VGGSOUND: A LARGE-SCALE AUDIO-VISUAL DATASET

Honglie Chen, Weidi Xie, Andrea Vedaldi and Andrew Zisserman

VGG, Department of Engineering Science, University of Oxford, UK
{hchen,weidi,vedaldi,az}@robots.ox.ac.uk

ABSTRACT

Our goal is to collect a large-scale audio-visual dataset with low label noise from videos 'in the wild' using computer vision techniques. The resulting dataset can be used for training and evaluating audio recognition models. We make three contributions. First, we propose a scalable pipeline based on computer vision techniques to create an audio dataset from open-source media. Our pipeline involves obtaining videos from YouTube; using image classification algorithms to localize audio-visual correspondence; and filtering out ambient noise using audio verification. Second, we use this pipeline to curate the VGGSound dataset consisting of more than 200k videos for 300 audio classes. Third, we investigate various Convolutional Neural Network (CNN) architectures and aggregation approaches to establish audio recognition baselines for our new dataset. Compared to existing audio datasets, VGGSound ensures audio-visual correspondence and is collected under unconstrained conditions. Code and the dataset are available at http://www.robots.ox.ac.uk/~vgg/data/vggsound/.

Index Terms— audio recognition, audio-visual correspondence, large-scale, dataset, convolutional neural network

1. INTRODUCTION

Large-scale datasets [1, 2] have played a crucial role. in many deep learning recognition tasks [3, 4, 5]. However in the audio domain, while several audio datasets have been released in the past few years [6, 7, 8, 9], the data collection process usually requires extensive human efforts, making it unscalable and often limited to narrow domains. *AudioSet* [10], is a large-scale audio-visual dataset containing over 2 million clips in unconstrained conditions. This is a valuable dataset, but it required extensive human verification in order to construct it. In contrast to these manually curated datasets, recent papers have demonstrated the possibility of collecting high-quality human speech datasets in an automated and scalable manner by using computer vision algorithms [11, 12, 13].

In this paper, our objective is to collect a large-scale audio dataset, similar to *AudioSet*, containing various sounds in the natural world and obtained 'in the wild' from unconstrained open-source media. We do this using a pipeline based on computer vision techniques that guarantees audio-visual correspondence (*i.e*. the sound source is visually evident) and low label noise, but requires only minimal manual effort.

Our contributions are three-fold: The first is to propose an automated and scalable pipeline for creating an 'in the wild' audio-visual dataset with low label noise. By using existing image classification algorithms, our method can generate accurate annotations, circumventing the need for human annotation. Second, we use this method to curate VGGSound, a large-scale dataset with over 200k video clips (visual frames and audio sound) for 300 audio classes, from YouTube videos. Each 10s clip contains frames that *show* the object making the sound, and the audio track contains the the *sound* of the object. There are at least 300 clips for each audio class. Our third contribution is to establish several baselines for audio recognition on the new dataset. To this end, we investigate different architectures, VGGish [3, 10] and ResNet [4] networks, as well as different aggregation approaches, global average pooling and NetVLAD [14, 15], for training deep CNNs on spectrograms extracted directly from the audio files with little pre-processing.

We expect VGGSound to be useful for both audio recognition and audio-visual prediction tasks. The goal of audio recognition is to determine the semantic content of an acoustic signal, *e.g*. recognizing the sound of a car engine, or a dog barking, *etc*. In addition, VGGSound is equally well suited for studying multi-modal audio-visual analysis tasks, for example, *audio grounding* aims to localize a sound spatially, by identifying in an image the object(s) emitting it [16, 17]. Another important task is to separate the sound of specific objects as they appear in a given frame or video clip [18, 19].

2. RELATED WORK

Audio and audio-visual datasets. Several audio datasets exist, as shown in Table 1. The UrbanSound dataset [6] contains more than 8k urban sound recordings for 10 classes drawn from the urban sound taxonomy. The Mivia Audio Events Dataset [7] focuses on surveillance applications and contains 6k audio clips for 3 classes. The Detection and Classification of Acoustic Scenes and Events (DCASE) community organizes audio challenges annually, for example, the authors of [20] released a dataset containing 17 classes with more than 56k audio clips. These datasets are relatively clean, but the scale is often too small to train the data-hungry Deep Neural

Networks (DNNs).

