A CAPSULE NEURAL NETWORKS BASED APPROACH FOR BIRD AUDIO DETECTION

Technical Report

Fabio Vesperini, Leonardo Gabrielli, Emanuele Principi, Stefano Squartini

Department of Information Engineering,
Università Politecnica delle Marche, Ancona, Italy
f.vesperini@pm.univpm.it

ABSTRACT

We propose a system for bird audio detection based on the innovative CapsNet architecture. It is our contribution to the third task of the DCASE2018 Challenge. The task consists on a binary detection of presence/absence of bird sounds on audio files belonging to different datasets. Spectral acoustic features are extracted from the acoustic signals, successively a deep neural network which comprehend capsule units is trained by means of supervised learning using binary annotations of bird song activity as target vector in combination with the dynamic routing mechanism. This procedure has the aim to incentive the network to learn global coherence implicitly and to identify part-whole relationships between capsules, thereby improving generalization performance in detecting the presence bird songs from various environmental conditions. We achieve a harmonic mean of the Area Under Roc Curve (AUC) score equal to 85.08 from the cross-validation performed on the development dataset, while we obtain an AUC equal to 84.43 as preview score from a subset of the unseen evaluation data.

Index Terms— Bird Audio Detection, CapsNet, CNN, DCASE2018, Acoustic Monitoring

1. INTRODUCTION

Automatic wildlife monitoring is a key concern nowadays. Climatic changes, the effects of pollution and alteration of the ecosystems have to be closely controlled in order to be the litmus test for the future sustainable technological and political guidelines.

In this context, bird audio analysis is an important task of the bioacoustics for wildlife and biodiversity monitoring, which can easily embrace the deep learning concept. In particular, detecting the presence of bird calls in audio recordings is a very common required first step for a framework that can perform different kind of analysis (e.g. species classification, counting), and makes it possible to conduct work with large datasets (e.g. continuous 24h monitoring) by segmenting the data stream into regions of interests.

To encourage the research in automating this task, in 2016 Stowell et al. [1] organized a first edition Bird Audio Detection (BAD) challenge. It has been appreciated to such an extent that a new round has been included in one of the tasks of the 2018 IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE). In fact, Task 3 consists in determining a binary decision for the presence/absence of bird sounds on audio files recorded in very different conditions, comprehending dataset balancing, birds species, background sounds and recordings equipment. Specifically, participants are asked to build algorithms that predict whether a given 10-second recording contains any type of bird vocalization, regardless of the species.

The organizers invite to explore approaches that can either inherently generalize across different conditions (including conditions not seen in the training data), or which can self-adapt to new datasets. The deep neural network based approach we propose has the aim to counteract the generalization problem by means of an innovative learning procedure named "capsule routing" which has shown promising performances since it has been presented [2] and also in pioneering employments in audio tasks [3].

1.1. Related Works

In very recent years, a strong growth of deep learning algorithms devoted to the acoustic monitoring has been observed. In particular, works such as [4, 5, 6] represent milestones, involving Convolutional Neural Networks (CNN) for audio signals detection and classification. These deep neural architectures, combined with the increased availability of datasets and computational resources, have allowed large performance improvements, outperforming in most of the cases the human accuracy [7]. This has also motivated researchers to employ such architecture, eventually combined with recurrent units [8], in almost all of the tasks proposed in the recent editions of research challenges such as the DCASE [9]. These algorithms often result among the strongest-performing systems [10, 11].

Similar results came from the first edition Bird Audio Detection (BAD2017) challenge, which was held in 2016-2017. In this case different novel algorithms have been proposed to create robust and scalable systems able to automate the annotation process of audio sequences containing free-field recordings. The work of Grill and Schlüter [12] should be also mentioned, which obtained the highest score and which is based on CNNs trained on Mel-scaled log-magnitude spectrograms. The outcomes of the BAD2017 are reported in [13].

