SUPERVISED DEEP HASHING FOR EFFICIENT AUDIO EVENT RETRIEVAL

Arindam Jati[1*], *Dimitra Emmanouilidou*[2]

[1]University of Southern California, Los Angeles, CA, USA
[2]Audio and Acoustics Research Group, Microsoft Research, Redmond, WA, USA

ABSTRACT

Efficient retrieval of audio events can facilitate real-time implementation of numerous query and search-based systems. This work investigates the potency of different hashing techniques for efficient audio event retrieval. Multiple state-of-the-art weak audio embeddings are employed for this purpose. The performance of four classical unsupervised hashing algorithms is explored as part of off-the-shelf analysis. Then, we propose a partially supervised deep hashing framework that transforms the weak embeddings into a low-dimensional space while optimizing for efficient hash codes. The model uses only a fraction of the available labels and is shown here to significantly improve the retrieval accuracy on two widely employed audio event datasets. The extensive analysis and comparison between supervised and unsupervised hashing methods presented here, give insights on the quantizability of audio embeddings. This work provides a first look in efficient audio event retrieval systems and hopes to set baselines for future research.

Index Terms— Audio events, efficient retrieval, hashing, quantization, deep neural network.

1. INTRODUCTION

Audio Event Classification (AEC) is defined as the inherent ability of machines to assign a semantic label to a given audio segment [1, 2, 3]. It is an increasingly popular research area due to its numerous applications in audio content understanding [4], surveillance [5, 6], health monitoring [7], self-driving vehicles, accessibility devices, and advanced gaming systems. Audio events are generally given annotation labels by humans following some pre-defined ontology [2], and most of the state-of-the-art AEC systems [3, 8] utilize those semantic descriptors for supervised learning. In spite of multiple efforts in learning better and more robust representations (*embeddings*) of audio events [3], there is limited work for their efficient retrieval. Some classical audio retrieval algorithms have been proposed in [9, 10], but no work can be found on *efficient* retrieval techniques with modern audio embeddings, and detailed evaluation on publicly available datasets. Fast retrieval can facilitate near-real-time similarity search between a query sound and a database with millions of audio events, which, in turn, can effectuate the implementation of the aforementioned systems. The work presented here addresses the problem of efficient retrieval of audio events.

Efficient similarity search and retrieval can be performed through Approximate Nearest Neighbor (ANN) search which encompasses quantization [11] and hashing [12] algorithms. As the audio domain lacks relevant literature, most of the related work is found in computer vision. Hashing and quantization methods attempt to transform high-dimensional image embeddings into compact binary codes that can then be utilized for efficient image retrieval [13]. ANN algorithms can be broadly classified into two groups: unsupervised and supervised. Unsupervised methods include classical hashing and quantization algorithms like Locality Sensitive Hashing (LSH) [14] and Product Quantization (PQ) [11]. Unsupervised algorithms do *not* leverage human annotated data labels and thus might suffer from the semantic gap dilemma [15]. In contrast, supervised algorithms like [16, 17] utilize available semantic labels, and employ Deep Neural Networks (DNNs) to simultaneously learn audio embeddings and their hash codes. These algorithms try to preserve the data patterns in the hash codes, which can be helpful for accurate retrieval. This can be further expedited by incorporating an extra objective for learning the feature patterns. For example, Yue Cao *et. al.* [13] proposed a Deep Quantization Network (DQN) for efficient image retrieval in a multi-task learning framework where image similarity is learnt through a pairwise cosine loss and the hash codes are generated via Product Quantization.

This work formulates the efficient audio event retrieval problem and proposes a deep supervised ANN algorithm suitable for the task. The novelties and contributions of this paper are as follows:

- We address and formulate the efficient audio event retrieval problem using deep audio embeddings. To the best of our knowledge, this work is the first of its kind.
- We employ multiple state-of-the-art audio embeddings for retrieval, and compare their performances.
- We apply classical unsupervised hashing algorithms and draw a detailed comparison.
- By consolidating existing methods and audio domain knowledge, we propose a supervised deep hashing model suitable for quantizing audio events with minimal training.
- We show that the proposed supervised method has significant benefits in retrieval performance across multiple datasets, even when using only a fraction of the available training instances.

