SELF-SUPERVISED LEARNING FROM AUTOMATICALLY SEPARATED SOUND SCENES

Eduardo Fonseca,[1,2*] *Aren Jansen,*[2] *Daniel P. W. Ellis*,[2] *Scott Wisdom,*[2] *Marco Tagliasacchi,*[2]
John R. Hershey,[2] *Manoj Plakal,*[2] *Shawn Hershey,*[2] *R. Channing Moore,*[2] *Xavier Serra*[1]

[1]Music Technology Group, Universitat Pompeu Fabra, Barcelona, eduardo.fonseca@upf.edu
[2]Google Research, New York, NY, Mountain View, CA, Cambridge, MA, and Zürich, CH

ABSTRACT

Real-world sound scenes consist of time-varying collections of sound sources, each generating characteristic sound events that are mixed together in audio recordings. The association of these constituent sound events with their mixture and each other is semantically constrained: the sound scene contains the union of source classes and not all classes naturally co-occur. With this motivation, this paper explores the use of unsupervised automatic sound separation to decompose unlabeled sound scenes into multiple semantically-linked views for use in self-supervised contrastive learning. We find that learning to associate input mixtures with their automatically separated outputs yields stronger representations than past approaches that use the mixtures alone. Further, we discover that optimal source separation is not required for successful contrastive learning by demonstrating that a range of separation system convergence states all lead to useful and often complementary example transformations. Our best system incorporates these unsupervised separation models into a single augmentation front-end and jointly optimizes similarity maximization and coincidence prediction objectives across the views. The result is an unsupervised audio representation that rivals state-of-the-art alternatives on the established shallow AudioSet classification benchmark.

Index Terms— Contrastive learning, audio representation learning, self-supervision, source separation

1. INTRODUCTION

The construction of human-labeled audio datasets for sound event recognition is notoriously time-consuming and subjective, imposing practical limitations on dataset size and quality. For example, label noise issues such as missing, incorrect or inconsistent labels are inherent to this process [1, 2, 3, 4]. In contrast, unsupervised representation learning can exploit much larger amounts of data without the need for additional labeling. One of the advantages of these representations is they are less specialized towards the often-biased human labels and thus may be better suited for generalization to other tasks [5, 6, 7, 8].

Self-supervised methods aim at learning representations without the need for external supervision. Absent explicit labels generated by humans, the success of these methods relies on the design of *proxy* learning tasks in which pseudo-labels are generated from patterns in the data. These methods solve proxy tasks on unlabeled data to learn mappings from inputs to low-dimensional representations, which can then be used for downstream tasks such as classification. This paradigm has seen major progress in computer vision [9, 10, 11] and in speech recognition [12, 13, 7]. For general-purpose audio, including a variety of environmental sounds beyond speech, the majority of works are based on contrastive learning [14, 15, 16, 17, 18, 19], where a representation is learned by comparing pairs of examples selected by some semantically-correlated notion of similarity [20]. Specifically, comparisons are made between positive pairs of "similar" and negative pairs of "dissimilar" examples, with the goal of learning a representation that pulls together positive pairs and thus reflects semantic structure. One of the first works in contrastive audio representation learning uses triplet loss [14], in which anchor-positive pairs are created by sampling neighboring audio frames as well as applying other audio transformations (e.g., adding noise). The proxy task of *coincidence prediction* learns a representation to support predicting whether a pair of examples occurs within a certain temporal proximity [15].

Recently, promising results have been attained by contrastive learning approaches that solve the proxy task of *similarity maximization* [17, 18, 19], following the seminal SimCLR work in visual representation learning [9]. This method consists of maximizing the similarity between differently-augmented *views* of the same input audio example. Critical to its success is the simultaneous use of a diversity of semantics-preserving, domain-specific augmentation methods [9, 18]. For audio modeling, proven augmentation strategies include sampling nearby audio frames [14, 15, 17, 18, 19], artificial example mixing [18, 19, 14], time/frequency masking [18, 19], random resized cropping [18] and time/frequency shifts [14, 18, 19]. In most cases, these augmentations introduce artificial, handcrafted transformations with hyperparameters that must be tuned to lie within a semantics-preserving range. However, typical real-world sound scene recordings already tend to be quite complex, involving mixtures of several sound sources at varying levels and unexpected channel distortions. Therefore, these artificial augmentation techniques risk introducing an unrealistic domain shift that hinders generalization to real-world applications.

In this work, we explore using unsupervised sound separation as an alternative path to generate views for contrastive learning. This provides a sort of inverse to traditional example mixing augmentation: instead of constructing artificial mixtures, we decompose a naturalistic sound scene into a collection of simpler channels that share semantic commonalities with the original recording and each other. In contrast to the previous approaches, this automatic separation approach is data-driven and input-dependent, producing ecologically valid views that eliminate the need for parameter tuning for the given dataset. We demonstrate that learning to associate sound mixtures with their constituent separated channels elicits semantic structure in the learned representation and is complementary to other commonly-used data augmentations when composed. Furthermore, we pair this

*Work done during an internship at Google Research.
This extended version of a WASPAA 2021 submission under same title has additional discussion for easier consumption.

augmentation procedure with a multitask objective that includes both similarity maximization and coincidence prediction, which exhibit complementary behavior for different downstream representation use cases. Finally, we discover that a wide range of separation model competencies enable useful (and complementary) augmentations, demonstrating that optimal sound separation performance is not essential for representation learning. The best representation learned with our sound-separation informed framework achieves an mAP of 0.326 on the downstream shallow-model AudioSet classification task. This exceeds previous results on this benchmark under the same evaluation protocol [14, 15], and is on par with the state-of-the-art under comparable evaluation settings [19].

2. SOUND SEPARATION AS DATA AUGMENTATION

Sound separation has been studied as preprocessing to improve supervised sound event detection [21, 22]. Here, we propose sound separation as an *augmentation* to generate pairs of positive examples for contrastive learning. In order to come up with good views for contrastive learning, previous work argues that their mutual information must be reduced while the downstream semantically-relevant information between the views is retained [23]. Another important observation is that, since contrastive learning pulls together representations of positive views, the proxy task attempts to ignore the transformations applied to create them. Consequently, how the pairs of views are generated determines the invariant properties promoted in the learned representation.

