Capsule Routing for Sound Event Detection

Turab Iqbal, Yong Xu, Qiuqiang Kong and Wenwu Wang
Centre for Vision, Speech and Signal Processing, University of Surrey
Email: {t.iqbal, yong.xu, q.kong, w.wang}@surrey.ac.uk

***Abstract*—The detection of acoustic scenes is a challenging problem in which environmental sound events must be detected from a given audio signal. This includes classifying the events as well as estimating their onset and offset times. We approach this problem with a neural network architecture that uses the recently-proposed capsule routing mechanism. A capsule is a group of activation units representing a set of properties for an entity of interest, and the purpose of routing is to identify part-whole relationships between capsules. That is, a capsule in one layer is assumed to belong to a capsule in the layer above in terms of the entity being represented. Using capsule routing, we wish to train a network that can learn global coherence implicitly, thereby improving generalization performance. Our proposed method is evaluated on Task 4 of the DCASE 2017 challenge. Results show that classification performance is state-of-the-art, achieving an F-score of 58.6%. In addition, overfitting is reduced considerably compared to other architectures.**

I. Introduction

Sound event detection (SED) is the task of classifying and localizing sound events in audio such that each detected event is assigned a class label as well as onset and offset times. Recently, the problem has received significant attention for environmental sounds in particular. For example, the series of challenges on the Detection and Classification of Acoustic Scenes and Events (DCASE) [1]–[4] has seen a rapid increase in participation since its first campaign in 2013. The number of applications that this area encompasses is extensive, and includes query-based sound retrieval [5], smart homes [6], smart cities [7], and bioacoustic scene analysis [8].

Compared to speech and music recognition, the general characteristics of environmental sounds are much broader, which means it is difficult to apply domain-specific knowledge. Thus, it is important that the method used is able to perform well despite little *a priori* knowledge. Supervised deep learning methods have largely satisfied this requirement, producing state-of-the-art results consistently in this task [9]–[12]. On the other hand, problems such as overfitting have not been completely eliminated, and this is especially severe for smaller datasets. To overcome this, we propose a neural network architecture based on grouping activation units into *capsules* and using a procedure called *routing* during inference.

The notion of a capsule was first introduced in [13] and very recently revisited in [14] with the addition of a routing mechanism. Simply put, a capsule represents a set of properties for a particular entity. The authors of [14] found that routing with capsules performed better than the state-of-the-art for digit recognition using the MNIST dataset [15]. The motivation for capsule routing is that it implicitly learns global coherence by enforcing part-whole relationships to be learned. For instance, a person's eye (the part) should be positioned sensibly relative to their face (the whole). In this case, we would like to associate a capsule representing the eye's position to a capsule representing a matching position for the face. If such an association cannot be made, it is less likely that a face has been identified.

As a result of this property, capsules overcome shortcomings of current solutions such as convolutional networks [15], which can only provide local translation invariance (via max-pooling, typically). In theory, routing can introduce invariances for any property captured by a capsule [14].

It is hypothesized that capsule routing will perform well for SED. One of the reasons is contemporary in that current datasets are relatively small, which means training is prone to overfitting. To compare with image recognition, ImageNet [16] has more than 14 million training samples, while most environmental sound datasets have thousands. Indeed, we demonstrate this issue in Section IV-A for a number of architectures. By utilizing capsules, we show that overfitting can be mitigated.

A more intrinsic rationale is that capsule routing can be considered as an attention mechanism. The idea of attention is to focus on the most salient parts of an input via weighting. It has been very successful in numerous applications, including machine translation [17], image captioning [18], and, notably, sound event detection [11], [12]. Attention is particularly useful for SED when training data is *weakly labeled*; ground truths for the onset and offset times are not available, so the learning algorithm must localize sound events without supervision. Routing implements attention by weighting the association between lower- and higher-level capsules.

In this paper, we focus on weakly-labeled event detection. It presents a challenge that is relevant to many applications, because collecting labeled data is often prohibitively costly. Nevertheless, we believe the main contributions of this paper easily apply to the strongly-labeled scenario too.