Experiments and results discussed here could serve as baseline for audio event retrieval tasks, and encourage future research in the field.

2. HASHING OF AUDIO EVENTS

The proposed supervised hashing is inspired by the DQN model and its recent success in computer vision applications [13]. We incorporate audio domain knowledge, and create a deep learning framework which consists of three sub-modules:

1. An audio event embedding generator that produces a fixed low-dimensional representation.
2. An (unsupervised) hashing module that quantizes a given embedding, and is characterized by a certain quantization error.
3. A supervised similarity preserver which exploits available audio labels, and increases similarity between the learned audio embeddings of the same class.

*The work was done while Arindam Jati was an intern at Microsoft Research Labs, Redmond, WA, USA.

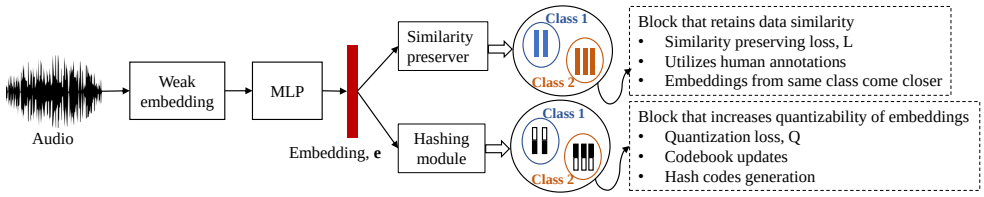

Fig. 1: Overview of the employed model for deep audio event hashing.

The overall model is trained in a multi-task learning environment to reduce the quantization error as well as optimize the similarity between the learned embeddings. Figure 1 depicts the overview of the proposed system. The individual modules are described below.

2.1. Audio event embeddings

Assume a set of audio samples $\mathcal{S} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where y_i denotes the semantic class label associated with audio $\mathbf{x}_i$. Generally, DNN audio event classification models are trained in a supervised fashion by optimizing the cross entropy loss across all audio classes. An audio embedding $\mathbf{f}(\mathbf{x}_i)$ is defined as the output of an intermediate layer of a pre-trained DNN model, and denotes a nonlinear transformation of $\mathbf{x}_i$. Audio event embeddings are known to capture the inherent semantic information associated with that particular audio sample. In this work, we will exploit weak-supervision pre-trained models since they have shown promising results in the recent literature [3, 8] and constitute state-of-the-art feature representations. We will denote these embeddings as weak embeddings, $\mathbf{e}_{\text{weak}} = \mathbf{f}(\mathbf{x})$, since they are trained with weak labels on *out-of-domain* data (see Section 3.2.1). The weak embeddings are used as features for the proposed supervised model to learn more quantizable embeddings.

To learn stronger audio embeddings by leveraging the human annotations available in the *in-domain* dataset (see Section 3.1), another stage of nonlinear transformation is applied via a Multi-layer Perceptron (MLP) parameterized by $\mathbf{\Theta}$:

$$\mathbf{e} = \mathbf{g}(\mathbf{e}_{\text{weak}}; \mathbf{\Theta}) \tag{1}$$

The two modules of the proposed system presented below operate on embedding $\mathbf{e}$.

2.2. Unsupervised hashing module

This module employs state-of-the-art Product Quantization (PQ) algorithm [11], a hashing method that is based on the general idea of Vector Quantization (VQ) [18].

2.2.1. *Vector Quantization (VQ)*

VQ maps a fixed dimensional vector, $\mathbf{e} \in \mathbb{R}^D$ into a *codeword* or *centroid*, $\mathbf{q}(\mathbf{e})$ which belongs to a finite set called the *codebook*, $\mathcal{C}$. More formally, if $\mathcal{I} = 1, 2, \ldots, K$ is a finite index set:

$$\mathbf{e} \rightarrow \mathbf{q}(\mathbf{e}) \in \mathcal{C} = \{\mathbf{c}_i : \forall i \in \mathcal{I}\} \tag{2}$$

The quantization error, $\mathcal{E}$ is defined as:

$$\mathcal{E} = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{e}_i - \mathbf{q}(\mathbf{e}_i)\|^2 \tag{3}$$

Under unconstrained conditions on the codebook, $\mathcal{C}$, minimizing this error leads to optimizing the K-Means clustering objective [11]:

$$\text{minimize } \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{e}_i - \mathbf{C}\mathbf{h}_i\|^2 \text{ s.t. } \|\mathbf{h}_i\|_0 = 1, \mathbf{h}_i \in \{0,1\}^K \tag{4}$$

Here, $\mathbf{C}$ is the matrix whose columns represent the codewords:

$$\mathbf{C}_{D\times K} = [\mathbf{c}_1|\mathbf{c}_2|\ldots|\mathbf{c}_K] \tag{5}$$

A K-codeword codebook produces $B = \log_2 K$ bits hash codes.

