AVHash: Joint Audio-Visual Hashing for Video Retrieval

Yuxiang Zhou
School of Computer Science, BUPT
Beijing, China
yuxiang.zhou@bupt.edu.cn

Zhe Sun
School of Computer Science, BUPT
Beijing, China
2021sunzhe@bupt.edu.cn

Rui Liu
School of Computer Science, BUAA
Beijing, China
lr@buaa.edu.cn

Yong Chen*
School of Computer Science, BUPT
Beijing, China
yong.chen@bupt.edu.cn

Dell Zhang
TeleAI, China Telecom
Shanghai, China
dell.z@ieee.org

ABSTRACT

Video hashing is a technique of encoding videos into binary vectors, facilitating efficient video storage and high-speed computation. Current approaches to video hashing predominantly utilize sequential frame images to produce semantic binary codes. However, videos encompass not only visual but also audio signals. Therefore, we propose a tri-level Transformer-based audio-visual hashing technique for video retrieval, named AVHash. It first processes audio and visual signals separately using pre-trained AST and ViT large models, and then projects temporal audio and keyframes into a shared latent semantic space using a Transformer encoder. Subsequently, a gated attention mechanism is designed to fuse the paired audio-visual signals in the video, followed by another Transformer encoder leading to the final video representation. The training of this AVHash model is directed by a video-based contrastive loss as well as a semantic alignment regularization term for audio-visual signals. Experimental results show that AVHash significantly outperforms existing video hashing methods in video retrieval tasks. Furthermore, ablation studies reveal that while video hashing based solely on visual signals achieves commendable mAP scores, the incorporation of audio signals can further boost its performance for video retrieval.

CCS CONCEPTS

• **Information systems** → **Top-k retrieval in databases**; • **Computing methodologies** → **Visual content-based indexing and retrieval**.

KEYWORDS

Learning to Hash, Video Retrieval, Audio-Visual, Transformer

ACM Reference Format:
Yuxiang Zhou, Zhe Sun, Rui Liu, Yong Chen, and Dell Zhang. 2024. AVHash: Joint Audio-Visual Hashing for Video Retrieval. In *Proceedings of the 32nd ACM International Conference on Multimedia (MM '24), October 28-November 1, 2024, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3664647.3681266

*Yong Chen is the **Corresponding** author at Beijing University of Posts and Communications (BUPT), China. By the way, the first two authors made equal contributions.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
MM'24, October 28–November 1, 2024, Melbourne, Australia
© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/10.1145/3664647.3681266

1 INTRODUCTION

In this digital age, short videos[1] have evolved into a prominent content form, emerging as a new narrative style, information source, communication mode, self-expression avenue, and interaction space, significantly influencing modern cognitive psychology, aesthetic experiences, and value preferences [27]. Therefore, the management and processing of videos are crucial yet challenging due to their vast scale and continuous expansion. Video hashing [28], the technique that encodes a video into a binary vector, can not only reduce memory requirements but also facilitate further indexing and computation, making it a potential tool for video analysis and understanding [1, 45].

Current video hashing methods begin by summarizing a given video through a series of keyframes, which are then modeled using non-temporal or temporal networks such as MLP [29, 36], CNN [21, 22], LSTM [7, 16], and Transformer [39]. To name just a few, MCMSH [13] models video feature contexts at three granularities, integrating them into an MLP-Mixer for a comprehensive representation; SRH [12] employs pre-trained CNN features of video keyframes as inputs to an LSTM network for binary code generation; DSVH [2] utilizes 3D convolutions on spatio-temporal keyframes via a 3DCNN to create $\{0, 1\}$-embeddings; ConMH [42] leverages a Transformer encoder-decoder on sequential pre-trained CNN features to learn binary codes of videos in a self-supervised fashion. It is evident that the underlying assumption of these approaches is as follows. The current computing capabilities face challenges, or are relatively insufficient, for processing all frames of a video clip, including short videos lasting just a few minutes. However, due to the high redundancy in video frames, summarizing a video's semantic content through keyframe extraction is an effective strategy.

Furthermore, we also observe that existing approaches to video hashing tend to view videos as *Silent Films*, neglecting a crucial element — audio signals — which are prevalent today and vitally important for full video comprehension. Neuroscience research [8] suggests that the combined effect of multiple sensory stimuli can produce an effect where "the whole is greater than the sum of its parts". For example, the concurrent stimulation of sight and sound

[1] https://digitaldelane.com/the-rise-of-short-form-video-content

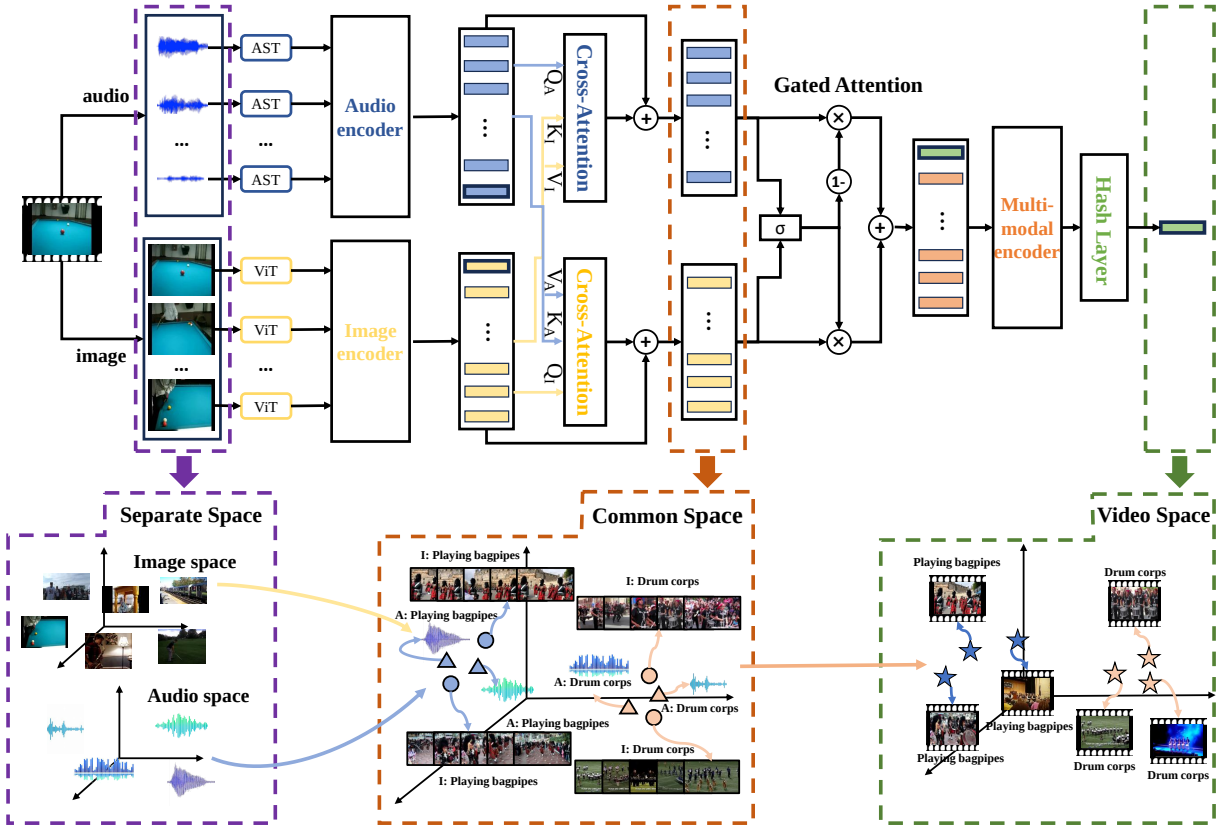


Figure 1: The architecture of our AVHash: from separate spaces to the common latent semantic space, and to the final video space.

during the viewing of TV shows or movies can evoke deep and holistic emotions, as well as enhance understanding of the content. In the context of machine learning, the principle that multi-modal learning works better than single-modal has also been mathematically analyzed and empirically validated by Huang et al. [17]. This inspires our investigation into video hashing that integrates both audio and visual signals, instead of solely relying on frame images.

