Improving label efficiency through multitask learning on auditory data

Anonymous Author(s)
Affiliation
Address
email

Abstract

Collecting high-quality, large scale datasets typically requires significant resources. The aim of the present work is to improve the label efficiency of large neural networks operating on audio data through multitask learning with self-supervised tasks on unlabeled data. To this end, we trained an end-to-end audio feature extractor based on WaveNet that feeds into simple, yet versatile task-specific neural networks. We describe three self-supervised learning tasks that can operate on any large, unlabeled audio corpus. We demonstrate that, in a scenario with limited labeled training data, one can significantly improve the performance of a supervised classification task by simultaneously training it with these additional self-supervised tasks. We show that one can improve performance on a diverse sound events classification task by nearly 8.94% when jointly trained with up to three distinct self-supervised tasks. This improvement scales with the number of additional auxiliary tasks as well as the amount of unlabeled data. We also show that incorporating data augmentation into our multitask setting leads to even further gains in performance.

1 Introduction

Deep neural networks (DNNs) [16] are the bedrock of state-of-the-art approaches to modeling and classifying auditory data [2, 16, 22, 39, 40]. However, these data-hungry neural architectures are not always matched to the available training resources, and the creation of large-scale corpora of audio training data is usually costly and time-consuming. While labeled datasets are quite scarce, we have access to virtually infinite sources of unlabeled data, which makes effective unsupervised learning an enticing research direction. Here we aim to develop techniques that enable models to generalize better by incorporating auxiliary self-supervised auditory tasks into the training phase [12, 13, 27].

Our main contributions in this paper are two fold: the successful identification of appropriate self-supervised audio-related tasks and the demonstration that they can be trained jointly with data-constrained supervised tasks in order to significantly improve performance. We also show how to use WaveNet as a general feature extractor capable of providing rich audio representations using raw waveform data as input. We hypothesize that by learning multi-scale hierarchical representations from raw audio, WaveNet-based models are capable of adapting to subtle variations within tasks in an efficient and robust manner. Our approach is quite general and flexible.

The remainder of the paper is organized as follows: after covering related work in section 2, we proceed to describe the model and the auditory tasks on which the model was trained in section 3. In section 4 we describe our experiments and report the results we obtained when training a shared acoustic model with multiple tasks in section 5. We close with a summary of the main takeaways of this work and propose some interesting future directions in section 6.

Submitted to 32nd Conference on Neural Information Processing Systems (NIPS 2018). Do not distribute.

2 Related Work

Principally, multitask learning is about learning two or more tasks simultaneously within a single shared model. A single model can only learn multiple tasks if they are related in some way. Task relatedness, as a concept, is poorly defined in the field, though it hinges on the presence of common structure within the input that is relied upon by each task [6]. Such structure has been described for decades in the literature on sensory environments, with Gabor filters and gammatone filters underlying much of visual and auditory processing, respectively [1, 21, 37]. This suggests that models trained to accomplish many tasks should be able to synergize to uncover this underlying structure, enabling better single-task performance with smaller amounts of data per-task. There are many ways in which models can be designed to uncover this common structure [23]. Most existing approaches to multitask learning attempt to learn a single non-trivial general-purpose representation [5]. While other intriguing approaches have been proposed [25], our work largely belongs to this first category, so we will focus our discussion there. For a more thorough review, see [23, 34].

Multitask learning [6] has been studied across several fields in machine learning. More recently it has been incorporated into a variety of deep neural network models, addressing problems in the domains of vision [5, 26, 33, 41], speech [7, 10, 36], natural language processing [8, 9, 15, 38], and reinforcement learning [3, 11, 17]. For instance, Bilen & Veldadi showed that a single visual model could learn 10 distinct visual tasks operating on 10 datasets [5]. The model described therein outperformed baseline single-task networks, suggesting that it was able to take advantage of the shared representation space to pool the error signals from seemingly disparate classification tasks. It has also been shown that noise-robust speech recognition performance can be improved by adding a denoising auxiliary task to the main classification task [32]. Though the utility of a shared representation space may not be surprising in these instances (one might expect that supplementary denoising should aid in producing a noise-free representation), modern deep learning models remain strikingly oriented toward single tasks. Shared representations are not only useful for single modality models. Kaiser et al. [18] have shown that a single, albeit very large, model is capable of jointly learning 8 tasks across 3 different modalities.