2.2.2. *Product Quantization (PQ)*

Note that VQ needs to keep a codebook of size 2^B for B bits codes. This restricts the number of bits it can accommodate, and for large value of B, VQ becomes intractable [11]. PQ was proposed to circumvent this issue, especially when an exponentially large number of codewords is desired. PQ decomposes embedding $\mathbf{e}$ into M lower dimensional subspaces, and then applies VQ independently on each subspace. If $\mathbf{e}_j \in \mathbb{R}^{D/M} = \mathbb{R}^{D^*}, \forall j = 1, 2, \cdots, M$, then:

$$\mathbf{e} = \left[\mathbf{e}_1^T|\mathbf{e}_2^T|\ldots|\mathbf{e}_M^T\right]^T, \tag{6}$$

and the optimization objective becomes:

$$\text{minimize} \quad Q = \frac{1}{N}\sum_{j=1}^{M}\sum_{i=1}^{N} \|\mathbf{e}_{ij} - \mathbf{C}_j\mathbf{h}_{ij}\|^2 \tag{7}$$

$$\text{subject to} \quad \|\mathbf{h}_{ij}\|_0 = 1, \mathbf{h}_{ij} \in \{0,1\}^{K^*} \tag{8}$$

Here, $\mathbf{C}_j$ contains K^* codewords in the j^{th} subspace:

$$[\mathbf{C}_j]_{D^*\times K^*} = [\mathbf{c}_{j1}|\mathbf{c}_{j2}|\ldots|\mathbf{c}_{jK^*}] \tag{9}$$

Note that the effective codebook, $\mathcal{C}$, becomes the cartesian product of the codebooks of all subspaces: $\mathcal{C} = \mathcal{C}_1 \times \mathcal{C}_2 \cdots \times \mathcal{C}_M$. Therefore, the effective codebook size becomes $K = (K^*)^M$, but we need to store only K^*M codewords. In this case, the codebook produces $B = M \log_2 K^*$ bits codes, for K^* codewords from M subspaces.

2.3. Supervised similarity preserver module

We seek to improve the distinctive characteristics of the learned hash codes by leveraging available audio labels [13]. The similarity preserver loss function increases the cosine similarity between two embeddings of the same class, and decreases similarity for embeddings of different classes. More formally, the objective is:

$$L = \sum_{t_{ij}\in\mathcal{T}} \left(t_{ij} - \frac{\langle \mathbf{e}_i, \mathbf{e}_j\rangle}{\|\mathbf{e}_i\|\|\mathbf{e}_j\|}\right)^2 \tag{10}$$

where $t_{ij} = +1$ if $y_i = y_j$, otherwise $t_{ij} = -1$.

2.4. Multi-task training

Equation 7 can be compactly written as:

$$\text{minimize} \quad Q = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{e}_i - \widetilde{\mathbf{C}}\widetilde{\mathbf{h}}_i\|^2 \tag{11}$$

where, $\widetilde{\mathbf{C}}$ is a diagonal block matrix containing all $\{\mathbf{C}_j\}_{j=1}^{M}$, $\widetilde{\mathbf{h}}_i = \left[\mathbf{h}_1^T|\mathbf{h}_2^T|\ldots\mathbf{h}_M^T\right]^T$. The overall minimization objective becomes:

$$\min_{\mathbf{\Theta}, \widetilde{\mathbf{C}}, \{\widetilde{\mathbf{h}}_i\}_{i=1}^{N}} L + \lambda Q \tag{12}$$

where, λ is a weight parameter on the quantization error[1]. This loss function is differentiable and can be minimized via minibatch SGD. We perform PQ at every epoch to update $\widetilde{\mathbf{C}}$ and $\{\widetilde{\mathbf{h}}_i\}_{i=1}^{N}$, while we update $\mathbf{\Theta}$ for each minibatch. Note, that the MLP weights and the codebook are randomly initialized before training.

