REAL-TIME SCENE UNDERSTANDING OF UNMANNED AERIAL VEHICLE IMAGERY WITH AND BEYOND VISULA SENSORS BY DEEP LEARNING

Maryam Rahnemoonfar,[*] Clay Sheppard,[†] and David Bridges[‡]

Deep Convolutional Neural Networks (CNNs) have emerged as a powerful model for classifying image content, and are widely considered in the computer vision community to be the de facto standard approach for most problems. Here we present a deep convolutional approach for classification of Aerial imagery taken by UAV. We applied our network on optical and infrared imagery taken with UAV RS-16 from Port Mansfield, TX. The proposed architecture is able to predict the labels for the images captured by UAVs in real time. We trained and tested our architecture on different combination of optical and IR Imagery. In our experiment it is harder for network to learn the characteristics of IR imagery in comparison to optical imagery. However, in the case that we enrich our IR training dataset with optical imagery, it reached a high accuracy similar to optical imagery. Experimental results in comparison with the ground-truth show the efficiency of our approach for the classification of UAV imagery across the spectrum.

INTRODUCTION

Unmanned Aerial Vehicles (UAVs) have seen unprecedented level of growth in recent years. UAVs are increasingly used for surveillance, fire detection, reconnaissance, mapping, cartography, landslide monitoring, inspection, traffic monitoring, search and rescue, to name a few application domains [1-3]. It is important for many of the aforementioned applications to perceive the scene in real-time. If a high altitude UAV can perform scene understanding by predicting the nature of landscape at any particular location, then a lower altitude UAV or a UGV (unmanned ground vehicle) can be send to the desired location for further detailed analysis for a specific application. The main purpose of the scene understanding using high altitude UAV is to remove the overhead of finding the desired location for the lower altitude UAV or UGVs for their specific purposes. Current computer vision algorithms and datasets are designed and evaluated on lab setting human centric photographs taken horizontally with a close distance to the object. For UAV imagery taken vertically in high altitude (10m to 100m) the objects of interest are relatively small and with less features, questioning the sustainability of current methods. For exam-

[*] Assistant Professor, Department of computer science, Texas A&M University-Corpus Christi
[†] Student, department of computer science, Texas A&M University-Corpus Christi.
[‡] Associate Professor, Department of Engineering, Texas A&M University-Corpus Christi.

ple an aerial image of building has only the top view of the building and except the roof, no distinguishable features are there. On the other hand, corresponding terrestrial image of the same building has many features like door, windows, walls which makes it easier for recognition even by human.

Several approaches are utilized for object detection in aerial images. In 1988, Huertas and Nevatia [4] proposed a technique to detect buildings according to the rectangular components and shadow information. Based on the similar concept, Sirmacek and Unsalan [5] presented an approach to detect buildings using invariant color features, edge, and shadow information. Cote and Saeedi [6], introduced a method using Harris corner detector for building detection. Manno-Kovacs and Sziranyi [7] proposed a framework based on region orientation with several steps to achieve building detection. The main drawback of aforementioned methods is that they are not suitable for real time applications. Moreover, traditional approaches cannot learn new features automatically [8-10]. Feature engineering is required to decide explicitly what features to learn. With the recent advances in GPU technology, deep learning has emerged as a feasible solution to real time applications. Deep learning consists of simultaneous learning of hierarchical models from multiple levels of representation that helps to identify input data [7, 11]. First, a complex task is decomposed into features that are fed to the next layer. Ideally, each layer generates results that approximate the expected solution. In the literature deep learning has been successfully exploited for object recognition [12-15], speech recognition [7, 16-19] and language processing [20, 21]. However limited work is performed for scene understanding using UAV images.