While shared representations allow models to pool data from different datasets, the problem persists that the cleanly labeled datasets that have permitted so much progress in deep learning are painstaking to come by. One proposed solution that has gained traction is to use self-supervised learning to take advantage of unlabeled data. Self-supervised tasks are those where the input, or a simple transformation of the inputs, provides its own label. Recent self-supervised work in the visual domain has shown promising results, leveraging unlabeled data using tasks like inpainting for image completion [28, 31], image colorization [20, 43], and motion segmentation [30].

In this work, we find that simultaneously training on multiple diverse self-supervised audio tasks yields strong performance gains on data-constrained supervised classification tasks. Though multitask learning shares much in common with transfer learning, it has no inherent task primacy. For the sake of expositional clarity, however, it is often easiest to think about multitask learning as being composed of a main task and a set of supporting auxiliary tasks. In the work described here, the main task is a supervised classification task, *viz.* sound events classification task which we refer to as audio tagging for the remainder of the paper, and the auxiliary tasks are three self-supervised tasks: next-step prediction, noise reduction, and upsampling.

3 Model Architecture

One approach to multitask learning via shared representations would be to enforce parameter sharing across tasks. In our setting, this is implemented using a network with a trunk comprised of a stack of layers shared across tasks, augmented by a set of specialized heads specific to individual tasks (see Figure 1 and Figure 2). The heads are standalone neural networks driven by inputs emitted by the trunk. We chose to keep the heads as “lightweight" as possible by giving them just enough capacity to solve their designated tasks, thus forcing the shared trunk to model as much of the shared representation space as possible. During training, task-specific input data is fed into the trunk, and in turn, the trunk’s output is routed to the appropriate task-specific head. The trunk’s parameters are simultaneously updated with respect to all tasks. While the parameters in the specific heads are not directly shared across tasks, they nonetheless interact with each other since the trunk’s parameters are updated using gradients contributed by all the heads.

3.1 Shared Trunk

Although audio tag classification does not require the fine temporal resolution found in raw audio waveforms, our chosen auxiliary tasks (or any arbitrary auditory task for which we may desire our model to be sufficient) require higher temporal resolutions. To satisfy this, we chose to build our model following the WaveNet architecture [39].

WaveNet models are autoregressive networks capable of processing high temporal resolution raw audio signals. Models from this class are ideal in cases where the complete sequence of input samples is readily available. WaveNet models employ causal dilated convolutions to process sequential inputs in parallel, making these architectures faster to train compared to RNNs which can only be updated sequentially.

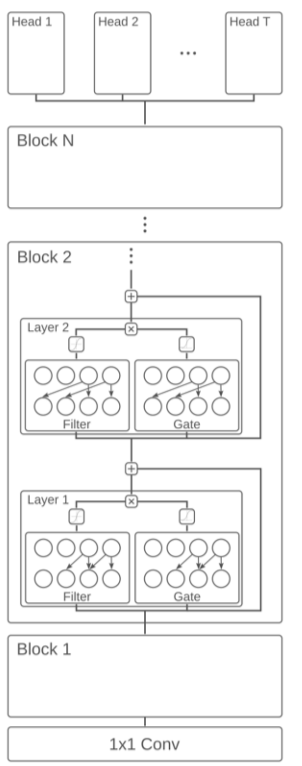

Figure 1: Model architecture. Multiple tasks are processed using small, task-specific neural networks built atop a task-agnostic trunk. The trunk architecture principally follows the structure of WaveNet, with several blocks of stacked, dilated, and causal convolutions between every convolution layer. Outputs from the trunk are fed into task-specific heads (details in Section 3.1).

As shown Figure 1, our WaveNet trunk is composed of N blocks, where each block consists of S dilated causal convolution layers, with dilation factors increasing from 1 to $2^S - 1$, residual connections and saturating nonlinearities. We label the blocks using $b = 1, \cdots, N$. We use indices $\ell \in [1 + (b-1)S, bS]$ to label layers in block b. Each layer, ℓ, of the WaveNet trunk consists of a "residual atom" which involves two computations, labeled as "Filter" and "Gate" in the figure. Each residual atom computation produces a hidden state vector $h^{(\ell)}$ and a layer output $x^{(\ell)}$ defined via

$$\begin{aligned} h^{(\ell)} &= \sigma\big(W_{gate}^{(\ell)} \circledast_\ell x^{(\ell-1)}\big) \odot \tanh\big(W_{filter}^{(\ell)} \circledast_\ell x^{(\ell-1)}\big) \\ x^{(\ell)} &= x^{(\ell-1)} + h^{(\ell)} \end{aligned}$$

where $\odot$ denotes element-wise products, $\circledast$ represents the regular convolution operation, $\circledast_\ell$ denotes dilated convolutions with a dilation factor of $2^{\ell \bmod bS}$ if ℓ is a layer in block $b+1$, σ denotes the sigmoid function and $W_{gate}^{(\ell)}$ and $W_{filter}^{(\ell)}$ are the weights for the gate and filter, respectively.

