This full text paper was peer-reviewed at the direction of IEEE Instrumentation and Measurement Society prior to the acceptance and publication.

Object classification on raw radar data using convolutional neural networks

Heejae Han[1], Jeonghwan Kim[2], Junyoung Park[3], YuJin Lee[4], Hyunwoo Jo[5], Yonghyeon Park[6], Eric T. Matson[7], and Seongha Park[8]*

[1] Industrial Systems Engineering, Dongguk University, Seoul, South Korea, heejae1213@dongguk.edu
[2] Computer Science, Handong University, Pohang, South Korea, jeonghwankim123@gmail.com
[3] Computer Science Engineering, Chungang University, Seoul, South Korea, jypark@uclab.re.kr
[4] Information Security, Seoul Women's University, Seoul, South Korea, skdisksksk@gmail.com
[5] Computer Engineering, Sejong University, Seoul, South Korea, whgusdn321@gmail.com
[6] Computer Science, Sungkyunkwan University, Seoul, South Korea, ryan8237@gmail.com
[7] Computer and Information Technology, Purdue University, West Lafayette, IN, USA, ematson@purdue.edu
[8] Mathematics and Computer Science, Argonne National Laboratory, Lemont, IL, USA, park708@purdue.edu

***Abstract*—This paper evaluates the classification of objects given their signal data via a simple convolutional neural network (CNN). Many of the signal processing neural networks involve sound frequency data or Doppler signatures that contain the characteristic features of each object. In this study, we use frequency-intensity data within range-time domain from a Frequency-Modulated Continuous-Wave (FMCW) radar to classify detected objects. The application of various data augmentation methods mitigated the scarcity of labeled data from our field experiments. Time stretching, frequency shifting and noise addition preserved the semantic information of each range-time data, further improving the models ability to generalize. Modifications applied to our data, which is then converted into a low-level log-scaled mel-spectrogram representation, are learned by CNN models with a set of convolutional and max-pooling layers along with fully-connected layers and selective residual module. Based on our experiments, we conclude that raw radar data can be used for training CNNs for classification and thus can be used to classify a car, a human, and an UAV.**

***Index Terms*—object classification, radar system, data augmentation, convolutional neural networks**

I. Introduction

CLASSIFICATION of objects by extracting features from data using convolutional neural networks (CNNs) has been studied with great interest over the past several years. Image data were mostly used in training procedures of neural networks, and most of them gave promising results by showing performances that easily surpassed human capabilities of cognizance [1], [12]. Several large datasets such as ImageNet [2] and UrbanSound8k [3] have made large contributions to improving the accuracy and overall performance of object classification and semantic segmentation [4], [9], [10], [12]. Nevertheless, many of the publicly available and well-established datasets mostly consist of images and sound files that contain explicitly characteristic features of respective objects. This indicates that most of the previous experiments with CNNs were extracting features that were apparent even from a human perspective. On the other hand, there are data without apparent features, such as raw radar data, which only specify frequency intensity information along range and time axis (range-time data). Such data do not seem to contain explicitly characteristic information of an object. Changes in the range of an object along the time axis do not have class-dependent information unlike other forms of data such as images. Hence, given distance and time alone, it is very difficult for even humans to show high classification performance. In contrast, deep convolutional neural networks can learn underlying characteristics of data given enough data samples. In other words, deep convolutional neural networks can recognize features of data that are not obvious to human. Thus deep convolutional neural networks are applied to classifying objects from raw radar data. The raw radar data was acquired from a compact and low power Frequency-Modulated Continuous-Wave (FMCW) radar, which can be performed at public places for monitoring purpose [5], [6].

