Hamming space. The bi-half hashing layer sorts all the elements of each hidden feature along the batch dimension and then assigns the top half of the elements to +1 and others to -1. However, they neglected the fact that hidden feature elements are not always evenly distributed in each dimension, which means forcing values to be divided in half may not accurately represent the learned distribution of the hidden feature components.

Due to these concerns, instead of using the above naive binarization functions or the bi-half hashing layer, we design a more flexible parameter-free hashing layer. This hashing layer, termed as min-max hashing layer, does the following forward mapping:

$$\mathbf{B} = \pi(\mathbf{H}) \tag{11}$$

where the $\mathbf{H}$ is $[H^1, H^2, ..., H^Z] \in \mathbb{R}^{m \times Z}$, and π is the mapping plan. Here $H^z, z = 1, 2, ..., Z$, represents the z-th dimension of the learned latent features in the batch. For each $H^z (z = 1, 2, ..., Z)$, we first find its max and min values (denoted as H^z_{max} and H^z_{min}). Then we calculate the distances between each element and the maximum and minimum values (denoted as d^z_{max} and d^z_{min}). Finally, the elements are assigned to +1 if they are close to H^z_{max}, and -1 otherwise:

$$\pi(H^z) = \begin{cases} +1, d^z_{max} \leq d^z_{min} \\ -1, d^z_{max} > d^z_{min} \end{cases} \tag{12}$$

To illustrate the advantages of the min-max hashing layer, three sample feature vectors a, b and c are shown in Figure 3 as an example. In the feature space, the cosine similarity between b and a is 0.998, and the cosine similarity between b and c is 0.874. Thus, the binary code of b should be more similar to a than c due to the

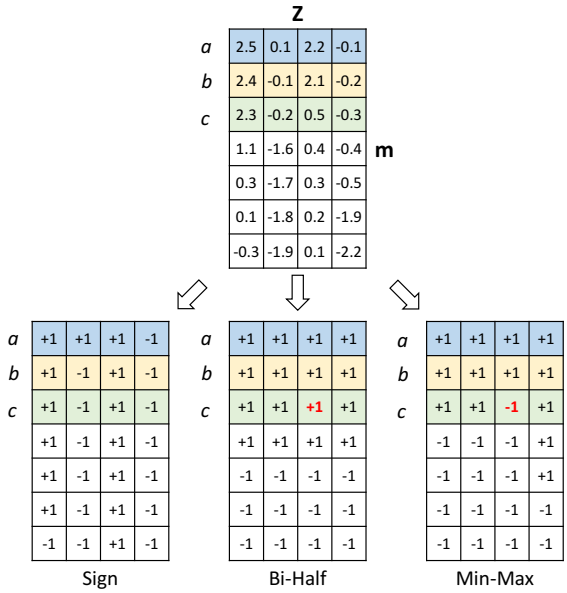

Figure 3: Illustrating comparison of different hashing layers. The features of sample a are in blue, b in yellow, c in green. The difference in binary codes of Bi-Half and Min-Max layers is highlighted in red.

difference in the 3-rd dimension of feature. Either **Sign** function or bi-half hashing layer could not preserve their relative relationships in the latent feature space. However, our min-max hashing layer can group the features with a dynamic threshold based on local statistics, better than forcibly assigning half +1 and −1 (bi-half layer) or by threshold 0 (**Sign**). The Min-Max Hashing module is termed **MMH** in the following sections for reference, which is shown as the pink dashed circle in Figure 2.

4 EXPERIMENTS

To evaluate the **CLIP4Hashing** approach, we conduct experiments on three video-text benchmark datasets: MSRVTT [36], DiDeMo [1], and MSVD [2] datasets.

4.1 Datasets and Evaluation Metrics

MSRVTT [36] is a large-scale video-text dataset that consists of 10,000 video clips with 20 captions for each one. The average video length is 15 seconds with a 30fps frame rate. Our model is trained by following the settings in [7, 38], and we report the results on the 1K-A test set [38].

DiDeMo [1] is a video-text dataset contains about 10K videos and 4 descriptions for each video. The average video length is about 30 seconds with a 30fps frame rate. Following [12, 19, 39], we use the same train/test split and concatenate all captions to form a longer sentence retrieval.

MSVD [2] contains 1970 videos in total and about 40 captions are associated with each video clip. The dataset is split into train, validation, and test sets with 1200, 100, and 670 videos. For the testing stage, we fixed the selection of the fifth captions among the 40 captions for the video-to-text retrieval evaluation (*i.e.*, the video-to-text is one-to-one retrieval).