To remedy this shortcoming, a large-scale dataset of video clips was released by Google. This dataset, called *AudioSet*, contains more than 2 million clips drawn from YouTube and is helpful not only for audio research, but audio-visual research as well, where the audio and visual modalities are analysed jointly. This dataset is a significant milestone, however, the process used to curate *AudioSet* requires extensive human rating and filtering. In addition, the authors of [21] manually curated a high-quality, but small dataset that guarantees audio-visual correspondence for multi-modal learning, where the objects or events that are the cause of a sound must also be observable in the visual stream.

Datasets	# Clips	Length	# Class	Video	AV-C
UrbanSound [6]	8k	8.75h	10	×	×
MIVIA [7]	6k	29h	3	×	×
DCASE2017 [20]	57k	89h	17	×	×
FSD [8]	24k	119h	398	×	×
AudioSet [10]	2.1m	5833h	527	✓	×
AVE [21]	4k	11.5h	28	✓	✓
VGGSound (Ours)	200k	560h	300	✓	✓

Table 1. Statistics for recent audio datasets. "# Clips", the number of clips in the dataset; "Length", the total duration of the dataset; "# Classes", number of classes in the dataset; "Video", whether videos are available; "AV-C", whether audios and videos correspond, in the sense that the sound source is always visually evident within the video clip.

Audio Recognition. Audio Recognition, namely the problem of classifying sounds, has traditionally been addressed by means of models such as Gaussian Mixture Models (GMM) [22] and Support Vector Machines (SVM) [23] trained by using hand-crafted low-dimension features such as the Mel Frequency Cepstrum Coefficients (MFCCs) or i-vectors [24]. However, the performance of MFCCs in audio recognition degrades rapidly in "unconstrained" environments that include real-world noise [25, 26]. More recently, the success of deep learning has motivated approaches based on CNNs [5, 27] or RNNs [28, 29, 30]. In this paper, rather than developing complex DNN architectures specific to audio recognition, we choose to illustrate the benefits of our new benchmark dataset by training baselines to serve as comparison for future research. To this end, we train powerful ResNet architectures with the NetVLAD aggregation method for audio recognition tasks [14, 15].

3. THE VGGSOUND DATASET

`VGGSound` contains over 200k clips for 300 different sound classes. The dataset is audio-visual in the sense that the object that emits each sound is also visible in the corresponding video clip. Figure. 1 shows example cropped image frames, corresponding audio waveforms, and a histogram details the statistics for each class. Each sound class contains 300–1000 10s clips, with no more than 2 clips per video. The set of sound labels is flat (*i.e*. there is no hierarchy as in *AudioSet*). Sound classes can be loosely grouped as: people, animals, music, sports, nature, vehicle, home, tools, and others. All clips in the dataset are extracted from videos downloaded from YouTube, spanning a large number of challenging acoustic environments and noise characteristics of real applications.

In the following sections, we describe the multi-stage approach that we have developed to collect the dataset. The process can be described as a cascade that starts from a large number of potential audio-visual classes and corresponding videos, and then progressively filters out classes and videos to leave a smaller number of clips that are annotated reliably. The number of classes and videos after each stage of this process is shown in Table 2. The process is extremely scalable and only requires manual input at a few points for well defined tasks.

Stages	Goal	# Classes	# Videos
1	Candidate videos	600	1m
2	Visual verification	470	550k
3	Audio verification	390	260k
4	Iterative noise filtering	300	200k

Table 2. The number of classes and videos after each stage of the generation pipeline. Note, classes with less than 100 videos are removed from the dataset.

Stage 1: Obtaining the class list and candidate videos. The first step is to determine a tentative list of sound classes to include in the dataset. We follow *three* guiding principles in order to generate this list. First, the sounds should be *in the wild*, in the sense that they should occur in real life scenarios, as opposed to artificial sound effects. Second, it must be possible to *ground and verify the sounds visually*. In other words, our sound classes should have a clear visual connotation too, in the sense of being predictable with reasonable accuracy from images alone. For instance, the sound 'electric guitar' is visually recognizable as it is generally possible to visually recognize someone playing a guitar, but 'happy song' and 'pop music' are not: these classes are too abstract for visual recognition and so they are not included in the dataset. Third, the classes should be *mutually exclusive*. Although we initialize the list using the label hierarchies in existing audio datasets [7, 8, 10] and other hierarchical on-line sources, our classes are only leaf nodes in these hierarchies. In this manner, the label set in `VGGSound` is flat and contains only one label for each clip. For instance, if the clip contains the sound of a car engine, then the label will be only "car engine"; the more general class "engine" is not included in the list.