A team at Google Brain recently has presented a new computational unit [2] called "CapsNet" with the intent to overcome two known limitations of the CNNs: the excessive information loss caused by the pooling and other down-scaling operations and the inability to infer part-whole relationships between the elements which the deep neural network (DNN) has to detect. In fact, the layers of a standard CNN are good at detecting space-invariant features which characterize an image (or a spectrogram in the case of audio spectrograms), but are less effective at exploring the spatial relationships among features (perspective, size, orientation). Capsule routing has the aim to learn global coherence implicitly, thereby improving generalization performance. In the BAD application, it means that the

DNN is driven to learn a general concept of the entities of "bird song" and "background sounds" without requiring extensive data augmentation or dedicated domain adaptation procedures, thus motivating the use of Capsules for the Task 3 of the DCASE 2018.

2. PROPOSED METHOD

The proposed system is a fully data-driven approach based on the CapsNet deep neural architecture presented by Sabour et al. [2]. The novel computational structure of the Capsules, combined to the routing mechanism allows to be invariant to intra-class affine transformations and to identify part-whole relationships between data features.

The whole system is composed of a feature extraction stage and a detection stage. The feature extraction stage transforms time-varying audio signal into acoustic spectral features, then the second stage takes the feature vector as input and maps them to a binary estimate of bird song presence. This latter stage is where we introduce the Capsule neural network architecture. The network parameters are obtained by supervised learning using annotations of bird song activity as one hot target vector.

2.1. Feature Extraction

The feature extraction stage operates on mono audio signals sampled at 44.1 kHz. For our purpose, we exploit *LogMels* as acoustic spectral representation, following results obtained in various audio tagging and sound event detection tasks. Firstly, the audio signals are down-sampled to 16 kHz, because the most relevant frequency bands related to bird songs are in the range from 2 kHz to 8 kHz [14]. Then, *LogMel* coefficients are obtained by filtering the magnitude spectrum of the STFT with a filter-bank composed of 40 filters evenly spaced in the mel frequency scale. The logarithm of the energy of each band is computed to match the human perception of loudness. In the STFT computation, the used frame size is equal to 40 ms and the frame step is equal to 20 ms. All of the datasets contain 10-second-long WAV files, thus the resulting feature matrix $\mathbf{x} \in \mathbb{R}^{D_1 \times D_2}$ has a shape 501×40. The range of feature values is then normalized according to the mean and the standard deviation computed on the training sets of the neural networks.

2.2. CapsNet Architecture

Conceptually, a CNN model uses multiple neurons (kernels) which act as translated replicas of learned feature detectors. As in that case, the whole input matrix is processed by repeated application of a function across its sub-regions, obtaining so-called *feature maps*. Practically, this is implemented by a convolution of the input data with a linear filter, adding a bias term and then applying a non-linear function.

Denoting with $\mathbf{W}_m \in \mathbb{R}^{K_{1m} \times K_{2m}}$ the m-th kernel and with $\mathbf{b}_m \in \mathbb{R}^{D_1 \times D_2}$ the bias vector of a generic convolutional layer, the m-th feature map $\mathbf{h}_m \in \mathbb{R}^{D_1 \times D_2}$ is given by:

$$\mathbf{h}_m = \varphi\left(\mathbf{W}_m * \mathbf{x} + \mathbf{b}_m\right), \tag{1}$$

where $*$ represents the convolution operation, $\varphi(\cdot)$ the non-linear activation function The dimension of the m-th feature map $\mathbf{h}_m$ depends on the zero padding of the input tensor: here, padding is performed in order to preserve the dimension of the input, i.e., $\mathbf{h}_m \in \mathbb{R}^{D_1 \times D_2}$. Commonly, Eq. (1) is followed by a pooling layer in order to be more robust against patterns shifts in the processed data. This allows them to share knowledge obtained at one position in an image to other positions, thus being invariant to spatial shifts and this has proven to be extremely helpful in image interpretation.

Following Hinton's preliminary works [15], in the CapsNet each layer in divided into many small groups of neurons called "capsules". The scalar-output feature detectors of CNNs are replaced with vector-output capsules and routing-by-agreement algorithm is used in place of max-pooling, in order to replicate learned knowledge across space.