In this paper, a deep convolutional neural network framework is implemented to achieve a fast and accurate result for classification of Aerial images taken with UAV. The network architecture comprises a series of convolution and pooling layers followed by fully connected layers. We applied our network on optical and infrared (IR) imagery taken with UAV RS-16 from Port Mansfield, TX. The proposed architecture is able to predict the labels for the images captured by UAVs in real time. We trained and tested our architecture on different combination of optical and IR Imagery. Our optical and IR imagery are not registered. In our experiment it is harder for network to learn the characteristics of IR imagery in comparison to optical imagery. However in the case that we enrich our IR training dataset with optical imagery and test it in IR imagery it reached a high accuracy similar to optical imagery. This result is really important for UAV flight experiment with limited payload. IR sensors can be used only on testing phase and still reach the high accuracy as optical imagery.

The rest of the paper is organized as follows. Background of Convolutional network is presented in section 2. Proposed methodology is explained in section 3. Experimental results are presented in section 4. Finally, the discussions and conclusions are drawn in section 5.

CONVOLUTIONAL NEURAL NETWORKS (CNN)

Convolutional Neural Networks (CNN) comprises various convolutional and pooling (subsampling) layers that resembles human visual system [22]. Generally, image data is fed to the CNN that constitute an input layer and produces a vector of reasonably distinct features associated to object classes in the form of an output layer. Between input and output layers there are hidden layers in the form of series of convolution and pooling layers followed by fully connected layers [23, 24].

The main building block of a CNN is convolutional layer. The parameters of this layer include a set of learnable filters (or kernels), which have a small receptive field, but spread through the full depth of the input. Every filter is convolved along the width and height of the input volume and produces a two-dimensional feature vector of that filter. All the feature vectors generated through different filters are stacked together along the depth dimension to form an output volume of the convolution layer [23]. In order to control the number of free parameters in convolutional layers, the parameters are shared.

The pooling layer is a form of non-linear down sampling applied to reduce the size of the feature vectors generated through convolution. The idea is to reduce the number of parameter required and number of computation required and therefore to control overfitting. Several non-linear functions are available to perform pooling such as max pooling, min pooling, and average pooling [23]. Most common approach to apply pooling is between successive convolution layers. After a series of convolutional and pooling layers, finally the abstract-level reasoning is performed using fully connected layer. In this layer, neurons have full connections to all the activations in the previous layer, similar to classical Neural Networks [23]. There can be many fully connected layers before the final output layer.

Generally, CNN models are trained using two different approaches. In first approach, the network is trained from scratch with randomly initialized weights. In second approach a pre-trained network is used with fine tuning [25, 26].

METHODOLOGY

In this section we explain the network architecture used in this work and the training methodology. The input to the network is 128X128 Aerial images.

Network architecture

The network architecture developed in this research is shown in Figure 1. As we can see in this figure, the first layer of the network is input layer containing the input image. The convolutional layer COV1 takes the 3 bands (RGB) in the input image and produces 6 different feature maps using a 5X5 kernel function. The convolution layer is followed by 2X2 max pooling layer (MP1 layer) with stride 2. 2X2 Max pooling reduces the dimensions of the image by taking the maximum value in a window at every depth slice in the feature map by 2 along both width and height. A stride of 2 indicates that the window is moved two pixels at a time. This condenses the information by reducing the features by half. Reducing the dimensions of the image reduces computation time and allows the