We propose to decompose an incoming audio clip, which in general is a *mixture* of multiple sound events, into its constituent sources. We then use the mixture and these separated channels to form positive pairs for contrastive learning. In particular, the comparison of the input mixture and one of the separated channels should meet the requirements established above: *(i)* the mutual information is reduced as, in principle, there is at least one input sound source that is no longer in the separated channel; *(ii)* some relevant semantics are preserved as the sound source(s) present in the separated channel is also present in the input mixture. Therefore, in theory, this comparison would be well suited for contrastive learning. Further, with this comparison we are promoting the learning of representations that are invariant to combinations of naturally co-occurring or overlapping sources—a valuable property for general-purpose audio recognition applications. We use this *mixture vs channel* comparison as the default contrastive setup for the majority of our experiments, as illustrated in the proposed learning framework depicted in Fig. 1. By contrast, the comparison between two separated channels would not be appropriate for the similarity maximization task as, in principle, each channel would contain different sources, thereby violating the semantic preservation requirement. However, this *channel vs channel* comparison could still be useful for the coincidence prediction task, where the semantic equality demand is relaxed to require only some consistency between the examples in order to support their coincidence prediction. We evaluate experimentally these hypotheses in Sec. 5, uncovering several nuances.

2.1. MixIT for Unsupervised Sound Separation

For a sound separation system we use a model trained with *mixture invariant training* (*MixIT*) [24]. MixIT is an unsupervised method in which training examples are constructed by mixing existing audio clips, and the model is tasked to separate the resulting mixtures into a number of latent sources, such that an optimal remix of the separated sources best approximates the original mixtures. The main advantage of MixIT compared to previous methods is that it does not require knowledge of ground truth source signals, which allows leveraging large amounts of unlabeled data. In addition, MixIT has shown great promise in the task of universal sound separation, that is, separating arbitrary sounds instead of specializing to (e.g.) speech [25, 26].

Our separation model was trained on AudioSet [27] while ignoring all available labels and using previously proposed training settings [24]. The separation model used is based on an improved time-domain convolutional network (TDCN++) [25], which is similar to a Conv-TasNet [28]. This model consists first of an encoder that maps short frames of the input waveform to a latent space. Then, separation is done in the latent space where M masks are predicted for the target sources. Finally, M separated waveforms are reconstructed through a decoder from the masked features. Separated sources are constrained to add up to the input mixture via a consistency projection layer [29]. Our main goal is to assess the benefits of MixIT separation pre-processing coupled with a contrastive learning back-end.

2.2. Composition of Augmentations

Using a single transformation to generate data views has been shown as inferior to the composition of several augmentations, which is essential to obtain effective representations for vision [9] and audio [18]. By composing multiple augmentations, the goal is to define a more challenging learning task so that higher-quality representations can emerge. However, not all compositions are necessarily valid; rather, the elements in the composition must adequately complement each other [9, 18].

Here, in order to construct a more challenging proxy task, we combine sound separation with *temporal proximity* sampling (TP) [14] and SpecAugment [30] (in this order). Temporal proximity sampling consists of randomly selecting two audio snippets (of 0.96s in our case) as basic units to construct pairs of examples, instead of leveraging entire AudioSet clips (of typically 10s). When randomly sampling audio snippets within a prescribed temporal proximity, we are likely to sample *(i)* the same sound sources emitting somewhat different acoustic patterns as they evolve over time; or *(ii)* different sources that are related semantically or casually with the initial ones. Thus the temporal coherence among neighboring audio snippets implies a natural form of data augmentation. This simple method has been proven effective in many contrastive audio representation learning works [14, 15, 18, 17, 19], analogous to the common practice of random cropping with images [9].

Then, we apply SpecAugment on the log-mel spectrograms of the selected audio snippets, with time warping and time/frequency masking [30]. SpecAugment has gained popularity as data augmentation in supervised classification, and has also been used to generate views for contrastive learning of speech [31] and audio [18]. The combination of TP and SpecAugment is represented by the data augmentation (DA) blocks in Fig. 1. Together with the preceding unsupervised sound separation stage, they form the front-end for the two proxy tasks. In Sec. 5.3 we further extend these compositions including different convergence states of the separation model which provide distinct transformations to the incoming audio.

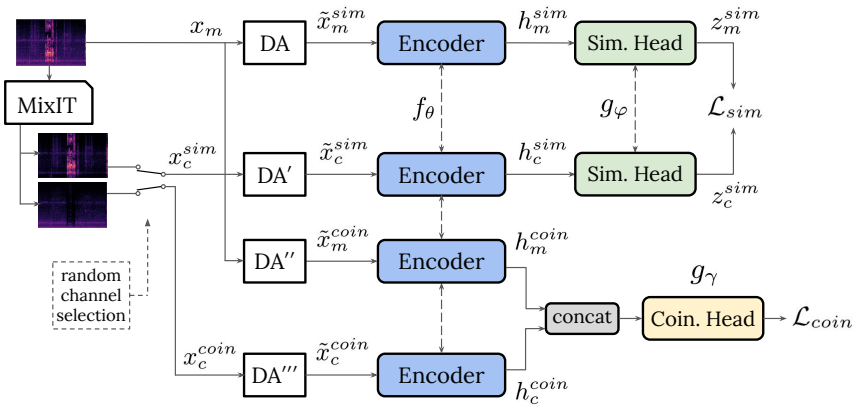

Figure 1: **Proposed contrastive learning framework**. It is composed of an unsupervised sound separation and augmentation front-end, a common encoder f_θ, and two task-specific heads, g_φ and g_γ, for the similarity maximization and coincidence prediction tasks respectively. Dashed lines between networks denote shared weights. Each separated channel feeding each proxy task (x_c^{sim} for similarity maximization or x_c^{coin} for coincidence prediction) is selected randomly between the two output channels from the MixIT separation model. The concat block stacks the latent representations for each view to define the input to the coincidence prediction head. Primes in the data augmentation (DA) blocks illustrate that each block is a different instance of the same augmentation policy, combining Temporal Proximity and SpecAugment. Note that the front-end illustrates the creation of pairs of positive examples—the pairs of negatives are constructed from different clips.