II. Capsule Routing

In general, a neural network is a function $f : \mathbf{x} \to \mathbf{y}$ that is composed of several lower-level functions $f_l : \mathbf{u} \to \mathbf{v}$, such that $f = f_L \circ \ldots \circ f_1$. Each lower-level function corresponds to a layer in the neural network, and is typically an affine transformation followed by a non-linearity, i.e.

$$\mathbf{s} = \mathbf{W}\mathbf{u} + \mathbf{b}, \tag{1}$$

$$\mathbf{v} = g(\mathbf{s}), \tag{2}$$

where $\mathbf{W}, \mathbf{b}$ are learned parameters and $g(\cdot)$ is a differentiable, non-linear function such as the rectifier (ReLU) [19].

ISBN 978-90-827970-1-5 © EURASIP 2018

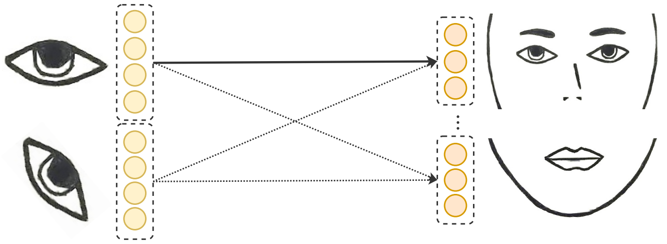


Fig. 1. A contrived illustration of the capsule routing concept. Activation units are shown as circles and capsules are shown as the dashed lines around them. The figure beside each capsule is the entity the capsule represents. The capsules in layer l are shown to the left while the capsules in layer $(l+1)$ are shown to the right. We see that the correctly-oriented eye associates well with the upper face, and is indicated by the thick arrow.

A capsule network applies the same transformations, but also introduces a routing mechanism that affects the learning dynamics. To derive this, we rewrite (1) as

$$\mathbf{s} = \begin{bmatrix} \mathbf{W}_{11}\mathbf{u}_1 + \ldots + \mathbf{W}_{1M}\mathbf{u}_M \\ \vdots \\ \mathbf{W}_{N1}\mathbf{u}_1 + \ldots + \mathbf{W}_{NM}\mathbf{u}_M \end{bmatrix}. \tag{3}$$

In (3), $\mathbf{s}$ has been partitioned into N groups, or *capsules*, so that each row in the column vector corresponds to an output capsule. Similarly, $\mathbf{u}$ has been partitioned into M capsules, where $\mathbf{u}_i$ denotes input capsule i, and $\mathbf{W}$ has been partitioned into submatrices. The bias term, $\mathbf{b}$, has been omitted for simplicity.

We now introduce *coupling coefficients*, α_{ij}, so that

$$\mathbf{s} = \begin{bmatrix} \alpha_{11}\mathbf{W}_{11}\mathbf{u}_1 + \ldots + \alpha_{M1}\mathbf{W}_{1M}\mathbf{u}_M \\ \vdots \\ \alpha_{1N}\mathbf{W}_{N1}\mathbf{u}_1 + \ldots + \alpha_{MN}\mathbf{W}_{NM}\mathbf{u}_M \end{bmatrix}. \tag{4}$$

Fixing these coefficients to $\alpha_{ij} = 1$ gives (3) and hence (1). Instead, we would like these coefficients to represent the amount of agreement between an input capsule and an output capsule. A capsule encompasses a set of properties, so if the properties of capsule i agree with the properties of capsule j in the layer above, α_{ij} should be relatively high.

These coefficients are not learned parameters; rather, their values are determined using an inference-time procedure called *routing*. The idea is based on assigning parts to wholes. Higher-level capsules should subsume capsules in the layer below in terms of the entity they identify. Routing attempts to find these associations using its notion of agreement, which causes the capsules to learn features that enable such a mechanism to result in correct predictions. Therefore, global coherencies can be learned implicitly, as exemplified in Fig. 1.