Formally, we can rewrite (1) as

$$\mathbf{h}_m = \begin{bmatrix} \alpha_{11}\mathbf{W}_{11}\mathbf{x}_1 + \ldots + \alpha_{M1}\mathbf{W}_{1M}\mathbf{x}_M \\ \vdots \\ \alpha_{1N}\mathbf{W}_{N1}\mathbf{x}_1 + \ldots + \alpha_{MN}\mathbf{W}_{NM}\mathbf{x}_M \end{bmatrix}, \tag{2}$$

In Eq. (2), $\mathbf{h}_m$ has been partitioned into N groups, or capsules, so that each row in the column vector corresponds to an output capsule. Similarly, $\mathbf{x}$ has been partitioned into M capsules, where $\mathbf{x}_i$ denotes an input capsule i, and $\mathbf{W}$ has been partitioned into submatrices called *transformation matrix*. Conceptually, a capsule incorporates a set of properties of a particular entity that is present in the input data. With this purpose, coefficients α_{ij} have been introduced. They are called *coupling coefficients* and they have the aim to represent the amount of agreement between an input capsule and an output capsule. α_{ij} measures how likely capsule i may activate capsule j, so if the properties of capsule i agree with the properties of capsule j in the layer above, α_{ij} should be relatively high. The coupling coefficients are calculated by the iterative dynamic routing process, which fulfills the idea of assigning parts to wholes.

Capsules in the higher layers should comprehend capsules in the layer below in terms of the entity they identify. Dynamic routing iteratively attempts to find these associations with its notion of agreement, and supports capsules to learn features that ensure their outputs are sent to the appropriate parents in the layer above.

2.2.1. Dynamic Routing

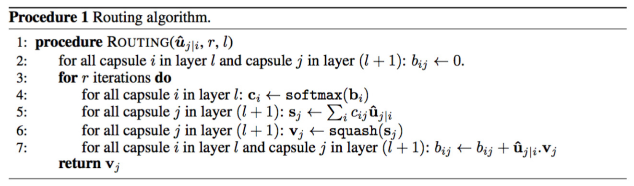
After giving an abstract description of routing, we describe the method used in [2] to compute the coupling coefficients. The activation of a capsule unit is a vector which holds in its direction the properties of the entity it represents. The vector's magnitude indicates instead the probability that the entity represented by the capsule is present in the current input. To ensure that the magnitude is a probability, a *squashing* non-linear function is used, which is given by:

$$\mathbf{v}_j = \frac{\|\mathbf{s}_j\|^2}{1 + \|\mathbf{s}_j\|^2}\frac{\mathbf{s}_j}{\|\mathbf{s}_j\|}, \tag{3}$$

where $\mathbf{v}_j$ is the vector output of capsule j and $\mathbf{s}_j$ is its total input. $\mathbf{s}_j$ is a weighted sum over all the outputs $\mathbf{u}_i$ of a capsule in the layer below multiplied by the coupling matrix $\mathbf{W}_{ij}$:

$$\mathbf{s}_j = \sum_i c_{ij}\hat{\mathbf{u}}_{j|i}, \; \hat{\mathbf{u}}_{j|i} = \mathbf{W}_{ij}\mathbf{u}_i. \tag{4}$$

The method used to compute the coupling coefficients is listed in Fig. 2.2.1. 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 $\alpha_{ij} = \mathbf{v}_j \cdot \hat{\mathbf{u}}_{j|i}$. 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



Procedure 1 Routing algorithm.

```
1: procedure ROUTING($\hat{\mathbf{u}}_{j|i}$, $r$, $l$)
2:     for all capsule $i$ in layer $l$ and capsule $j$ in layer $(l+1)$: $b_{ij} \leftarrow 0$.
3:     for $r$ iterations do
4:         for all capsule $i$ in layer $l$: $\mathbf{c}_i \leftarrow \texttt{softmax}(\mathbf{b}_i)$
5:         for all capsule $j$ in layer $(l+1)$: $\mathbf{s}_j \leftarrow \sum_i c_{ij}\hat{\mathbf{u}}_{j|i}$
6:         for all capsule $j$ in layer $(l+1)$: $\mathbf{v}_j \leftarrow \texttt{squash}(\mathbf{s}_j)$
7:         for all capsule $i$ in layer $l$ and capsule $j$ in layer $(l+1)$: $b_{ij} \leftarrow b_{ij} + \hat{\mathbf{u}}_{j|i}.\mathbf{v}_j$
    return $\mathbf{v}_j$
```

Figure 1: The routing algorithm. Source taken from [2].