One of many drawbacks of using deep neural networks (DNNs) is that it requires a huge amount of labeled, preprocessed data; however, this does not mean simply using any random data from random sources. Low quality data may either result in an underperforming neural network model or produce unexpected, irrelevant outcomes. The performance of a DNN is determined by how well the model generalizes on ample amount of data. The requirement of large dataset may discourage experiments with a small number of data, but such deprivation of data can be overcome by data augmentation [4]. Data augmentation is a technique that applies multiple forms of changes to data while preserving their semantic information. For images, image translation, horizontal reflection and grayscaling are conventional methods for augmenting image data [7], [8]. Since our data is in the frequency domain, several conventional augmentations, such as time stretching, pitch shifting and adding noise to frequency data were applied to counteract the scarcity of labeled data and increase the size

978-1-5386-7713-1/19/$31.00 ©2019 IEEE

of dataset. With the augmented dataset trained on CNNs with different number of layers and selective residual module, we achieved a reasonable performance in classifying objects. The goal of this paper is to explore the possibility of a new type of data (range-time data), which is unconventional, being trained on CNNs for object classification.

II. Methods

Using DNNs to extract features and learn data representations using an ample amount of data is possibly the most useful and effective way to model real-world data. DNNs are universal function approximators, which use dot products of input and learnable parameters (θ), along with backpropagation that calculates error terms based on gradients to optimize the cost function, J [9]. While there are several variants of deep neural networks that can model our data, we choose the convolutional neural network. CNNs innate ability to exploit the locality of data through multiple layers of convolution and pooling can act as automatic feature extractors. Such strengths of CNN work to discover characteristic features in each window as it slides over the input X in log-scaled mel-spectrogram representation.

A. Convolutional Neural Networks

Convolutional neural networks (CNN) use the conventional methods of convolution operation and pooling of maximum values from each feature map to train a deep neural network. It learns data representations that is within the given data by weight values that adapt to local features of the given data. These are called $filters$, or kernels. Given our input X, a number of $depth$ filters convolve over X to reduce the size of the input space into $feature\ maps$, which maintain the semantics and characteristic feature values of data before convolution. Max-pooling, which follows convolution operation in CNNs, produces feature maps with maximum values within each receptive fields, or kernels. Such pooling technique down-samples the features, additionally reduces the dimensionality of data, and preserves characteristic values and inherently cover more variations in the input image for a reduced computational cost. This sequence of processes improves robustness to translational shifts within the input space X. Thus, given a random two dimensional input x, filter w, and convolution operation, which is also known as cross-correlation, is represented as follows:

$$y[i,j] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} x[m-i, n-j]\ w[m,n] \qquad (1)$$

Then, max-pooling is applied to extract the maximum value from the resulting feature map y. The entire idea of sliding $m \times n$ kernels through the input space to extract the relevant features are based on the observation that locally useful features are also useful in other places of the input space [10]. This idea derives from the fact that a vision system that has learned certain characteristic features uses the same knowledge at all other locations in an image [11]. CNN became the primary model of our experiment because of the tendency of the model to exploit the locality of data in analyzing their features.

B. Residual Learning

Deep neural networks enable the integration of features of various levels and the depth of the network has been proven to be an important factor in model performance [13], [14]. The model learns multiple features from the lowest level to the higher, more abstract level of data representation. For example, a CNN model learning on images of dogs has its lower layers learning the lower level features of dogs face like horizontal or diagonal edges, and as the layer grows deeper it also learns the more abstract features like dogs round nose and protruding ears. However, stacking more layers does not assure high classification performance of a model. As a network grows deeper, the $degradation$ problem occurs. Previous experiment shows that when a deep network becomes too deep the accuracy starts to degrade, leading to higher training error and thus test error [14]. This suggests that such deterioration of performance results from the limited ability of the deep neural network to fully optimize its cost function. The nature of DNNs provides the ability to learn with gradient descent and back propagation. Yet, the increased depth proportionally increases the number of parameters that need to be optimized. To solve the problem, He et al. [14] proposed a $deep\ residual\ learning$ framework. Optimizing a deep network is a difficult task, considering how many parameters and underlying mappings should be learned to model the relationship between input space X and the label Y. The authors of $ResNet$ [14] reduce such a burden of learning on the network by explicitly connecting layers via $skip\ connection$. Let us denote the inputs to the first layer as x and underlying mapping the network performs as $H(x)$. If we feed forward the x through a skip connection, which performs identity mapping, the stacked layers only need to learn the residual function $F(x) = H(x) - x$ as shown in Figure 1.