Given a video with synchronized audio-visual signals, we first extract N (e.g., 25) keyframe images and divide its audio into N sound waveform segments; then, we map them to a shared latent space through separate visual-Transformer and audio-Transformer networks respectively; in the end, we combine the audio-visual embeddings into the ultimate video representation using a cross-attention mechanism and an av-Transformer. This completes our method for translating a video from its original signals to its final semantic representation. Given its novel integration of audio and visual signals, we designate this method as "AVHash", whose primary contributions are highlighted below:

- It distinguishes itself by learning to hash videos with both audio and visual components, not merely frame images, and introduces a three-tier Transformer network to map its separate original signals to a shared latent space and finally to the video representation. To the best of our knowledge, this is the first work that fully leverages both audio and visual elements for video hashing.
- It engages in video-oriented contrastive learning within the final video space and also ensures semantic alignment between audio and visual signals in the common latent space. This approach stems from the understanding that a video's audio-visual signals can be considered not only separately but also collectively as a whole.
- It is validated through comprehensive experiments on two public datasets, ActivityNet and FCVID, under various conditions against a number of state-of-the-art competitors. Notably, AVHash demonstrates significant improvements over those existing methods, showing that while visual-based video hashing already achieves high mAP scores, incorporating audio signals further enhances these metrics.

2 RELATED WORK

Video hashing [26, 34, 35, 43, 44, 46] aims to generate a semantic binary code for each video, and therefore significantly reduce storage space and enhance computation speed. This process would be of great benefit to downstream applications including video retrieval, analysis, and understanding.

Current video hashing methods first extract a video's representative image frames and then apply non-temporal or temporal models to create $\{0, 1\}$ embeddings. The early approaches typically leverage a pre-trained CNN to extract features from video keyframes first, and then average them to obtain an overall representations. Subsequently,

these methods employ image hashing techniques such as LSH [9], ITQ [11], SDH [24, 33], COSDISH [19], SCDH [4], CSQ [47], LTH [3], and MDSH [40] to produce a binary vector, which serves as the video's hash code. More recent approaches predominantly use deep neural networks. For example, DVH [28] utilizes temporal pooling across sequential MLP outputs from video keyframes; SRH [12] captures keyframe CNN features and feeds them into an LSTM network for supervised learning; DSVH [2] extends 2DCNN to 3DCNN for modeling spatio-temporal keyframes; ConMH [42] inputs pre-trained CNN features of sequential keyframes into a Transformer encoder-decoder network, enabling self-supervised learning to hash.

To sum up, existing learning-to-hash methods for video retrieval only utilize visual signals (i.e., frame images) of videos, overlooking a crucial component: audio waveforms. By contrast, our proposed AVHash technique learns to project videos into Hamming space using both visual and audio signals jointly.

In addition, while AVHash utilizes multi-modal signals, it is fundamentally different from existing multi-modal hashing methods such as CLIP4Hashing [49], S^2BIN [31], EDMH [5, 6], and SDAH [30]. Those multi-modal hashing techniques aim to learn hash codes for each individual modality (e.g., text, image, or video) for cross-modal retrieval tasks (e.g., image-to-text, text-to-image, video-to-text, text-to-video searches). However, AVHash just learns one-modality's embeddings by fully utilizing its audio-visual components and only performs video-to-video retrieval task. Conversely, AVHash focuses on learning binary embeddings for a single modality, video, while leveraging its audio-visual components, for the purpose of performing video-to-video retrieval better.

3 THE PROPOSED METHOD

3.1 Problem Statement

Given a set of V videos, denoted as $\mathcal{S} = \{\mathbf{v}_i\}_{i=1}^{V}$, our AVHash aims to learn a hash function that maps each video to a compact binary code.

Specifically, each video is represented by its N keyframes $\{\mathbf{I}_j\}_{j=1}^{N}$ and N audio segments $\{\mathbf{A}_j\}_{j=1}^{N}$. We extract ViT features [32] $\{\mathbf{f}_j^I\}_{j=1}^{N}$ for keyframes and AST features [10] $\{\mathbf{f}_j^A\}_{j=1}^{N}$ for audio segments, where I and A correspond to the image and audio modalities respectively. These modal features are then fed into their respective branches of the AVHash architecture (see Fig. 1), where the features sequentially pass through separate Transformer encoders to generate image/audio embeddings, a cross-attention for inter-modal interactions, a gated-attention for feature fusion, a multimodal Transformer encoder, and conclude at a hash layer to produce the video's embedding. This embedding is then binarized into the final hash code $\mathbf{b}_i \in \{-1, +1\}^q$ using a Sign(·) function for video $\mathbf{v}_i$, where q represents the code length.

3.2 Model Architecture

Our AVHash consists of the following modules: "Input", " Modality-Specific Encoder", "Cross Attention", "Gated Attention", "Multi-Modal Encoder", "Hash Layer", and "Loss Function" (see Fig. 1).

Input. To enable the model to process data from these modalities, we use the AST [10] and CLIP [32] models to extract features from audios and images, respectively, resulting in $\{\mathbf{f}_j^A\}_{j=1}^{N} \in \mathbb{R}^{N\times 768}$ and $\{\mathbf{f}_j^I\}_{j=1}^{N} \in \mathbb{R}^{N\times 768}$. Before feeding them into the subsequent modules, a CLS token needs to be added in front of the feature sequence, i.e.,

$$\mathbf{z}_A^0 = [\mathrm{cls}^A; \mathbf{f}_1^A; \mathbf{f}_2^A; \cdots; \mathbf{f}_N^A] + \mathbf{E}_{\mathrm{pos}}^A, \tag{1}$$

$$\mathbf{z}_I^0 = [\mathrm{cls}^I; \mathbf{f}_1^I; \mathbf{f}_2^I; \cdots; \mathbf{f}_N^I] + \mathbf{E}_{\mathrm{pos}}^I, \tag{2}$$

where $\mathbf{E}_{\mathrm{pos}}^A$ and $\mathbf{E}_{\mathrm{pos}}^I$ correspond to the position encodings of time-series audios and temporal keyframes, respectively.

Table 1: The configurations of our AVHash architecture.