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.

2.3. CapNet for Bird Audio Detection

The architecture of the neural network is shown in Fig. 2.3. The first stages of the model are traditional CNN blocks which act as feature extractors on the input *LogMel* coefficients. After each block, max-pooling is used to halve the dimensions. The feature maps obtained by the CNN layers are then fed to the Primary Capsule Layer that represents the lowest level of multi-dimensional entities. Basically it is a convolutional layer whose output is reshaped and squashed using (3). The final layer, is a capsule layer and it is composed of two densely connected capsule units. Since the previous layer is also a capsule layer, the dynamic routing algorithm is used to compute the output. The model predictions are obtained computing the the Euclidean length of each output capsule, which represent the probabilities that an input feature vector $\mathbf{x}$ belongs to the background or the bird audio class, thus we consider only the latter as system output prediction.

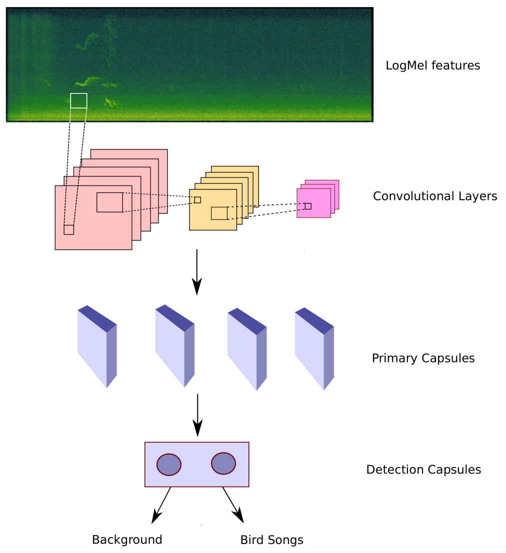


Figure 2: Flow chart of the proposed neural network architecture.

3. EXPERIMENTAL SET-UP

The network hyperparameters optimization was obtained by means of a *random search* strategy [16]. The number and the shape of convolutional layers, the non-linear activation function, the regularizers in addition to the capsules dimensions and the maximum number of routing iterations have been varied for a total of 100 configurations. Details of searched hyperparameters and their ranges are reported in Table 1. The neural networks training was accomplished by the AdaDelta stochastic gradient-based optimisation algorithm [17] for a maximum of 100 epochs and batch size equal to 20 on the margin loss function. The optimizer hyperparameters were set according to [17] (i.e., initial learning rate $lr = 1.0$, $\rho = 0.95$, $\epsilon = 10^{-6}$). It was chosen because it is well-suited for dealing with sparse data and its robustness to different choices of model hyperparameters. Furthermore no manual tuning of learning rate is required.

An early stopping strategy was employed in order to avoid overfitting. Thus if the validation score does not increase for 20 consecutive epochs, the training is stopped and the last saved model is selected as the final model. In addition, dropout and L2 (with $\lambda = 0.01$) have been used as weights regularization techniques [18]. The algorithm has been implemented in the Python language using Keras [19] and Tensorflow [20] as deep learning libraries.

3.1. Dataset

According to the DCASE 2018 guidelines, the performance of the proposed algorithm has been assessed firstly by using the development dataset for training and validation of the system. Then, a blind test on the provided evaluation dataset was performed with the models which achieved the highest performance and submitted to the organizers of the challenge. The complete dataset is composed of recordings belonging to five different collections. Further details are reported below:

- “freefield1010”: a collection of 7690 excerpts from field recordings around the world;
- “warblrb10k”: a crowsourced dataset recorded with the *Warblr*[1] smartphone app. It covers a wide distribution of UK locations and environments and includes weather noise, traffic noise, human speech and even human bird imitations; 8000 samples are used in the development dataset while a held-out set of 2,000 recordings from the same conditions is included in the evaluation split;
- “BirdVox-DCASE-20k”: 20000 files containing remote monitoring flight calls collected from recordings units placed near Ithaca, NY, USA during the autumn of 2015;
- “Chernobyl”: dataset collected from unattended remote monitoring equipment in the Chernobyl Exclusion Zone (CEZ). A totoal of 6620 audio files cover a range of birds and includes weather, large mammal and insect noise sampled across various CEZ environments, including abandoned village, grassland and forest areas;
- “PolandNFC”: 4000 recordings obtained from a project of monitoring of autumn nocturnal bird migration. They were collected every night, from September to November 2016 on the Baltic Sea coast, Poland, using Song Meter SM2 units with microphones mounted on 3–5 m poles.