The first ($\ell = 0$) layer – represented as the initial stage marked "1×1 Conv" in Figure 1 – applies causal convolutions to the raw audio waveforms $X = (X_1, X_2, \cdots, X_T)$, sampled at 16 kHz, to produce an output $x^{(0)} = W^{(0)} \circledast X$.

Given the structure of the trunk laid out above, any given block b has an effective receptive field of $1 + b(2^S - 1)$. Thus the total effective receptive field of our trunk is $\tau = 1 + N(2^S - 1)$. Following an extensive hyperpameter search over various configurations, we settled on [$N = 3$] blocks comprised of [$S = 6$] layers each for our experiments. Thus our trunk has a total receptive field of $\tau = 190$, which corresponds to about 12 milliseconds of audio sampled at 16kHz.

3.2 Task-specific Heads

As indicated above, each task-specific head is a simple neural network whose input data is first constrained to pass through a trunk that it shares with other tasks. Each head is free to process this input to its advantage, independent of the other heads.

Each task also specifies its own objective function, as well as a task-specific optimizer, with customized learning rates and annealing schedules, if necessary. We arbitrarily designate supervised tasks as the primary tasks and refer to any self-supervised tasks as auxiliary tasks. In the experiments reported below, we used "audio tagging" as the primary supervised classification task and "next-step prediction", "noise reduction" and "upsampling" as auxiliary tasks training on various amounts of unlabeled data. The parameters used for each of the task specific heads can be found in Table 3 of the accompanying supplement to this paper.

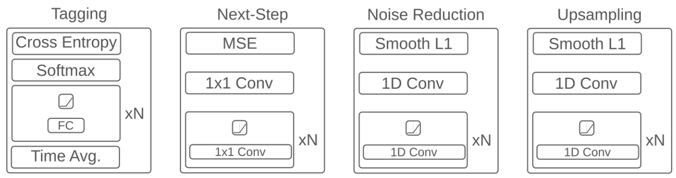

Figure 2: The head architectures were designed to be simple, using only as few layers as necessary to solve the task. Simpler head architectures force the shared trunk to learn a representation suitable for multiple audio tasks.

3.2.1 Next-Step Prediction

The next-step prediction task can be succinctly formalized as follows: given a sequence $\{x_{t-\tau+1}, \cdots, x_t\}$ of frames of an audio waveform, predict the next value x_{t+1} in the sequence. This prescription allows one to cheaply obtain arbitrarily large training datasets from an essentially unlimited pool of unlabeled audio data.

Our next-step prediction head is a 2-layer stack of 1×1 convolutional layers with ReLU nonlinearities in all but the last layer. The first layer contains 128 units, while the second contains a single output unit. The head takes in τ frames of data from the trunk, where τ is the trunk's effective receptive field, and produces an output which represents the model's prediction for the next frame of audio in the sequence. The next-step head treats this as a regression problem, using the mean squared error of the difference between predicted values and actual values as a loss function, i.e. given inputs $\{x_{t-\tau+1}, \cdots, x_t\}$, the head produces an output y_t from which we compute a loss $\mathcal{L}_{\text{MSE}}(t) = (y_t - x_{t+1})^2$ and then aggregate over the frames to get the total loss.

We would like to note that the original WaveNet implementation treated next-step prediction as a classification problem, instead predicting the bin-index of the audio following a μ-law transform. We found that treating the task as a regression problem worked better in multitask situations but make no claims on the universality of this choice.