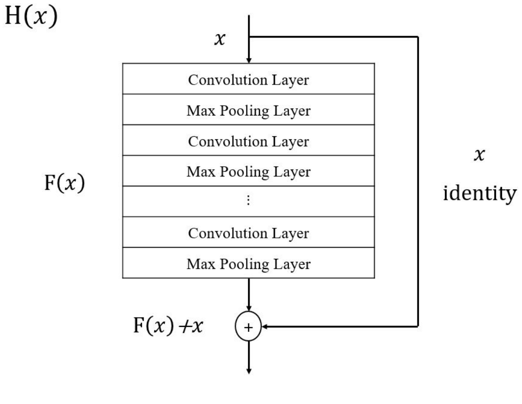

Fig. 1. Residual Learning.

Since it is easier to optimize the residual function than a regular, stacked network, degradation of performance can be avoided by establishing explicit skip connections instead of expecting the stacked network to learn the "underlying mappings" [14]. This additional connection also projects forward the information straight from input space X to the output of the desired layer. In CNNs, peripheral information lost from convolution and pooling process can be complemented by this mapping. The results of various studies [14], [15], [16] support this claim.

III. Object Classification

Field experiments were conducted to collect raw data to train an object classification model. A number of experiments were conducted with a fixed FMCW radar detecting moving objects of respective classes: UAV, car and human. Each object moved in front of the radar within a 30 meter range, since the maximum detection distance of our radar was 30 meters. The experiments yielded desired raw data of UAV, car and humans distance along the time domain. Despite the collection of raw data, classification of objects by training CNNs on radar signal intensity data in range-time domain acquired from our field experiments was challenged by lack of data and low computational power. We have acquired radar signal intensity data in range-time domain in a text file format, which was converted in a form of wav file to be preprocessed into a form of log-scaled mel-spectrogram [10] using the "librosa" implementation. Time stretching, frequency shifting and noise addition applied on our data derived from previous experiments [4], in which environmental sounds containing objects of respective classes are classified with CNN. The concern of applying such feature extraction and data augmentation methods is that the data collected may be in audio file format, which is ".wav", showing characteristics that are analogous to environmental sound files. However, the nature of each datum is a range-time information of each object. This implies that manually extracting features using conventional audio features such as Mel-Frequency Cepstral Coefficients (MFCC) [13] may not work properly on the collected data. Thus, rather than using manually extracted features of traditional methods, we use the data representation learning nature of CNN to acquire the underlying features of our data to be used in object classification.

A. Data Augmentation

Before extracting features from our range-time data, we augment our data using three different augmentation methods that will reform our data to a certain extent, causing deformations that still maintain the important semantic information. Choosing the right parameters for time stretching, frequency shifting and adding noise to augment our dataset should be done based on results from previous experiments [4], [10]. Each of the following alterations are applied before they are converted into our input representation used to train the network as explained below.

1) Frequency shifting: Increments or decrements frequency values while maintaining the relationship between values along the time domain. Similar to pitch shifting in sounds that preserve the harmonic relationships among pitches of different scales, frequency shifting does not disturb the inter-frequency distances. This deformation is applied by frequency shifting the waveform by n half-steps. Each data sample is pitch shifted by 4, 8, 16 and 32 half-steps, causing shift only in frequency values while preserving the time interval and amplitude. Frequency shifting is applied to both the data and sync channels of each sample.

2) Noise Addition: This method modifies data samples by generating random noise to modify each data by a random value, w. Randomly generated noise w from uniform distribution in half-open range $[2, 5)$ is reshaped to fit the length of amplitude values of each sample, multiplied by a scale c to control the size of w and is added to the data channel of the sample, while leaving the sync channel to maintain its original value. While noise addition may appear to be similar to frequency shifting, it affects the amplitude value instead of frequency and time. Adding a noise constant preserves the differences among values while translating the values along the amplitude axis.

3) Time Stretching: Stretches data and sync along the time axis by a fixed rate. After converting the given data to a frequency domain with Short-time Fourier Transform (STFT), we use phase vocoder to time-stretch by rate r. This augmentation alters both the data and sync channels.

The augmentation techniques explained above are applied to each sample before converting them into spectral features. Log-scaled mel-spectrogram that results from feature extraction is the input representation we employ in this experiment to train CNNs correctly classify objects of 4 classes: $\{$*UAV, human, car, others*$\}$.