A. Dynamic Routing

Until now, we have given an abstract description of routing. In this section, we describe the method used in [14] to compute the coupling coefficients. Noting that a capsule is a vector of activation units, we can consider the direction of a capsule as representing its properties. In addition, the magnitude of a capsule can be used to indicate how likely it is to represent an entity of interest. To ensure that the magnitude is a probability, a squashing function is used, and is given by

$$\mathbf{v}_j = \frac{\|\mathbf{s}_j\|}{1 + \|\mathbf{s}_j\|^2} \frac{\mathbf{s}_j}{\|\mathbf{s}_j\|}. \tag{5}$$

```
Input: Prediction vectors $\hat{\mathbf{u}}_{j|i}$, layer $l$, max iterations $r$
Output: Layer $(l+1)$ capsules $\mathbf{v}_j$
Initialization: $\beta_{ij} = 0$
for $r$ iterations do
    $\boldsymbol{\alpha}_i = \mathrm{softmax}(\boldsymbol{\beta}_i)$
    $\mathbf{s}_j = \sum_i \alpha_{ij} \hat{\mathbf{u}}_{j|i}$
    $\mathbf{v}_j = \mathrm{squash}(\mathbf{s}_j)$                ▷ cf. (5)
    $\beta_{ij} = \beta_{ij} + \mathbf{v}_j \cdot \hat{\mathbf{u}}_{j|i}$
end for
```

Fig. 2. Routing algorithm. Whenever the indices i and j are encountered, it should be assumed that it is for all $i = 1 \ldots M$ and $j = 1 \ldots N$, respectively.

The method used to compute the coupling coefficients is listed in Fig. 2. It is a procedure that iteratively applies the softmax function to log prior probabilities, β_{ij}. These logits are initially set to $\beta_{ij} = 0$ to compute $\mathbf{v}_j$ and then updated based on an agreement computation $a_{ij} = \mathbf{v}_j \cdot \hat{\mathbf{u}}_{j|i}$, where $\hat{\mathbf{u}}_{j|i} = \mathbf{W}_{ji}\mathbf{u}_i$. The agreement value is a measure of how similar the directions of capsules i and j are. The use of the softmax function ensures that $\sum_j \alpha_{ij} = 1$. Thus, α_{ij} can be seen as the probability that the entity represented by capsule i is a part of the entity represented by capsule j as opposed to any other capsule in the layer above.

III. Proposed Method for SED

We model the SED task as being comprised of a feature extraction stage and a detection stage. Feature extraction refers to transforming the time-varying audio signal into a feature vector that is appropriate for subsequent detection. The detection stage takes the feature vector as input and attempts to detect the sound events that occur and provide timestamps for the start and end of each event. This latter stage is where we introduce our neural network architecture.

A. Feature Extraction

The input feature vectors are extracted by transforming them to produce a logarithmic Mel-frequency (logmel) representation, which is essentially a short-time Fourier transform followed by a Mel filterbank and a log nonlinearity. After this, each resulting feature vector is padded to ensure that the inputs to the neural network are of the same dimension. Finally, the feature vectors are standardized to zero mean and unit variance. The mean and variance parameters used to accomplish this are computed from the training set.

The use of a logmel representation, or the closely-related Mel-frequency cepstrum coefficients (MFCC), is standard in the literature due to its good performance [20]. Compared to older techniques such as Gaussian mixture models, deep learning benefits from the additional information that logmel retains over MFCC. For this reason, we have chosen logmel.