3. PROXY LEARNING TASKS

This section describes the two proxy tasks used in our framework: a similarity maximization task, and a coincidence prediction task. For both, a pair of positive examples is constructed by selecting two audio snippets within the *same* 10 s AudioSet clip, either from the input mixture or from the resulting separated channels. Analogously, a pair of negative examples is constructed by selecting two snippets from *different* clips (mixtures or separated channels). This is based on the assumption that two snippets within a given temporal proximity are much more likely to be semantically related than two snippets from independent recordings.

3.1. Similarity Maximization

The similarity maximization task consists of maximizing the agreement between differently-augmented views of the same audio example. After deriving the different views as above, their corresponding embedding representations are compared using a contrastive loss. This loss attempts to co-locate the two representations in the same spot of the embedding space, thus promoting invariance to the transformations applied to generate the views. This task is based on recent work on visual representation learning, SimCLR [9], and its block diagram is depicted in the top half of Fig. 1.

Front-end. The pipeline starts with one input mixture x_m (i.e., one AudioSet clip). Using the separation model, every incoming mixture x_m is separated into two output channels, from which one is randomly selected for this proxy task, x_c^{sim}. Next, x_m and x_c^{sim} undergo TP and SpecAugment transformations (see Sec. 2.2). Note that each example x_m or x_c^{sim} undergoes a different instance of the same transformation policy (indicated by DA and DA′ in Fig. 1).

Encoder Network. The outputs from the DA blocks, $\tilde{x}_m^{sim}$ and $\tilde{x}_c^{sim}$, feed a convolutional encoder f_θ in order to extract low-dimensional embeddings h. Specifically, for the top branch we obtain $h_m^{sim} = f_\theta\left(\tilde{x}_m^{sim}\right)$, where h_m^{sim} is the representation after a d-dimensional embedding layer and θ are the encoder's parameters. Once the training is over and the encoder has converged, the representation h is evaluated on downstream tasks.

Similarity Head. We use a simple MLP, g_φ with parameters φ, to map the representation h to the final metric embedding z, the domain in which the contrastive loss is applied [9]. This head is used to allow the representation h to back away from the representation at the training objective. Previous work reports better downstream performance using h instead of z [9], something we also observed in preliminary experiments.

Contrastive Loss. To compare a positive pair of examples, x_m and x_c^{sim}, we adopt the normalized temperature-scaled cross-entropy (*NT-Xent*) loss [9, 20] given by

$$\mathcal{L}_{sim_{i,j}} = -\log \frac{\exp\left(\text{sim}(z_i, z_j)/\tau\right)}{\sum_{v=1}^{2N} \mathbb{1}_{v\neq i} \exp\left(\text{sim}(z_i, z_v)/\tau\right)} \qquad (1)$$

where z_i and z_j are the metric embeddings corresponding to x_m and x_c^{sim}, $\text{sim}(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u}^\top \boldsymbol{v}/\|\boldsymbol{u}\|\|\boldsymbol{v}\|$ represents cosine similarity whose sensitivity is adjusted by a temperature value $\tau \in (0, 1]$, $\mathbb{1}_{v\neq i} \in \{0, 1\}$ is a function that returns 1 when $v \neq i$, and N is the batch size. Since two views are generated from each incoming audio clip, the batch size is extended from N to $2N$ elements, allowing for one pair of positive examples and $2(2N-2)$ pairs of negative examples for every input mixture—the overall loss includes both $\mathcal{L}_{sim_{i,j}}$ and $\mathcal{L}_{sim_{j,i}}$. By minimizing the objective in (1), parameters θ and φ are adjusted to maximize the numerator (i.e., the agreement between embeddings of positives, assigning them to neighboring representations) while simultaneously minimizing the denominator (i.e., the similarity between embeddings of negatives, forcing them to distant spots in the embedding space).

3.2. Coincidence Prediction

The coincidence prediction task relies on the *slowness prior* in representation learning [32]. Audio waveforms of sound sources can vary quickly, whereas the corresponding perceived semantics change at a much slower rate. Consequently, there should be a relatively stable latent representation in order to explain the semantic perception. This representation would support the prediction of whether a pair of examples are coinciding within a given temporal proximity. Note this task is a generalization of the correspondence prediction task

proposed for audio-visual multimodal learning [33], where the task is to predict time *correspondence* between audio and video frames. Here, we relax the time scale requirement to predict *coincidence* within a prescribed temporal proximity, specifically within the (maximum) 10 s of AudioSet clips. The task diagram is depicted in the bottom half of Fig. 1.

Front-end and Encoder network. In previous work, this proxy task has been applied directly to audio snippets drawn from the same/different temporal proximity [15]. Here, we adopt the augmentation front-end described in Sec. 2, which is the same as for the similarity maximization task, leading to the augmented examples $\tilde{x}_m^{coin}$ and $\tilde{x}_c^{coin}$ from x_m and x_c^{coin}. Then, we use a convolutional encoder to extract d-dimensional embedding representations h—the same encoder network is shared across both proxy tasks, f_θ.

Coincidence Head. Once the embedding representations for a pair of examples are obtained, we use a coincidence network, g_γ with parameters γ, tasked to predict the (non)-coincidence between the pair. More specifically, we feed g_γ with the concatenation of the two embeddings $[h_m^{coin}, h_c^{coin}] \in \mathbb{R}^{2d}$. The coincidence head consists of an MLP with one output unit, mapping the concatenated embedding representation to the probability that the input pair is coinciding—a binary classification task.

Loss. In a generic batch of N pairs of within-clip coinciding examples (i.e., positive pairs), $X = \left\{(x_1^i, x_2^i)\right\}_{i=1}^N$, we define $N-1$ pairs of negative examples per each pair of positives. This is done by pairing the non-coinciding examples (x_1^i, x_2^j) for $i \neq j$. In this setting, for a given batch X and focusing on our goal of optimizing the representation h, the coincidence loss function follows the class-balanced binary cross entropy expression [15]:

$$\mathcal{L}_{coin}(X) = -\frac{1}{N}\sum_{i=1}^{N} \log g_\gamma([h_m^{coin,i}, h_c^{coin,i}]) - \frac{1}{N(N-1)} \sum_{\substack{1\leq i,j\leq N \\ j\neq i}} \log\left[1 - g_\gamma([h_m^{coin,i}, h_c^{coin,j}])\right]. \quad (2)$$

3.3. Joint Optimization

We conjecture that jointly optimizing the two objectives above in a multi-task setting can favor learning complementary information for semantic representation learning. Both proxy tasks share the ultimate goal of contrastive learning, that is, supporting relationships between pairs of positives and pairs of negatives so as to force a semantically structured embedding space. However, each task pursues this goal in a slightly different way, in terms of underlying principle and implementation.