[1] $\lambda \in [10^{-5}, 1]$ is chosen through cross-validation inside training set.

Table 1: Performance (% mAP) of audio event retrieval algorithms on DCASE dataset using VGGish and TLWeak weak embeddings

		TLWeak (1024 dimensional)					VGGish (128 dimensional)				
Type of training	Algorithm[4]	8 bits	16 bits	24 bits	32 bits	64 bits	8 bits	16 bits	24 bits	32 bits	64 bits
Unsupervised (full database)	SH	8.14	10.97	11.93	12.84	13.31	9.23	11.34	12.18	12.85	12.90
	ITQ	8.34	12.03	14.17	15.52	17.62	10.10	12.18	14.25	14.61	15.64
	AGH	9.40	13.03	15.13	15.35	16.49	13.37	15.04	15.52	16.34	15.41
	PQ	15.06	16.15	16.30	16.39	16.36	16.12	16.34	16.23	16.24	15.65
Supervised (10% of database)	Audio DQN (proposed)	**39.07**	**44.24**	**45.50**	**45.77**	**46.83**	33.84	38.93	39.68	40.31	41.43

Table 2: Performance (% mAP) of audio event retrieval algorithms on ESC-50 dataset using VGGish and TLWeak weak embeddings

		TLWeak (1024 dimensional)					VGGish (128 dimensional)				
Type of training	Algorithm[4]	6 bits	12 bits	18 bits	24 bits	48 bits	6 bits	12 bits	18 bits	24 bits	48 bits
Unsupervised (full database)	SH	11.38	16.15	18.34	20.79	22.75	10.59	15.69	18.42	19.13	20.05
	ITQ	12.48	21.17	25.43	28.22	32.92	8.72	13.82	18.11	19.92	21.67
	AGH	13.46	25.43	30.72	33.27	34.98	13.68	21.15	24.47	24.42	25.38
	PQ	27.00	28.40	29.40	29.90	30.10	18.55	18.31	19.00	18.80	17.33
Supervised (12% of database)	Audio DQN (proposed)	**32.15**	**39.72**	**40.92**	**40.80**	**43.48**	21.25	24.91	25.50	25.18	26.73

2.5. Approximate Nearest Neighbor search (ANN)

We employ Asymmetric Quantizer Distance (AQD) for ANN search [11]. Given a query audio, we compute its embedding representation, $\mathbf{e}_{\text{query}}$, using equation 1. Then, an exhaustive search[2] is performed by computing AQD between $\mathbf{e}_{\text{query}}$ and all database samples, and finding the sample that gives minimum AQD:

$$\mathbf{x}_{\text{ANN}} = \underset{\mathbf{x}_i \in \mathcal{S}}{\operatorname{argmin}} \|\mathbf{e}_{\text{query}} - \widetilde{\mathbf{C}}\widetilde{\mathbf{h}}_i\|^2 \tag{13}$$

A perfect retrieval system would put all the positive *hits* at the top of the list ordered by AQD distance (see Section 3.3).

3. EXPERIMENTAL SETTING

3.1. Datasets

2018 DCASE challenge, Task-2 [20]: The dataset contains $\sim$ 9.5K training and $\sim$ 1.6K test audio files from 41 audio classes, annotated using the Google AudioSet Ontology [2]. We employ this dataset for detailed analysis and ablation studies of the proposed system.
ESC-50 [21]: To evaluate the generalization capability of the proposed system, we also employ ESC-50 dataset [21]. This dataset is much smaller comprising of 2000 audio samples from 50 classes. The performance is measured through five-fold cross-validation.

3.2. Models

3.2.1. Weakly supervised audio embeddings

We employ two state-of-the-art weak embeddings (Section 2.1): "VGGish" [3] and "TLWeak"[3] [8]. VGGish are 128-dimensional embeddings trained on $\sim$ 5 M hours of weakly labeled YouTube videos [3]. TLWeak are 1024-dimensional embeddings trained on class-balanced AudioSet dataset [2]. The compact representations of both types of embeddings make them suitable for transfer learning.