B. Feature Extraction

Since our raw radar data share similar properties with audio data in that it is in a form of frequency, we converted our range-time data into wav file format (sampled to $5,682Hz$). Then, to explore unseen features of the data, we applied Fast Fourier Transform (FFT) to get a power spectrum from each wav files. The power spectrum and mel-filter banks were multiplied to produce a mel-scaled frequency, then the mel-scaled frequency values were applied with log to derive a log-scaled mel-spectrogram. Parameters used to extract the spectral features are as follows: window size of $512 \times (frames - 1)$ with hop length equal to 512, mel-bands set to 60 along with 41 frames. However, since training each sample without any alterations resulted in a limited number of training data, we split log-scaled spectrograms into N frames (default value of $N = 41$), which consists of 50% overlapping portions with the previous and following frames. As a result, the original raw dataset consists of 1,937 segments and we increased the number of segments up to 3,944 by applying aforementioned data augmentation methods to the raw dataset. Along with these segments of $M \times N$ shape, in which M represents the value of mel-bands, we concatenated their delta values to produce two input channels: log-scaled spectrogram and delta [8]. Extracting spectral features and their corresponding values were implemented by the "librosa" library.

C. Experimental Setup

Setting the parameters of deep neural networks is a heuristic and complicated procedure, especially on atypical data without previous experiments. There is no golden rule to the depth of network, filter size, the number of filters, stride size, and other

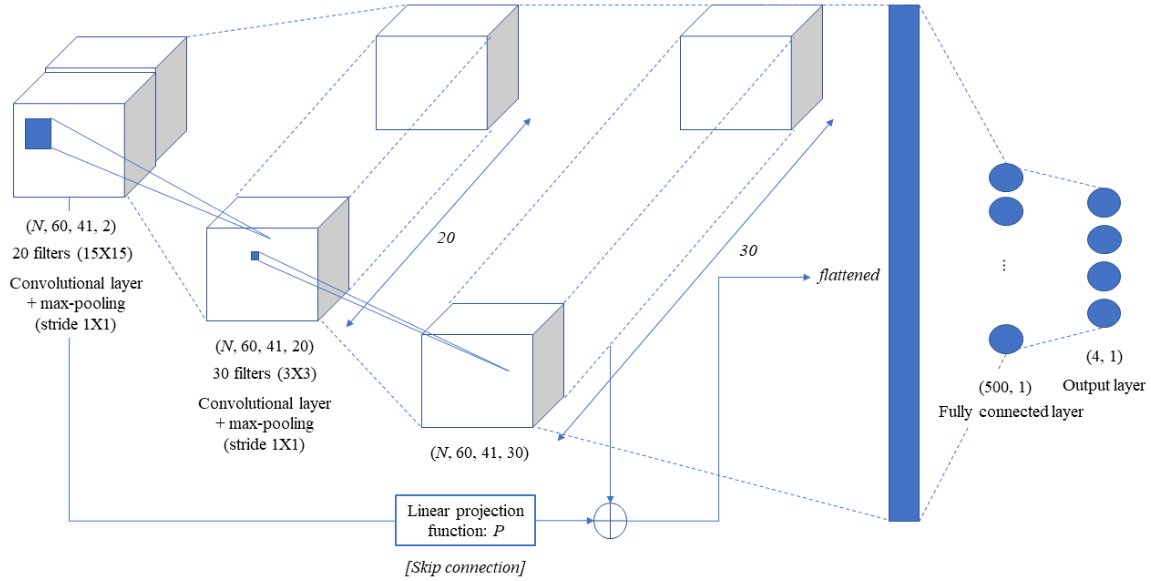

Fig. 2. Double layer CNN architecture with skip connection.

hyperparameters. Hyperparameters such as depth of the network should be set according to the size of dataset and type of data. Due to the limited amount of time, computing power and data, not all possible combinations of hyperparameters could be tested. Within the available conditions, we experimented with several feasible combinations.