4.2 Implementations Details

The pre-trained CLIP (ViT-B/32) model is utilized for feature extraction and is not updated during training. For each video, 6 frames for MSVD, 12 frames for MSRVTT, and 24 frames for DiDeMo are uniformly selected. The training epoch is 200, the batch size is 16, and the initial learning rate is 0.01. The learning rate is decreased by a factor of 0.1 at the 150th epochs. The SGD optimizer is employed with 0.9 momentum and 0.0005 weight decay. The trade-off values are set as $\lambda_1 = 0.1$, $\lambda_2 = 1$, $\lambda_3 = 2$ to make those three losses are with similar magnitude. The method is implemented with the PyTorch [21] and trained on the NVIDIA V100 GPU.

4.3 Compared Baseline Methods

Four unsupervised deep hashing algorithms, **DJSRH** [32], **JDSH** [17], **DGCPN** [37], and **S^{2}Bin [23]** are included in the evaluation to compare with our proposed **CLIP4Hashing**. These four deep hashing algorithms leverage the similarity/affinity matrix as the guidance in learning, yet each involves several empirical hyper-parameters. To gain a better grasp of these reference hashing networks, please refer to the original papers. For **S^{2}Bin [23]**, we only quote the performance of the **MSRVTT** on text-to-video retrieval since the splitting strategy for video-to-text testing dataset is different.

Several state-of-the-art non-hashing video-text retrieval methods, including **MMT** [7], **ClipBERT** [12],**HiT** [16], **CE** [19], **FSE** [39], **SUPPORT-SET** [22], and the zero-shot learning of the original **CLIP** model [24], are utilized as comparison with **CLIP4Hashing** as well. The performance of those methods is quoted directly from the corresponding papers. Moreover, the evaluation metric of non-hashing methods is based on cosine similarity, while the hashing ones is based on Hamming distance.

4.4 Performance Comparison and Analysis

For a fair comparison, the same extracted features from the pre-trained CLIP model are provided as the input for all the compared hashing methods. The average recall at different K (R@1, R@5, and R@10) and median rank (MdR) are used to evaluate the performance of all the methods. The bit sizes 256, 512, 1024, 2048 are used since larger bit sizes (like 256 or bigger) are more suitable for maintaining the semantic information for video clips and have been reported to achieve good performance in recent video hashing works [23, 27, 30]. Small bit sizes (*e.g.*, 128 or smaller) can also be employed with our model. However, too compact binary codes will make the well-defined semantic space collapse and the performance will drop catastrophically. Therefore, the bit size selection can be a trade-off between performance and efficiency.

The results on MSRVTT, DiDeMo, and MSVD datasets are shown in Tables 1 to 3. Our approach achieves the best performance compared with other hashing methods. With the increase of bit size, the performance of our **CLIP4Hashing** improves more significantly than others. Moreover, with a large bit size (*e.g.*, the case of 2048), **CLIP4Hashing** achieved competitive performance compared with some non-hashing based video-text retrieval methods. In addition, the performance of other unsupervised hashing-based methods demonstrates that the simple adoption of image-text retrieval methods cannot have satisfactory results in the video-text domain.

In Figure 4, we show an illustration of the R@5 results of the designed method and **DJSRH** on the query text "a cat is playing and playing a turtle" from the MSVD dataset. The green bounding box stands for the correctly retrieved data, while the red represents incorrect data samples. **CLIP4Hashing** can retrieve the correct video at the top one from the testing split of the MSVD dataset. Compared with **CLIP4Hashing**, even though **DJSRH** can capture the information of the "cat" and "playing", it misses the information of "playing a turtle", which results in the degradation of performance.

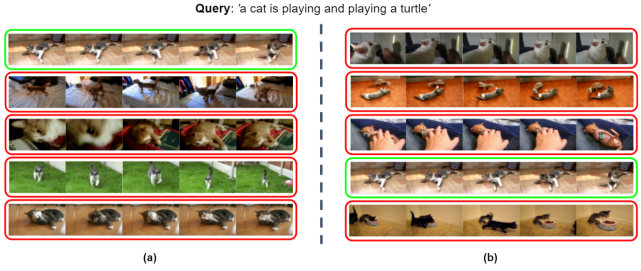

Figure 4: Illustrating results of (a) CLIP4Hashing vs (b) DJSRH on the Query Text: "A Cat is Playing and Playing a Turtle".

Table 4: Ablation Study of different modules on CLIP4Hashing (MSRVTT dataset, 1024 bit size).