The initial list of classes, constructed in this manner, had 600 items. Each class name is used as a search query for YouTube to automatically download corresponding candidate videos. In order to increase the chance of obtaining relevant videos, the class names are further transformed to generate variants of each textual query as follows: (1) form-

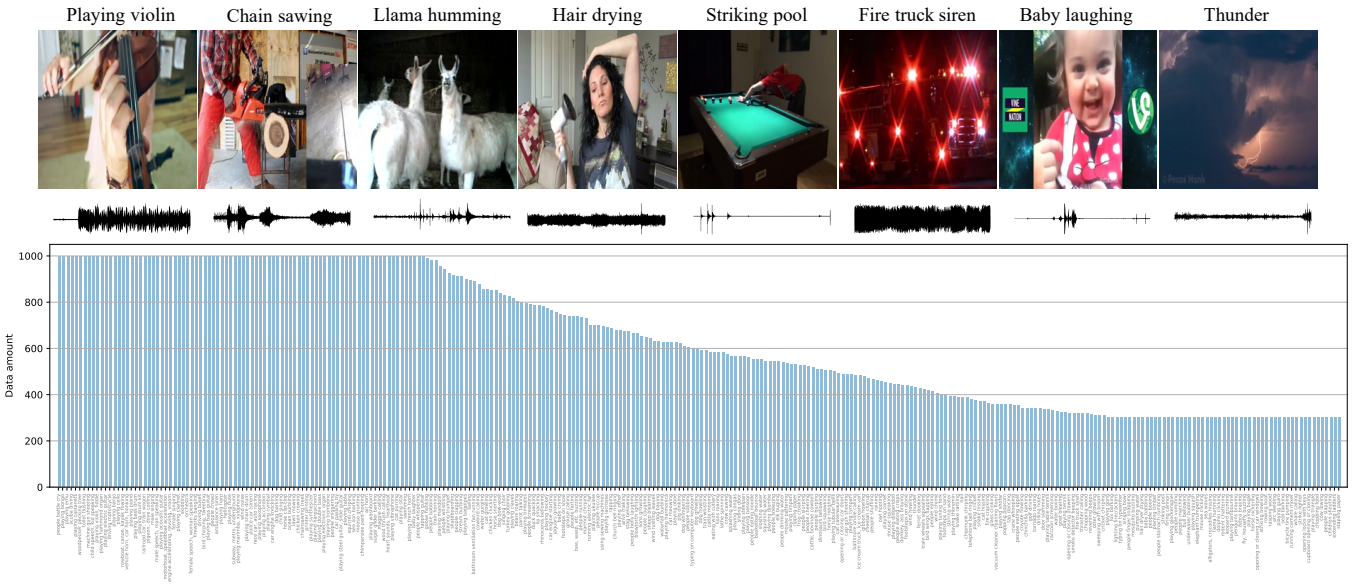

Fig. 1. The top row in this figure shows example video frame and audio pairs of VGGSound classes, the bottom bar chart demostrates VGGSound classes with sizes of each audio class sorted by descending order.

ing 'verb+(ing) object' sentences, *e.g.* 'playing electric guitar','ringing church bells', *etc.* (2) submitting the query after translation to different languages, as is done in [31], such as English, Spanish and Chinese, etc; (3) adding possible synonym phrase which specify the same sounds, *e.g.* 'steam hissing: water boiling, liquid boiling, *etc.*' In total, over 1m videos were downloaded from YouTube in this manner.

Stage 2: Visual verification. The purpose of this stage is to verify and localize the visual signature in the downloaded videos. In detail, for each VGGSound class, the corresponding visual signature is given by image classifiers. For example, 'playing violin' and 'cat meowing' in VGGSound can be matched directly to the OpenImage classifiers [32] 'violin' and 'cat'. These associations are proposed automatically by matching keywords and then verified manually.

