EdgeL^3: Compressing L^3-Net for Mote-Scale Urban Noise Monitoring

Sangeeta Kumari*§, Dhrubojyoti Roy*§, Mark Cartwright†‡, Juan Pablo Bello†‡, Anish Arora*
* Department of Computer Science and Engineering, The Ohio State University, USA
† Music and Audio Research Laboratory (MARL), New York University, USA
‡ Center for Urban Science and Progress (CUSP), New York University, USA
{kumari.14, roy.174}@osu.edu, {mark.cartwright, jpbello}@nyu.edu, arora.9@osu.edu

***Abstract*—Urban noise sensing in deeply embedded devices at the edge of the Internet of Things (IoT) is challenging not only because of the lack of sufficiently labeled training data but also because device resources are quite limited. Look, Listen, and Learn (L^3), a recently proposed state-of-the-art transfer learning technique, mitigates the first challenge by training self-supervised deep audio embeddings through binary Audio-Visual Correspondence (AVC), and the resulting embeddings can be used to train a variety of downstream audio classification tasks. However, with close to 4.7 million parameters, the multi-layer L^3-Net CNN is still prohibitively expensive to be run on small edge devices, such as "motes" that use a single microcontroller and limited memory to achieve long-lived self-powered operation.**

In this paper, we comprehensively explore the feasibility of compressing the L^3-Net for mote-scale inference. We use pruning, ablation, and knowledge distillation techniques to show that the originally proposed L^3-Net architecture is substantially overparameterized, not only for AVC but for the target task of sound classification as evaluated on two popular downstream datasets. Our findings demonstrate the value of fine-tuning and knowledge distillation in regaining the performance lost through aggressive compression strategies. Finally, we present EdgeL^3, the first L^3-Net reference model compressed by 1-2 orders of magnitude for real-time urban noise monitoring on resource-constrained edge devices, that can fit in just 0.4 MB of memory through half-precision floating point representation.

I. Introduction

Real-time monitoring and mitigation of noise complaints in an urban setting such as New York City [1]–[3] is a topical problem that impacts the physical health and emotional well being of citizens. However, urban deployment of sensing nodes that support city operations combat noise pollution poses technical as well as managerial challenges that make the solution quite hard to scale. System longevity in particular is of paramount importance: the audio sensing nodes must last for months, if not years, with minimal human intervention and often in the absence of wall power. This means that these devices are required to be significantly more resource constrained than wearable-scale Systems on Chips (SoCs) or similar low power variants, such as the Raspberry Pi Zero [4], PocketBeagle [5], or Intel Edison [6]. Even smaller, so-called mote-scale, IoT platforms such as NXP's Vybrid VF6xx [7] or STM's Cortex M7 [8] offer as low as 1-1.5 MB of on-chip RAM, which must then accommodate all sensing, classification, networking and management modules. The unfortunate consequence of this fact is that the inference engine, usually a deep neural net, cannot be allocated say more than 500 KB of run-time memory.

Noise complaint monitoring is well suited to machine learning. An important requirement for on-device machine-learned classification of noise complaints in the urban setting is the generalizability to a variety of contexts, so as to tolerate the dynamic and unforeseen nature of a new deployment environment. This necessitates training the model with a huge corpus of labeled sound data, which is generally unavailable due to the lack of sufficiently labeled domain-specific datasets and the challenges in manually annotating audio sources [9], [10]. Recently, a new Convolutional Neural Net (CNN) architecture called Look, Listen, and Learn (L^3-Net) [11] has been proposed to effectively train an audio embedding by learning associations between audio snippets and video frames (Audio-Visual Correspondence, henceforth referred to as AVC), without the need for explicit labels. As shown in [12], an embedding derived from the L^3 audio subnetwork can outperform other state-of-the-art approaches such as SoundNet [13] or VGGish [14] networks on a variety of downstream tasks with limited data, making it a suitable model to be adopted for our problem. However, the proposed embedding architecture has 4,688,066 parameters. Excluding the downstream classifier, the embedding alone requires 18 MB in the single precision floating point (FP) format, and therefore must be reliably compressed by at least 1-2 orders of magnitude before it can be considered for a realistic edge deployment.

Contributions of this paper. We present (to the best of our knowledge) the first analysis of magnitude-based sparsification, width reduction as well as depth reduction on the L^3 audio embedding for severely resource-constrained edge audio sensing applications. Our experimental evaluation of the accuracy and degree of compression of the L^3 audio model is in terms of the AVC task as well as representative downstream tasks that respectively use two open source public datasets

§ Co-primary authors

The authors would like to thank Jason Cramer, Ho-Hsiang Wu and Justin Salamon for their valuable feedback. This work is supported by NSF award 1544753.