In our experiments, we trained networks of varying depth, from 2-layer to 8-layer networks with different sets of hyperparameters. For each network variant, we experimented with and without skip connections [14] connected straight from input space X to the output of the last pooling layer F through identity mapping. Such skip connection was applied to see if feature maps with much of the peripheral information lost from convolution and pooling processes could be complemented by the original input X and thus lead to improved classification performance. To match the output of the last pooling layers dimension with the dimension of input space X, we applied zero-padding and a stride of 1 to all of the convolutional and pooling layers. 2-layer network with skip connection is depicted in Figure 2 as an example. Training the model was based on mini-batch gradient descent with batch normalization after each layer, which comprises of a convolution layer followed by pooling layer, to reduce the training time and variance among each mini-batch (batch size of 32). The networks were trained on both the raw and augmented dataset.

The final models used for evaluation are 2-layer and 6-layer CNNs, both of which are trained with and without skip connections. The best performing model among those is the 6-layer CNN without skip connection. Each layer, L, represents an ith layer of the 6-layer CNN along with the specification of hyperparameters. $L1$ to $L6$ are convolutional layers followed by $L7$, the fully-connected layer and $L8$, the output layer:

$L1$: The first convolutional layer consisted of 12 filters with the size of 9×9 (1×1 stride). This is followed by rectified linear unit (ReLU) activation function and max-pooling (pool size 9×9, pool stride 1×1).

$L2$: The second convolutional layer consisted of 12 filters with the size of 5×5 (1×1 stride). This is followed by ReLU activation function and max-pooling (pool size 5×5, pool stride 1×1).

$L3$: The third convolutional layer consisted of 24 filters with the size of 5×5 (1×1 stride). This is followed by ReLU activation function and max-pooling (pool size 5×5, pool stride 1×1).

$L4$: The fourth convolutional layer consisted of 24 filters with the size of 3×3 (1×1 stride). This is followed by ReLU activation function and max-pooling (pool size 3×3, pool stride 1×1).

$L5$: The fifth convolutional layer consisted of 48 filters with the size of 3×3 (1×1 stride). This is followed by ReLU activation function and max-pooling (pool size 3×3, pool stride 1×1).

$L6$: The sixth convolutional layer consisted of 48 filters with the size of 3×3 (1×1 stride). This is followed by ReLU activation function and max-pooling (pool size 3×3, pool stride 1×1).

$L7$: The output of L6 is fed into the hidden layer of 500 fully connected nodes, followed by a sigmoid activation function.

$L8$: The output layer consisted of 4 output nodes which represent each class. Each is activated with softmax function to calculate the output probability of the 4 classes.

IV. Results

For k-fold cross validation, extracted log-scaled mel-spectrogram features from each wav file were divided equally

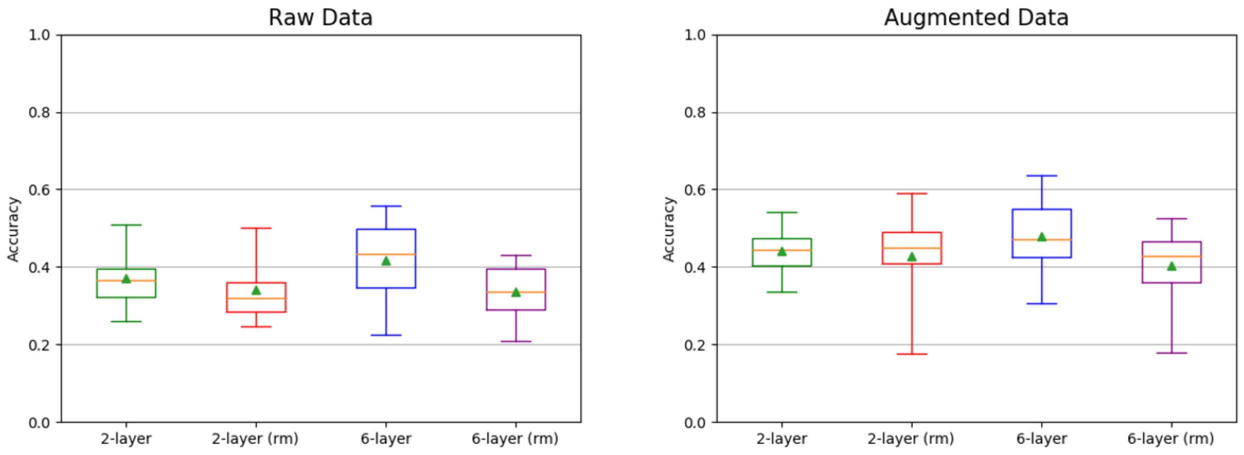

Fig. 3. Comparison of classification accuracy on different datasets. “rm” means with residual module.