However, half of VGGSound classes (*e.g.* 'hail', 'playing ukulele') could not be matched directly to OpenImage classifiers in this manner. To tackle this issue, we relaxe the way sound labels are matched to visual labels via semantic word embeddings. Specifically, we convert our 600 sound classes and the 5000 OpenImage classes to word2vec embeddings [33]. These embedding have 512 dimensions, so this step results in matrices $S \in \mathcal{R}^{600\times512}$ and $O \in \mathcal{R}^{5000\times512}$, respectively for sound and image labels. We then compute the cosine similarity between the two matrices, resulting in an affinity matrix $A \in \mathcal{R}^{600\times5000} = SO^{\top}$ that represents the strength of the similarity between sound and image classes. The top 20 OpenImage classes for each of the 600 sound classes are then selected as the visual signature of the corresponding sound. For example, 'hail' was matched to 'nature, nature reserve, rain and snow mixed, lightning, thunderstorm, *etc.*' and 'playing electric guitar' to 'electric guitar, guitar, acoustic-electric guitar, musical instrument, *etc.*'.

After determining these associations, the OpenImage pretrained classifier are run on the downloaded videos, and the 10 frames in the video that receive the highest prediction score are selected provided the score is above an absolute confidence threshold of 0.2. The frames that pass this test are assumed to contain the visual content selected by the classifier. Clips are then created by taking 5 seconds at either side of these representative frames. After this stage, the number of sound classes is reduced from the original 600 to 470, due either an initial scarcity of potential video matches or by failed visual verification.

Stage 3. Audio verification to remove negative clips. Despite visual verification, our clips are still not guaranteed to contain the desired sound, as an object being visible does not imply that it emits a sound at the same time; in fact, we found that many clips where the correct object was in focus, contained instead generic sounds from humans, such as a narrator describing an image or video, or background music. Since these issues are fairly specific, we address them by finetuning the VGGish model with only three sound classes: speech, music and others. The finetuned classifier is typically reliable as most of the existing datasets offers higly clean data of these classes. We use it to reject clips. For example, using a threshold 0.5, in 'playing bass guitar' videos, we reject any clip for which "speech" is greater than the threshold, but allow music; while for 'dog barking' videos, both speech and music are rejected. After this stage, there are 390 classes left with at least 300 validated video clips. Note that our selection process aims to reject "false positive" *i.e.* inappropriate sounds in each class, we do not attempt to use an audio classifier to select positive clips as that risks losing hard positive audio samples.

Stage 4: Iterative noise filtering. For this final clean up

stage of the process, 20 video clips are randomly sampled from each class and manually checked (both visually and on audio) that they belong to the class. Classes with at least 50% correct are kept and the other classes are discarded. The total set of video clips remaining forms our candidate dataset. Note that, at this stage, the candidate videos can be categorized by audio as one of three types: (i) audios that are clearly of the correct category, *i.e.* easy positives; (ii) audios of the correct category, but with a mixture of sounds, *i.e.* hard positives; or (iii) incorrect audios, *i.e.* false positive.

To further curate the candidate dataset, we make three assumptions: First, there is no systematic bias in the noisy samples, by that we mean, the false positives are not from the same category. Second, Deep CNNs tend to end up with different local minimas and prediction errors, ensembling different models can therefore result in a prediction that is both more stable and often better than the predictions of any individual member model. Third, when objects emit sound, there exists particular visual patterns, *e.g.* a "chimpanzee pant-hooting" will mostly happen with moving bodies.

Exploiting the first two assumptions, the videos of each class are randomly divided into two sets, and an audio classifier is trained on half the candidate videos and used to predict the class of the other half. This process is done twice so that each clip has 2 predictions. To obtain relatively easy and precise postives, we keep the clips whose actual class-label falls into the top-3 of the predictions from the ensembled models. In order to mine the harder positives, we exploit the third assumption by computing visual features for the positive clips, and perform visual retrieval from the rest of data that has been rejected by the audio classifiers. Using a visual classifier can result in similar looking visual clips but disparate hard-positive audio clips. Lastly, we train a new audio classifier (ResNet18) with all easy and hard clips, and retrieve more data from that rejected so far. This increases the number of video clips and forms our final dataset: VGGSound with 300 classes of over 200k videos, and each class contains 300–1000 audio-visual corresponding clips.