US8K [2] and ESC-50 [15]. The main contributions of this work are as follows:

1) We find that the original L^3 architecture is substantially over-parameterized, since it is possible to achieve up to 95% compression with just 0.22% drop in AVC accuracy and ~1.5% compromise in the downstream tasks.
2) We compare the effects of pruning, fine-tuning, and knowledge distillation in AVC as well as downstream tasks. We find that aggressive compression of the audio subnetwork reduces the AVC and downstream accuracies as expected, but surprisingly, post hoc fine-tuning can help regain the lost performance almost entirely. Similarly, knowledge distillation using the pruned model as a student shows comparable improvements in performance.
3) Orthogonally, we report a series of ablation experiments on the L^3 audio subnetwork in the form of depth and width reduction. On the former, we show the feasibility of reducing 50% of the model size by extracting embeddings out of the penultimate convolutional layer at runtime. On the latter, which involves significant architectural changes to the network and incurs significant degradation in downstream accuracies, we find that fine-tuning can recoup the lost performance by as much as 106%.

As a concrete first step in optimizing the classifier resource utilization on the aforementioned class of constrained edge devices, we are able to successfully compress the L^3 embedding by 1-2 orders of magnitude and produce a 0.4 MB model with half precision FP. We discuss the currently existing implementation challenges of EdgeL^3 and motivate future work towards successful mote-scale realization that combines sparsities with ablation. Finally, we note that we have open-sourced all of our models and code[1].

II. Related Work

CNNs have been used in state-of-the-art models for a wide range of visual and acoustic applications [15]–[19]. While the accuracy of CNNs have been seen to improve with increased width and depth, it is difficult to store and run large models in edge sensing applications. For instance, we are not aware of any significant CNN implemented for a single microcontroller platform. Size compression of CNNs has been an area of active study over in the past few years: methods such as pruning [20]–[26], quantization [27]–[29], low-rank approximation [30], [31], knowledge distillation [32]–[34], efficient convolution techniques [35], [36], hash encoding [37], [38] have been considered by many researchers. Our approach to shrinking the audio model in L^3 involves the use of pruning with fine-tuning as well as knowledge distillation with a pruned student model.

Early works on pruning such Optimal Brain Damage [39] and Optimal Brain Surgeon [40] reduced the number of connections using the local curvature (in terms of the Hessian) of the loss function. Given the overhead of the Hessian computation, Han et al. [27] instead omit the low-weight connections and are able to compress AlexNet by 9X and VGG-16 by 13X. Our work adopts this magnitude-based neuron pruning method. Wen et al. [41] use Lasso regularization on different structures such as filters, channels, filter shapes, and layer depth to learn a compressed structure of deep CNNs. Li et al. [21] prune channels with small incoming weights in trained CNNs, and then fine-tune the network to regain accuracy pruning whole filter/channels. One of our ablation studies uses this approach. Our other studies include different levels of sparsity, not unlike Mao et al. [42] who study different levels in between fine-grained sparsity [43] and very coarse-grained sparsity [21] and observe their impact on inference accuracy.

Knowledge distillation aims at training a smaller "student" network with the knowledge extracted from a comparatively larger "teacher" model. Knowledge itself is variously defined in the literature, for instance, in terms of feature representation [32], class distribution [33] or output distribution [34]. We use knowledge distillation as a retraining strategy. Specifically, we let the pruned model that approximates the function of the original model serve as the student in knowledge distillation.

III. Look, Listen, and Learn (L^3-Net)

Look, Listen, Learn (L^3) is an approach to training audio and visual embedding through self-supervised learning via the AVC task. This auxiliary task aims to predict whether a 1 s audio segment and a single video image frame come from the same video and also overlap in time. While the AVC task itself has limited application, it can be trained on a large amount of unlabeled data, learning how to generate powerful intermediate embedding representations that can then subsequently be used as input features on a target task for which limited data are available. When used to extract audio embeddings, this is a form of transfer learning that researchers have shown to be effective for machine listening tasks such as audio classification [11].

The AVC correspondence model consists of audio and video subnetworks whose outputs are subsequently input to a fusion network which predicts a binary AVC output. The inputs to the network are a single image video frame (RBG) resized to 224×224 pixels and 1 s of audio sampled at 48 kHz. Both the audio and video subnetworks are simple convolutional neural network architectures constructed of 4 convolutional blocks each of which contains 2 layers of 3×3 convolutional filters, followed by a 2×2 max-pooling layer with stride of 2. Each individual convolutional layer is also followed by batch normalization [44] and rectified linear unit (ReLU) activations [45]. Both subnetworks are max-pooled across spatial locations to produce 512 dimensional (512-D) feature vectors. In the fusion network, the 512-D audio and visual feature vectors are concatenated into a 1024-D vector and processed by a 128-unit dense layer with ReLU activations, followed by a dense, 2-D output layer with softmax activations. See Fig. 1 for details.