3.2.2. MLP model and PQ parameters

The proposed MLP model has three dense layers of $[512, 256, E]$ units, where E is the dimension of the output embedding $\mathbf{e}$ (equation 1). In this work $E = 2 \times B$ where B is the required number of bits, a parameter of the model. Similarly, we set the default number of codewords in each subspace to be $K^* = 256$ (see equation 9).

The number of subspaces M depends on the number of bits through $B = M \log_2 K^* = M \log_2 256 = 8 \times M$. For example: for a 32-bits code, we need $M = B/8 = 4$ subspaces. Therefore, the effective codebook size is $(K^*)^M = 256^4 \approx 4.3$B; but the actual required storage is $K^* \times M = 256 \times 4 = 1024$ codewords. If VQ was used instead of PQ, we would need to store the full 4.3B-word codebook, at which point the problem becomes intractable.

3.3. Evaluation metric

For evaluating the audio retrieval task, we adopt Mean Average Precision (mAP@R) metric where R is the number of retrieved items [13, 16]. Metric mAP varies $\in [0, 1]$, and is high when positive retrievals (*hits*) are at the top. Setting R equal to the elements in the entire database is common in retrieval literature [22]. It incorporates *hits* even at the end of the list but at the cost of a higher penalty. The average mAP performance is reported for all experiments, over 10 repetitions of random MLP model and codebook initialization.

4. RESULTS AND DISCUSSIONS

4.1. Comparison between different algorithms

The retrieval experiments incorporate three sets of samples: *train*, *test*, and *database*. The unsupervised algorithms do not require labels for training and are trained on the entire database. The proposed supervised audio DQN method assumes the availability of a fraction of the annotated database ($\sim$ 10%), which is used for training. The exact percentage may vary slightly, depending on rounding effects for each random sampling of the dataset. The train set is pooled randomly from the classes of the database via uniform sampling. The test set is the same as in the official instructions of the individual datasets. Note that the off-the-shelf unsupervised algorithms are applied on the weak embeddings, $\mathbf{e}_{\text{weak}}$; while the proposed supervised model works on the learned embedding, $\mathbf{e}$ (see Section 2.1).

Table 1 compares performance of the different unsupervised algorithms and the proposed audio DQN method, for an increasing number of hash code bits B (and increasing number of subspaces M, see Section 3.2.2). Here mAP is computed over all retrieved

[2]Non-exhaustive search can be implemented by standard algorithms like the inverted multi-index [19].

[3]We will use this term throughout the rest of the paper.

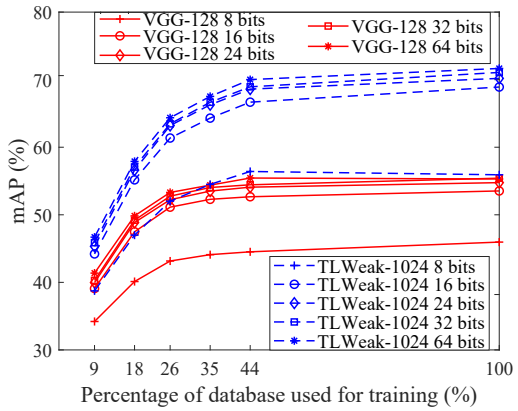

Fig. 2: Effect of training data size on retrieval performance for DCASE. mAP is computed on the full retrieved list (Section 3.3).

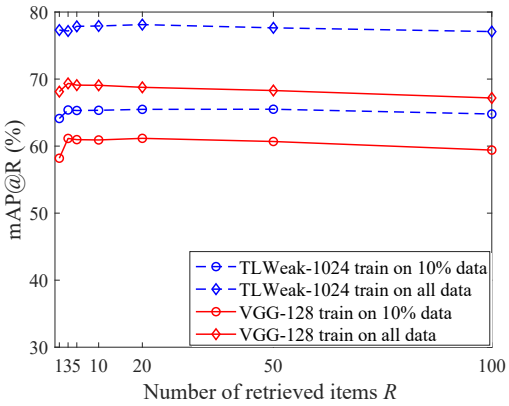

Fig. 3: MAP@R for increasing values of R, for DCASE.

and ranked items. An extensive set of classical unsupervised hashing algorithms was computed[4] [23], but due to space limitation we only report the top performing four methods in Table 1: Spectral Hashing (SH) [24], Iterative Quantization (ITQ) [25], Anchor Graph Hashing (AGH) [26], and Product Quantization (PQ) [11]. PQ outperforms all unsupervised methods for both VGGish and TLWeak embeddings.