Layer	Configurations
0: Input	Audio Segments: $\{A_1, \cdots, A_N\}$; Sequential Keyframes: $\{I_1, \cdots, I_N\}$.
1: Feature	Image: CLIP Visual Branch (L=24, H=1024, A=16); FC layer (1024 × 768); Audio: AST (L=12, H=768, A=12).
2: Modality-Specific Encoder	Image: (L_S=1, H_S=768, A_S=12); Audio: (L_S=1, H_S=768, A_S=12).
3: Cross-Attention	I → A: ($L_{I\to A}$=1, $H_{I\to A}$=768, $A_{I\to A}$=12); A → I: ($L_{A\to I}$=1, $H_{A\to I}$=768, $A_{A\to I}$=12).
4: Gated Attention Fusion Module	FC layer (1536 × 2); **Tanh**(·); Softmax(·).
5: Multi-Modal Encoder	Transformer Blocks (L_M=1, H_M=768, A_M=12).
6: Hash Layer (Binary-like)	FC layer(768 × q); **Tanh**(·).
7: Output (Bianry Codes)	**Sign**(·).

Modality-Specific Encoder. AVHash comprises two modality-specific encoders, designed to capture the unique characteristics inherent to each modality. We employ standard Transformer as audio and visual encoders. Each encoder consists of L_S Transformer layers, and each Transformer layer is mainly composed of Multi-Head Self-Attention (MSA) (with the number of multi-heads as A_S) and an MLP layer. The output of $(\ell - 1)$-th MLP layer is fed to the (ℓ)-th Multi-Head Self-Attention (MSA) layer. Layer Normalization (LN) is applied before each layer, and a residual connection is incorporated after each layer. The MLP layer contains two linear fully connected (FC) sub-layers with a non-linear GELU activation function:

$$\mathbf{z}_m^{\ell'} = \mathrm{MSA}(\mathrm{LN}(\mathbf{z}_m^{\ell-1})) + \mathbf{z}_m^{\ell-1}, \tag{3}$$

$$\mathbf{z}_m^{\ell} = \mathrm{MLP}(\mathrm{LN}(\mathbf{z}_m^{\ell'})) + \mathbf{z}_m^{\ell'}, \tag{4}$$

where $\ell = 1, \ldots, L_S$, $m \in \{A, I\}$ denotes the audio or visual modality. By following the above procedure, we could get the latent representation $\mathbf{z}_m^{L_S}$ of video $\mathbf{v}_i$'s audio and keyframe features.

Notice that the output $\mathbf{z}_m^{L_S}$ of each encoder contains $(1 + N)$ parts corresponding to the preset token CLS and N audio/visual features. Then, we use the first part's embedding, dubbed $\mathbf{z}_m^{L_S}[0]$, as the holistic representation and the second part's embedding, dubbed $\mathbf{z}_m^{L_S}[1{:}]$, as the representation for each feature.

Cross-Modality Attention. After unimodal feature encoding, we employ a cross attention to capture the semantic interactions between audio and visual modalities. Concretely, we use a multi-head cross-attention (MCA) with a linear fully connected (FC) layer:

$$\mathbf{z}_{m_2\to m_1} = \mathrm{FC}(\mathrm{MCA}(\mathbf{z}_{m_1}^{L_S}[1:], \mathbf{z}_{m_2}^{L_S}[1:])) + \mathbf{z}_{m_1}^{L_S}[1:], \tag{5}$$

Table 2: Dataset Statistics

Datasets	#Training	#Validation	#Testing	#Total
ActNet-20	930	237	228	1,395
ActNet-50	2,544	666	636	3,846
ActNet-all	10,023	2,512	2,413	14,948
FCVID-20	3,060	1,520	1,520	6,100
FCVID-50	7,993	3,973	3,983	15,899
FCVID-all	45,508	22,754	22,754	91,016

where m_1 denotes the target modality and m_2 is the source modality, and the MCA is given as follow:

$$\mathrm{MCA}(\mathbf{z}_{m_1}, \mathbf{z}_{m_2}) = \mathrm{Concat}(\mathrm{head}_1, \ldots, \mathrm{head}_{A_c})\mathbf{W}, \tag{6}$$

$$\mathrm{head}_i = \mathrm{Softmax}\left(\frac{\mathbf{Q}_i^{m_1} \times \mathbf{K}_i^{m_2 T}}{\sqrt{d}}\right)\mathbf{V}_i^{m_2}, i = 1, \ldots A_c, \tag{7}$$

and

$$\mathbf{Q}_i^m = \mathbf{z}_m \times \mathbf{W}_i^Q, \tag{8}$$

$$\mathbf{K}_i^m = \mathbf{z}_m \times \mathbf{W}_i^K, \tag{9}$$

$$\mathbf{V}_i^m = \mathbf{z}_m \times \mathbf{W}_i^V, \tag{10}$$

where $\mathbf{W}$, $\mathbf{W}_i^Q$, $\mathbf{W}_i^K$, and $\mathbf{W}_i^V$ are learnable matrice, and A_c equals to $A_{A\to I}$ or $A_{I\to A}$. Thus, we obtain representations enriched with cross-modal information $\mathbf{z}_{A\to I}$, $\mathbf{z}_{I\to A}$, by the following formula:

$$\mathbf{z}_{A\to I} = \mathrm{FC}(\mathrm{MCA}(\mathbf{z}_I^{L_S}[1:], \mathbf{z}_A^{L_S}[1:])) + \mathbf{z}_I^{L_S}[1:], \tag{11}$$

$$\mathbf{z}_{I\to A} = \mathrm{FC}(\mathrm{MCA}(\mathbf{z}_A^{L_S}[1:], \mathbf{z}_I^{L_S}[1:])) + \mathbf{z}_A^{L_S}[1:]. \tag{12}$$

Gated-Attention Fusion Module. To fuse semantics of different modalities, we introduce a gated attention to determine the importance of each modality. For simplicity, we implement this module via a linear layer followed by a Tanh activation function. This module accepts two modalities' features as input and concatenates the features of each modality by rows. The output of the module is fed into the Softmax function to ensure that the sum of weights across two modalities equals to 1:

$$[\alpha, 1-\alpha] = \mathrm{Softmax}(\tanh(\mathrm{FC}([\mathbf{z}_{I\to A}, \mathbf{z}_{A\to I}]))); \tag{13}$$

the final fused features are then formulated as:

$$\mathbf{z}_{AI} = \alpha \cdot \mathbf{z}_{I\to A} + (1-\alpha) \cdot \mathbf{z}_{A\to I}. \tag{14}$$

Table 3: The number of parameters of different methods.

Methods	#Parameters	Methods	#Parameters
MCMSH [13]	1.2M	DSVH [2]	63M
BTH [25]	2M	SRH [12]	4.5M
DKPH [23]	4M	AVH [41]	22M
SSTH [48]	5M	AVHash (Ours)	28.4M
ConMH [42]	11.3M	—	—

Multi-Modal Encoder. After fusing the two modality features, to obtain the final representation of the video, we introduce a multi-modal Transformer encoder. This encoder consist of L_M Transformer layers, with each composed of Multi-head Self-Attention (MSA) (the number of multi-heads is A_M) and an MLP layer. Before feeding $\mathbf{z}_{AI}$, a **cls** token is added at the beginning of $\mathbf{z}_{AI}$, and its corresponding output serves as the video's final representation:

$$\mathbf{z}_V^0 = [\mathrm{cls}^{AI}; \mathbf{z}_{AI}] + \mathbf{E}_{\mathrm{pos}}^{AI}, \tag{15}$$

$$\mathbf{z}_V^{\ell'} = \mathrm{MSA}(\mathrm{LN}(\mathbf{z}_V^{\ell-1})) + \mathbf{z}_V^{\ell-1}, \tag{16}$$

$$\mathbf{z}_V^{\ell} = \mathrm{MLP}(\mathrm{LN}(\mathbf{z}_V^{\ell'})) + \mathbf{z}_V^{\ell'}, \tag{17}$$

where $\ell = 1, \ldots, L_M$. By following the above procedure, we can get the latent representation $\mathbf{z}_V^{L_M}$ of video $\mathbf{v}_i$. Note that the output $\mathbf{z}_V^{L_M}$ of each encoder also contains $(1+N)$ parts corresponding to the preset token CLS and N multimodal features. We employ the first part's embedding, i.e., $\mathbf{z}_V^{L_M}[0]$, as its final representation.