The audio model spans 8 convolutional layers arranged in 4 sequential blocks, with 64, 128, 256 and 512 filters per layer in block respectively as mentioned in Table I. While the original

[1]https://github.com/ksangeeta2429/l3embedding/tree/dcompression

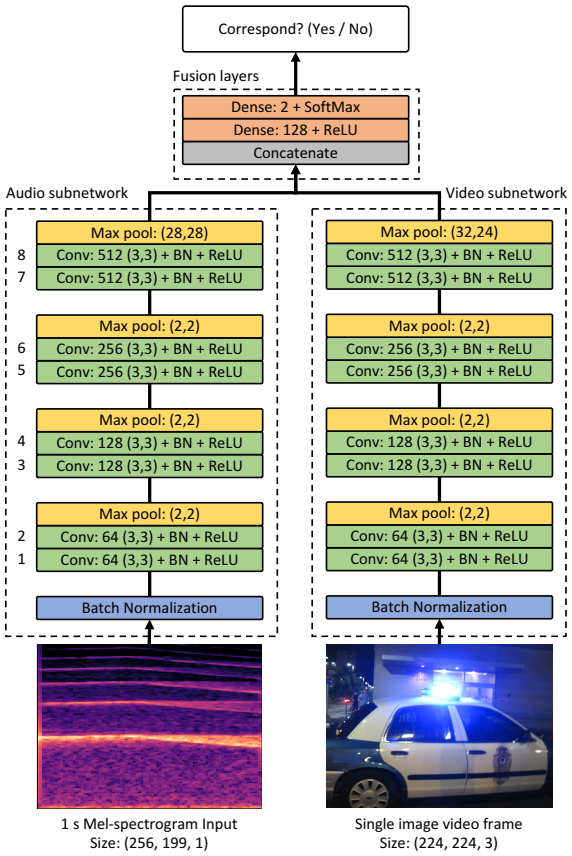


Fig. 1: The architecture for training the L^3-Net embedding models on audio-visual correspondence (AVC)

L^3-Net transformed the time domain audio input using a linear-frequency log-magnitude spectrogram (0.01 s windows with 50% overlap, 257 frequency bands), subsequent work by other researchers [12] found that an L^3 embedding trained with 256-band Mel-frequency (quasi-logarithmic) spectrograms input achieved higher performance on downstream environmental sound classification tasks.

These researchers also found that training on a music-related subset of AudioSet [13] videos rather than the Flickr dataset [46] used in the original paper also resulted in improved downstream performance on downstream environmental sound

TABLE I: Number of filters and trainable parameters per convolution layer in the original L^3-Net audio embedding model

Layer	Num. Params.	Num. Filters
conv1	640	64
conv2	36,928	64
conv3	73,856	128
conv4	147,584	128
conv5	295,168	256
conv6	590,080	256
conv7	1,180,160	512
conv8	2,359,808	512
Total	**4,684,224**	**1,920**

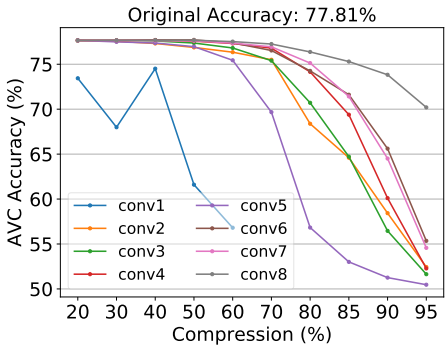


Fig. 2: Impact of pruning individual L^3-Net audio convolution layers on AVC accuracy (first layer not pruned beyond 60 % because of it's sensitivity)

classification tasks. In this work, we build upon these findings and use both Mel-spectrograms as input and train on the music-related subset of AudioSet videos.

L^3 [11] as well as L^3 More [12] maxpool the last convolutional layer of the audio subnetwork such that the audio features obtained are 6144-D. We, on the other hand, use audio features of 512-D.

IV. Experimental Design

Typically, L^3-Net is trained against the "upstream" AVC task using the *music* subset of AudioSet [47]. The audio subnetwork's last convolution layer (*conv8*) is used to extract an embedding for "downstream" tasks. For the downstream classifier, we use a multi-layer perceptron (MLP) with two fully-connected hidden layers of size 512 and 128 respectively, followed by an output layer whose size correspond to the number of classes in the dataset being evaluated. The MLP is trained to predict the class of a 1 s audio clip [2].

We explore the following fine-grained and coarse-grained pruning strategies:

(i) Pruning individual weights of the filters, resulting in sparse models.

(ii) Dropping entire filters or entire layers, resulting in non-sparse models but with reduced width and depth, respectively.

A. Sparse Audio Model

As in [27] we prune potentially unimportant connections by nulling the weights whose absolute magnitude is less than a constant threshold, which is calculated from the desired sparsity level for that layer.