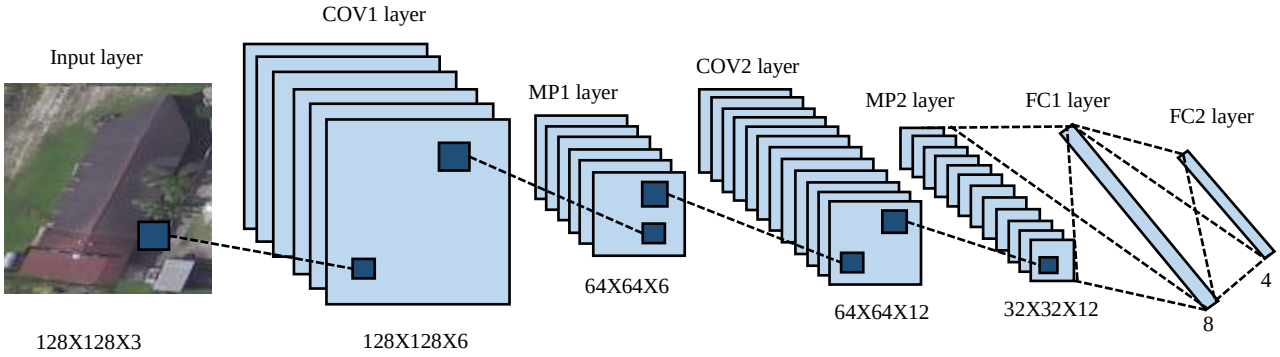


Figure 1: Network architecture used for scene understanding

model to fit into GPU memory [27]. After first pooling layer we applied another layer of convolution (COV2 layer). This convolutional layer takes 6 output features from the previous layer as input and maps them to 12 feature maps using a 5X5 kernel function. The convolution layer is again followed by a 2X2 max pooling layer (MP2 layer) with stride 2. This max pooling layer is followed by a fully connected layer (FC1 layer). As can be seen in Figure 3 the size of this fully connected layer is 8. We used dropout to prevent the network from over-fitting at this stage. Moreover, it helps network to learn fast. According to this technique, some units are randomly dropped along with their connections [28]. How many connections will be dropped off is decided by the percentage of dropout. In our research 50% of connections were randomly dropped off while training the network. Finally, the last fully connected layer (FC2) after the dropout layer gives the prediction for classification with the output size 4 because we have used four classes in this research namely, building, ground, road, and tree. Softmax was used after the final output layer. Softmax activation is the normalized exponential probability of class observations represented as neuron activations. It is used for the output layer to ensure that the sum of the components of output vector is equal to 1. Batch normalization was performed after every convolution to remove internal covariate shift [29]. To minimize the error an Adam optimizer is used [30] because it requires little tuning of hyperparameters. The learning rate for the Adam optimizer is set at a constant 1e-3. Cross entropy [31] is used as the cost function. Rectified linear function [32] is used as an activation function.

EXPERIMENTAL RESULTS

Dataset

The dataset used in this work comprises images captured by both optical and IR cameras mounted on a UAV. The study area for this research is Port Mansfield, TX and data was captured on March 4-6, 2015. Figure 2 shows the mosaic created by IR images. Figure 3 shows the google earth image of the study area along with the sample image captured by UAV in the inset. The UAV platform used in this work is Recon System™ (RS-16) Unmanned Aircraft System. The aircraft is a multi-payload, long endurance system capable of performing safe and successful civil missions in remote locations. Details of the aircraft, payload, and operations can be found in [33]. The optical camera used in this work is FCBEH6300, 3.27 Megapixel, 20x Zoom, HD color block camera. The resolu-

tion of the captured images is 1920 x 1080. The thermal camera used in this research is FLIR TAU2 which is a long wave Infrared thermal Camera. The spectral band is between 7.5 and 13.5 micrometer. The red circle in the Figure 3 shows the limit of our range based on the C2 radio link between the GCS and the RS-16, and the blue box shows the air-space limitations of our COA. A sample image captured by UAV can also be seen in the inset.