TABLE I
ACCURACY EVALUATED ON RAW DATASET

	Plain	Residual Module
2-layer	37.05%	34.09%
6-layer	41.73%	33.56%

TABLE II
ACCURACY EVALUATED ON AUGMENTED DATASET

	Plain	Residual Module
2-layers	44.18%	42.93%
6-layers	47.96%	40.44%

TABLE III
PER-CLASS CLASSIFICATION ACCURACY OF 6-LAYER PLAIN MODEL EVALUATED ON AUGMENTED DATASET

	UAV	Human	Car	Others
Per class accuracy	32.10%	72.59%	43.58%	36.37%

into folds of 5. four fifths of folds were used for training and the last fold of choice was used for model validation. To demonstrate the effectiveness of data augmentation, CNN models are learned and evaluated on both raw dataset and augmented dataset.

The total classification accuracy of the proposed CNNs is in Figure 3. According to the result, the 6-layer model without residual module showed the best performance. This indicates that depth of neural network affects the performance of neural network, supporting the conventional claim [15] that the deeper the network, the better it performs. However, the existence of residual module did not seem to have a meaningful effect on the performance of the neural network. It rather decreased the classification performance possibly due to added number of parameters along with the linearly projected mapping between input space X and the output of the last pooling layer. Such phenomenon can be addressed by the decreased accuracy in Table I and Table II. In addition, data augmentation methods we applied turned out to be effective in refining the dataset and improving the performance of the neural network. Augmented data led to a model stabilization and better convergence of our cost function. Conclusively, increasing the depth of our neural network improved our overall performance, whereas skip connections had an adverse effect on our model.

The best performing model is the 6-layer plain neural network and its per class accuracy is presented in Table 3. Objects that are large in size like humans and cars and thus show high reflection signal have high classification accuracy. However, the small size of UAVs makes it hard to be detected by the radar and the resulting low reflection signal gives the data less features. Therefore, UAVs tend to show low classification accuracy compared to that of humans or cars.

Such result can be interpreted in several ways. Underfitting could have resulted because of our lack of data. Our dataset consisted of approximately 4,000 instances only and this indicates that the network might have underfitted during training because of data scarcity. The range of accuracy in Figure 3 supports this claim. One other cause of such performance issues may have been the lack of high capacity GPUs. Lack of high computing power allowed us to extend up to only an 8-layer network and evaluate limited number of hyperparameter combinations. With more computing power, deeper models with various combinations of hyperparameters could be trained and higher performance is expected.

V. Conclusion

This paper elaborates our attempt to classify objects on raw radar data using CNNs. The absence of previous works on using range-time data to train neural network models for classification provided us a new ground for research on a new type of data.

The goal of our research was testing our hypothesis that range-time data hold enough characteristic features of each object to be used in training deep neural networks for classification. Although classification accuracy of UAV seems relatively low, those of car and human show the possibility of using the raw radar data to train deep neural networks for classification. Considering that lack of data, especially for UAV and car classes, has led to such outcome, collecting more data and providing class-dependent refines and augmentations could improve performance of the models. Another, our experiment was limited to CNN models of shallow depth and several combinations of hyperparameters due to lack of high capacity computing power. Upon being provided enough computing power capable of training deeper network architectures and testing various combinations of hyperparameters, higher classification accuracy could be anticipated. Moreover, since our dataset is in time-series, models that are effective for interpreting time-series data, such as recurrent neural network (RNN) based models, could unveil temporal dependencies among frequency values that could not be learned by CNNs.

In this paper, we propose that conventional data such as image and sound data are not the only type of data capable of holding unique, object-dependent features, but other seemingly featureless data such as range-time data can hold features that could be exploited by deep neural networks. We anticipate that our research contributes to new ground for research on a new type of data.

