# *Performance Evaluation of Hybrid CNN for SIPPER Plankton Image Classification*

*Hussein A. Al-Barazanchi, Abhishek Verma, and Shawn Wang*
Department of Computer Science
California State University
Fullerton, CA, U.S.A.
hussein_albarazanchi@csu.fullerton.edu, averma@fullerton.edu, xwang@fullerton.edu

*Abstract*— **Plankton are a diverse group of organisms that live in large bodies of water. They are a major source of food for fishes and other larger aquatic organisms. The distribution of plankton plays an important role in the marine ecosystem. The study of plankton distribution relies heavily on classification of plankton images taken by underwater imaging systems. Since plankton are very different in both sizes and shapes, plankton image classification poses a significant challenge. In this paper we proposed hybrid classification algorithms based on convolutional neural networks (CNN). In particular, we provide an in depth comparison of the experimental results of CNN with Support Vector Machine and CNN with Random Forest. Unlike traditional image classification techniques these hybrid CNN based approaches do not rely on features engineering and can be efficiently scaled up to include new classes. Our experimental results on the SIPPER dataset show improvement in classification accuracy over the state of the art approaches.**

***Keywords— SIPPER Plankton Images; Convolutional Neural Network; SVM; Random Forest; Image Search***

## I. Introduction

Plankton are a highly diverse collection of organisms that exist in water columns. They are the main source of food for fish and other larger organisms in their environment. Plankton are known to play a very critical role for the survival of various other life forms in oceans. Plankton has two main types: phytoplankton and zooplankton. Phytoplankton are plant type while the zooplankton are animal type. Phytoplankton are also known to be a major contributor in the carbon fixation cycle. Plankton population of either type is very sensitive to changes in their environment. For example, their population may change rapidly in response to changes in the level of pollution. Therefore, the change of plankton population in the marine ecosystem is often used as an indication of environmental issues. The study of plankton distribution in terms of both numbers and types can improve our understanding of the marine ecosystem and facilitate the research on many environmental issues.

In the early days, scientists were limited to the use of traditional techniques to investigate the distribution of plankton, such as Niskin bottles, towed nets, or pumps to collect samples. The analysis and classification were done manually by an expert. This process was very laborious and time consuming. The invention and advances in underwater imaging systems provided an alternative approach and dramatically improved the study of plankton distribution. Several systems were developed for plankton sample collection using underwater imaging systems. The three major imaging systems are Video Plankton Recorder (VPR) [1], under water holographic camera system (HOLOMAR) [2], and Shadowed Image Particle Profiling and Evaluation Recorder (SIPPER) [3]. These systems enabled scientists to continuously collect huge amount of plankton samples. However, for many years, the analysis and classification of the sample plankton images remained a manual process. More recently, with rapid advances in computer vision techniques, there were a number of algorithms developed to automatically recognize and classify plankton images.

One of the early works on plankton image classification can be traced back to Tang et al. [4] where they used samples collected from VPR devices. Their approach used Fourier descriptors with gray-scale morphological granulometries and invariant moment features. In 2005, Lue et al. [5] developed a system that achieved 90% accuracy on plankton images obtained by SIPPER system. Their technique utilized active learning with Support Vector Machines (SVM). In the following year, a new technique named normalized multilevel dominant eigenvector estimation was presented by Tang et al. [6]. Their approach used shape descriptor and achieved 91% accuracy. In 2009, Zhao et al. [7] proposed a system based on multiple classifiers and random sampling. They improved the classification accuracy to around 93%. More recently Li et al. [8] developed a technique called PNDA to reach the accuracy level of 95%.

All the aforementioned algorithms rely on features engineering. The reliability of accuracies depends on the design and extraction of features in the existing data. This poses a challenge in the integration of a new class to the existing system. Each new class may require intensive work to find new features that can represent those new classes. Depending on the quality of feature design, maintaining the accuracy may not always be possible. In this paper we propose hybrid classification algorithms based on convolutional neural networks (CNN). In particular, we provide an in depth comparison of the experimental results of CNN with Support Vector Machine and CNN with Random Forest. These hybrid CNN based approaches do not rely on features engineering and can be efficiently scaled up to include new classes. Our experimental results on the SIPPER dataset show improvement in classification accuracy over state of the art approaches.