Figure 2: the mosaic of Infrared Imagery in our experiment

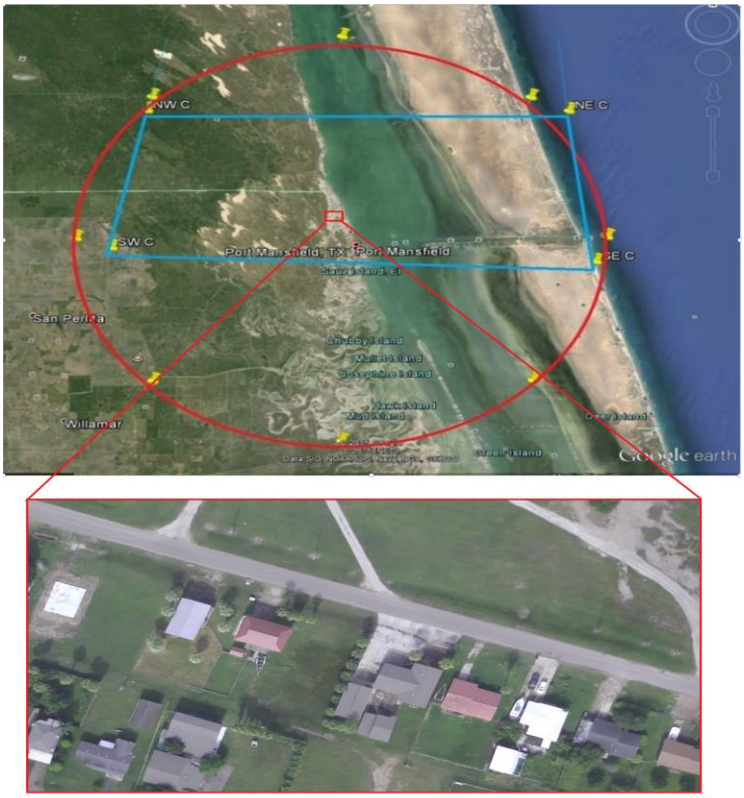


Figure 3: Location of the study area on Google earth along with the sample image captured by UAV in the inset

Training of the network

The network was implemented using TensorFlow [34] running on an NVidia 980Ti GPU. We used different combination of training and testing images for 5 different experiments according to Table 1.

Table 1.Number of images that we used in 5 different experiments

experiments	Training (number of Images)	Testing (number of Images)
Exp1:Opt-Opt	3505	1502
Exp2:Opt-IR	5007	313
Exp3:IR-Opt	313	5007
Exp4:IR-IR	251	62
Exp5: OptIR-IR	5258	62

In the first experiment (Opt-Opt), 3505 optical images were used for training and 1502 optical images for testing. In the second experiment (Opt-IR), 5007 optical images were used for training and 313 infrared imagery for testing. In the third experiment (IR-Opt), 313 IR images were used for training and 5007 optical images for testing. In the fourth experiment (IR-IR) 251 IR images were used for training and 62 IR images for testing. In the fifth experiment (OptIR-IR) 5258 optical and infrared images were used for training and 62 IR images for testing.

Network testing and validation

We tested and validated our algorithm on various numbers of optical and IR imagery according to Table1. Figure 4 shows the training accuracy where horizontal axis represents number of steps and vertical axis represents accuracy. The network was trained with the dropout value of 50 which indicates that 50% connections were dropped off randomly from the fully connected part of the network while training. The average running time for each image is around 12 milliseconds. Figure 5 shows some of the optical and IR images in our dataset for four different classes which are building, road, ground and tree. We calculated the overall and classwise accuracy according to equation 1.

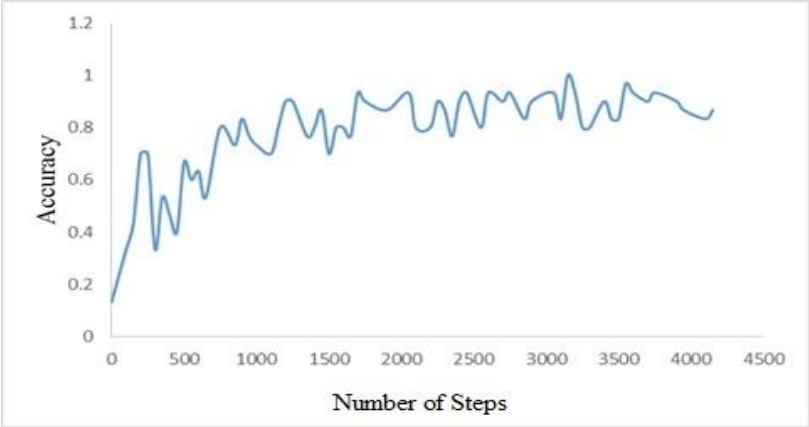


Figure 4: Training accuracy at dropout value 50

$$Accuracy = \frac{\sum TP + \sum TN}{\sum TP + TN + FP + FN} \quad (1)$$

where TP is true positive, TN is true negative, FP is false positive and FN is false negative.