Hash Layer. To obtain the final hash codes of video $\mathbf{v}_i$, we add a hash layer, i.e., a FC network with the $\mathrm{Tanh}(\cdot)$ function, on top of the Multi-Modal Encoder. Formally, given video $\mathbf{v}_i$, we transform its Multi-Modal Encoder's output $\mathbf{z}_V^{L_M}[0]$ into a q-dimensional binary-like real-valued vector $\mathbf{g}_i$ via:

$$\mathbf{g}_i = \mathbf{Tanh}(\mathbf{FC}(\mathbf{z}_V^{L_M}[0])) \in \mathbb{R}^q, \tag{18}$$

which is then forwarded to the $\mathbf{Sign}(\cdot)$ function for the final binary code $\mathbf{b}_i$, i.e.,

$$\mathbf{b}_i = \mathbf{Sign}(\mathbf{g}_i) \in \{-1, +1\}^q. \tag{19}$$

Loss Function. We leverage the InfoNCE loss [14, 37] to train our network. InfoNCE loss consists of three types samples: an anchor sample $\mathbf{a}$, a positive sample $\mathbf{p}$, and n_o negative samples $\{\mathbf{n}_i\}_{i=1}^{n_o}$. The anchor and positive samples belong to the same class, while the anchor and negative samples belong to different classes. This InfoNCE loss maximizes the agreement between positive pairs and the dis-agreement between positive and negative samples, which is formulated as:

$$\mathbf{L}(\mathbf{a}, \mathbf{p}, \{\mathbf{n}_i\}_{i=1}^{n_o}) = -\log \frac{\exp\left(\frac{\mathbf{sim}(\mathbf{a},\mathbf{p})}{\tau}\right)}{\sum_{i=1}^{n_o} \exp\left(\frac{\mathbf{sim}(\mathbf{a},\mathbf{n}_i)}{\tau}\right) + \exp\left(\frac{\mathbf{sim}(\mathbf{a},\mathbf{p})}{\tau}\right)}, \tag{20}$$

where $\mathbf{sim}(\cdot,\cdot)$ denotes the similarity function (e.g., cosine similarity), τ and n_o are two pre-set hyper-parameters. In our experiments, we build 5 types of positive and negative sample pairs:

$$\begin{aligned}\mathbf{L_1} = {} & \mathbf{L}_A(\mathbf{a}^A, \mathbf{p}^A, \{\mathbf{n}_i^A\}_{i=1}^{n_o}) + \mathbf{L}_I(\mathbf{a}^I, \mathbf{p}^I, \{\mathbf{n}_i^I\}_{i=1}^{n_o}) \\ & + \mathbf{L}_{IA}(\mathbf{a}^I, \mathbf{p}^A, \{\mathbf{n}_i^A\}_{i=1}^{n_o}) + \mathbf{L}_{AI}(\mathbf{a}^A, \mathbf{p}^I, \{\mathbf{n}_i^I\}_{i=1}^{n_o}),\end{aligned} \tag{21}$$

and

$$\mathbf{L_2} = \mathbf{L}_V(\mathbf{a}^V, \mathbf{p}^V, \{\mathbf{n}_i^V\}_{i=1}^{n_o}), \tag{22}$$

where $\mathbf{L_1}$ is the semantic alignment between audio and visual signals in the common latent semantic space, and $\mathbf{L_2}$ represents the video-based contrastive loss in the video space. Then, the overall objective can be constructed as:

$$\mathbf{L} = \alpha\mathbf{L_1} + \mathbf{L_2}, \tag{23}$$

where α is hyperparameter. By minimizing the InfoNCE loss, AVHash can learn discriminative hash codes, thereby realizing efficient large-scale video retrievals.

Table 4: Different methods' mAP@100 values on ActivityNet and FCVID datasets with 32 and 64 bits.

Method/mAP/ Dataset	ActivityNet						FCVID					
	-20		-50		-all		-20		-50		-all	
	32 bits	64 bits	32 bits	64 bits	32 bits	64 bits	32 bits	64 bits	32 bits	64 bits	32 bits	64 bits
MCMSH [13]	0.1059	0.2149	0.1508	0.2846	0.1488	0.2849	0.2705	0.2963	0.3642	0.3930	0.3442	0.3723
BTH [25]	0.1835	0.2283	0.1845	0.2634	0.1845	0.2721	0.2762	0.3458	0.3083	0.3680	0.2987	0.3593
DKPH [23]	0.1668	0.2251	0.1648	0.2252	0.1528	0.2283	0.2937	0.3535	0.3235	0.3820	0.3256	0.3731
SSTH [48]	0.1636	0.2124	0.1765	0.2408	0.1727	0.2395	0.2017	0.2671	0.2428	0.3057	0.2275	0.3037
ConMH [42]	0.2536	0.2973	0.2887	0.3268	0.2974	0.3312	0.3784	0.4020	0.4040	0.4304	0.3899	0.4232
DSVH [2]	0.1463	0.2200	0.2003	0.2903	0.2056	0.2979	0.2205	0.2900	0.2560	0.3373	0.2616	0.3683
SRH [12]	0.2659	0.3116	0.3538	0.4574	0.3205	0.4266	0.3511	0.4768	0.3102	0.4708	0.3699	0.4783
AVH [41]	0.3320	0.4203	0.3756	0.4621	0.3576	0.4722	0.3273	0.4580	0.3227	0.4745	0.3318	0.4818
AVHash (Ours)	**0.9559**	**0.9731**	**0.9392**	**0.9440**	**0.8530**	**0.8676**	**0.9592**	**0.9622**	**0.9507**	**0.9566**	**0.9208**	**0.9213**

Table 5: The mAP@100 values of AVHash with different components on ActivityNet and FCVID datasets with 32 and 64 bits.

Method/mAP/ Dataset	ActivityNet						FCVID					
	-20		-50		-all		-20		-50		-all	
	32 bits	64 bits	32 bits	64 bits	32 bits	64 bits	32 bits	64 bits	32 bits	64 bits	32 bits	64 bits
audio	0.6546	0.6709	0.4071	0.4135	0.2113	0.2175	0.7819	0.7863	0.6495	0.6551	0.3956	0.4007
visual	0.9372	0.9474	0.9164	0.9370	0.8209	0.8443	0.9184	0.9246	0.9308	0.9361	0.9038	0.9076
audio+visual	**0.9559**	**0.9731**	**0.9392**	**0.9440**	**0.8530**	**0.8676**	**0.9592**	**0.9622**	**0.9507**	**0.9566**	**0.9208**	**0.9213**

3.3 Training and Inference

The specific configurations of AVHash's architecture is provided in Table 1, and AVHash is implemented and trained (from Layer-0 to Layer-6 in Table 1) via pytorch.