TABLE I. PLANKTON TYPES AND THEIR DISTRIBUTION

| Class No. | Class Name | # Samples | Average Height | Average Width |
|---|---|---|---|---|
| 0 | Acantharia | 131 | 46 | 70 |
| 1 | Calanoid | 172 | 319 | 346 |
| 2 | Chaetognath | 450 | 273 | 160 |
| 3 | Doliolid | 485 | 45 | 68 |
| 4 | Larvacean | 529 | 43 | 61 |
| 5 | Radiolaria | 563 | 69 | 110 |
| 6 | Trichodesmium | 789 | 97 | 107 |
| **Total Samples** | | **3,119** | **138** | **142** |

The rest of this paper is organized as follows. In section II, we give a brief description of the SIPPER image dataset used in our experiments. We discuss the implementation details of convolutional neural networks in section III. We present the experimental setup and results in section IV. We conclude the paper and suggest future work in section V.

## II. PLANLKTON IMAGE DATASET

The plankton image dataset that we used for this research is obtained from the SIPPER system. The image resolution produced by SIPPER is 3-bit grayscale [9]. The dataset is provided by the University of South Florida (Tampa, FL, USA). All the samples were collected from Gulf of Mexico during the time period between 2010 and 2014. The entire image collection consists of more than 750 thousand images. They are classified into 81 types of plankton by the researchers of marine science at USF. We choose from this dataset a total of seven types of plankton for our experiments. The reason for our choice is to allow us to compare our classification results with those obtained in [5-8]. Those are the same class types used in the previous studies with exactly the same distribution per class. Table I gives details of the seven types and their distributions.

Analysis of the data revealed two major challenges. The first challenge was low quality of images due to their low resolution and high noise. The second challenge was due to the similarity between different classes and high variations within the same class. Image occlusion and deformation posed additional challenges within this dataset. Fig. 1 shows a set of four randomly selected images for each of the seven plankton classes.

To overcome aforementioned challenges we propose hybrid system that employs deep learning algorithms, and which are end to end feature learning and classification techniques. Since the problem we are dealing with is visual in nature we chose convolutional neural networks (CNN). Amongst the group of deep learning algorithms CNN is empirically proven to be the best solution for this type of classification problems.

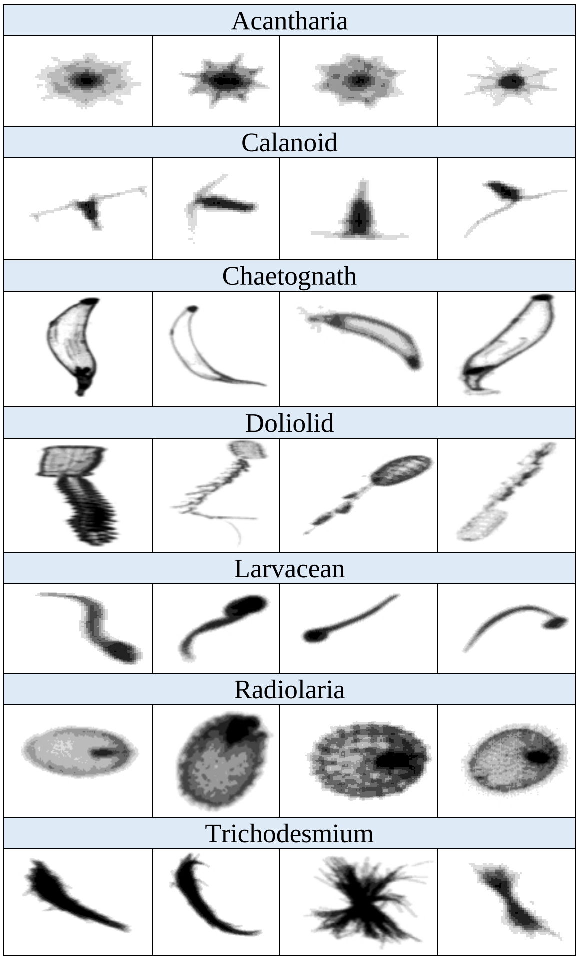


Fig. 1. Random samples from seven classes of the SIPPER dataset.

## III. CONVOLUTIONAL NEURAL NETWORK

The first step in visual image recognition requires construction of a classification model that can classify images to an acceptable level of accuracy. The model consists of invariant distinguishing features from an image to achieve higher levels of accuracy. The main challenge lies in being able to identify such features. One very popular method for identifying features is features engineering. This method studies the differences between the different classes and the similarity within the same class to design specific features. It requires intensive analysis to find good quality features and cannot be efficiently scaled up for each newly discovered class. An alternate method is to allow the model to dynamically learn those features and thereby provide a more reliable and scalable system. Several approaches have been invented to allow the model to learn features. One of them is to simulate animals' vision mechanism. This approach is known as convolutional neural networks (CNN) and is proven to be one of the best systems this far.