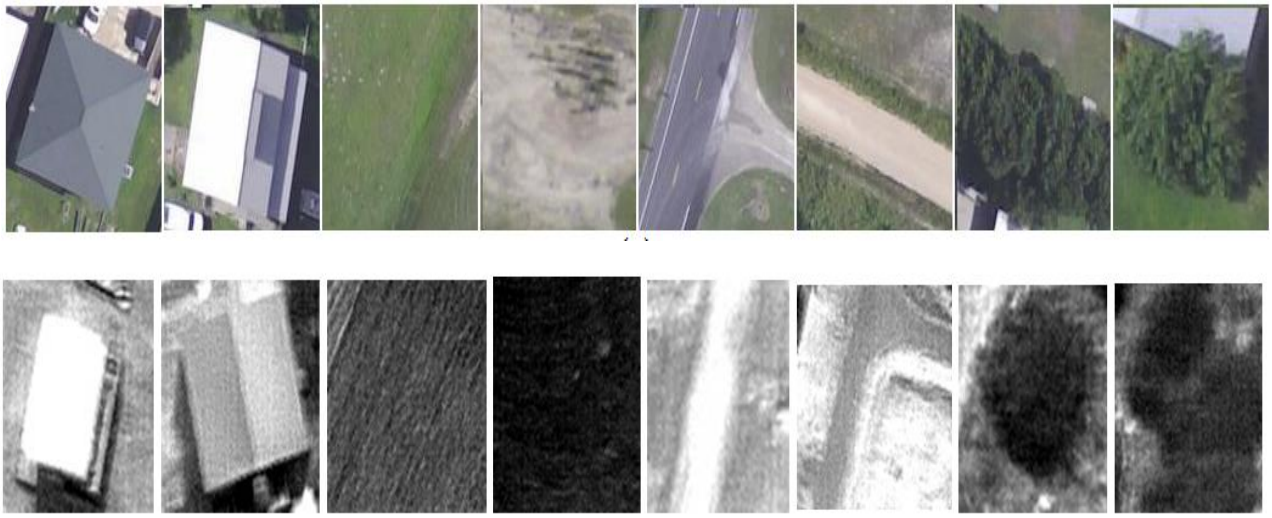


Figure 5: Sample images from building, ground, road and tree classes in both optical (top) and IR (bottom) dataset

The overall accuracy for five different experiments is listed in table 2.

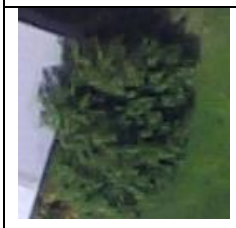
Table 2: overall accuracy for five different experiments

experiments	Overall accuracy (%)
Exp1:Opt-Opt	93.6
Exp2:Opt-IR	90.75
Exp3:IR-Opt	77.25
Exp4:IR-IR	87.00
Exp5: OptIR-IR	92.75

The overall accuracy for the first experiment where it was trained and tested on optical imagery is 93.6%. In the second experiment which was trained on optical and tested in IR imagery, we reached the overall accuracy of 90.75%. While the network is trained on optical imagery, it is able to learn the characteristic of IR images. However the opposite is not quite true. When trained on IR imagery and tested on optical imagery, the learning parameters are not quite transferable and we got the low accuracy of 77.25%. This is logical because IR imagery have lower resolution and less features in comparison to optical imagery. In the case that we train and test the network on IR images, the accuracy is 87% which is higher than previous experiment (training on IR and testing on Opt) but still it is lower than the first experiment. In the fifth experiment we added some of our IR imagery to the optical imagery that we used in the second experiment and we tested on another set of IR imagery. This time we reached the accuracy of 92.75%. In none of the experiments we used registered or correspondent optical and thermal imagery.