To determine the sensitivity of each layer, we prune each layer independently with a range of sparsity values and test the resulting pruned network's accuracy on the validation set for the AVC problem. Notably, we find the first convolution layer (*conv1*) to be particularly sensitive to pruning: a sparsity value as low as 30% in *conv1* alone leads to a 9.81% drop in accuracy. Besides *conv1*, we also notice *conv5* to be highly

[2]Implementation details of both tasks are in [12].

TABLE II: Decomposition plan for L^3-Net audio subnetwork layerwise pruning: per-layer sparsities, overall decrease in model parameters and the resulting model size. The first row corresponds to the original model with 18MB weights. The highlighted row corresponds to the EdgeL^3 audio model for reasons we discuss in the Results section

Sparsities in Audio Convolution Layers (%)								Reduction in Weights (%)	Memory (MB)
conv1	conv2	conv3	conv4	conv5	conv6	conv7	conv8		
0	0	0	0	0	0	0	0	NA	18
0	30	40	50	30	50	50	60	53.49	8.317
0	40	50	60	40	60	60	70	63.48	6.530
0	40	50	60	40	70	70	80	72.29	4.955
0	60	60	70	50	70	70	80	73.55	4.730
0	70	70	75	60	80	80	85	80.87	3.421
0	80	80	85	40	85	85	95	87.08	2.310
30	85	85	90	60	90	90	95	90.51	1.697
0	85	85	85	75	95	98	98	95.45	0.814
0	93	94	96	97	95.97	98	97	97.00	0.536
0	95	96	97	97	98	98.65	98	98.00	0.357
0	94	96	99	99	99	99	99.2	99.00	0.179

sensitive to sparsity values more than 50%. The impact of sparsity on each layer is seen in Fig. 2.

While pruning across multiple layers, we prune the less sensitive layers more than the sensitive layers, i.e., *conv1* and *conv5*. Taking into account the sensitivity of *conv1*, we avoid sparsifying it in most other experiments. Table II shows various sparsity combinations we considered. The audio model with 95.45% reduction in weights with a memory footprint of 0.814MB corresponds to the new audio model for EdgeL^3-Net. We also experiment with extreme sparsity levels so as to achieve >95% and higher reductions in model weights to understand how loss in performance grows.

Moreover, to compensate for the loss in performance at high reduction levels, we re-learn the audio model's weights in two ways, one by using fine-tuning and the other by using knowledge distillation.

1) Fine-Tuning: We fine-tune the model by retraining the L^3-Net with the pruned audio model while freezing the video model and enabling an early stopping of the training (with patience level of 10, i.e., training stops when loss does not improve for 10 epochs). The training converges in ~50 epochs whereas the original model was trained up to 300 epochs [12].

2) Knowledge Distillation: To approximate the original feature extractor for the L^3-Net, we explore the teacher-student paradigm with the pruned audio model as student as shown in Fig. 3. We formulate the learning objective to minimize the Mean Square Error (MSE) Loss.

B. Non-Sparse Audio Model

Reductions other model sparsification are useful, for instance, because their efficient realization is not necessarily dependent upon implementation assumptions, such as hardware accelerators or software libraries which eschew wasteful Multiply-Accumulate (MAC) operations in convolution. Here, we specifically explore the dimensions of architectural reductions in width and depth.

1) Depth Reduction: Since we find *conv8*, also the audio embedding layer, to be least sensitive to sparsity, we test audio embeddings extracted out of earlier layers like *conv7*, *conv6* and *conv5*.

2) Width Reduction: Hao Li et al [21] consider filters with smaller kernel weights produce feature maps with weaker

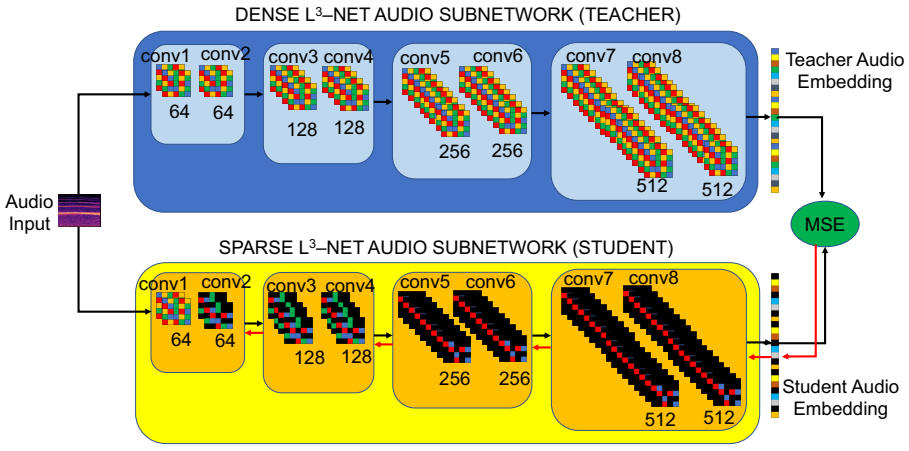


Fig. 3: Knowledge Distillation setup with original L^3 audio as teacher and pruned audio model as student for audio embedding approximation. Blackened cells signify sparsities in filters. Forward propagation is marked with black arrows, while red arrows indicate backpropagation in student