Acknowledgment

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the National Program for Excellence in SW supervised by the IITP(Institute for Information & communications Technology Promotion). (2016-0-00017)

References

[1] S. Dodge and L. Karam, “A study and comparison of human and deep learning recognition performance under visual distortions”, in *Computer Communication and Networks: 26th International Conference on Computer Communication and Networks, ICCCN 2017, Vancouver, Canada, July 31-August 3*, pp. 1-7, 2017.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks”, *Advances in neural information processing systems, Lake Tahoe, NV, USA, December 3-8*, pp. 1097-1105, 2012.

[3] J. Salamon, C. Jacoby, and J. P. Bello, “A dataset and taxonomy for urban sound research”, *ACM international conference on Multimedia: Proceedings of the 22nd ACM international conference on Multimedia. ACM 2014, Orlando, FL, USA, November 3-7*, pp. 1041-1044, 2014.

[4] J. Salamon, and J. P. Bello, “Deep convolutional neural networks and data augmentation for environmental sound classification”, *IEEE Signal Proc. Let.*, vol. 24-3, pp. 279-283, Jan. 2017.

[5] S. Park, Y. Kim, E. T. Matson, and H. A. Smith, “Accessible synthetic aperture radar system for autonomous vehicle sensing”, *In IEEE Sensors Applications Symposium (SAS)*, Cantana, Italy, April 2016, pp. 1-6.

[6] S. Park, S. Shin, Y. Kim, E. T. Matson, K. Lee, P. J. Kolodzy, J. C. Slater, M. Scherreik, M. Sam, J. C. Gallagher, B. R. Fox, and M. Hopmeier, “Combination of radar and audio sensors for identification of rotor-type unmanned aerial vehicles (uavs)”, *In IEEE Sensors*, November 2015, pp. 1-4.

[7] S. C. Wong, A. Gatt, V. Stamatescu, and M. D. McDonnell, “Understanding data augmentation for classification: when to warp?”, *arXiv preprint arXiv:1609.08764*, 2016.

[8] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation”, *Medical Image Computing and Computer-Assisted Intervention: Proceedings on the 18th International Conference on Medical Image Computing and Computer-Assisted Intervention, MICCAI 2015, Munich, Germany, October 5-9*, pp 234-241, 2015.

[9] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives”, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35-8, pp. 1798-1828, Mar. 2013.

[10] K. J. Piczak, “Environmental sound classification with convolutional neural networks”, *Machine Learning for Signal Processing: Proceedings of IEEE 25th International Workshop on Machine Learning for Signal Processing, MLSP 2015, Boston, MA, USA, September 17-20*, pp. 1-6, 2015.

[11] G. E. Hinton, S. Sabour, and N. Frosst, “Matrix capsules with EM routing”, *International Conference on Learning Representations: Proceedings of the seventh International Conference on Learning Representations, ICLR 2018, New Orleans, LA, May 6-9*, pp. 1-15, 2018.

[12] K. He, X. hang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”, *International Conference on Computer Vision: Proceedings of the IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 13-16*, pp. 1026-1034, 2015.

[13] C. V. Cotton, and D. P. W. Ellis, “Spectral vs. spectro-temporal features for acoustic event detection”, *Applications of Signal Processing to Audio and Acoustics: Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, WASPAA 2011, New Paltz, NY, USA, October 16-19*, pp. 69-72, 2011.

[14] K. He, X. Zhang, S. Ren, J. Su, “Deep residual learning for image recognition”, in *Computer Vision and Pattern Recognition: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30*, pp. 770-778, 2016.

[15] K. Simonyan, and A. Zisserman, “Very deep convolutional networks for large-scale image recognition”, in *arXiv preprint arXiv:1409.1556*, 2014.

[16] C. Szegedy, W. Liu, Y. Jia, P. ermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions” in *Computer Vision and Pattern Recognition: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12*, pp. 1-9, 2015.

[17] W. Yin, H. Schtze, B. Xiang, B. Zhou, “ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs”, in *arXiv preprint arXiv:1512.05193*, 2015.