The organizers recommended a 3-way cross-validation (CV) for the algorithms development, thus in each fold we used two sets for training and the other one as validation set in order to have scores comparable with the others challenge participant.

[1]https://www.warblr.co.uk/

Parameter	Range	Distribution	CapsNet1	CapsNet 2	CapsNet3
CNN layers Nr.	[1 - 4]	uniform	3	4	4
CNN kernels Nr.	[4 - 64]	log-uniform	[64,16,8]	[32,16,16,32]	[32,64,4,64]
CNN kernels dim.	[3×3 - 8×8]	uniform	3×3	5×5	6×6
Pooling dim.	[1×1 - 2×5]	uniform	[1×5],[1×4], [1×4]	[1×5],[1×4], [1×2],[1×2]	[1×4],[1×2], [1×2],[1×2]
CNN activation	[tanh - relu]	random choice	tanh	relu	relu
CNN dropout	[0 - 0.5]	uniform	0	0	0
CNN L2	[yes - no]	random choice	no	yes	yes
Primary Capsules channels Nr.	[2 - 8]	uniform	6	2	8
Primary Capsules kernels dim.	[3×3 - 5×5]	uniform	4×4	4×4	3×3
Primary Capsules dimension	[2 - 16]	uniform	8	8	2
Capsules dimension	[2 - 16]	uniform	2	15	10
Capsules dropout	[0 - 0.5]	uniform	0	0.1	0.3
Max routing iterations	[1 - 5]	uniform	2	3	2
Batch Normalization	[yes - no]	random choice	yes	yes	yes
Trainable Params	-	-	113K	282K	424K

Table 1: Hyper-parameters optimized in the random-search phase and the resulting best performing models.

3.2. Baseline

The baseline system is an adapted version of the method winner of the BAD2017 [12]. The peculiarity of this algorithm is its double training procedure. In a first run, the network is trained on the whole training data. Binary predictions are obtained for the testing data. The more confident predictions (the ones closer to 0 or 1) are then added to the training data as so-called "pseudo-labeled" samples. Thus, a second training run is performed on this extended training set and the final predictions are yielded.

3.3. Metric

The performance metric of the DCASE 2018 on this task is the "Area Under the ROC Curve" (AUC). More precisely, it is a stratified AUC: the score is computed separately for each fold of the evaluation set, then the partial scores are averaged. This procedure allows to adapt the "detection threshold" to each dataset conditions, then the performance across datasets are combined in an explicit weighted fashion, thus the final score is not merely influenced by the number of files in each subset.

4. RESULTS

4.1. Results on Development dataset

Results reported in Table 2 show both the best performance we obtained on the single CV fold, and the best averaged AUC. We obtain a harmonic mean for AUC equal to 83.72 for a single configuration, whilst if we consider the mean of the best performing models on the single folds we achieve an AUC equal to 85.08.

4.2. Preview Score on Evaluation dataset

We considered as candidates for the test on the evaluation procedure [21] both the best performing setups on the single CV folds (submission label **Vesperini_UnivPM_task3_1**), and the setups with the best averaged AUC (submission label **Vesperini_UnivPM_task3_2**). For the latter, we trained a new model with the same hyperparameters on the whole development dataset before performing the predictions on the evaluation dataset.

The DCASE 2018 featured a submission site where contestants could upload their predictions and compute a "preview score" for a subset of around 1000 files from the test set. With an ensemble of the single fold best models trained during the CV procedure we obtain an AUC score equal to 84.43, while for the model with the best averaged AUC we obtain an AUC score equal to 81.43.