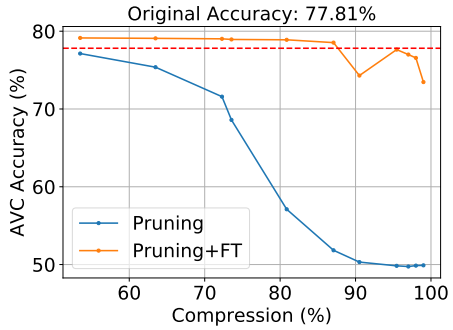


Fig. 4: Improvement in L^3-Net AVC through fine-tuning (FT). The red dotted line corresponds to the baseline model performance

activations as compared to the other filters in that layer. Following a similar reasoning, we drop kernels whose absolute weight sum is less than a threshold value.

C. Downstream Tasks

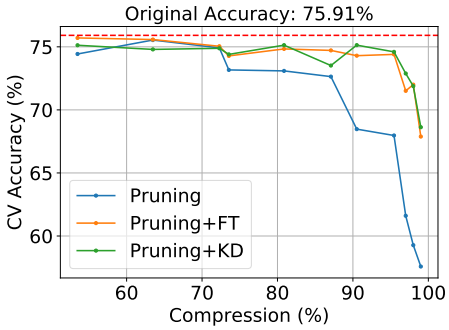
We choose urban sound and environmental sound classification as representative downstream tasks for our target application. These are evaluated on the following two open source datasets:

- **UrbanSound8K (US8K)** [2], which consists of 8732 audio clips of up to 4s with 10 sound categories. The dataset contains 10 equally-sized cross-validation folds. The baseline downstream classifier for US8K gives a mean cross-validation accuracy of 75.91%.
- **Environmental Sound Classification Dataset (ESC-50)** [48], which consists of 2000 5s audio clips with 50 environmental sound categories. The dataset contains 40 data points in each category and comes separated into 5 equally-sized cross-validation folds. The downstream classifier for ESC-50 has a mean cross-validation accuracy of 73.65%.

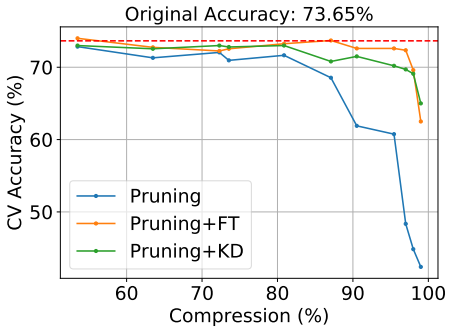
V. Results

A. Pruning Results

Fig. 4 shows the degradation pattern in L^3-Net AVC accuracies with progressively higher sparsifications as per Table II. It also shows how much of the lost performance is regained respectively with the fine tuning and knowledge distillation based variants. Surprisingly, we find that with regard to AVC, it is possible to compress L^3 by up to 87% essentially for free; in fact, doing so with the help of fine tuning reduces overfitting and actually increases AVC validation accuracy by 0.9-1.7%. Pruning by itself is somewhat less effective, and the untuned AVC validation accuracy drops sharply beyond a 70% compression. The AVC quality degrades with compressions of an order of magnitude or higher, but as mentioned before fine-tuning helps recover it significantly, so that even at 95% we are only about 0.22% short of the original AVC accuracy. Similar benefits are observed in the downstream classification



(a) UrbanSound8K



(b) ESC-50

Fig. 5: Improvements in L^3-Net pruned audio transfer learning on downstream tasks through fine-tuning (FT) and knowledge distillation (KD). Note that CV accuracy is the mean over the folds. The red dotted line corresponds to the baseline model performance

tasks (Fig. 5). With the aid of fine-tuned embeddings, there is only a 2% (1.4%) accuracy loss on US8K (ESC-50), when the model is compressed by 95%.

In accordance with the observations in Fig. 2, compensating for the AVC performance loss becomes difficult when the first layer *conv1* is pruned. This is exemplified through the (deliberate) 30% sparsification of conv1 on row 8 of Table II: we get as high as ∼4.5% reduction in accuracy with 90% pruning post fine-tuning, whereas the 95% pruned model with conv1 intact only shows a 0.2% drop. However, this is not reflected in the downstream tasks where we find both the above models to perform nearly the same. This, and other observations in the 90-99% range, leads us to postulate that the relationship between the quality of AVC and that of the learned embedding is significantly weakened as the compression ratio increases beyond an order of magnitude.