4. EXPERIMENTS

4.1. Experimental setup and Evaluation

Experimental setup. We investigate the audio recognition task on both *AudioSet* and our new VGGSound dataset. As the two datasets contain different sound vocabularies, at training time, we use subsets of common categories in both datasets. Similarly, at testing time, we select the intersection of *AudioSet* and VGGSound to form a single testset called AStest (and remove any videos in AStest that are in the training sets of *AudioSet* or VGGSound). This leads roughly 15k clips in AStest. In addition, we investigate how audio recognition performs on VGG and ResNet backbone networks with/without NetVLAD aggregation using our new VGGSound datasets.

Evaluation metrics. For evaluation metrics, we adopt the evaluation metrics of [5], *i.e.* mean average precision (mAP), AUC and equivalent d-prime class separation.

4.2. Implementation details

During training, we follow [5] for data preprocessing for models trained with VGGish models. For models trained on ResNet18, we randomly sample 5s from the 10s audio clip and apply a short-time Fourier transform on the sample, resulting a 257×500 spectrogram. During testing, we directly feed the 10s audio into the network.

All experiments were trained using the Adam optimizer with cross entropy loss. The learning rate starts with 10^{-3} and is reduced by a factor of 10 after training plateaus. We use a sigmoid layer when training on *AudioSet* data since each video clip has multiple labels. For models trained on VGGSound data, we use a softmax layer in the last layer.

4.3. Results

From the experimental results in Table 3, we can draw the following conclusions: First, when we adopt a pretrained model from [5] and finetune on *AudioSet* (Model-A) and VGGSound (Model-B), despite *AudioSet* (2m) containing more data than VGGSound (200k) , model-B still outperforms model-A on all metrics, we conjecture that this is because the noise ratio of VGGSound is lower than that of *AudioSet*, as we wished in our initial design objective. Second, training model-C (a VGGish model from scratch) on VGGSound, gets slightly worse performance on all metrics than model-B, which shows that pretraining on a large dataset helps boost the model's performance. Third, when comparing model-D (trained with average pooling) and model-E (trained with NetVLAD), we demonstrate the significant effectiveness of NetVLAD aggregation over the naïve global average pooling also beats the pretrained VGGish model (model-B).

Model	Description	Train	mAP	AUC	d-prime
A	VGGish pretrain+ft	AudioS	0.286	0.899	1.803
B	VGGish pretrain+ft	VGGS	0.326	0.916	1.950
C	VGGish scratch	VGGS	0.301	0.910	1.900
D	ResNet18 AVG	VGGS	0.328	0.923	2.024
E	ResNet18 VLAD	VGGS	0.369	0.927	2.058

Table 3. We compare the results using various combination of models, training sets and test on AStest, "VLAD" means NetVLAD; "AVG" means Global Average Pooling; "AudioS' refers to *AudioSet*; "VGGS" means VGGSound.

5. CONCLUSION

In this paper, we propose an automated pipeline for collecting a large-scale audio-visual dataset – VGGSound, which contains more than 200k videos and 300 classes for "unconstrained" conditions. We also compare CNN architectures and aggregation methods to provide baseline results for audio recognition on VGGSound.

Acknowledgement. Financial support was provided by the EPSRC Programme Grant Seebibyte EP/M013774/1.

References

[1] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes (VOC) challenge," *IJCV*, vol. 88, no. 2, pp. 303–338, 2010.

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. CVPR*, 2009.

[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. ICLR*, 2015.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016.

[5] S. Hershey, S. Chaudhuri, D Ellis, J Gemmeke, A. Jansen, C Moore, M Plakal, D Platt, R Saurous, B Seybold, M Slaney, R Weiss, and K Wilson, "CNN architectures for large-scale audio classification," in *Proc. ICASSP*. 2017.

[6] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *Proc. ACMM*, 2014.

[7] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, "Reliable detection of audio events in highly noisy environments," *Pattern Recognition Letters*, 2015.

[8] F. Eduardo, P. Jordi, F. Xavier, F. Frederic, B. Dmitry, F. Andrés, O. Sergio, P. Alastair, and S. Xavier, "Freesound datasets: a platform for the creation of open audio datasets," in *ISMIR 2017*, 2017.