We can see a significant boost in performance for the proposed audio DQN model compared to the unsupervised methods. We hypothesize the creation of more relevant embeddings for the task, as the similarity preserver module forces the embeddings to come closer for audio events of the same class (Section 2.3). This, in turn, assists the inherent PQ algorithm to find more compact clusters (or codewords). The further incorporation of quantization error in the loss function encourages the model to produce better hash codes which result in more accurate retrieval. A comparison of the weak audio embeddings shows that TLWeak outperforms VGGish for the task. This can be attributed to the higher dimension of the TLWeak embeddings that generally help retain more detailed feature information; such finer details can be absent in a smaller embedding space.

Table 2 draws a similar analysis on ESC-50 dataset. Among the unsupervised algorithms, interestingly, AGH outperforms PQ for a high number of bits. Also note that we employ only 12% of the database for supervised training in ESC-50. This gives only 4 samples per class, hence, in total $50 \times 4 = 200$ training samples. Due to the limited amount of training data, we reduced the value of K^* from the default 256 to 64 (see Section 3.2.2) so that enough training samples exist for the K-Means algorithm ($K^* < 200$). Even with this strictly limited labelled data, the proposed supervised method still outperforms the unsupervised algorithms. Comparing results between datasets, the lower performance achieved for ESC-50 can be attributed to the availability of fewer per-class examples.

4.2. Effect of training set size

In this experiment, we analyze the retrieval performance for a varying training set size (assume that we have additional labeled samples in the database, beyond $\sim 10\%$). Figure 2 shows gradual improvement in mAP as we increase the amount of labeled training data. The gap in performances between the two sets of features increases with training samples; this can be attributed to the better generalization of high-dimensional TLWeak embeddings in this case.

[4]Locality Sensitive Hashing (LSH), Kernelized LSH (KLSH), Binary Reconstructive Embedding (BRE), Density Sensitive Hashing (DSH) obtained even lower performance and are not reported here.

4.3. Ordering of the retrieved items

Figure 3 shows the variation of mAP@R for increasing R for DCASE dataset at 64 bits. Metric mAP@R corresponds to performance when looking only at the top R retrieved items in the list. Low R values are important in applications such as web-based search engines, where hits are desirable at the top of the retrieved list. The case of $R = 1$ is basically a hit or miss scenario. Notice the jump after $R = 1$, which can be expected as increased R values are more probable to contain hits. The fairly flat performance illustrates the strength of the proposed method even when we look at the lower elements in the retrieved list (high R). Note that the map@R for 10% of data will not remain flat beyond $R = 100$ items. For instance, the red line (solid line with circle marker) that has 60% mAP@R for $R = 100$, will go down to $\sim 40\%$ when R matches the entire database size as shown in Figure 2.

5. CONCLUSION AND FUTURE DIRECTIONS

This work provides a first look into retrieval and ranking of audio events, outlines potential challenges and illustrates its benefits for real system architectures, and creates baselines that can encourage future research in the field. We employed two state-of-art audio embeddings, and analyzed performance of classical unsupervised hashing algorithms on two audio event datasets. By combining weak audio embeddings with a supervised similarity objective and a quantization module, we proposed a deep hashing framework particularly suitable for efficient audio retrieval. Extensive experiments show the usefulness of the proposed supervised training even when we allow annotations for just a small portion of the database.

As a future step, training directly on lower level features can help better understand the importance of transfer learning for audio event retrieval. A closer look into the model layout can help understand the contribution of using concurrent hash learning as opposed to having a sequential optimization framework. Finally, the incorporation of non-exhaustive search algorithms combined with ranking modules suited for audio events could substantially decrease retrieval time, while increase precision of retrieved results: for example, generating hash codes that preserve hierarchical information of audio events [2, 27] might be extremely useful for fast and semantically-driven retrieval, even for unseen classes of audio events.