Figure 5 shows that knowledge distillation (KD) is also competitive on the downstream tasks as well, given our MSE loss teacher-student formulation explained in Section IV. While we explored KD purely from a feasibility standpoint in this paper, these initial results are motivating enough for us to pursue this direction as future work, since it opens up a vast space of small, non-sparse student models that can be trained

TABLE III: Downstream accuracy of L^3-Net depth reduction experiments (fine-tuning does not apply here)

Reduction In	Num. Filters in Audio Convolution Layers								Reduction in Weights (%)	Accuracy (%)	
	conv 1	conv 2	conv 3	conv 4	conv 5	conv 6	conv 7	conv 8		US8K	ESC-50
Original	64	64	128	128	256	256	512	512	NA	75.91	73.65
Depth	64	64	128	128	256	256	512		50.42	74.38	74
	64	64	128	128	256	256			72.34	71.74	68.7
	64	64	128	128	256				86.66	68.77	66.6

TABLE IV: AVC and downstream accuracy of L^3-Net width reduction experiments, before and after fine-tuning

Reduction In	Num. Filters in Audio Convolution Layers								Reduction in Weights (%)	Acc. Before FT (%)			Acc. With FT (%)		
	conv 1	conv 2	conv 3	conv 4	conv 5	conv 6	conv 7	conv 8		AVC	US8K	ESC-50	AVC	US8K	ESC-50
Original	64	64	128	128	256	256	512	512	NA	77.81	75.91	73.65	NA	NA	NA
Width	64	48	64	64	128	128	256	256	43.14	NA	51.39	34	77.51	74	71.25
	64	48	64	64	128	128	128	128	64.54	NA	52.16	33.3	75.4	71.45	64.7
	64	48	64	64	64	64	128	128	69.89	NA	48.87	32.6	76.40	72.46	67.2

to mimic L^3-Net with even greater compressions than what was achieved here.

To summarize, the key observation we make through our pruning experiments is that the originally proposed L^3-Net embedding is highly over-parameterized for AVC as well as transfer learning tasks. Even though this is established with downstream datasets closest to our application, we believe our findings hold true for other context as well.

B. Ablation Results

The ablation experiments are designed to evaluate non-sparse reductions of audio models. As mentioned above, our approach is two-fold: in one, we derive the embedding out of a layer earlier than *conv8* in the sequence, and in the other, we reduce the number of filters in each layer. We considered fine-tuning for the latter approach alone and not for the former; fine-tuning is not expected to be meaningful for depth reduction since we remove entire layers including *conv8*, which was originally intended to be the audio embedding layer. Similarly, AVC testing without fine-tuning is not applicable for the latter case because, due to change in the output shape of audio model, the fusion layers of L^3 as shown in Fig. 1 need to be redesigned and retrained from scratch. This corresponds to the NA values in Table III.

1) Depth Reduction: The results of the first approach are summarized in Table III. Taking the embedding out of the penultimate layer (*conv7*) of a trained L^3 audio model works reasonably well, the compression achieved is 50% as the last layer alone contributes over half of the trainable parameters. This has the surprising implication that the entire *conv8* can be removed safely post-training. The network does start to underperform as depth is reduced further, and downstream accuracies go down by 9.5% when the embedding is taken out of *conv5* producing 86.6% reduction in model size. This indicates that depth *is* relatively important in preserving the quality of the audio model with the current L^3-Net architecture.

2) Width Reduction: For the second approach, we find that un-tuned filter dropping shows a huge performance loss on both the downstream tasks (Table IV): for a mere 69.9% compression, the US8K (ESC-50) dataset incurs an accuracy drop of 35.63% (55.74%). However, the power of fine-tuning becomes quite apparent in width reduction experiments as we get the biggest improvements in the entire study (48.3% and 106.13% for US8K and ESC-50 respectively, for the aforementioned datapoint).

In this work, the ablation experiments have been presented as an orthogonal direction to sparsification. While the results currently underperform the sparsified models, cf. Fig. 5, the prospects of being able to remove 50% of the model parameters through depth reduction, as well as recovering almost the entire performance loss in filter dropping through fine-tuning are quite attractive. As future work we are motivated to explore this avenue much more comprehensively in conjunction with fine-tuning and knowledge distillation, specifically due to the advantages of small, non-sparse models from an implementation standpoint (Section VI).

VI. EdgeL^3 Mote-Scale Implementation and Challenges

EdgeL^3 provides a useful reference model for approximating L^3 audio for transfer learning. In this section we explore its utility for implementation on mote-scale devices.