5. CONCLUSION AND OUTLOOK

In this paper, we have presented an algorithm for bird audio detection based on the CapsNet architecture. We feed a deep neural network which uses the dynamic routing procedure during the training with the *LogMel* extracted from the audio signals in order to obtain predictions on unseen data recorded in various conditions possibly also very different from the training set. To assess the performance of the algorithm we conducted experiments on the development dataset from the DCASE 2018, obtaining an AUC score equal to 85.08 with respect to an AUC equal to 83.00 of the baseline system. For future work, variants [22] or strategy to customize the dynamic routing can be considered.

6. ACKNOWLEDGMENT

We acknowledge the CINECA award under the ISCRA initiative, for the availability of high performance computing resources and support.

Conf ID	Fold 1	Fold 2	Fold 3	Avg	Preview Score
Baseline	-	-	-	83.00	89.18
CapNet1	**88.22**	72.78	74.16	78.39	-
CapNet2	81.77	**80.90**	85.52	82.73	-
CapNet3	86.59	78.46	**86.11**	83.72	81.83
Ensemble	-	-	-	**85.08**	**84.43**

Table 2: Results on Development dataset in terms of AUC (%).

7. REFERENCES

[1] D. Stowell, M. Wood, Y. Stylianou, and H. Glotin, "Bird detection in audio: a survey and a challenge," *arXiv preprint arXiv:1608.03417*, 2016.

[2] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Advances in Neural Information Processing Systems*, 2017, pp. 3856–3866.

[3] T. Iqbal, Y. Xu, Q. Kong, and W. Wang, "Capsule routing for sound event detection," *arXiv preprint arXiv:1806.04699*, 2018.

[4] I. McLoughlin and Y. Song, "Low frequency ultrasonic voice activity detection using convolutional neural networks," in *Proc. of Interspeech*, Dresden, Germany, Sep. 6-10 2015.

[5] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *Machine Learning for Signal Processing (MLSP), 2015 IEEE 25th International Workshop on*. IEEE, 2015, pp. 1–6.

[6] 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.

[7] H. B. Sailor, D. M. Agrawal, and H. A. Patil, "Unsupervised filterbank learning using convolutional restricted boltzmann machine for environmental sound classification," *Proc. Interspeech 2017*, pp. 3107–3111, 2017.

[8] E. Cakir, G. Parascandolo, T. Heittola, H. Huttunen, T. Virtanen, E. Cakir, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 25, no. 6, pp. 1291–1303, 2017.

[9] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, E. Vincent, B. Raj, and T. Virtanen, "DCASE 2017 challenge setup: Tasks, datasets and baseline system," in *Proc. of DCASE*, 2017.

[10] H. Lim, J. Park, and Y. Han, "Rare sound event detection using 1D convolutional recurrent neural networks," in *Proc. of DCASE*, 2017, pp. 80–84.

[11] M. Valenti, S. Squartini, A. Diment, G. Parascandolo, and T. Virtanen, "A convolutional neural network approach for acoustic scene classification," in *2017 International Joint Conference on Neural Networks (IJCNN)*, May 2017, pp. 1547–1554.

[12] T. Grill and J. Schlüter, "Two convolutional neural networks for bird detection in audio signals," in *2017 25th European Signal Processing Conference (EUSIPCO)*, Aug 2017, pp. 1764–1768.

[13] D. Stowell, Y. Stylianou, M. Wood, H. Pamuła, and H. Glotin, "Automatic acoustic detection of birds through deep learning: the first bird audio detection challenge," *arXiv preprint arXiv:1807.05812*, 2018.

[14] R. B. Payne, "Handbook of the birds of the world," *The Wilson Journal of Ornithology*, vol. 122, no. 3, pp. 627–629, 2010.

[15] G. E. Hinton, A. Krizhevsky, and S. D. Wang, "Transforming auto-encoders," in *International Conference on Artificial Neural Networks*. Springer, 2011, pp. 44–51.

[16] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

[17] M. D. Zeiler, "AdaDelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[19] F. Chollet *et al.*, "Keras," https://github.com/keras-team/keras, 2015.

[20] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[21] http://dcase.community/challenge2018/.

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