3.2.2 Noise-Reduction

In defining the noise reduction task, we adopt the common approach of treating noise as an additive random process on top of the true signal: if $\{x_t\}$ denotes the clean raw audio waveform, we obtain the noisy version via $\hat{x}_t := x_t + \xi_t$ where ξ_t an arbitrary noise process. For the denoising task, the model is trained to predict the clean sample, x_t, given a window $\left\{\hat{x}_{t-\frac{1}{2}(\tau-1)}, \cdots, \hat{x}_{t+\frac{1}{2}(\tau-1)}\right\}$ of noisy samples. Formally speaking, the formulation of the next-step prediction and denoising tasks are nearly identical, so it should not be surprising to find that models with similar structures are well-adapted to solving either task. Thus, our noise reduction head has a structure similar to the next-step head. It is trained to minimize a smoothed L1 loss between the clean and noisy versions of the waveform inputs, i.e. for each frame t, the head produces an output $\hat{y}_t$, and we compute the loss

$$\mathcal{L}_{\text{smooth L1}}(t) = \begin{cases} \frac{1}{2}|\hat{y}_t - x_t|^2 & \text{if } |\hat{y}_t - x_t| < 1 \\ |\hat{y}_t - x_t| - \frac{1}{2} & \text{if } |\hat{y}_t - x_t| \geq 1 \end{cases} \tag{1}$$

and then aggregate over frames to obtain the total loss. We used the smooth L1 loss because it provided a more stable convergence for the denoising task than mean squared error.

3.2.3 Upsampling

In the same spirit as the denoising task, one can easily create an unsupervised upsampling task by simply downsampling the audio source. The downsampled signal serves as input data while the original source serves as the target. Upsampling is an analog of the "super-resolution" task in computer vision.

For the upsampling task, the original audio was first downsampled to 4 kHz using the resample method in the librosa python package [24]. To keep the network operating at the same time scale for all auxiliary tasks, we repeated every time-point of the resampled signal 4 times so as to mimic the original signal's 16 kHz sample rate. The job of the network is then to infer the high frequency information lost during the transform.

Again, given the formal similarity of the upsampling task to the next-step prediction and noise-reduction tasks, we used an upsampling head with a structure virtually identical to those described above. As with the denoising task, we used a smooth L1 loss function (see eqn. (1) above) to compare the estimated upsampled audio with the original.

3.2.4 Audio Tag Classification

All of the tasks described above are entirely self-supervised and can make use of vast amounts of unlabeled data. In contrast, the audio tagging task is a classification problem that requires labeled data for training.

Since the WaveNet trunk produces outputs with a temporal structure, our audio tagging head first reduces the trunk's output across the time axis to produce a single output vector for the entire audio sequence. This is done using a global mean pooling layer, which simply averages over the time axis. On top of this pooling, we use a multilayer perceptron with ReLU nonlinearities and finally a softmax output layer. Training is done by minimizing the cross entropy between the softmax outputs and one-hot encoded audio tag vectors, i.e. if we use $\hat{p}_k$ to denote the one-hot vector corresponding to the kth tag label, and p_k to represent the corresponding softmax output, then $\mathcal{L}_{\text{cross-entropy}} = -\sum_{k\in[1,K]} \hat{p}_k \ln p_k$, where K is the total number of tag labels.

4 Experiments

4.1 Datasets

4.1.1 FSDKaggle2018

FSDKaggle2018 [14] is a dataset collected through Freesound, a sound sharing site with a heterogeneous audio content including sounds from a wide range of real-world environments. The complete dataset contains a total of 11,073 files provided as uncompressed PCM 16 bit, 44.1 kHz, mono audio files which is further subdivided into a training set and a test set. The duration of these audio clips ranges from 300ms to 30s. The training set is composed of 9473 audio clips corresponding to approximately 18 hours of audio which is unequally distributed among 41 categories. The ground truth labels of the training data have varying degrees of reliability, with only 3710 of the audio clips having manually-verified labels and the remaining 5763 having non-verified labels, meaning they were automatically categorized using user-provided metadata. The test set is composed of 1600 audio clips with manually-verified labels which are used for the final scoring.

4.1.2 Librispeech

The Librispeech dataset[1] (comprised of read English speech sampled at 16 kHz) was used as a proxy for a large unlabeled dataset. The models described below were trained using clips from either the "train-clean-100" or "train-other-500 versions". Models trained with 5, 50 and 100 hours of unlabeled data were sourced from "train-clean-100", while the model trained with 500 hours was sourced entirely from "train-other-500". Due to memory constraints, we limited the duration of each utterance to 2 seconds which we obtained by cropping from a random position in the original clip. This dataset was only used to train the auxiliary tasks.