In terms of model storage, the L^3 audio subnetwork requires ~18 MB for its 4,688,066 parameters, whereas EdgeL^3 requires only 0.814 MB for its 213,491 parameters. This compressed model result in a negligible loss of 0.22% in AVC performance and 1.4% (1.9%) drop in the ESC-50 (US8K) downstream classification tasks. Quantization techniques are orthogonal to network pruning techniques and can therefore be complementary. Changing single-precision to half-precision FP representation alone can reduce the EdgeL^3 model size to 0.407 MB. [27] [49] use fixed-point quantization at no cost in accuracy. Absence of floating-point not only means a smaller model but also produces more energy efficient EdgeL^3. However, further experiments are needed to evaluate the effect of quantization on downstream tasks.

Consideration of dynamic memory resources is also warranted for the edge sensing context. Although the EdgeL^3

audio model has a sufficiently small static memory footprint, the memory required for the activations is substantial at run time, with *conv1* and *conv2* having the highest activation memory, of ~12 MB. Recall the sensitivity of *conv1*, as is seen in Fig. 4, which significantly reduces its sparsifying potential and thereby impacts the sparsity of its output activation state, making it the main bottleneck to address. Fortunately, as is illustrated by the case of 90.51% reduction, the accuracy of downstream tasks is not affected when sparsifying *conv1* by 30%, as is seen in Fig. 5. This, along with the sparse representation techniques, provides a path for managing dynamic memory consumption by activation.

From a compute perspective, considering the dominance of zero weights and activations in all of our methods, approaches that eschew multiplications and additions with zero inputs materially improve performance and energy efficiency of CNNs. Several hardware accelerators and software techniques [50]–[53] have been proposed to handle sparse convolutions.

VII. Conclusion

We introduce EdgeL^3, a 95% sparsified version of L^3-Net, as the first reference model for bringing state-of-the-art robust machine listening to the edge. We establish the value of using structured sparsity to compress the original model, which is redundant in the number of parameters as well as the number of layers. We show the benefits of fine-tuning and knowledge distillation in recovering the lost performance of compressed models.

However, more work needs to be done for a realistic mote scale realization of EdgeL^3. As explained in Section VI, reducing the dynamic memory used by model activations at runtime is of particular importance. Based on the individual merits of sparsification as well as width or depth reduction we have explored in this paper, an intelligent combination of these techniques could be an interesting future direction in this regard. We wish to investigate the effectiveness of fine-tuning and knowledge distillation in bridging the performance gap of these small but powerful approximations of L^3-Net designed for edge machine listening.

References

[1] J. P. Bello, C. Silva, O. Nov, R. L. DuBois, A. Arora, J. Salamon, C. Mydlarz, and H. Doraiswamy, "SONYC: A system for monitoring, analyzing, and mitigating urban noise pollution," *Communications of the ACM*, vol. 62, no. 2, pp. 68–77, 2019.

[2] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *22nd ACM Int. Conf. on Multimedia*. ACM, 2014, pp. 1041–1044.

[3] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[4] Raspberry PI Zero. [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-zero/

[5] PocketBeagle. [Online]. Available: https://beagleboard.org/pocket

[6] Intel Edison. [Online]. Available: https://software.intel.com/en-us/iot/hardware/discontinued

[7] Vybrid VF6xx. [Online]. Available: https://www.nxp.com/docs/en/fact-sheet/VYBRIDVF6FS.pdf

[8] STM32H7. [Online]. Available: https://www.st.com/en/microcontrollers/stm32h7-series.html

[9] M. Cartwright, A. Seals, J. Salamon, A. Williams, S. Mikloska, D. MacConnell, E. Law, J. P. Bello, and O. Nov, "Seeing sound: Investigating the effects of visualizations and complexity on crowdsourced audio annotations," *Proceedings of the ACM on Human-Computer Interaction*, vol. 1, no. 1, 2017.

[10] B. Kim and B. Pardo, "A human-in-the-loop system for sound event detection and annotation," *ACM Trans. on Interactive Intelligent Systems (TiiS)*, vol. 8, no. 2, p. 13, 2018.

[11] R. Arandjelović and A. Zisserman, "Look, listen and learn," in *IEEE ICCV*, 2017, pp. 609–617.

[12] J. Cramer, H.-H. Wu, J. Salamon, and J. P. Bello, "Look, listen, and learn more: Design choices for deep audio embeddings," in *IEEE ICASSP*, 2019.

[13] Y. Aytar, C. Vondrick, and A. Torralba, "Soundnet: Learning sound representations from unlabeled video," in *NIPS*, 2016, pp. 892–900.

[14] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. Gemmeke, A. Jansen, C. Moore, M. Plakal, D. Platt, R. Saurous, B. Seybold, M. Slaney, R. Weiss, and K. Wilson, "CNN architectures for large-scale audio classification," in *IEEE ICASSP*, New Orleans, LA, USA, Mar. 2017, pp. 131–135.

[15] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *IEEE MLSP*, 2015, pp. 1–6.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv 1409.1556*, Sep 2014.