Underlying principle. The similarity maximization task essentially aims to co-locate the representations of both examples in a positive pair at the same point in the embedding space. Therefore, for successful representation learning, it is usually required that some semantic relationship is preserved between the two examples, e.g., the two examples share some class label(s). By contrast, the coincidence prediction task is based on a weaker condition. Instead of co-locating representations, the goal is to assign a representation that supports coincidence prediction, establishing a clear relationship between the representations for both examples, but not necessarily requiring their collocation.

Implementation. The NT-Xent loss of (1) follows a canonical version of contrastive loss, explicitly measuring similarity of embeddings as the scoring function [20]. By contrast, the binary cross entropy loss of (2) is not a contrastive loss per se, but rather a loss typically used for classification, fed with probabilities. In this case, one could argue that the coincidence head serves as a learned similarity measure between two points in the embedding space, conceptually analogous to the handcrafted scoring functions typically present in the canonical contrastive losses (e.g., cosine similarity in the NT-Xent loss).

4. EXPERIMENTAL SETUP

4.1. Evaluation Methodology

By optimizing the training objectives of (1) and (2), the goal is to learn semantically discriminative audio representations. We train our framework using a superset of the AudioSet training set consisting of around 3M audio clips, while ignoring all available labels. To evaluate the learned representation h we use the trained encoder f_θ as a feature extractor for the two evaluation methodologies considered in past work [14, 15]:

Query by Example Retrieval (QbE). Given a small subset of AudioSet with around 100 examples per class, cosine distance is computed between all the within-class target pairs, and all (present, not-present) pairs as non-target trials. Then, we sort the resulting distances in ascending order and compute per-class average precision (AP) of ranking target over non-target trials. Averaging per-class AP leads to the reported *QbE* mAP. This is a direct measurement of the representation semantic consistency without requiring further training.

Downstream Classification with Shallow Model. This is a supervised classification task carried out by training and evaluating a shallow architecture on top of the fixed embeddings previously learned. In particular, we use an MLP with one 512-unit hidden layer and ReLU activation, followed by a 527-way classification layer with sigmoid activation. For this purpose, we use the entire AudioSet training set version and report *classification* mAP. This measures the usefulness of the learned representation for a large-vocabulary downstream supervised classification task.

For every experiment, we train our framework until QbE convergence, which typically occurs between 400k and 600k steps, after which QbE mAP plateaus. We select an encoder checkpoint from this plateau and report the QbE mAP. Then, we use this checkpoint to extract features for the entire AudioSet and conduct the shallow classifier evaluation. After L2-normalizing the embeddings, we train on the AudioSet training set, allowing 5% for validation where we optimize mAP, then report classification mAP on the evaluation set.

4.2. Implementation and Training Details

A critical parameter in the proposed framework is the number of output waveforms in the separation model M, which must be defined in advance. After experimenting with $M = \{2, 4\}$, we decided to use $M = 2$, as results with $M = 4$ were slightly worse for both proxy tasks. We attribute this to the fact that when $M = 4$ it is common to produce near-empty output channels, which is problematic for the creation of positive pairs. (This issue is further explained in Appendix A.) Note that sound separation is only used during the learning of the representation—in our downstream tasks no separation is applied. The DA blocks in Fig. 1 consist first of

Temporal Proximity sampling, i.e., random selection of 0.96 s waveform snippets within the (maximum) 10 s AudioSet clips. Snippets are transformed to log-mel spectrogram patches using a 25 ms Hann window with 10 ms hop, and 64 mel log-energy bands, leading to time-frequency patches of $T \times F = 96 \times 64$. SpecAugment is then applied using *(i)* two frequency masks and two time masks, with a max width of 10 bands or frames, respectively; and *(ii)* time warping with 8 frames as maximum warp [30].

For the encoder we use a convolutional network based on CNN14 from previous work [34]. Our modifications from the original CNN14 include removing Batch Normalization [35] and Dropout [36], which was not found to be beneficial in our experiments. In addition, we substitute the classifier layer and the preceding fully-connected layer by an embedding convolutional layer with d filters, followed by a global max pooling operation to produce the d-dimensional representation h, which is used for downstream tasks. We use $d = 128$ unless stated otherwise. The resulting encoder network has 76M weights. The similarity head consists of an MLP with one hidden layer of 256 units and ReLU non-linearity, followed by an output layer with 128 units, which is the dimension for the metric embeddings z feeding the NT-Xent loss. The coincidence head consists of an MLP with two hidden layers of 512 units and ReLU nonlinearities, followed by an output layer with one single unit to produce coincidence predictions feeding the class-balanced binary cross entropy loss.

Experiments are carried out considering each proxy task individually as well as the full framework trained jointly. When both tasks are trained jointly, the two objectives are optimized from scratch and equally weighted obtaining a joint loss $\mathcal{L}_{joint} = \mathcal{L}_{sim} + \mathcal{L}_{coin}$, using one optimizer to update all the networks. We use the Adam optimizer [37] with a learning rate of 1e-4 whenever the coincidence prediction task is involved, or 3e-4 when only the similarity maximization task is considered. The temperature parameter in (1) is set to $\tau = 0.3$. Learning rates and τ are tuned by optimizing QbE mAP on a validation set different from that used to report results.

The framework is trained on Google Cloud TPUs of 32 cores with a global batch size of 2048, which means local batches of 64 examples per core. Loss contributions and gradients are computed locally in each replica, then aggregated across replicas before applying the gradient update. Contrastive learning approaches typically benefit from comparison with multiple negative examples. In our framework, negative examples are drawn from clips within the current batch at every iteration, as done in previous works [9, 18, 15, 19]. This approach is more practical than relying on a memory bank [38], a memory queue [39], or negative mining techniques to find suitable negatives [14]. However, with this simple approach the quality and diversity of negatives are limited by the batch size (in our case, the local batch size). Recent works show how increasing batch sizes provide solid improvements in visual [9] and audio [19] contrastive representation learning, the latter work utilizing batch sizes of up to 32k examples. Here, we do not explore this avenue and evaluate our proposed approach using a more usual batch size. Based on previous literature [9, 19, 20], if our approach shows promise using the batch size selected for our experiments, it is expected to provide better performance under more favorable conditions given by larger batches.