Table 3 presents some difficult images for which our algorithm predicts accurately. As can be seen in Table 3, the network is able to predict well on some difficult images. One of the building images presented here is round shaped, and despite being not trained to predict round shaped buildings explicitly, the network was able to predict it well. In the ground images and one of the tree images mentioned in the table, there can be observed road like structures, but based on the dominant feature present in the image the network was able to predict it accurately. The road images presented in Table 3 have no proper boundaries but still network could detect the roads.

Table 3. Correctly classified images and their label

Sample images		Predicted Label
		Building
		Ground
		Road
		Tree

Class wise Accuracy Assessment

In this section we calculated class wise statistics to analyze the performance of our method on individual classes. We computed accuracy for all the classes as shown in Table 4.

Table 4: Classwise accuracy for five different experiments

experiments	Building accuracy (%)	Ground accuracy (%)	Road accuracy (%)	Tree accuracy (%)
Exp1: Opt-Opt	98	90	91	96
Exp2: Opt-IR	88	93	85	97
Exp3: IR-Opt	81	59	80	89
Exp4: IR-IR	84	87	77	100
Exp5: OptIR-IR	89	95	89	98

From Table 4, it can be observed that building and tree classes have higher accuracy almost in all five experiments compared to ground and road. There are some false positives and false negatives observed in ground and road classes that are responsible for relatively lower accuracy as compared to building and tree classes. Further investigations by looking at the images in the first experiment which are not classified correctly, revealed that few samples from ground are classified as road and vice versa. The reason is few samples in the road class are mud roads (dirt roads) that have similar texture with ground.

Building detection has the lowest accuracy in the third experiment which is trained on IR and tested on optical imagery. By looking at figure 5 and comparing building classes in IR and optical imagery, it is obvious that optical images have richer features in comparison to IR images. Buildings in IR images are like a white box but in optical images the roof structure is more distinct. Therefore when the network is trained on thermal imagery it has fewer features to learn for building class and therefore when it is tested on optical imagery which more features it is less accurate. Road detection is less accurate o fourth experiment when it is trained and tested on IR imagery. However for class trees, in the fourth experiment we reached 100% accuracy. It shows that tree detection has the best results when it is trained and tested on IR imagery. This experiment proves that the CNN network is able to learn parameters across spectrum. This is beneficial when there are limitations on UAV payload or on the number of sensors that can be used simultaneously. According to our experiments optical sensors are the best for detecting buildings while IR sensors are the best for detecting trees. For ground and road classes that had similar features in our experiment, combining both IR and optical imagery in the training time will increase the accuracy for both classes.

CONCLUSION

We presented a deep convolutional neural network framework for classification of UAV imagery with both optical and infrared sensors. Traditional methods require feature engineering to learn features explicitly. The main advantage of using deep learning is that, unlike traditional methods, it automatically learns features. Semantic outputs are generated to classify various objects such as building, tree, ground, and road without additional translation. Our network architecture comprises series of convolution and pooling layers followed by fully connected layers. We applied our network on optical imagery taken with UAV RS-16 from Port Mansfield, TX. The proposed architecture is able to predict the labels for the images captured by UAVs with different sensors. We tested our network on the combination of different set of optical and IR imagery. This experiment proves that the CNN network is able to learn parameters across spectrum. According to our experiments optical sensors are the best for detecting buildings while IR sensors are the best for detecting trees. Although the images in our experiment are not registered in any sense, still our network is able to detect different classes across spectrum. When there is a limitation on UAV payload, we can use just one of the sensors at testing time and still reach a high accuracy.

REFERENCES

[1] K. P. Valavanis, *Advances in unmanned aerial vehicles: state of the art and the road to autonomy* vol. 33: Springer Science & Business Media, 2008.

[2] U. Niethammer, M. James, S. Rothmund, J. Travelletti, and M. Joswig, "UAV-based remote sensing of the Super-Sauze landslide: Evaluation and results," *Engineering Geology,* vol. 128, pp. 2-11, 2012.

[3] S. Nebiker, A. Annen, M. Scherrer, and D. Oesch, "A light-weight multispectral sensor for micro UAV—Opportunities for very high resolution airborne remote sensing," *The international archives of the photogrammetry, remote sensing and spatial information sciences,* vol. 37, pp. 1193-1199, 2008.