Given a new video $\mathbf{v}_{oos}$, we can use the trained AVHash model to compute its real-valued binary-like vector:

$$\mathbf{g}_{oos} = \mathbf{AVHash}(\mathbf{v}_{oos}) \tag{24}$$

first and then binarize it into the final hash code:

$$\mathbf{b}_{oos} = \mathbf{Sign}(\mathbf{g}_{oos}). \tag{25}$$

4 EXPERIMENTS

4.1 Datasets

We assess AVHash using two public video datasets: ActivityNet[2] [15] and FCVID[3] [18]. ActivityNet comprises 14,948 videos, which are categorized into 200 classes. From these, we randomly selected 20 and 50 categories to create two smaller datasets, named ActNet-20 and ActNet-50, respectively. The complete dataset is referred to as ActNet-all. Consequently, we have three ActNet datasets for our experiments. Likewise, FCVID includes 91,016 videos classified into 239 categories, from which we formed three variants: FCVID-20, FCVID-50, and FCVID-all.

[2]http://activity-net.org/
[3]https://fvl.fudan.edu.cn/dataset/fcvid/list.htm

Each dataset is split into three partitions: training, validation, and testing sets. Their statistics are presented in Table 2.

4.2 Metrics and Competitors

We evaluate the retrieval performance of different approaches using the most widely-adopted metric, mean Average Precision at top-K results (mAP@K) [5, 12, 41, 42]. For our study, we set K to 100.

When conducting video retrieval experiments, we designate the testing samples as the query set, with the validation and training sets serving as the video database. Additionally, we order the retrieved results by their Hamming distances from the queries and record the performances of different methods with code lengths of 32 as well as 64 bits.

Regarding baseline approaches, we include SSTH [48], BTH [25], DKPH [23], MCMSH [13], ConMH [42], SRH [12], AVH [41], and DSVH [2], chosen for their competitive retrieval performances. The first five methods are unsupervised, while the last three are supervised.

4.3 Implementation Details

Image processing. Consistent with SSTH [48], BTH [25], and ConMH [42], we uniformly sample 25 frames from each video in the ActivityNet and FCVID datasets. Each image is resized to 336×336×3 and segmented into 14×14 patches. These images are then processed using the visual branch of CLIP, which features a depth of 24 layers, 16 multi-head attentions, and a hidden size of

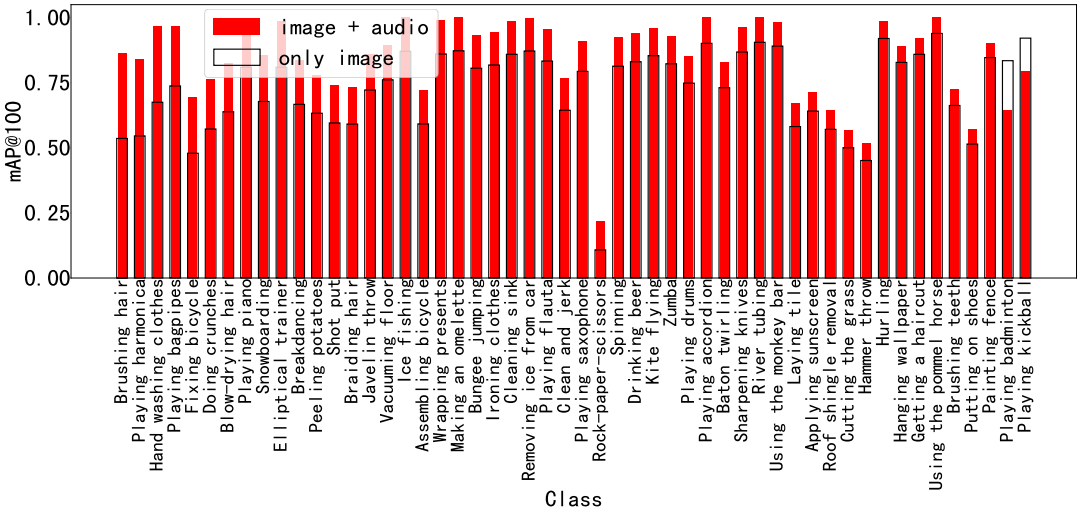


Figure 2: The mAP@100 values of each class in the ActNet-all dataset (the illustration selects partial classes). The red bars indicate the results of AVHash with audio and visual signals, while the grey bars indicate the results of AVHash only with visual signals.

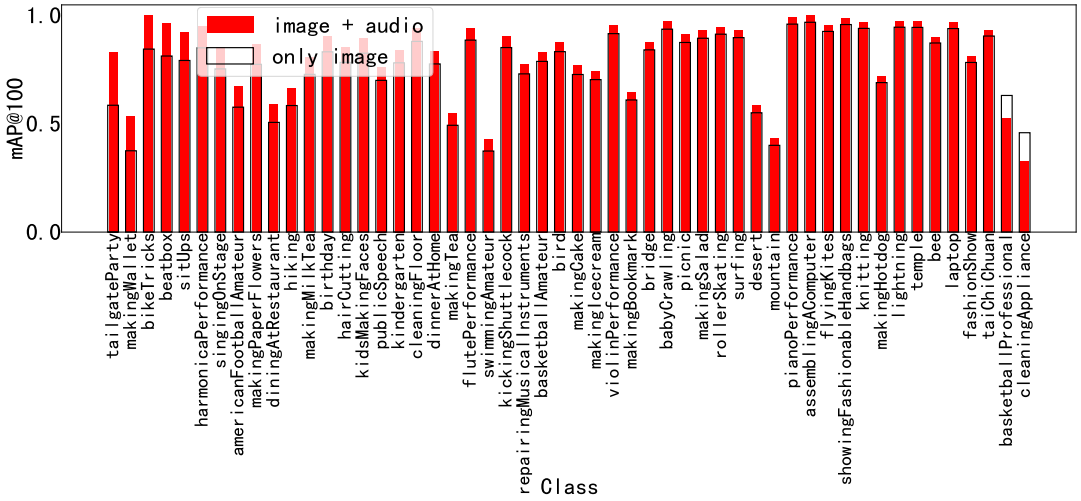


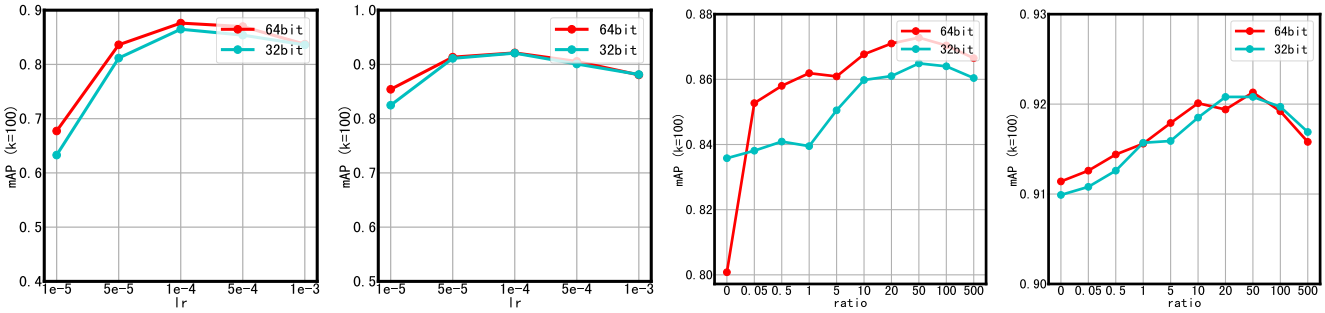
Figure 3: The mAP@100 values of each class in the FCVID-all dataset (the illustration selects partial classes). The red bars indicate the results of AVHash with audio and visual signals, while the grey bars indicate the results of AVHash only with visual signals.