5. EXPERIMENTS

This Section describes the experiments run using the framework of Fig. 1, or portions of it. For simplicity, in the following Tables the similarity maximization task and the coincidence prediction task are sometimes referred to as *SimCLR* and *CP*, respectively.

Table 1: mAP without sound separation in the front-end (i.e., using only the input mixture). SA = SpecAugment, TP = Temporal Proximity, CP = Coincidence Prediction.

Representation	QbE mAP	Classif. mAP
Log-Mel Spectrogram (baseline)	0.423	0.065
simCLR & SA	0.551	0.196
simCLR & TP	0.591	0.248
simCLR & TP & SA	**0.613**	0.265
CP & TP & SA	0.599	**0.286**

Table 2: mAP using sound separation (SSep) in the front-end and the *SimCLR* back-end. TP is always applied; SpecAugment (SA) is applied as specified.

Comparison	SSep	SA	QbE mAP	Classif. mAP
Mix vs mix (baseline)	-	-	0.591	0.248
Mix vs mix (baseline)	-	✓	0.613	0.265
Mix vs chan	✓	-	0.631	0.272
Mix vs chan	✓	✓	**0.640**	**0.282**
Mix vs any	✓	✓	0.638	0.279
Chan vs chan	✓	✓	0.611	0.254

5.1. Baseline Experiments

Table 1 lists the performance when sound separation is ablated from the front-end in Fig. 1, which is equivalent to all DA blocks being fed by the input mixture x_m. We use log mel spectrogram as a baseline handcrafted representation. As expected, both SpecAugment and TP with the similarity maximization task as back-end substantially outperform the naive mel spectrogram. The effectiveness of TP is noteworthy considering its simplicity; Initially proposed in previous work [14], it has been widely adopted in contrastive learning works [15, 17, 18, 19], some of which use it as the sole augmentation [15, 17]. Note that the approach simCLR & TP is conceptually comparable to the recent COLA [17]. Combining TP and SpecAugment outperforms either one alone, thus validating the composition in the DA blocks of the front-end. Finally, results indicate different tendencies for the two proxy tasks, with the similarity maximization providing better QbE mAP, and the coincidence prediction attaining better classification mAP.

5.2. Sound Separation for Contrastive Representation Learning

We now report the experiments including the unsupervised sound separation block in the front-end, as depicted in Fig. 1. We assess various comparisons enabled by sound separation preprocessing, namely: *(i)* comparing the input mixture with one of the separated channels (*mix vs chan*); *(ii)* comparing the two separated channels (*chan vs chan*); or *(iii)* comparing the input mixture with anything else, i.e., either with the input mixture or with one of the separated channels (*mix vs any*). Table 2 shows the performance with the similarity maximization (*SimCLR*) task as back-end. By looking at the first rows of Table 2, we can benchmark SpecAugment and sound

Table 3: mAP using sound separation in the front-end and the CP back-end. TP and SpecAugment are applied.

Comparison	QbE mAP	Classif. mAP
Mix vs mix (baseline)	0.599	0.286
Mix vs chan	**0.619**	**0.293**
Chan vs chan	0.590	0.283

separation. We see that sound separation preprocessing (third row) provides a bigger boost in both metrics compared with SpecAugment (second row), yet the best performance is obtained from their composition (fourth row). This trend for the *mix vs chan* comparison also holds for the other contrastive setups.

Results indicate that comparing the input mixture with the separated channels provides substantially better representations than the baseline approach of leveraging only the input mixture. This confirms the usefulness of sound separation preprocessing for contrastive learning of audio representations. Allowing the input mixture to be compared with itself in addition to the separated channels (*mix vs any*) does not lead to performance boosts. Generally, the performance of *mix vs any* and *mix vs chan* were very similar across the experiments we ran. Hence, we adopt *mix vs chan* as best setup in order to focus on the effect of sound separation. Finally, comparing both separated channels (*chan vs chan*) performs significantly worse, on par with the non-separated baseline (for QbE mAP) or even worse (for classification mAP). If we assume the separation has successfully isolated independent sources in each output, this comparison violates the semantic preservation principle (thus hindering the learning of semantic representations), so we might have expected decrements even larger than the $\approx$0.03 mAP with respect to *mix vs chan* for both metrics. After inspection of a few dozen separation examples, we identify two potential explanations for this observation: First, the result of the separation algorithm is not always perfect. This depends on the complexity of the input mixture—this is to be expected considering the great diversity of AudioSet clips. When this happens, the same source can be present in both separated channels. Second, even when the separation is satisfactory, there are some classes that retain a semantic relationship, e.g., two different instruments from the same family, or two different vocalizations from the same or similar animals. When used as a pair of positives, their relationship may still provide a useful learning signal compared to pairs of unrelated negative examples.

Table 3 shows the performance with the coincidence prediction (*CP*) task as back-end. Similar to the *SimCLR* back-end (Table 2), we again observe that the *mix vs chan* comparison yields top performance, outperforming the no-separation baseline. Comparing both separated channels (*chan vs chan*) again leads to the worst results, in this case underperforming the baseline for QbE mAP, while being on par in terms of classification mAP. In addition to corroborating the utility of sound separation in the front-end, these results also demonstrate that *CP* benefits from composing augmentations, which was not explored in previous work [15], where only TP is used.

Comparing performance across both proxy tasks, we notice that *SimCLR* always yields the best QbE mAP while *CP* produces top classification mAP. This could be due to a better alignment between *SimCLR*'s underlying principle (maximizing/minimizing the cosine similarity between positives/negatives) and the QbE retrieval mAP (computed by ranking pairwise cosine distances). Finally, regarding the performance using the *chan vs chan* comparison, *CP* shows significantly better classification mAP than similarity maximization (specifically, a mAP on par with the latter's best case). This accords with our intuition that *CP* is tolerant of semantic differences between positives due to its weaker assumptions. However, for QbE mAP, the opposite behaviour is observed, presumably because this tolerance does not help the QbE objective.