[1] http://www.openslr.org/12/

4.2 Training

We trained the model using raw audio waveform inputs taken from the FSDKaggle2018 and Librispeech datasets. All code for the experiments described here was written in the PyTorch framework [29]. All audio samples were first cropped to two seconds in duration and downsampled to 16 kHz. To normalize for the variation in onset times for different utterances, the 2 seconds were randomly selected from the original clip. Samples shorter than 2 seconds were zero padded. We then scaled the inputs to lie in the interval $[-1, 1]$. The noise-reduction task required noisy inputs which we obtained by adding noise sampled from ChiME3 datasets [4] at a randomly chosen SNR from 10dB to 15dB. The noise types include booth (BTH), on the bus (BUS), cafe (CAF), pedestrian area (PED), and street junction (STR)) . Starting with the main task, we first performed a hyperparameter search over the number of blocks in the trunk, the number of layers per block, the number of layers and units of the main task head, and the learning rate. We tried several values for the number of blocks in the trunk, ranging from 2 to 5. We also varied the number of dilated convolution layers in each block from 3 to 8. We found that the performance and training characteristics of the network were largely unaffected by the exact architecture specifications, though learning rate was often important. We then searched over the depth and width of each auxiliary task head, as well as the learning rate for the head. These searches were done by pairing each task individually with the main task. The final choice of hyper-parameters was made by picking values which gave the best possible performance on both the main task and the auxiliary tasks, heuristically favoring performance on the main task.

We jointly trained the model on all tasks simultaneously by performing a forward pass for each task, computing the loss function for each task, and then calculating the gradients based on a weighted sum of the losses, *viz.* $\mathcal{L}_{\text{total}} = \sum_i \alpha_i \mathcal{L}_i$, where the sum runs over all the tasks. We used a uniform weighting strategy in our current experiments. More advanced weighting strategies showed no benefit for the tagging task (see section 6).

We used the “Adam” optimizer [19] with parameters $\beta_0 = 0.9$, $\beta_1 = 0.99$, $\varepsilon = 10^{-8}$. The learning rate was decayed by a factor of .95 every 5 epochs, as this was found to improve convergence. We used a batch size of 48 across all experiments, since it was the largest batch size permissible by the computational resources available to us. Adding the noise reduction and upsampling tasks required a separate forward propagation of the noisy and downsampled audio, respectively. Exact values for all important parameters of the model can be found in Table 3 of the accompanying supplement to this paper.

5 Results

As discussed above, we used audio tagging as the main task to investigate whether supervised classification of audio could be improved by the addition of self-supervised tasks. The datasets used for these tasks are detailed in Section 4.1. The benchmark model provided by [14] used a 3-layer CNN with log mel spectrogram features as input and obtained a mean average precision at 3 (MAP@3) score of 0.69 on the test set. For our experiments, we also used the MAP@3 [14] along with top-1 classification accuracy as the performance metric.

First, we trained a purely supervised model on 2 seconds of non-silence audio extracted using random cropping from the FSDKaggle2018 dataset. This model was trained using 90% of the training data and the remaining training data was set aside for validation. The final scores were reported on the test set. At the end of training, this baseline model with a single task of audio tagging as the head obtained an MAP@3 score of 0.637. It is not surprising that the baseline model achieves a slightly lower score than the benchmark model. This can be attributed to the fact that the benchmark model does the time averaging of the entire audio signal during training as well as inference. Due to limitations in memory requirements we constrained our sample length to 2 seconds in our model.

5.1 Addition of self-supervised tasks

In this experiment, we added each of the self-supervised tasks to the baseline model discussed above, simultaneously training them using 100 hours of unlabeled data sampled from the Librispeech dataset along with the main supervised task.

We notice that, addition of any self-supervised task showed an average improvement of 4.6% to the MAP@3 score compared to the main task's baseline performance. Adding a pair of tasks gave an average improvement of 4.55% over baseline, showing no improvement over adding a single task. Training with three additional tasks yielded the best results with an improvement of 5.33% over the main task. Looking at MAP@3 scores throughout training showed that convergence in every multitask setting was stable, with gradual improvements for increasing number of tasks. The best performance values on the test sets for a sequence of task additions can be found in Table 1.

The set of experiments described above demonstrate that, for a fixed amount of unlabeled data (100 hours), simultaneously training a supervised task with various self-supervised tasks yields a significant improvement in the main task's performance.