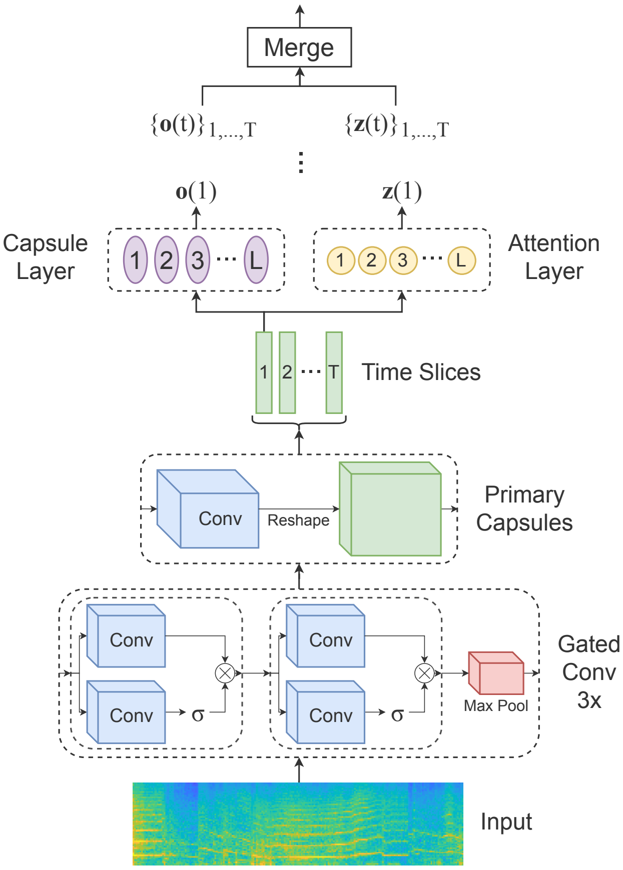


Fig. 3. Diagram of the proposed neural network architecture. After the primary capsule layer, the output is divided into time slices. These slices are transformed by the subsequent layers to give $o(t)$ and $z(t)$, which are then merged.

B. Neural Network Architecture

The architecture of the neural network is shown in Fig. 3. In contrast to the ReLU convolutional layer used in [14], the initial layers of the network are gated convolutions [21], [12]. Experiments showed that a gated nonlinearity improves the performance and that having several such layers is beneficial. There are two such layers per block and three blocks in total. After each block, max-pooling is used to halve the dimensions. The convolutions use 128 filters (64 linear, 64 sigmoidal), a kernel width of 3, and a stride of 1.

Following these initial layers is the primary capsule layer, which is a ReLU convolutional layer that has been reshaped into a $T \times \cdot^{1} \times U$ tensor and squashed using (5). T is the same temporal dimension prior to reshaping and $U = 4$ is the capsule size. In other words, each capsule is a $1 \times 1 \times 4$ slice from the output. The convolution uses 64 filters and a kernel width of 3. The stride is set to 1 for the temporal dimension and 2 for the frequency dimension.

After this, each $1 \times \cdot \times 4$ time slice is treated as a separate input to the layers that follow. Indeed, the slices are given as inputs to two adjacent layers (cf. Fig. 3): a capsule layer and a 'temporal attention' (TA) layer. The capsule layer is densely connected with $U = 8$ and L capsules, where L is the number of classes (sound events). Since the previous layer is also a capsule layer, the dynamic routing algorithm (Fig. 2) is used to compute the output. Lastly, the Euclidean length of each output capsule is computed. This gives a vector of activations for each time slice t, denoted $\mathbf{o}(t) \in \mathbb{R}^L$.

The TA layer is somewhat of a novelty that is not present in the original capsule routing paper [14]. It is used to implement an attention mechanism via the saliency of the time slices, and is based on the attention scheme described in [11], [12]. The layer is densely connected with L units and a sigmoid activation. The output is $\mathbf{z}(t) \in \mathbb{R}^L$. We can then merge $\mathbf{o}(t)$ and $\mathbf{z}(t)$ across t so that each prediction, y_l, for class l, is given by

$$\begin{aligned} y_l &= \frac{\sum_{t=1}^{T} o_l(t) z_l(t)}{\sum_{t=1}^{T} z_l(t)} \\ &= \mathbb{E}_{t \sim q_l(t)}[o_l(t)], \end{aligned} \tag{6}$$

where $q_l(t) = \mathrm{softmax}(\log \mathbf{Z}_l)$ and $\mathbf{Z}_l \in \mathbb{R}^T$ is the collation of $\{z_l(t)\}_{t=1...T}$. As such, y_l can be considered as the expected length of the capsule with respect to the probability distribution derived from the TA layer. Since $q_l(t)$ is normalized across t, there is an implicit assumption that the sound event is present in a single time slice only. Although this is restrictive, we justify this choice as a practical compromise, since including the TA layer led to better performance in our experiments. In any case, it is important to choose an appropriate granularity for the time slices because of this.

Choosing a probability threshold, τ_1, a sound event l is present if $y_l > \tau_1$. To calculate onset and offset times, we threshold the probabilities of $o_l(t)$ with another value, τ_2, and apply a morphological closing operation. The purpose of closing is to reduce fragmentation and remove noise. The onset and offset times can then be determined from the start and end points of the resulting binary regions.

IV. Experiments

To evaluate the performance of the proposed method, we used the weakly-labeled dataset provided for Task 4 of the DCASE 2017 challenge [4]. This dataset is comprised of 17 sound event classes, of which nine are warning sounds and eight are vehicle sounds. It is divided into a training set, a validation set, and an evaluation set, where the former contains 51,172 audio clips. Each clip is up to ten seconds in duration, and corresponds to one or more sound events that may overlap.