1024, resulting in high-level image representations within a 1024-dimensional feature space. These features are subsequently transformed to 768-dimensional vectors via a fully connected layer.

Audio processing. We also evenly divide each audio track from the ActivityNet and FCVID datasets into 25 segments, corresponding to the 25 frames. These audio segments are then processed using AST [10], which is characterized by an embedding dimension of 768, 12 layers, and 12 heads, generating high-level audio representations within a 768-dimensional feature space.

Network Architecture. Our architecture comprises three primary components: the Audio Encoder, the Image Encoder, and the Multi-modal Encoder. Each component employs a Transformer encoder architecture, configured with a single layer, 12 multi-head attentions, and a hidden layer size of 768. Notably, we limit each component to one layer to ensure that our AVHash network, which has 28.4M learnable parameters, remains comparable in size to competitors such as AVH [41] with 22M parameters and DSVH [2] with 63M parameters, as illustrated in Table 3.

Model training. During the training phase, we employ the Adam optimizer [20] for gradient descent, setting the learning rate at 10^{-4} and the batch size at 128. In accordance with Eq. (20), we use four negative samples (n_o=4) and a temperature parameter (τ) of 0.1. Our AVHash model is implemented using PyTorch on an Nvidia Tesla V100 GPU (32GB). Additional details is available in our code release on Github website: https://github.com/iFamilyi/AVHash. For comparative purposes, the source codes and parameters of competing models are accessible on Github or provided courtesy of their authors.



(a) **ActivityNet**, learning rate (b) **FCVID**, learning rate (c) **ActivityNet**, loss ratio α (d) **FCVID**, loss ratio α

Figure 4: The mAP@100 values w.r.t. different settings of hyper-parameter lr and α on ActNet-all and FCVID-all datasets.

4.4 Results

Table 4 presents the mAP@K (K=100) values of all the methods in comparison on the ActivityNet (-20, -50, and -all) and FCVID (-20, -50, and -all) datasets. Our AVHash method consistently outperforms the others, including AVH [41], SRH [12], and ConMH [42], across all configurations. Specifically, AVHash achieves impressively high mAP scores — generally exceeding 0.9 — while the scores of competing methods seldom reach 0.5. For instance, at a code length of 32 bits, the mAP@100 scores for AVHash (0.9559 and 0.9731) are approximately 2.88 and 2.32 times higher than those of AVH (0.3320 and 0.4203) on the ActNet-20 and FCVID-20 datasets, respectively. These results strongly suggest the superior efficacy of our proposed approach to video hashing.

One may question why the performance gap between 32-bit and 64-bit configurations is marked for the baseline methods but negligible for our AVHash. This difference can be attributed to the architecture of AVHash (Fig. 1), where the only variation between the 32-bit and 64-bit models is in their final fully connected layer (768×32 vs. 768×64), with both configurations having a similar scale of learnable parameters.

4.5 Ablation Studies

We would like to explore the impact of visual and audio signals on the performance of AVHash in video retrieval tasks. This curiosity leads us to conduct comparative experiments: AVHash (with both signals) vs. AVHash with visual input only vs. AVHash with audio input only, denoted as "audio+visual", "visual", and "audio" respectively, as detailed in Table 5. The findings reveal a hierarchy: "audio+visual" outperforms the others, followed by "visual", and "audio" ranks last. It is noteworthy that while visual signals alone could enable AVHash to achieve commendably high mAP@100 scores, the addition of audio signals would further improve AVHash's performance, which affirmatively highlights the positive contribution of audio.

Moreover, we selectively analyze subsets of ActNet-all and FCVID-all, illustrating the mAP@100 values for AVHash under both "audio+visual" and "visual" settings in Fig. 2&3. These figures demonstrate that audio signals generally enhance video comprehension, as indicated by the predominance of red bars over white ones. However, there are instances where audio waveforms might confuse the interpretation of frame images, suggesting an opportunity for the development of more sophisticated models.

4.6 Parameter Sensitivity

Two critical parameters, the learning rate lr and regularization factor α (Eq. (23)), significantly influence our model's performance. Fig. 4 compiles and illustrates their impacts upon mAP@100 scores. Insights from Fig. 4(a)&4(b) reveal that an lr of 1e-4 enables AVHash to attain high performance on both the ActivityNet and FCVID datasets. Likewise, analysis from Fig. 4(c)&4(d) suggests setting α to 50 for optimal results, regardless of whether the dataset is ActivityNet or FCVID. Consequently, we adopt lr=1e-4 and α=50 as the default settings for our AVHash method.

4.7 Convergence Curves

Fig. 5 displays the mAP@100 scores and loss curves for AVHash on both datasets, utilizing 32-bit and 64-bit codes. Each subplot indicates the iteration number on the x-axis, while the left and right y-axes, in blue and red respectively, denote the mAP@100 scores and normalized values of the objective function[4]. The plots clearly demonstrate that AVHash achieves convergence within 300 iterations, thereby confirming the efficacy of AVHash's training process.

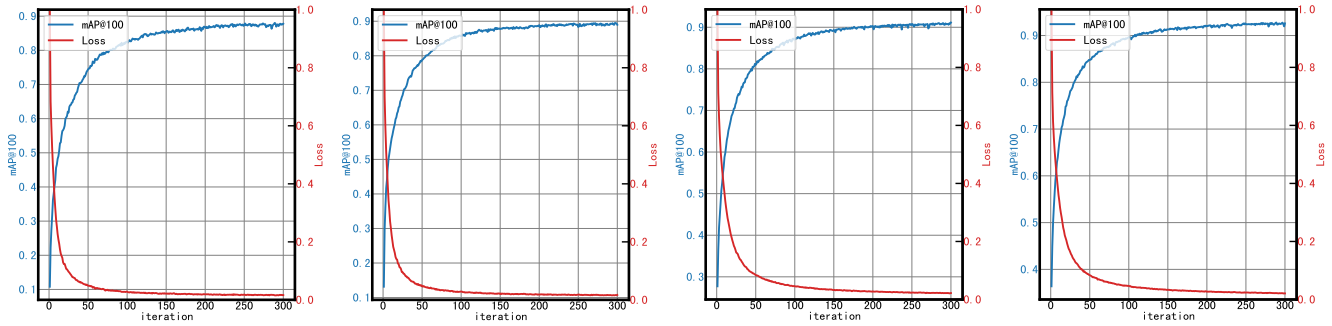
4.8 t-SNE Visualizations

We selected 10 classes at random, each with 50 samples from the ActNet-all and FCVID-all datasets, and projected them into a 2D space using t-SNE [38], as illustrated in Fig. 6&7. Notably, our AVHash method clusters these videos more distinctly than both the competitive supervised methods AVH/DSVH and the unsupervised approach MCMSH. This demonstrates AVHash's superior capabilities in representation learning.

5 CONCLUSIONS

This paper sets itself apart from previous studies on video hashing by utilizing both audio and visual components of videos in a tri-level Transformer architecture, named AVHash, for creating binary embeddings of videos. Specifically, AVHash first maps separate audio and video signals to a shared latent semantic space, before projecting them onto the final video space. It then establishes a contrastive loss in the video space, along with a regularization constraint for aligning audio and visual signals in the common latent semantic space, to guide model training. Extensive experiments conducted

[4]The loss for each iteration is normalized via dividing it by the loss of the first iteration.