5.2 Varying amounts of unlabeled data

To further test how performance changes with increasing amounts of data, we re-trained our model while varying the amount of unlabeled data used to train the auxiliary tasks. We noticed that even without any additional unlabeled data, the MAP@3 score with three additional
238 tasks was significantly better than the score obtained on a single task. This demonstrates that addition of self-supervised tasks improves the performance of main task.

Increasing the size of the unlabeled data for the auxiliary tasks increases the size of the multitask benefit (Figure 3).The MAP@3 Scores at different levels of unlabaled data showed progressive improvement to 0.656, 0.669, with 5 and 50 hours respectively. We observed a peak MAP@3 score of 0.694 with 500 hours of unlabeled data, which is an improvement of 8.94% over the main task's baseline performance.

5.3 Comparison with Data Augmentation

Next, we explore several approaches to data augmentation and compare them with multitask learning. Previous work has demonstrated the effectiveness of data augmentation through simple techniques, such as noise injection, and pitch shifting [35, 42, 44]. We compared our proposed method with traditional data augmentation strategies by retraining our model only for the main task after applying the aforementioned augmentations to the FSDKaggle2018 training data.

The MAP@3 values for the data augmentation experiments on the test sets can be found in Table 2. We observed a peak MAP@3 score of 0.703 with pitch shifting augmentation which is similar in scale to that of our best multitask performance gains. In an attempt to observe how both the techniques work together, we combined data augmentation with multitask learning and obtain an MAP@3 score of 0.726 which was the best score among all the experiments we conducted.

Table 1: Results showing multitask learning performance gains with audio tagging as the primary classification task along with 100 hours of unlabeled data. TAG=Audio tagging, UP=upsampling, NS=next-step prediction, NR=noise-reduction.

	MAP@3 Score	Classification Accuracy(%)
TAG	0.637	55.31
TAG + NS	0.665	58.15
TAG + NR	0.665	57.77
TAG + UP	0.669	58.54
TAG + NS + NR	0.664	57.88
TAG + UP + NR	0.664	58.14
TAG + NS + UP	0.669	58.27
TAG + NS + UP + NR	0.671	58.40

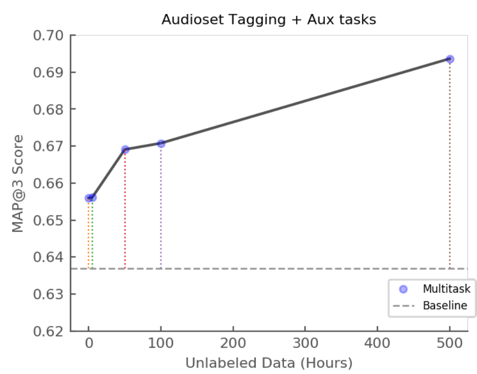

Figure 3: Improved MAP@3 scores with increasing amounts of unlabeled data. Shown are the MAP@3 scores on test set when the main task is trained with 3 auxiliary tasks with 0, 5, 50, 100, and 500 hours of unlabeled data respectively. The amount of labelled data is held constant for the whole experiment. We see a smooth increase in performance with increasing amounts of unlabeled data.

Table 2: Results showing performance gains with data augmentation on audio tagging task. MTL100=Multitask learning with all auxiliary tasks and 100 hours of unlabeled data, NI=noise injection, PS=pitch shifting.

	MAP@3 Score	Classification Accuracy(%)
NI	0.661	57.31
PS	0.703	62.60
PS + MTL100	0.726	64.87

6 Discussion

We investigated our multitask learning framework under two specific evaluation settings: sequentially adding various self-supervised tasks and adding more unlabeled data. We have shown that jointly training a supervised classification task together with multiple self-supervised tasks using a WaveNet-based architecture can significantly improve the performance of the supervised task in situations where one has a limited quantity of labeled data. We have also shown that the performance of the supervised task improves by increasing either the number of auxiliary self-supervised tasks or the quantity of unlabeled data or both. We attain a peak performance boost of 8.94% over the baseline with the inclusion of 3 self-supervised tasks when trained with additional 500 hours of unlabeled data. Finally, our multitask learning scheme further benefits when the training data for the data-constrained task is augmented using standard techniques. Since our results suggest that the performance gain with our approach is additive when used with data augmentation, it may be interesting to use multitask learning with other augmentation approaches to observe if they complement each other in different settings.