[9] A. Mesaros, T. Heittola, and T. Virtanen, "TUT database for acoustic scene classification and sound event detection," in *EUSIPCO*, 2016.

[10] J Gemmeke, D Ellis, D Freedman, A Jansen, W Lawrence, C Moore, M Plakal, and M Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. ICASSP*, 2017.

[11] A. Nagrani, J. S. Chung, and A. Zisserman, "VoxCeleb: a large-scale speaker identification dataset," in *INTERSPEECH*, 2017.

[12] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, "VoxCeleb: Large-scale speaker verification in the wild," *Computer Speech & Language*, 2020.

[13] J. S. Chung, A. Nagrani, and A. Zisserman, "VoxCeleb2: Deep speaker recognition," in *INTERSPEECH*, 2018.

[14] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *Proc. CVPR*, 2016.

[15] W. Xie, A. Nagrani, J.S. Chung, and A. Zisserman, "Utterance-level aggregation for speaker recognition in the wild," in *Proc. ICASSP*, 2019.

[16] R. Arandjelović and A. Zisserman, "Look, listen and learn," in *Proc. ICCV*, 2017.

[17] E. Kidron, Y. Schechner, and M. Elad, "Pixels that sound," in *Proc. CVPR*, 2005.

[18] A. Owens, P. Isola, J.H. McDermott, A. Torralba, E.H. Adelson, and W.T. Freeman, "Visually indicated sounds.," in *Proc. CVPR*, 2016.

[19] H. Zhao, C. Gan, A. Rouditchenko, C. Vondrick, J. McDermott, and A. Torralba, "The sound of pixels," in *Proc. ECCV*, 2018.

[20] A. Mesaros, A. Diment, B. Elizalde, T. Heittola, E. Vincent, B. Raj, and T. Virtanen, "Sound event detection in the DCASE 2017 challenge," *IEEE/ACM TASLP*, 2019.

[21] Y. Tian, J. Shi, B. Li, Z. Duan, and C. Xu, "Audio-visual event localization in unconstrained videos," in *Proc. ECCV*, 2018.

[22] X. Zhuang, X. Zhou, M. Hasegawa-Johnson, and T. Huang, "Real-world acoustic event detection," *Pattern Recognition Letters*, 2010.

[23] A. Temko and C. Nadeu, "Classification of acoustic events using svm-based clustering schemes," *Pattern Recognition*, 2006.

[24] Z. Huang, Y. Cheng, K. Li, V. Hautamki, and C. Lee, "A blind segmentation approach to acoustic event detection based on i-vector," in *INTERSPEECH*, 2013.

[25] U. Yapanel, X. Zhang, and J. Hansen, "High performance digit recognition in real car environments.," in *INTERSPEECH*, 2002.

[26] J. Hansen, R. Sarikaya, U. Yapanel, and B.L. Pellom, "Robust speech recognition in noise: an evaluation using the spine corpus.," in *INTERSPEECH*, 2001.

[27] T. Naoya, G. Michael, P. Beat, and V. Luc, "Deep convolutional neural networks and data augmentation for acoustic event detection," in *INTERSPEECH*, 2016.

[28] G. Parascandolo, H. Huttunen, and T. Virtanen, "Recurrent neural networks for polyphonic sound event detection in real life recordings," in *Proc. ICASSP*, 2016.

[29] K. Choi, G. Fazekas, M. Sandler, and K. Cho, "Convolutional recurrent neural networks for music classification," in *Proc. ICASSP*, 2017.

[30] Y. Xu, Q. Kong, W. Wang, and M. D. Plumbley, "Large-scale weakly supervised audio classification using gated convolutional neural network," in *Proc. ICASSP*, 2018.

[31] J. Carreira, E. Noland, A. Banki-Horvath, C. Hillier, and A. Zisserman, "A short note about kinetics-600.," *arXiv preprint arXiv:1808.01340*, 2018.

[32] I. Krasin, T. Duerig, N. Alldrin, A. Veit, S. Abu-El-Haija, S. Belongie, D. Cai, Z. Feng, V. Ferrari, V. Gomes, A. Gupta, D. Narayanan, C. Sun, G. Chechik, and K. Murphy, "Open-images: A public dataset for large-scale multi-label and multi-class image classification.," 2016.

[33] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013.