5.3. Separation Processing at Different Convergence States

In the previous Section we show that sound separation is beneficial for our tasks, even when the separation is less than perfect as can occur when input mixtures are difficult to separate. This leads us to ask whether the processing provided by a separation model *before* convergence can also be a valid form of augmentation for contrastive representation learning. To answer this, we experiment with separation examples generated by multiple training checkpoints of a single separation network. We view the separation checkpoints as *audio processors* that implement complex modifications on the incoming audio. A qualitative assessment of output streams as learning progresses indicates four types of processors corresponding to four convergence states, and we empirically characterize their behavior as follows.[1] (Fig. 2 shows example spectrograms of the separated channels for each of these processors given the same input mixture.)

- **Separation after full convergence (S2, 1.7M steps)**. This is the separation model used for experiments in Sec. 5.2.
- **Separation before convergence (S1, 5k steps)**. Separation performance is more limited.
- **Filtering with early training model (F, 500 steps)**. Outputs are produced by the separation model very early during training. After $\sim$500 steps, sources are not separated, and the output channels are differently filtered versions of the input. Most sources in the input are present in all outputs, but often with different levels/spectral content, such as different spectro-temporal modulations.
- **Noise with untrained model (N, 0 step)**. Outputs are produced by the separation model untrained. They feature a clearly audible, wideband structured noise, correlated with the input signal. Audio artifacts are sometimes present. Both output channels are very similar.

Table 4 shows results of substituting S2 with the other identified processors, while keeping the rest of the framework as in Fig. 1. By looking at the top left section of Table 4, two observations can be made: First, the four processors all provide valid forms of augmentation to generate positive views for contrastive learning. While sound separation is beneficial (S2), a poorer separation is also valuable (S1) and the earlier checkpoints of the separation network (which do not actually provide separation) are also useful (F and N). Second, an untrained, quasi-random TDCN++ provides structured noise that surprisingly yields the best single-checkpoint performance (N).

Following the common practice of composing augmentations to achieve more powerful representations [9, 18], we investigate combining the processors. The bottom left section of Table 4 shows the best results obtained when combining processors using the *OR* rule, that is, applying only one randomly selected processor at a time. It can be seen that sound separation and the quasi-random TDCN++ noise turn out to be complementary augmentations, result-

[1] Note that the description of every *processor* is approximate and somewhat dependent on the input's complexity. For example the S1 model could provide a good separation when fed with an easy mixture.

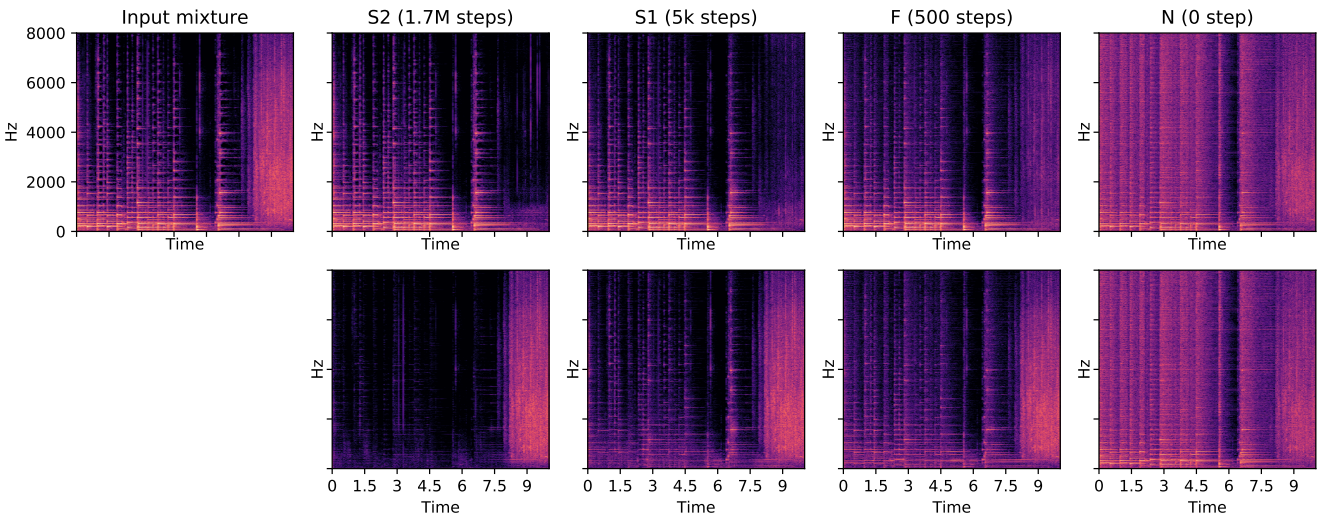

Figure 2: Spectrograms of the two separated channels obtained with four checkpoints (S2, S1, F, N) of the same separation model, given one input mixture (top left). The input mixture contains a guitar melody (up to ≈8 s) followed by applause. For illustration purposes, this is a simple case where the separation is purely temporal (i.e., sources do not overlap). The general case features overlapping sources.

Table 4: mAP using different checkpoints of the separation model as learning progresses (top), as well as some combinations (bottom). As back-end, the *SimCLR* task is used (left), as well as the two proxy tasks trained jointly (right). TP and SpecAugment are applied. Comparison is always *mix vs chan*.

Models	**SimCLR**		**SimCLR & CP**	
	QbE	**Classif.**	**QbE**	**Classif.**
S2 (1.7M)	0.640	0.282	0.649	0.289
S1 (5k)	0.639	0.283	0.651	0.293
F (500)	0.651	0.280	0.659	0.297
N (0)	**0.659**	**0.286**	**0.663**	**0.301**
S2 ∨ F	0.653	0.283	0.658	0.300
S2 ∨ N	0.660	**0.297**	0.671	0.306
S2 ∨ F ∨ N	**0.667**	0.285	**0.672**	**0.310**

ing in a more beneficial composition. Adding the F processor seems to provide lift for QbE mAP, but not for classification mAP. Applying two processors in cascade to every example does not improve performance.

5.4. Joint Learning Framework

Lastly, the right side of Table 4 lists the results when training the entire framework of Fig. 1, jointly optimizing both proxy tasks. We observe trends similar to using the *SimCLR* back-end alone (left side of Table 4), but with increased performance. When compared to the *CP* back-end alone (i.e., S2 in Table 4 vs second row of Table 3), QbE mAP is improved by a large margin whereas classification mAP is on par. Overall, while the boost from jointly optimizing both tasks is sometimes not very large, it is consistent across almost all cases considered, both for individual processors as well as their combinations. We also note that the changes needed in the framework to accommodate a second task are minimal—only an additional MLP head and corresponding loss function—and the training setup carries no modifications—both tasks are trained jointly from scratch using one optimizer. Adopting a curriculum learning instead could enhance performance [15].