For this dataset, two tasks were evaluated: audio tagging and sound event detection. The former is for detecting which sound events occur in an audio clip, while the latter also requires providing onset and offset times. For both tasks, performance was evaluated using micro-averages of precision, recall, and F-scores. For SED, a segment-based error rate with a one-second time resolution was computed too. We used the *sed_eval* toolbox [22] for evaluation of the SED task. The reader is referred to [22] for a description of these metrics.

[1] '·' denotes that this dimension can be inferred from the others.

TABLE I
PERFORMANCE RESULTS OF AUDIO TAGGING SUBTASK

Method	F-score	Precision	Recall
GCCaps	58.6%	59.2%	57.9%
GCNN	57.2%	59.0%	57.2%
GCRNN	57.3%	53.6%	59.6%
EMSI	52.6%	69.7%	42.3%

TABLE II
PERFORMANCE RESULTS OF SOUND EVENT DETECTION SUBTASK

Method	F-score	Precision	Recall	Error Rate
GCCaps	46.3%	58.3%	38.4%	0.76
GCNN	37.5%	46.6%	31.1%	0.88
GCRNN	43.3%	57.9%	34.8%	0.79
EMSI	55.5%	-	-	0.66

A. System Setup

Prior to extracting the features, we resampled each clip to 16 KHz. The logmel features were computed using a 64 ms frame length, 20 ms overlap, and 64 Mel-frequency bins per frame. For a 10-second clip, this gives a 240 × 64 feature vector.

To reduce overfitting, we applied batch normalization [23] followed by dropout [24], [25] after each gated convolutional layer as well as the primary capsule layer. The dropout rate (fraction of units to drop) was set to 0.2 for the gated layers and 0.5 for the primary capsule layer. For capsule routing, the number of iterations was set to $r = 3$ following [14].

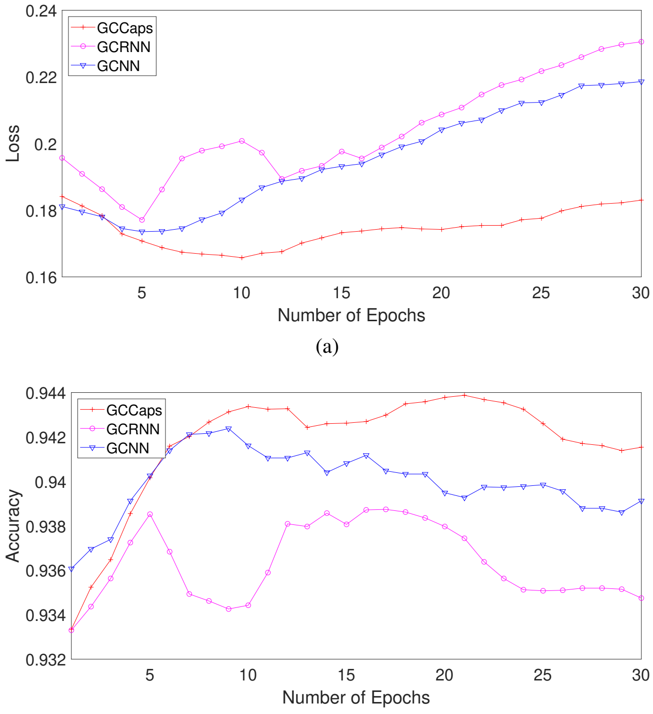
To train the network, we used binary cross-entropy as the loss function and Adam [26] as the gradient descent algorithm. The gradient was computed using mini-batch sizes of 44. The initial learning rate was set to 0.001 and decayed by a factor of 0.9 every two epochs. We trained the network for 30 epochs, with learned weights being saved per epoch.

The dataset used in the experiments has a large amount of class imbalance, which can lead to bias in the classification. To alleviate this issue, we used the data balancing technique suggested in [12] to ensure that every mini-batch contains a fair number of samples from each class.

During inference, the five models (epochs) that achieved the highest accuracy on the validation set were selected and their predictions were averaged. The detection thresholds were set to $\tau_1 = 0.3$ and $\tau_2 = 0.6$ for our system. For SED, the dilation and erosion sizes were set to 10 and 5, respectively. As with the other hyperparameters, these values were determined based on experiments on the validation set.