Module names					Text-to-video retrieval				Video-to-text retrieval			
CLIPbase	SHM	DW	MMH	$\mathcal{L}_{con}$	R@1 ↑	R@5↑	R@10↑	MdR↓	R@1 ↑	R@5↑	R@10↑	MdR↓
✓					8.6	22.6	29.8	39	7.0	20.1	28.0	53
✓	✓				11.7	27.7	37.1	23	11.5	28.1	38.2	22
✓	✓	✓			14.6	32.9	41.5	18	13.9	32.3	42.0	18
✓	✓	✓	✓		31.0	50.7	58.6	**5**	32.7	50.8	59.2	**5**
✓	✓	✓	✓	✓	**32.3**	**52.1**	**61.3**	**5**	**32.9**	**51.5**	**59.7**	**5**

Table 5: Comparison of bi-half and min-max hashing layer on CLIP4Hashing (MSRVTT dataset, 1024 bit size).

Module name	Text-to-video retrieval			
	R@1 ↑	R@5↑	R@10↑	MdR↓
CLIP4Hashing with bi-half	20.0	38.3	47.2	14
CLIP4Hashing with min-max	**32.3**	**52.1**	**61.3**	**5**

Table 6: Comparison of DW, MMH and $\mathcal{L}_{con}$ modules on DJSRH (MSRVTT dataset, 1024 bit size).

Method	Text-to-video retrieval			
	R@1 ↑	R@5↑	R@10↑	MdR↓
DJSRH	5.4	16.2	23	57
DJSRH with DW	7.6	22.0	32.8	32
DJSRH with MMH	14.8	30.5	40.3	20
DJSRH with $\mathcal{L}_{con}$	7.5	22.7	31.3	34.5

4.5 Ablation Study

To comprehensively analyze the impact of all designed modules in our framework, we present in Table 4 the results on the MSRVTT dataset with the 1024-bit setting. It is easy to observe that the single hashing module (SHM) boosts the performance with our proposed affinity matrix $\mathbf{S}$ (CLIPbase) by more than 3% for the R@1 metric. Therefore, we evaluate other modules based on the structure of CLIPbase+SHM to make the evaluation more distinct. From the second and third rows, we can find that the dynamic weighting (DW) process can improve over CLIPbase+SHM as well. Furthermore, the min-max hashing layer (MMH) improves the performance along with the proposed affinity matrix. The last row shows the consistency preserving loss term $\mathcal{L}_{con}$ enhances the performance of the R@1 metric and reduces the differences between text-to-video and video-to-text retrieval tasks. From all the rows in Table 4, it is evident that each of our proposed modules helps improving the performance, and together they deliver a remarkable performance boost.

In addition, we compare the bi-half hashing layer with our min-max hashing layer within **CLIP4Hashing**. Table 5 shows that the min-max hashing layer design outperforms the bi-half hashing layer since it can better preserve the semantic relationships. We also applied the designed DW, MMH and $\mathcal{L}_{con}$ modules in other method like DJSRH. Table 6 shows that either dynamic weighting, min-max hashing layer or consistency preserving loss designs can boost the performance even on other hashing methods.

4.6 Hashing Codes Efficiency Analysis

As shown in Tables 1-3 and Figure 1, **CLIP4Hashing** will be improved with a larger bit size. However, a larger bit size requires more memory storage and computation time. Even though usage of a 2048 bit size in our approach still saves roughly 87.5% storage space compared to the CLIP feature, which has a dimension of 512 for each vector. Moreover, for the search speed, the bit-wise logic operation for hashing-based methods is much faster than the matrix multiplication of the non-hashing methods. For the 1 million number of our 2048 bit hashing codes and the CLIP feature vectors, a rough similarity computation time estimate is 4.3 seconds against 156.9 seconds. Thus, even with 2048 bit size, the hashing-based codes are still able to save around 97% searching time.

5 CONCLUSION

In this paper, we propose a novel unsupervised deep hashing approach named **CLIP4Hashing** for video-text retrieval. To our best knowledge, it is the first approach utilizing the CLIP model for video-text hashing with a single hashing network. Two parameter-free modules, dynamic weighting and min-max hashing layer, are designed, which help drastically boost the performance of the framework. Extensive experiments demonstrate **CLIP4Hashing** outperforms existing unsupervised hashing methods on three video-text benchmarks, establishing new performance records on these datasets. With a larger bit size like 2048, **CLIP4Hashing** even has competitive performance compared with non-hashing-based methods.