We have strived to systematically present our results within a coherent multitask learning framework. For the most part, our methodology follows a straightforward extension of the techniques used in related approaches like transfer learning and self-supervised learning. There is, however, one challenging aspect that deserves more attention: how to best simultaneously optimize a set of arbitrary objective functions. For example, in our setup, the auxiliary tasks are inherently temporal in nature while the supervised classification task does not make use of the temporal aspects of the audio waveform. It is quite plausible that a naive combination of loss functions associated with tasks operating on very different time scales leads to sub-optimal results. While in all our experiments we have simply added the task specific losses to design our final objective, we believe that a better understanding of multiple objective optimization will improve the performance further.

While we have shown that one can effectively utilize multitask learning with unlabeled audio data, many questions remain to be answered. We want to explore if there is a limit to the number of auxiliary tasks that can be added to a main task in the multitask setting and if we can place an upper bound on the amount of improvement that we can expect from such a setting. A more principled notion of task similarity/relationship still need to be investigated with regard to multitask learning in order to know which tasks should be preferred. Intuitively, we expect that when our multitask model learns to simultaneously forecast frames of audio, remove noise from the audio and perform upsampling, it must have formed a representation of the audio. What is this representation? Can it be extracted or distilled? A proper exploration of these questions should enable us to handle a broader range of auditory tasks, hopefully providing a useful tool for tackling deep learning in the limited-data regime.

Supplementary Material

Table 3: Important hyperparameter values for all experimental runs

	Parameter	Value
Trunk	# Blocks	3
	# Layers	6
	# Units	64
Optimizer	Type	Adam
	Learning rate	3×10^{-4}
	Epochs per step	5
	Schedule multiplier	0.95
Audio Tagging Head	# Layers	1
	# Units - hidden	512
	# Units - output	41
	Learning rate	5.37×10^{-5}
	Epochs per step	5
	Schedule multiplier	0.95
Next-step Head	# Layers	2
	# Units - hidden	128
	# Units - output	1
	Learning rate	5×10^{-3}
	Epochs per step	5
	Schedule multiplier	0.95
Noise Reduction Head	# Layers	2
	# Units - output	128
	Filter width	11
	Learning rate	5×10^{-3}
	Epochs per step	5
	Schedule multiplier	0.95
Upsampling Head	# Layers	2
	# Units	128
	# Units - output	1
	Filter width	11
	Learning rate	5×10^{-3}
	Epochs per step	5
	Schedule multiplier	0.95

References

[1] Bruno A. Olshausen and David Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. 381:607–9, 07 1996.

[2] Dario Amodei, Rishita Anubhai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Jingdong Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Erich Elsen, Jesse Engel, Linxi Fan, Christopher Fougner, Tony Han, Awni Y. Hannun, Billy Jun, Patrick LeGresley, Libby Lin, Sharan Narang, Andrew Y. Ng, Sherjil Ozair, Ryan Prenger, Jonathan Raiman, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Yi Wang, Zhiqian Wang, Chong Wang, Bo Xiao, Dani Yogatama, Jun Zhan, and Zhenyao Zhu. Deep speech 2: End-to-end speech recognition in english and mandarin. *CoRR*, abs/1512.02595, 2015.

[3] Jacob Andreas, Dan Klein, and Sergey Levine. Modular multitask reinforcement learning with policy sketches. *CoRR*, abs/1611.01796, 2016.

[4] J. Barker, R. Marxer, E. Vincent, and S. Watanabe. The third 'chime' speech separation and recognition challenge: Dataset, task and baselines. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 504–511, Dec 2015.

[5] Hakan Bilen and Andrea Vedaldi. Universal representations: The missing link between faces, text, planktons, and cat breeds. *CoRR*, abs/1701.07275, 2017.

[6] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, Jul 1997.

[7] D. Chen and B. K. W. Mak. Multitask learning of deep neural networks for low-resource speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(7):1172–1183, July 2015.

[8] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 160–167, New York, NY, USA, 2008. ACM.

[9] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. Natural language processing (almost) from scratch. *CoRR*, abs/1103.0398, 2011.

[10] Amit Das, Mark Hasegawa-Johnson, and Karel Veselý. Deep auto-encoder based multi-task learning using probabilistic transcriptions. In *Proc. Interspeech 2017*, pages 2073–2077, 2017.

[11] Coline Devin, Abhishek Gupta, Trevor Darrell, Pieter Abbeel, and Sergey Levine. Learning modular neural network policies for multi-task and multi-robot transfer. *CoRR*, abs/1609.07088, 2016.