B. Results

In addition to our system (GCCaps)[2], we also evaluated the model proposed in [12] (GCRNN), which won 1st place in the audio tagging subtask of Task 4. It is similar to our proposal, with the difference being an additional gated convolutional layer and recurrent layers [27] as opposed to capsule layers. The same model without the recurrent layers (GCNN) is also compared as an ablation study for both GCCaps and GCRNN. Moreover, the results for [28] (EMSI) are listed too, albeit much of the setup for that system is not the same, so it is not a direct comparison between the architectures. It is included because it achieved 1st place in the SED subtask.



(a)

(b)

Fig. 4. Performance as a function of the number of epochs for (a) loss and (b) accuracy. The GCCaps model has the highest accuracy and the lowest loss, which also does not diverge as severely.

We present our results in Table I and II for audio tagging and sound event detection, respectively. For audio tagging, our method performs the best overall with an F-score of 58.6%. EMSI has the highest precision, but its recall score is much lower, and, as a result, it has the lowest F-score. GCRNN and GCNN perform the same in this subtask.

For SED, the recurrent layers clearly improve localization for GCRNN, as it scores considerably higher compared to GCNN. Meanwhile, GCCaps performs marginally better than GCRNN with an F-score of 46.3% and an error rate of 0.76. We can deduce that the capsule layers in GCCaps are a good substitute for the recurrent layers in GCRNN. EMSI performs the best by a large margin, but it should be emphasized that much of the system is different, including the use of ensemble techniques to utilize multiple feature vectors, which demonstrably [28] improves its performance significantly.

To obtain greater insight, we also compared the performance of these models (excluding EMSI) on the validation set as a function of the number of epochs. As evident in Fig. 4, our proposal achieved the lowest loss and highest accuracy, which supports our earlier results. It can be seen in Fig. 4a that all of the models eventually diverge in terms of the value of the loss function. In Fig. 4b, it can be seen that the accuracy decreases

[2] Code available online: https://github.com/turab95/gccaps

after a number of epochs. These issues are not observed with the training set, which suggests that the models are overfitting. However, as shown in the figures, the extent of this problem is greatly reduced when using capsule routing. This is reassuring, because it indicates that the network can differentiate between fundamental features and training-specific features.

These results demonstrate that a dynamic routing mechanism can improve the generalization abilities of a neural network. Although it remains to be seen, we are confident that this applies to other datasets too. Investigating deeper layers of capsules or different capsule networks, such as convolutional capsule networks [29], [30], is a natural direction to take in the future. It is also of interest to explore different routing algorithms, such as that proposed in [29].

V. Conclusion

In this paper, we have proposed a neural network architecture based on capsule routing for the detection of sound events. The motivation was that capsules can learn to identify global structures in the data that alternatives such as convolutional networks cannot. Our system was evaluated on a weakly-labeled dataset from Task 4 of the DCASE 2017 challenge. We found that the method was considerably less prone to overfitting compared to other architectures. For the audio tagging subtask, we achieved a best-in-class F-score of 58.6%, while for the event detection subtask, an F-score of 46.3% and an error rate of 0.76. These are promising results, and suggest that capsule routing should be further investigated.

References

[1] D. Giannoulis, E. Benetos, D. Stowell *et al.*, "Detection and classification of acoustic scenes and events: An IEEE AASP challenge," in *2013 IEEE Workshop Appl. Signal Process. Audio, Acoustics (WASPAA)*, New Paltz, NY, 2013, pp. 1–4.

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

[3] A. Mesaros, T. Heittola, E. Benetos *et al.*, "Detection and classification of acoustic scenes and events: Outcome of the DCASE 2016 challenge," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 26, no. 2, pp. 379–393, Nov. 2018.

[4] A. Mesaros, T. Heittola, A. Diment *et al.*, "DCASE 2017 challenge setup: Tasks, datasets and baseline system," in *Proc. Detection, Classification Acoust. Scenes, Events 2017 (DCASE2017)*, Munich, Germany, 2017.

[5] F. Font, G. Roma, and X. Serra, "Sound sharing and retrieval," in *Computational Analysis of Sound Scenes and Events*, 1st ed., T. Virtanen, M. D. Plumbley, and D. Ellis, Eds. Cham: Springer, 2018, pp. 279–301.