ACKNOWLEDGMENTS

Y.Zhuo and B.Li were supported in part by an ONR grant (# N00014-19-1-2119). Any opinions expressed in this material are those of the authors and do not necessarily reflect the views of ONR.

REFERENCES

[1] Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. 2017. Localizing moments in video with natural language. In *Proceedings of the IEEE international conference on computer vision*. 5803–5812.
[2] David Chen and William B Dolan. 2011. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*. 190–200.
[3] Zhen-Duo Chen, Chuan-Xiang Li, Xin Luo, Liqiang Nie, Wei Zhang, and Xin-Shun Xu. 2019. SCRATCH: A scalable discrete matrix factorization hashing

framework for cross-modal retrieval. *IEEE Transactions on Circuits and Systems for Video Technology* 30, 7 (2019), 2262–2275.

[4] Cheng Deng, Zhaojia Chen, Xianglong Liu, Xinbo Gao, and Dacheng Tao. 2018. Triplet-based deep hashing network for cross-modal retrieval. *IEEE Transactions on Image Processing* 27, 8 (2018), 3893–3903.

[5] Pak Lun Kevin Ding, Yikang Li, and Baoxin Li. 2018. Mean local group average precision (mLGAP): a new performance metric for hashing-based retrieval. *arXiv preprint arXiv:1811.09763* (2018).

[6] Yixian Fang, Huaxiang Zhang, and Yuwei Ren. 2019. Unsupervised cross-modal retrieval via Multi-modal graph regularized Smooth Matrix Factorization Hashing. *Knowledge-Based Systems* 171 (2019), 69–80. https://doi.org/10.1016/j.knosys.2019.02.004

[7] Valentin Gabeur, Chen Sun, Karteek Alahari, and Cordelia Schmid. 2020. Multi-modal transformer for video retrieval. In *European Conference on Computer Vision*. Springer, 214–229.

[8] Vijetha Gattupalli, Yaoxin Zhuo, and Baoxin Li. 2019. Weakly supervised deep image hashing through tag embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10375–10384.

[9] Hengtong Hu, Lingxi Xie, Richang Hong, and Qi Tian. 2020. Creating something from nothing: Unsupervised knowledge distillation for cross-modal hashing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3123–3132.

[10] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*. PMLR, 4904–4916.

[11] Lu Jin, Zechao Li, and Jinhui Tang. 2020. Deep semantic multimodal hashing network for scalable image-text and video-text retrievals. *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[12] Jie Lei, Linjie Li, Luowei Zhou, Zhe Gan, Tamara L Berg, Mohit Bansal, and Jingjing Liu. 2021. Less is more: Clipbert for video-and-language learning via sparse sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7331–7341.

[13] Chao Li, Cheng Deng, Lei Wang, De Xie, and Xianglong Liu. 2019. Coupled cyclegan: Unsupervised hashing network for cross-modal retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 176–183.

[14] Yunqiang Li and Jan van Gemert. 2021. Deep Unsupervised Image Hashing by Maximizing Bit Entropy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 2002–2010.

[15] Haomiao Liu, Ruiping Wang, Shiguang Shan, and Xilin Chen. 2016. Deep supervised hashing for fast image retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2064–2072.

[16] Song Liu, Haoqi Fan, Shengsheng Qian, Yiru Chen, Wenkui Ding, and Zhongyuan Wang. 2021. HiT: Hierarchical Transformer With Momentum Contrast for Video-Text Retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 11915–11925.

[17] Song Liu, Shengsheng Qian, Yang Guan, Jiawei Zhan, and Long Ying. 2020. Joint-Modal Distribution-Based Similarity Hashing for Large-Scale Unsupervised Deep Cross-Modal Retrieval. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, New York, NY, USA, 1379–1388. https://doi.org/10.1145/3397271.3401086

[18] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, and Shih-Fu Chang. 2012. Supervised hashing with kernels. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2074–2081.

[19] Yang Liu, Samuel Albanie, Arsha Nagrani, and Andrew Zisserman. 2019. Use What You Have: Video retrieval using representations from collaborative experts. In *Proceedings of the British Machine Vision Conference (BMVC)*, Kirill Sidorov and Yulia Hicks (Eds.). BMVA Press, Article 73, 14 pages. https://doi.org/10.5244/C.33.73

[20] Norman Mu, Alexander Kirillov, David Wagner, and Saining Xie. 2021. SLIP: Self-supervision meets Language-Image Pre-training. *arXiv preprint arXiv:2112.12750* (2021).

[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019), 8026–8037.