(a) **ActivityNet**, 32bits (b) **ActivityNet**, 64bits (c) **FCVID**, 32bits (d) **FCVID**, 64bits

Figure 5: The mAP@100 and loss curves w.r.t. training iterations on ActNet-all and FCVID-all datasets.

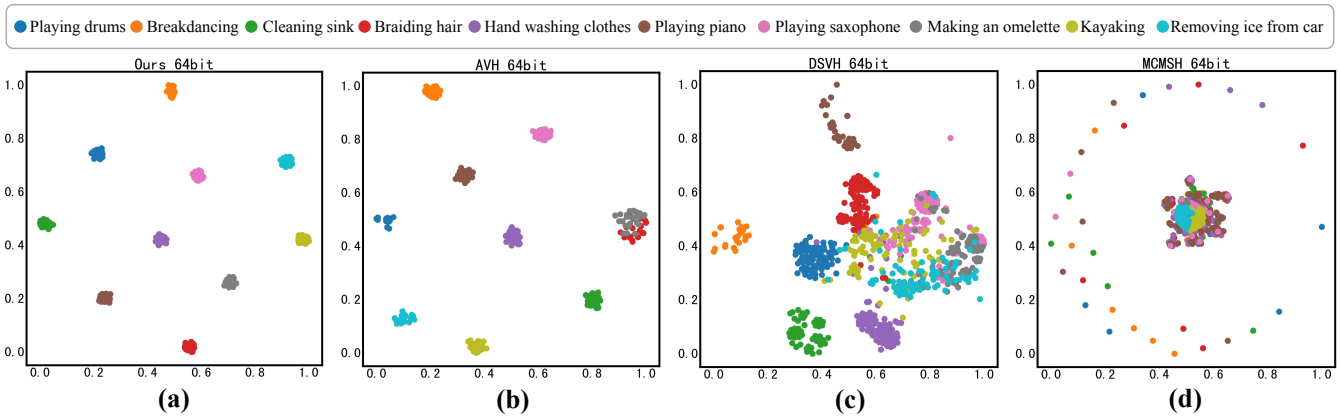


Figure 6: t-SNE visualizations of AVHash and 3 most competitive methods on ActNet-all dataset (10 classes are randomly selected).

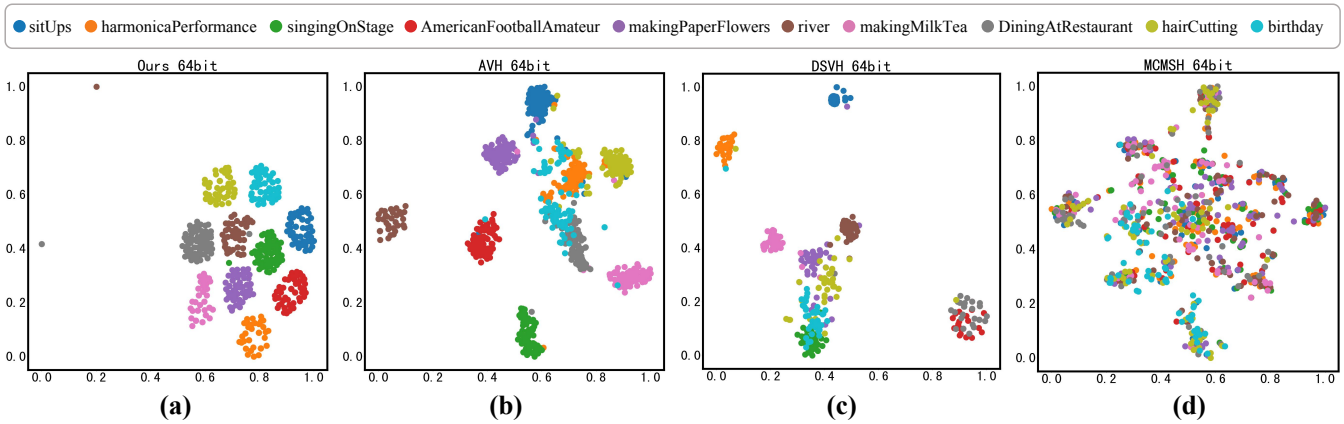


Figure 7: t-SNE visualizations of AVHash and 3 most competitive methods on FCVID-all dataset (10 classes are randomly selected).

on two widely used large video datasets demonstrate that AVHash significantly outperforms existing video hashing techniques in video retrieval tasks. Our findings indicate that, while a high mAP score for video retrieval could be achieved using visual signals alone in video hashing, incorporating audio signals effectively would further improve the system's performance.

ACKNOWLEDGMENTS

This work is supported in part by National Natural Science Foundation of China (NSFC, Grant No. 62372054, 62006005) and National Key Research and Development Program of China (Grant No. 2022YFC3302200). It is also supported in part by Super Computing Platform of Beijing University of Posts and Telecommunications (BUPT).

REFERENCES

[1] Xiaojun Chang, Xiaodan Liang, Yan Yan, and Liqiang Nie. 2020. Guest editorial: Image/video understanding and analysis. *Pattern Recognit. Lett.* 130 (2020), 1–3.

[2] Hanqing Chen, Chunyan Hu, Feifei Lee, Chaowei Lin, Wei Yao, Lu Chen, and Qiu Chen. 2021. A Supervised Video Hashing Method Based on a Deep 3D Convolutional Neural Network for Large-Scale Video Retrieval. *Sensors* 21, 9 (2021), 1–15.

[3] Yong Chen, Yuqing Hou, Shu Leng, Qing Zhang, Zhouchen Lin, and Dell Zhang. 2021. Long-Tail Hashing. In *SIGIR*. ACM, 1328–1338.

[4] Yong Chen, Zhibao Tian, Hui Zhang, Jun Wang, and Dell Zhang. 2020. Strongly Constrained Discrete Hashing. *IEEE Trans. Image Process.* 29 (2020), 3596–3611.

[5] Yong Chen, Hui Zhang, Zhibao Tian, Jun Wang, Dell Zhang, and Xuelong Li. 2022. Enhanced Discrete Multi-Modal Hashing: More Constraints Yet Less Time to Learn. *IEEE Trans. Knowl. Data Eng.* 34, 3 (2022), 1177–1190.

[6] Yong Chen, Hui Zhang, Zhibao Tian, Jun Wang, Dell Zhang, and Xuelong Li. 2023. Enhanced Discrete Multi-modal Hashing: More Constraints yet Less Time to Learn (Extended Abstract). In *ICDE*. IEEE, 3857–3858.

[7] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *CoRR* abs/1412.3555 (2014).

[8] Leonardo G. Cohen, Pablo Celnik, Alvaro Pascual-Leone, Brian Corwell, Lala Faiz, James Dambrosia, Manabu Honda, Norihiro Sadato, Christian Gerloff, M. Dolores Catala, and Mark Hallett. 1997. Functional relevance of cross-modal plasticity in blind humans. *Nature* 389 (1997), 180–183.

[9] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni. 2004. Locality-sensitive hashing scheme based on p-stable distributions. In *SCG*. 253–262.

[10] Yuan Gong, Yu-An Chung, and James R. Glass. 2021. AST: Audio Spectrogram Transformer. In *Interspeech*. 571–575.