[17] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, June 2015.

[18] O. Abdel-Hamid, A. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 10, pp. 1533–1545, Oct 2014.

[19] Y. Zhang, M. Pezeshki, P. Brakel, S. Zhang, C. Laurent, Y. Bengio, and A. Courville, "Towards end-to-end speech recognition with deep convolutional neural networks," in *Interspeech*, 09 2016, pp. 410–414.

[20] M. Zhu and S. Gupta, "To prune, or not to prune: exploring the efficacy of pruning for model compression," *CoRR*, vol. abs/1710.01878, 2017. [Online]. Available: http://arxiv.org/abs/1710.01878

[21] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," *NIPS*, Aug 2016.

[22] H. Hu, R. Peng, Y. Tai, and C. Tang, "Network trimming: A data-driven neuron pruning approach towards efficient deep architectures," in *IEEE ICCV*, July 2016.

[23] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," *IEEE ICCV*, pp. 1398–1406, 2017.

[24] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," *IEEE ICCV*, pp. 2755–2763, 2017.

[25] J. Luo and J. Wu, "Autopruner: An end-to-end trainable filter pruning method for efficient deep model inference," *CoRR*, vol. abs/1805.08941, 2018.

[26] Y. Hu, S. Sun, J. Li, X. Wang, and Q. Gu, "A novel channel pruning method for deep neural network compression," *CoRR*, vol. abs/1805.11394, 2018.

[27] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," in *ICLR*, Oct 2016.

[28] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: Imagenet classification using binary convolutional neural networks," in *ECCV*, Oct 2016.

[29] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *ICML*, 2015.

[30] Y. Ioannou, D. Robertson, J. Shotton, R. Cipolla, and A. Criminisi, "Training CNNs with low-rank filters for efficient image classification," May 2016.

[31] X. Yu, T. Liu, X. Wang, and D. Tao, "On compressing deep models by low rank and sparse decomposition," *IEEE CVPR*, pp. 67–76, 2017.

[32] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "FitNets: Hints for thin deep nets," *CoRR*, vol. abs/1412.6550, 2014.

[33] S. W. Kim and H. Kim, "Transferring knowledge to smaller network with class-distance loss," *ICLR*, 2016.

[34] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *CoRR*, vol. abs/1503.02531, 2015.

[35] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *IEEE CVPR*, pp. 1800–1807, 2017.

[36] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, 2017.

[37] Z. Cao, M. Long, J. Wang, and P. Yu, "HashNet: Deep learning to hash by continuation," in *IEEE ICCV*, Oct 2017, pp. 5609–5618.

[38] H. Bagherinezhad, M. Rastegari, and A. Farhadi, "LCNN: Lookup-based convolutional neural network," *IEEE CVPR*, pp. 860–869, 2017.

[39] Y. Lecun, J. Denker, and S. Solla, "Optimal brain damage," vol. 2, Jan 1989, pp. 598–605.

[40] B. Hassibi and D. G. Stork, "Second order derivatives for network pruning: Optimal brain surgeon," in *NIPS*, 1992.

[41] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," in *NIPS*, 2016, pp. 2074–2082.

[42] H. Mao, S. Han, J. Pool, W. Li, X. Liu, Y. Wang, and W. J. Dally, "Exploring the regularity of sparse structure in convolutional neural networks," *CoRR*, vol. abs/1705.08922, 2017.

[43] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural networks," in *NIPS*, June 2015.

[44] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[45] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010, pp. 807–814.

[46] R. Arandjelović and A. Zisserman, "Objects that sound," in *ECCV*, Munich, Germany, Sep. 2018, pp. 451–466.

[47] 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 *IEEE ICASSP*, 2017, pp. 776–780.

[48] K. J. Piczak, "ESC: Dataset for environmental sound classification," in *ACM Int. Conf. on Multimedia*, 2015, pp. 1015–1018.

[49] S. Anwar, K. Hwang, and W. Sung, "Fixed point optimization of deep convolutional neural networks for object recognition," in *IEEE ICASSP*, April 2015, pp. 1131–1135.

[50] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, "EIE: Efficient inference engine on compressed deep neural network," *ACM SIGARCH Computer Architecture News*, vol. 44, Feb 2016.

[51] A. Parashar, M. Rhu, A. Mukkara, A. Puglielli, R. Venkatesan, B. Khailany, J. Emer, S. W. Keckler, and W. J. Dally, "SCNN: An accelerator for compressed-sparse convolutional neural networks," in *2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA)*, June 2017, pp. 27–40.

[52] Y. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, Jan 2017.

[53] D. Kim, J. Ahn, and S. Yoo, "ZeNA: Zero-aware neural network accelerator," *IEEE Design Test*, vol. 35, no. 1, pp. 39–46, Feb 2018.