Table 5: Comparison with previous work using shallow model classification. mAP reported is classification mAP. MM = Multimodal approach.

Method	d	**MM**	**mAP**
Unsupervised triplet [14]	128	-	0.244
C^3 [15]	128	✓	0.285
Separation-based framework (ours)	128	-	0.310
CPC [16]	512	-	0.277
Separation-based framework (ours)	1024	-	0.326
MMV [40]	2048	✓	0.309
Multi-format [19]	2048	-	**0.329**
L^3 [33]	6144	✓	0.249
Supervised PANN [34]	-	-	0.439
Supervised PSLA [41]	-	-	0.474

Results suggest that the key ingredient is not the quality of the sound separation process, but rather the combination of diverse processing provided by the separation model as its learning progresses. While training a separation model requires a certain effort, once it is done several non-parametric augmentation generators become available, facilitating the generation of useful positive examples. While we choose a MixIT-based TDCN++, any source separation methodology could be used (and there may be additional benefit to using supervised systems).

5.5. Comparison with Previous Work

Table 5 compares our best setup with previous work on the downstream classification task (see Sec. 4.1). Works are grouped by ascending embedding dimensionality, d. Results are strictly comparable only in the top section as those works are the only ones using the same training data, evaluation protocol and downstream embedding dimensionality, $d = 128$. Note that C^3 is based on audio-video

multimodality for representation learning [15], while our proposed framework outperforms it using only audio. We also compare our system with works that use somewhat different evaluation settings in terms of, e.g., training data, embedding dimensionality or shallow classifier type, thus hindering a fair comparison. For example, most previous works use larger d values, ranging from 512 to 6144; we expect performance to improve to some extent as d increases [34]—the embedding representation contains more information that can be leveraged by the shallow model in the downstream task. We confirm this by increasing our d from 128 to 1024, which yields an absolute increase of 0.016 mAP. Some of these works leverage multimodal data for training such as audio-video (L^3 [33]) or audio-video-text (MMV [40]), while reporting worse performance than our lower-d audio-only framework. The current unsupervised state-of-the-art on this task is achieved by a contrastive learning setup that maximizes the agreement between raw audio and its spectral representation [19]. Among several variants proposed by the authors, we select the one that is more comparable to our proposed framework (i.e., using only log-mel as input and one encoder). Our reported performance is on par with this approach (0.326 vs 0.329) despite it leveraging higher d (2048 vs our 1024) and a much larger batch size (32768 vs our 64), potentially having an impact on performance as discussed in Sec. 4.2. Better results are reported in previous work [19] by combining two different encoders (one per audio format) and concatenating their output representations. Finally, for reference, we include the current supervised state-of-the-art on this task. PANN is based on data balancing and augmentation [34], whereas PSLA makes use of a collection of training techniques to boost performance (ImageNet pretraining, data balancing and augmentation, label enhancement and model aggregation) [41].

6. CONCLUSION

We have presented a sound separation-based contrastive learning framework for unsupervised audio representation learning. We show that sound separation can be seen as a valid augmentation to generate positive views for contrastive learning, and that learning to associate sound mixtures with their constituent separated channels elicits semantic structure in the learned representation, outperforming comparable systems without separation. We demonstrate that sound separation can be successfully combined with other commonly-used augmentations to define more challenging proxy tasks. We discover that the transformations provided by different checkpoints of the same separation model as learning progresses are valid, and sometimes complementary, forms of augmentation for generating positives. In addition, we show the benefit of jointly training the proxy tasks of similarity maximization and coincidence prediction. By appropriately combining several separation processors followed by the joint optimization of the two proxy tasks, we obtain downstream AudioSet classification results competitive with the state-of-the-art in unsupervised representations, and outperforming multimodal approaches.

7. ACKNOWLEDGMENT

The authors would like to thank Malcolm Slaney (Google Research) and Luyu Wang (DeepMind) for helpful feedback and discussions.

8. REFERENCES

[1] E. Fonseca, S. Hershey, M. Plakal, D. P. Ellis, A. Jansen, and R. C. Moore, "Addressing missing labels in large-scale sound event recognition using a teacher-student framework with loss masking," *IEEE Signal Processing Letters*, vol. 27, pp. 1235–1239, 2020.

[2] B. Zhu, K. Xu, Q. Kong, H. Wang, and Y. Peng, "Audio tagging by cross filtering noisy labels," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2073–2083, 2020.

[3] E. Fonseca, M. Plakal, D. P. W. Ellis, F. Font, X. Favory, and X. Serra, "Learning Sound Event Classifiers from Web Audio with Noisy Labels," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2019.

[4] E. Fonseca, F. Font, and X. Serra, "Model-agnostic Approaches to Handling Noisy Labels When Training Sound Event Classifiers," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2019.

[5] K. Kawakami, L. Wang, C. Dyer, P. Blunsom, and A. van den Oord, "Learning robust and multilingual speech representations," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, 2020, pp. 1182–1192.

[6] M. Rivière, A. Joulin, P.-E. Mazaré, and E. Dupoux, "Unsupervised pretraining transfers well across languages," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2020, pp. 7414–7418.

[7] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *arXiv preprint arXiv:2006.11477*, 2020.

[8] J. Shor, A. Jansen, R. Maor, O. Lang, O. Tuval, F. de Chaumont Quitry, M. Tagliasacchi, I. Shavitt, D. Emanuel, and Y. Haviv, "Towards learning a universal non-semantic representation of speech," in *Proc. INTERSPEECH*, 2020.

[9] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning (ICML)*, 2020, pp. 1597–1607.

[10] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv preprint: 2003.04297*, 2020.

[11] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, *et al.*, "Bootstrap your own latent: A new approach to self-supervised learning," *arXiv preprint arXiv:2006.07733*, 2020.

[12] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint: 1807.03748*, 2018.

[13] A. Baevski, S. Schneider, and M. Auli, "vq-wav2vec: Self-supervised learning of discrete speech representations," *arXiv preprint arXiv:1910.05453*, 2019.