[22] Mandela Patrick, Po-Yao Huang, Yuki Asano, Florian Metze, Alexander G Hauptmann, Joao F. Henriques, and Andrea Vedaldi. 2021. Support-set bottlenecks for video-text representation learning. In *International Conference on Learning Representations*. https://openreview.net/forum?id=EqoXe2zmhrh

[23] Mengshi Qi, Jie Qin, Yi Yang, Yunhong Wang, and Jiebo Luo. 2021. Semantics-Aware Spatial-Temporal Binaries for Cross-Modal Video Retrieval. *IEEE Transactions on Image Processing* 30 (2021), 2989–3004.

[24] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*. PMLR, 8748–8763.

[25] Nikhil Rasiwasia, Jose Costa Pereira, Emanuele Coviello, Gabriel Doyle, Gert R.G. Lanckriet, Roger Levy, and Nuno Vasconcelos. 2010. A New Approach to Cross-Modal Multimedia Retrieval. In *Proceedings of the 18th ACM International Conference on Multimedia* (Firenze, Italy) *(MM '10)*. Association for Computing Machinery, New York, NY, USA, 251–260. https://doi.org/10.1145/1873951.1873987

[26] Fumin Shen, Yan Xu, Li Liu, Yang Yang, Zi Huang, and Heng Tao Shen. 2018. Unsupervised deep hashing with similarity-adaptive and discrete optimization. *IEEE transactions on pattern analysis and machine intelligence* 40, 12 (2018), 3034–3044.

[27] Ling Shen, Richang Hong, Haoran Zhang, Xinmei Tian, and Meng Wang. 2019. Video Retrieval with Similarity-Preserving Deep Temporal Hashing. *ACM Trans. Multimedia Comput. Commun. Appl.* 15, 4, Article 109 (dec 2019), 16 pages. https://doi.org/10.1145/3356316

[28] Yuming Shen, Jie Qin, Jiaxin Chen, Mengyang Yu, Li Liu, Fan Zhu, Fumin Shen, and Ling Shao. 2020. Auto-encoding twin-bottleneck hashing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2818–2827.

[29] Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. 2021. FLAVA: A Foundational Language And Vision Alignment Model. *arXiv preprint arXiv:2112.04482* (2021).

[30] Jingkuan Song, Hanwang Zhang, Xiangpeng Li, Lianli Gao, Meng Wang, and Richang Hong. 2018. Self-supervised video hashing with hierarchical binary auto-encoder. *IEEE Transactions on Image Processing* 27, 7 (2018), 3210–3221.

[31] Shupeng Su, Chao Zhang, Kai Han, and Yonghong Tian. 2018. Greedy Hash: Towards Fast Optimization for Accurate Hash Coding in CNN. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2018/file/13f3cf8c531952d72e5847c4183e6910-Paper.pdf

[32] Shupeng Su, Zhisheng Zhong, and Chao Zhang. 2019. Deep joint-semantics reconstructing hashing for large-scale unsupervised cross-modal retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 3027–3035.

[33] Ruikui Wang, Ruiping Wang, Shishi Qiao, Shiguang Shan, and Xilin Chen. 2020. Deep position-aware hashing for semantic continuous image retrieval. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2493–2502.

[34] Gengshen Wu, Zijia Lin, Jungong Han, Li Liu, Guiguang Ding, Baochang Zhang, and Jialie Shen. 2018. Unsupervised Deep Hashing via Binary Latent Factor Models for Large-scale Cross-modal Retrieval.. In *IJCAI*. 2854–2860.

[35] De Xie, Cheng Deng, Chao Li, Xianglong Liu, and Dacheng Tao. 2020. Multi-task consistency-preserving adversarial hashing for cross-modal retrieval. *IEEE Transactions on Image Processing* 29 (2020), 3626–3637.

[36] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. 2016. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5288–5296.

[37] Jun Yu, Hao Zhou, Yibing Zhan, and Dacheng Tao. 2021. Deep Graph-neighbor Coherence Preserving Network for Unsupervised Cross-modal Hashing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4626–4634.

[38] Youngjae Yu, Jongseok Kim, and Gunhee Kim. 2018. A joint sequence fusion model for video question answering and retrieval. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 471–487.

[39] Bowen Zhang, Hexiang Hu, and Fei Sha. 2018. Cross-modal and hierarchical modeling of video and text. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 374–390.

[40] Peng-Fei Zhang, Yadan Luo, Zi Huang, Xin-Shun Xu, and Jingkuan Song. 2021. High-order nonlocal Hashing for unsupervised cross-modal retrieval. *World Wide Web* 24, 2 (2021), 563–583.