[6] S. Krstulović, "Audio event recognition in the smart home," in *Computational Analysis of Sound Scenes and Events*, 1st ed., T. Virtanen, M. D. Plumbley, and D. Ellis, Eds. Cham: Springer, 2018, pp. 335–371.

[7] J. P. Bello, C. Mydlarz, and J. Salamon, "Sound analysis in smart cities," in *Computational Analysis of Sound Scenes and Events*, 1st ed., T. Virtanen, M. D. Plumbley, and D. Ellis, Eds. Cham: Springer, 2018, pp. 373–397.

[8] D. Stowell, "Computational bioacoustic scene analysis," in *Computational Analysis of Sound Scenes and Events*, 1st ed., T. Virtanen, M. D. Plumbley, and D. Ellis, Eds. Cham: Springer, 2018, pp. 303–333.

[9] E. Çakır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 25, no. 6, pp. 1291–1303, May 2017.

[10] M. Valenti, S. Squartini, A. Diment, G. Parascandolo, and T. Virtanen, "A convolutional neural network approach for acoustic scene classification," in *Proc. 2017 Int. Jt. Conf. Neural Netw. (IJCNN)*, Anchorage, AK, 2017, pp. 1547–1554.

[11] Y. Xu, Q. Kong, Q. Huang, W. Wang, and M. Plumbley, "Attention and localization based on a deep convolutional recurrent model for weakly supervised audio tagging," in *Interspeech 2017*, Stockholm, Sweden, 2017, pp. 3083–3087.

[12] Y. Xu, Q. Kong, W. Wang, and M. D. Plumbley, "Large-scale weakly supervised audio classification using gated convolutional neural network," *arXiv preprint arXiv:1710.00343*, Oct. 2017.

[13] G. E. Hinton, A. Krizhevsky, and S. D. Wang, "Transforming auto-encoders," in *21st Int. Conf. Artificial Neural Netw. (ICANN)*, Espoo, Finland, 2011, pp. 44–51.

[14] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Adv. Neural Inf. Process. Syst. (NIPS)*, Long Beach, CA, 2017, pp. 3859–3869.

[15] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[16] J. Deng, W. Dong, R. Socher *et al.*, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conf. Comput. Vis. Pattern Recognit.*, Miami, FL, 2009, pp. 248–255.

[17] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *3rd Int. Conf. Learn. Repr. (ICLR)*, San Diego, CA, 2015.

[18] K. Xu, J. Ba, R. Kiros *et al.*, "Show, attend and tell: Neural image caption generation with visual attention," in *Proc. 32nd Int. Conf. Mach. Learn. (ICML)*, vol. 37, Lille, France, 2015, pp. 2048–2057.

[19] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, Haifa, Israel, 2010, pp. 807–814.

[20] D. Barchiesi, D. Giannoulis, D. Stowell, and M. D. Plumbley, "Acoustic scene classification: Classifying environments from the sounds they produce," *IEEE Signal Process. Mag.*, vol. 32, no. 3, pp. 16–34, Apr. 2015.

[21] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, vol. 70, Sydney, Australia, 2017, pp. 933–941.

[22] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Appl. Sci.*, vol. 6, no. 6-162, 2016.

[23] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Mach. Learn. (ICML)*, Lille, France, 2015, pp. 448–456.

[24] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, Jul. 2012.

[25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res. (JMLR)*, vol. 15, no. 1, pp. 1929–1958, Jun. 2014.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd Int. Conf. Learn. Repr. (ICLR)*, San Diego, CA, 2015.

[27] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.

[28] D. Lee, S. Lee, Y. Han, and K. Lee, "Ensemble of convolutional neural networks for weakly-supervised sound event detection using multiple scale input," DCASE2017 Challenge, Tech. Rep., 2017.

[29] G. E. Hinton, S. Sabour, and N. Frosst, "Matrix capsules with EM routing," in *6th Int. Conf. Learn. Repr. (ICLR)*, Vancouver, BC, 2018.

[30] R. LaLonde and U. Bagci, "Capsules for object segmentation," *arXiv preprint arXiv:1804.04241*, Apr. 2018.

ISBN 978-90-827970-1-5 © EURASIP 2018