[12] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. *CoRR*, abs/1708.07860, 2017.

[13] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin A. Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. *CoRR*, abs/1406.6909, 2014.

[14] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar 2017.

[15] Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. A joint many-task model: Growing a neural network for multiple NLP tasks. *CoRR*, abs/1611.01587, 2016.

[16] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, Nov 2012.

[17] Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z. Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. *CoRR*, abs/1611.05397, 2016.

[18] Lukasz Kaiser, Aidan N. Gomez, Noam Shazeer, Ashish Vaswani, Niki Parmar, Llion Jones, and Jakob Uszkoreit. One model to learn them all. *CoRR*, abs/1706.05137, 2017.

[19] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[20] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual understanding. *CoRR*, abs/1703.04044, 2017.

[21] Michael S. Lewicki. Efficient coding of natural sounds. *Nature Neuroscience*, 5:356–363, 2002.

[22] Chao Li, Xiaokong Ma, Bing Jiang, Xiangang Li, Xuewei Zhang, Xiao Liu, Ying Cao, Ajay Kannan, and Zhenyao Zhu. Deep Speaker: an End-to-End Neural Speaker Embedding System. 2017.

[23] Mingsheng Long and Jianmin Wang. Learning multiple tasks with deep relationship networks. *CoRR*, abs/1506.02117, 2015.

[24] Brian McFee, Matt McVicar, Oriol Nieto, Stefan Balke, Carl Thome, Dawen Liang, Eric Battenberg, Josh Moore, Rachel Bittner, Ryuichi Yamamoto, Dan Ellis, Fabian-Robert Stoter, Douglas Repetto, Simon Waloschek, CJ Carr, Seth Kranzler, Keunwoo Choi, Petr Viktorin, Joao Felipe Santos, Adrian Holovaty, Waldir Pimenta, and Hojin Lee. librosa 0.5.0, February 2017.

[25] Elliot Meyerson and Risto Miikkulainen. Beyond shared hierarchies: Deep multitask learning through soft layer ordering. *CoRR*, abs/1711.00108, 2017.

[26] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. *CoRR*, abs/1604.03539, 2016.

[27] T. Nathan Mundhenk, Daniel Ho, and Barry Y. Chen. Improvements to context based self-supervised learning. *CoRR*, abs/1711.06379, 2017.

[28] Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. Representation learning by learning to count. *CoRR*, abs/1708.06734, 2017.

[29] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[30] Deepak Pathak, Ross B. Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. *CoRR*, abs/1612.06370, 2016.

[31] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. Context encoders: Feature learning by inpainting. 2016.

[32] Y. Qian, M. Yin, Y. You, and K. Yu. Multi-task joint-learning of deep neural networks for robust speech recognition. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 310–316, Dec 2015.

[33] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. *CoRR*, abs/1705.08045, 2017.

[34] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *CoRR*, abs/1706.05098, 2017.

[35] Justin Salamon and Juan Pablo Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *CoRR*, abs/1608.04363, 2016.

[36] M. L. Seltzer and J. Droppo. Multi-task learning in deep neural networks for improved phoneme recognition. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6965–6969, May 2013.

[37] Evan Campbell Smith. *Efficient Auditory Coding*. PhD thesis, Pittsburgh, PA, USA, 2006. AAI3228006.

[38] G. Tur. Multitask learning for spoken language understanding. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 1, pages I–I, May 2006.

[39] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *CoRR*, abs/1609.03499, 2016.

[40] W Xiong, L Wu, F Alleva, J Droppo, X Huang, and A Stolcke. The Microsoft 2017 Conversational Speech Recognition System. (August), 2017.

[41] Yongxin Yang and Timothy M. Hospedales. Deep multi-task representation learning: A tensor factorisation approach. *CoRR*, abs/1605.06391, 2016.

[42] Shi Yin, Chao Liu, Zhiyong Zhang, Yiye Lin, Dong Wang, Javier Tejedor, Thomas Fang Zheng, and Yinguo Li. Noisy training for deep neural networks in speech recognition. *EURASIP Journal on Audio, Speech, and Music Processing*, 2015(1):2, Jan 2015.

[43] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful image colorization. *CoRR*, abs/1603.08511, 2016.

[44] Yingbo Zhou, Caiming Xiong, and Richard Socher. Improved regularization techniques for end-to-end speech recognition. *CoRR*, abs/1712.07108, 2017.