[14] A. Jansen, M. Plakal, R. Pandya, D. P. W. Ellis, S. Hershey, J. Liu, R. C. Moore, and R. A. Saurous, "Unsupervised learning of semantic audio representations," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2018.

[15] A. Jansen, D. P. Ellis, S. Hershey, R. C. Moore, M. Plakal, A. C. Popat, and R. A. Saurous, "Coincidence, categorization, and consolidation: Learning to recognize sounds with minimal supervision," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2020, pp. 121–125.

[16] L. Wang, K. Kawakami, and A. van den Oord, "Contrastive predictive coding of audio with an adversary," *Proc. Interspeech 2020*, pp. 826–830, 2020.

[17] A. Saeed, D. Grangier, and N. Zeghidour, "Contrastive learning of general-purpose audio representations," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2021.

[18] E. Fonseca, D. Ortego, K. McGuinness, N. E. O'Connor, and X. Serra, "Unsupervised contrastive learning of sound event representations," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2021.

[19] L. Wang and A. van den Oord, "Multi-format contrastive learning of audio representations," in *Self-Supervised Learning for Speech and Audio Processing Workshop, NeurIPS*, 2020.

[20] P. H. Le-Khac, G. Healy, and A. F. Smeaton, "Contrastive representation learning: A framework and review," *IEEE Access*, vol. 8, pp. 193 907–193 934, 2020.

[21] N. Turpault, S. Wisdom, H. Erdogan, J. R. Hershey, R. Serizel, E. Fonseca, P. Seetharaman, and J. Salamon, "Improving sound event detection in domestic environments using sound separation," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, 2020, pp. 205–209.

[22] N. Turpault, R. Serizel, S. Wisdom, H. Erdogan, J. Hershey, E. Fonseca, P. Seetharaman, and J. Salamon, "Sound event detection and separation: a benchmark on Desed synthetic soundscapes," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2021.

[23] Y. Tian, C. Sun, B. Poole, D. Krishnan, C. Schmid, and P. Isola, "What Makes for Good Views for Contrastive Learning?" *arXiv preprint: 2005.10243*, 2020.

[24] S. Wisdom, E. Tzinis, H. Erdogan, R. J. Weiss, K. Wilson, and J. R. Hershey, "Unsupervised sound separation using mixture invariant training," in *Advances in Neural Information Processing Systems*, 2020.

[25] I. Kavalerov, S. Wisdom, H. Erdogan, B. Patton, K. Wilson, J. Le Roux, and J. R. Hershey, "Universal sound separation," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2019, pp. 175–179.

[26] S. Wisdom, H. Erdogan, D. Ellis, R. Serizel, N. Turpault, E. Fonseca, J. Salamon, P. Seetharaman, and J. Hershey, "What's all the FUSS about free universal sound separation data?" in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2021.

[27] 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 International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2017.

[28] Y. Luo and N. Mesgarani, "Conv-TasNet: Surpassing ideal time–frequency magnitude masking for speech separation," *IEEE/ACM transactions on audio, speech, and language processing*, vol. 27, no. 8, pp. 1256–1266, 2019.

[29] S. Wisdom, J. R. Hershey, K. Wilson, J. Thorpe, M. Chinen, B. Patton, and R. A. Saurous, "Differentiable consistency constraints for improved deep speech enhancement," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2019.

[30] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," *Proc. Interspeech 2019*, pp. 2613–2617, 2019.

[31] A. Nandan and J. Vepa, "Language Agnostic Speech Embeddings for Emotion Classification," in *International Conference on Machine Learning (ICML) Workshop*, 2020.

[32] L. Wiskott and T. J. Sejnowski, "Slow feature analysis: Unsupervised learning of invariances," *Neural computation*, vol. 14, no. 4, pp. 715–770, 2002.

[33] R. Arandjelovic and A. Zisserman, "Look, listen and learn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 609–617.

[34] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "PANNs: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020.

[35] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning (ICML)*, 2015, pp. 448–456.

[36] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[38] Z. Wu, Y. Xiong, S. Yu, and D. Lin, "Unsupervised Feature Learning via Non-Parametric Instance Discrimination," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[39] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 9729–9738.

[40] J.-B. Alayrac, A. Recasens, R. Schneider, R. Arandjelović, J. Ramapuram, J. De Fauw, L. Smaira, S. Dieleman, and A. Zisserman, "Self-supervised multimodal versatile networks," *arXiv preprint arXiv:2006.16228*, 2020.

[41] Y. Gong, Y.-A. Chung, and J. Glass, "PSLA: Improving audio event classification with pretraining, sampling, labeling, and aggregation," *arXiv preprint arXiv:2102.01243*, 2021.

A. SELECTION OF NUMBER OF SEPARATED CHANNELS

A critical parameter in the proposed framework is the number of output waveforms in the separation model, M, which must be defined at train time. Upon inspection of a few AudioSet clips selected

randomly, we realize that many clips contain one or two dominant sources (i.e., in the foreground, lasting long within the clip), sometimes accompanied by additional sources (either in the foreground but very short, e.g., impact sounds, or in the background).[2] We therefore ran preliminary experiments with $M = \{2, 4\}$ and saw that results using $M = 4$ were slightly worse for both proxy tasks. We attribute this to the fact that when $M = 4$ it is not uncommon to find output channels which are almost empty, filled with mild background noise, or with sound sources active only in a very short period of time. We hypothesize that using these channels to create positive pairs can hurt performance.

To confirm our hypothesis, we designed simple heuristics (based on energy and cosine similarity) to detect these quasi-empty channels, in order to allow discarding the "worst" channel in every contrastive setup, thus keeping only the other 3 channels from where to pool positive examples. This led to a small but consistent performance improvement, confirming our initial hypothesis, yet still underperforming results with $M = 2$. While further optimizations to allow using $M = 4$ could be pursued, for simplicity we decided to adopt $M = 2$ for our experiments, which is the minimal separation possible. In some cases, the two output waveforms coming out of the separation model will contain one source each, whereas in other cases they will contain several sources each. Consequently, we use the term separated *channels* (and not *sources*) as it is deemed more appropriate. We believe $M = 2$ is sufficient to evaluate our hypothesis of sound separation serving as a valid transformation for view generation in contrastive learning.

[2]The main exception to this rule is music segments.