[11] Yunchao Gong, Svetlana Lazebnik, Albert Gordo, and Florent Perronnin. 2013. Iterative Quantization: A Procrustean Approach to Learning Binary Codes for Large-Scale Image Retrieval. *IEEE Trans. Pattern Anal. Mach. Intell.* 35, 12 (2013), 2916–2929.

[12] Yun Gu, Chao Ma, and Jie Yang. 2016. Supervised Recurrent Hashing for Large Scale Video Retrieval. In *ACM Multimedia*. 272–276.

[13] Yanbin Hao, Jingru Duan, Hao Zhang, Bin Zhu, Pengyuan Zhou, and Xiangnan He. 2022. Unsupervised Video Hashing with Multi-granularity Contextualization and Multi-structure Preservation. In *ACM Multimedia*. 3754–3763.

[14] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In *CVPR*. 9726–9735.

[15] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. 2015. ActivityNet: A large-scale video benchmark for human activity understanding. In *CVPR*. 961–970.

[16] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (1997), 1735–1780.

[17] Yu Huang, Chenzhuang Du, Zihui Xue, Xuanyao Chen, Hang Zhao, and Longbo Huang. 2021. What Makes Multi-Modal Learning Better than Single (Provably). In *NeurIPS*. 10944–10956.

[18] Yu-Gang Jiang, Zuxuan Wu, Jun Wang, Xiangyang Xue, and Shih-Fu Chang. 2018. Exploiting Feature and Class Relationships in Video Categorization with Regularized Deep Neural Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* 40, 2 (2018), 352–364.

[19] Wang-Cheng Kang, Wu-Jun Li, and Zhi-Hua Zhou. 2016. Column Sampling Based Discrete Supervised Hashing. In *AAAI*. 1230–1236.

[20] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*. 1–15.

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *NeurIPS*. 1106–1114.

[22] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.

[23] Pandeng Li, Hongtao Xie, Jiannan Ge, Lei Zhang, Shaobo Min, and Yongdong Zhang. 2022. Dual-Stream Knowledge-Preserving Hashing for Unsupervised Video Retrieval. In *ECCV*, Vol. 13674. 181–197.

[24] Qi Li, Zhenan Sun, Ran He, and Tieniu Tan. 2017. Deep Supervised Discrete Hashing. In *NeurIPS*. 2482–2491.

[25] Shuyan Li, Xiu Li, Jiwen Lu, and Jie Zhou. 2021. Self-Supervised Video Hashing via Bidirectional Transformers. In *CVPR*. 13549–13558.

[26] Shuyan Li, Xiu Li, Jiwen Lu, and Jie Zhou. 2022. Structure-Adaptive Neighborhood Preserving Hashing for Scalable Video Search. *IEEE Trans. Circuits Syst. Video Technol.* 32, 4 (2022), 2441–2454.

[27] Chen Lin. 2023. Exploring the Impact of Short Videos on Society and Culture: An Analysis of Social Dynamics and Cultural Expression. 6, 3 (2023), 115–118.

[28] Venice Erin Liong, Jiwen Lu, Yap-Peng Tan, and Jie Zhou. 2017. Deep Video Hashing. *IEEE Trans. Multim.* 19, 6 (2017), 1209–1219.

[29] Luke Melas-Kyriazi. 2021. Do You Even Need Attention? A Stack of Feed-Forward Layers Does Surprisingly Well on ImageNet. *CoRR* abs/2105.02723 (2021).

[30] Min Meng, Jiaxuan Sun, Jigang Liu, Jun Yu, and Jigang Wu. 2024. Semantic Disentanglement Adversarial Hashing for Cross-Modal Retrieval. *IEEE Trans. Circuits Syst. Video Technol.* 34, 3 (2024), 1914–1926.

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

[32] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *ICML*, Vol. 139. 8748–8763.

[33] Fumin Shen, Chunhua Shen, Wei Liu, and Heng Tao Shen. 2015. Supervised Discrete Hashing. In *CVPR*. 37–45.

[34] Xiaobo Shen, Yue Zhou, Yun-Hao Yuan, Xichen Yang, Long Lan, and Yuhui Zheng. 2023. Contrastive Transformer Hashing for Compact Video Representation. *IEEE Trans. Image Process.* 32 (2023), 5992–6003.

[35] Jingkuan Song, Yi Yang, Zi Huang, Heng Tao Shen, and Richang Hong. 2011. Multiple feature hashing for real-time large scale near-duplicate video retrieval. In *ACM Multimedia*. 423–432.

[36] Ilya O. Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. 2021. MLP-Mixer: An all-MLP Architecture for Vision. In *NeurIPS*. 24261–24272.

[37] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation Learning with Contrastive Predictive Coding. *CoRR* abs/1807.03748 (2018).

[38] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing High-Dimensional Data Using t-SNE. *Journal of Machine Learning Research* 9 (2008), 2579–2605.

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NeurIPS*. 5998–6008.

[40] Liangdao Wang, Yan Pan, Cong Liu, Hanjiang Lai, Jian Yin, and Ye Liu. 2023. Deep Hashing with Minimal-Distance-Separated Hash Centers. In *CVPR*. 23455–23464.

[41] Yingxin Wang, Xiushan Nie, Yang Shi, Xin Zhou, and Yilong Yin. 2021. Attention-Based Video Hashing for Large-Scale Video Retrieval. *IEEE Trans. Cogn. Dev. Syst.* 13, 3 (2021), 491–502.

[42] Yuting Wang, Jinpeng Wang, Bin Chen, Ziyun Zeng, and Shu-Tao Xia. 2023. Contrastive Masked Autoencoders for Self-Supervised Video Hashing. In *AAAI*. 2733–2741.

[43] Gengshen Wu, Jungong Han, Yuchen Guo, Li Liu, Guiguang Ding, Qiang Ni, and Ling Shao. 2019. Unsupervised Deep Video Hashing via Balanced Code for Large-Scale Video Retrieval. *IEEE Trans. Image Process.* 28, 4 (2019), 1993–2007.

[44] Ke Xia, Yuqing Ma, Xianglong Liu, Yadong Mu, and Li Liu. 2017. Temporal Binary Coding for Large-Scale Video Search. In *ACM Multimedia*. 333–341.

[45] Zhongwen Xu. 2017. *Large-scale video analysis and understanding*. Ph.D. Dissertation. University of Technology Sydney, Australia.

[46] Guangnan Ye, Dong Liu, Jun Wang, and Shih-Fu Chang. 2013. Large-Scale Video Hashing via Structure Learning. In *ICCV*. 2272–2279.

[47] Li Yuan, Tao Wang, Xiaopeng Zhang, Francis E. H. Tay, Zequn Jie, Wei Liu, and Jiashi Feng. 2020. Central Similarity Quantization for Efficient Image and Video Retrieval. In *CVPR*. 3080–3089.

[48] Hanwang Zhang, Meng Wang, Richang Hong, and Tat-Seng Chua. 2016. Play and Rewind: Optimizing Binary Representations of Videos by Self-Supervised Temporal Hashing. In *ACM Multimedia*. 781–790.

[49] Yaoxin Zhuo, Yikang Li, Jenhao Hsiao, Chiuman Ho, and Baoxin Li. 2022. CLIP4Hashing: Unsupervised Deep Hashing for Cross-Modal Video-Text Retrieval. In *ICMR*. 158–166.