6. REFERENCES

[1] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley, "Detection and classification of acoustic scenes and events," *IEEE Transactions on Multimedia*, vol. 17, no. 10, pp. 1733–1746, 2015.

[2] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*.

[3] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, *et al.*, "Cnn architectures for large-scale audio classification," in *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*, 2017, pp. 131–135.

[4] Yao Wang, Zhu Liu, and Jin-Cheng Huang, "Multimedia content analysis-using both audio and visual clues," *IEEE Signal Processing Magazine*, vol. 17, no. 6, pp. 12–36, 2000.

[5] G. Valenzise, L. Gerosa, M. Tagliasacchi, F. Antonacci, and A. Sarti, "Scream and gunshot detection and localization for audio-surveillance systems," in *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2007, pp. 21–26.

[6] P. K. Atrey, N. C. Maddage, and M. S. Kankanhalli, "Audio based event detection for multimedia surveillance," in *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*, vol. 5, 2006.

[7] T. Hao, G. Xing, and G. Zhou, "isleep: unobtrusive sleep quality monitoring using smartphones," in *Proc. of the 11th ACM Conference on Embedded Networked Sensor Systems*, 2013.

[8] A. Kumar, M. Khadkevich, and C. Fügen, "Knowledge transfer from weakly labeled audio using convolutional neural network for sound events and scenes," in *ICASSP*, 2018, pp. 326–330.

[9] S. Z. Li, "Content-based audio classification and retrieval using the nearest feature line method," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 5, pp. 619–625, 2000.

[10] Z. Liu and Q. Huang, "Content-based indexing and retrieval-by-example in audio," in *IEEE International Conference on Multimedia and Expo. (ICME).*, vol. 2. IEEE, 2000, pp. 877–880.

[11] H. Jegou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 33, no. 1, pp. 117–128, 2010.

[12] J. Wang, T. Zhang, N. Sebe, H. T. Shen, *et al.*, "A survey on learning to hash," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 40, no. 4, pp. 769–790, 2017.

[13] Y. Cao, M. Long, J. Wang, H. Zhu, and Q. Wen, "Deep quantization network for efficient image retrieval," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, ser. AAAI'16. AAAI Press, 2016, pp. 3457–3463.

[14] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proc. of the 34th annual ACM symposium on Theory of computing*, 2002, pp. 380–388.

[15] A. W. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, no. 12, pp. 1349–1380, 2000.

[16] H. Lai, Y. Pan, Y. Liu, and S. Yan, "Simultaneous feature learning and hash coding with deep neural networks," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3270–3278.

[17] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan, "Supervised hashing for image retrieval via image representation learning," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, ser. AAAI'14. AAAI Press, 2014, pp. 2156–2162.

[18] R. M. Gray and D. L. Neuhoff, "Quantization," *IEEE transactions on information theory*, vol. 44, no. 6, pp. 2325–2383, 1998.

[19] A. Babenko and V. Lempitsky, "The inverted multi-index," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 37, no. 6, pp. 1247–1260, 2014.

[20] E. Fonseca, M. Plakal, F. Font, D. P. W. Ellis, X. Favory, J. Pons, and X. Serra, "General-purpose tagging of freesound audio with audioset labels: Task description, dataset, and baseline," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, November 2018, pp. 69–73.

[21] K. J. Piczak, "Esc: Dataset for environmental sound classification," in *Proc. of the 23rd ACM international conference on Multimedia*, 2015, pp. 1015–1018.

[22] B. Liu, Y. Cao, M. Long, J. Wang, and J. Wang, "Deep triplet quantization," *arXiv preprint arXiv:1902.00153*, 2019.

[23] D. Cai, "A revisit of hashing algorithms for approximate nearest neighbor search," *arXiv preprint arXiv:1612.07545*, 2016.

[24] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Advances in neural information processing systems*, 2009, pp. 1753–1760.

[25] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 35, no. 12, pp. 2916–2929, 2012.

[26] W. Liu, J. Wang, S. Kumar, and S.-F. Chang, "Hashing with graphs," in *International Conference on Machine Learning (ICML)*, 2011.

[27] A. Jati, N. Kumar, R. Chen, and P. Georgiou, "Hierarchy-aware loss function on a tree structured label space for audio event detection," in *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*, 2019, pp. 6–10.