[4] A. Huertas and R. Nevatia, "Detecting buildings in aerial images," *Computer Vision, Graphics, and Image Processing,* vol. 41, pp. 131-152, 1988.

[5] B. Sirmacek and C. Unsalan, "Building detection from aerial images using invariant color features and shadow information," in *Computer and Information Sciences, 2008. ISCIS'08. 23rd International Symposium on*, 2008, pp. 1-5.

[6] M. Cote and P. Saeedi, "Automatic rooftop extraction in nadir aerial imagery of suburban regions using corners and variational level set evolution," *IEEE transactions on geoscience and remote sensing,* vol. 51, pp. 313-328, 2013.

[7] A. Manno-Kovacs and T. Sziranyi, "Orientation-selective building detection in aerial images," *ISPRS Journal of Photogrammetry and Remote Sensing,* vol. 108, pp. 94-112, 2015.

[8] H. Zhang, A. C. Berg, M. Maire, and J. Malik, "SVM-KNN: Discriminative nearest neighbor classification for visual category recognition," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, 2006, pp. 2126-2136.

[9] Y. LeCun, F. J. Huang, and L. Bottou, "Learning methods for generic object recognition with invariance to pose and lighting," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, 2004, pp. II-97-104.

[10] D. G. Lowe, "Object recognition from local scale-invariant features," in *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, 1999, pp. 1150-1157.

[11] S.-h. Zhong, Y. Liu, and Y. Liu, "Bilinear deep learning for image classification," in *Proceedings of the 19th ACM international conference on Multimedia*, 2011, pp. 343-352.

[12] Y. Kawano and K. Yanai, "Food image recognition with deep convolutional features," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, 2014, pp. 589-593.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097-1105.

[14] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.

[15] K. Sohn, D. Y. Jung, H. Lee, and A. O. Hero, "Efficient learning of sparse, distributed, convolutional feature representations for object recognition," in *2011 International Conference on Computer Vision*, 2011, pp. 2643-2650.

[16] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 609-616.

[17] S. Xue, O. Abdel-Hamid, H. Jiang, L. Dai, and Q. Liu, "Fast adaptation of deep neural network based on discriminant codes for speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, pp. 1713-1725, 2014.

[18] Z. Huang, M. Dong, Q. Mao, and Y. Zhan, "Speech emotion recognition using CNN," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 801-804.

[19] J. Markoff, "Scientists see promise in deep-learning programs," *New York Times*, 2012.

[20] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 160-167.

[21] S. Gouws, "Deep unsupervised feature learning for natural language processing," in *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Student Research Workshop*, 2012, pp. 48-53.

[22] S. Filipe and L. A. Alexandre, "From the human visual system to the computational models of visual attention: a survey," *Artificial Intelligence Review*, vol. 39, pp. 1-47, 2013.

[23] S. L. Phung and A. Bouzerdoum, "MATLAB library for convolutional neural networks," *University of Wollongong, Tech. Rep., URL: http://www. elec. uow. edu. au/staff/sphung*, 2009.

[24] T. Liu, S. Fang, Y. Zhao, P. Wang, and J. Zhang, "Implementation of Training Convolutional Neural Networks," *arXiv preprint arXiv:1506.01195*, 2015.

[25] D. S. Maitra, U. Bhattacharya, and S. K. Parui, "CNN based common approach to handwritten character recognition of multiple scripts," in *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, 2015, pp. 1021-1025.

[26] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, pp. 142-158, 2016.

[27] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv preprint arXiv:1603.07285*, 2016.

[28] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929-1958, 2014.

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

[30] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[31] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of operations research*, vol. 134, pp. 19-67, 2005.

[32] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 8609-8613.

[33] D. H. Bridges and D. Yoel, "Unmanned Aircraft Operations at Texas A&M University–Corpus Christi," in *AIAA Infotech@ Aerospace*, ed, 2016, p. 0745.

[34] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.