Convolutional neural networks (CNN) was introduced by Hubel et al. [10] during their work on cats' visual cortex. The first software that simulated this process could be traced back to Fukushima in his work titled Neocognitron [11]. In 1998, LeCun et al [12] applied the algorithm to recognition of

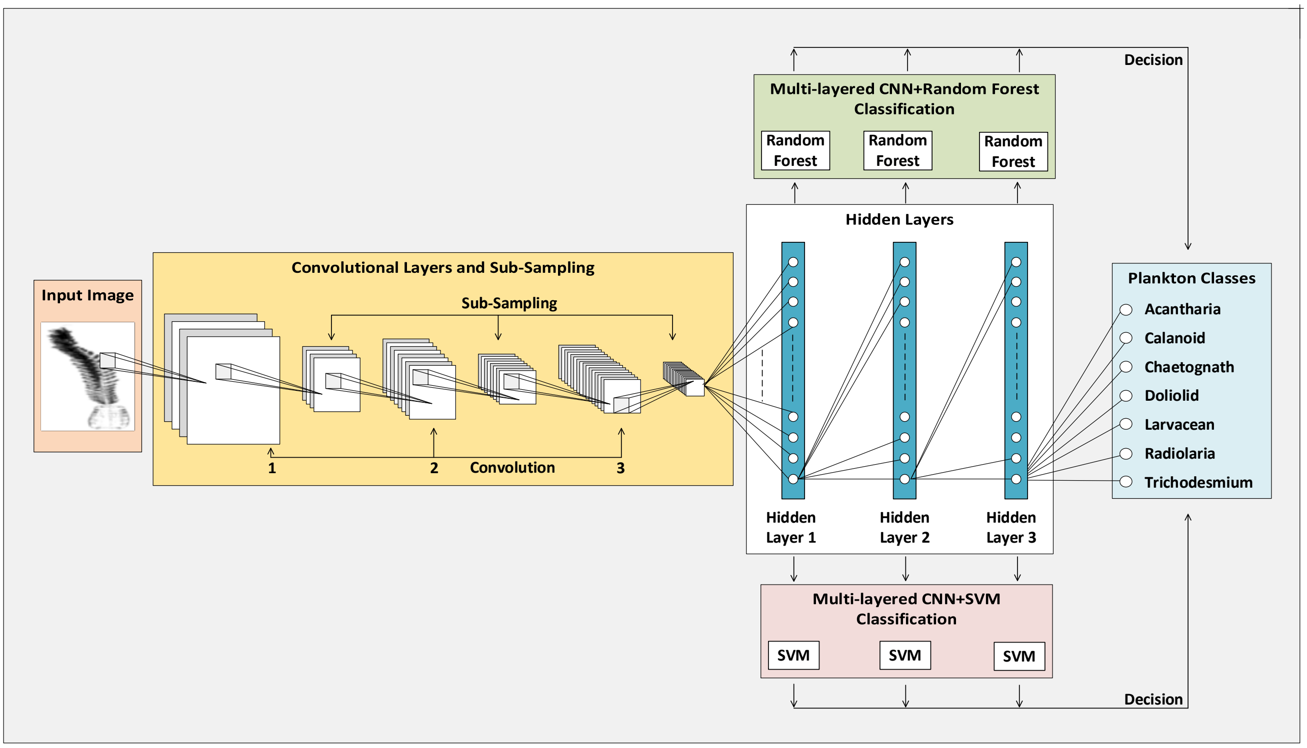


Fig. 2. Hybrid CNN architecture with SVM and Random Forest.

handwritten characters. In the past, for a number of years, the use of CNN had stayed limited due to its high computational cost. Recently, the availability of high performance computational hardware with GPGPU made CNN more feasible. The remarkable success of CNN in ImageNet competition [13] furthered its popularity.

The construction of a model based on convolutional neural networks consists of two main parts. The first part is the convolutional layers, which work as feature extractor. The features are automatically learnt by the system in the convolutional layers. Each convolutional layer consists of three main components, namely filter layer, non-linearity, and feature pooling [12-14]. The purpose of using several layers of CNN is to allow the system to learn feature hierarchies. First, the low level features such as pixel intensities are learnt at initial layers. Next, mid-level features such as edges are learnt in middle layers. And finally, the high level features for instance objects are learnt at final set of layers. The second part is the classifier that is typically a fully connected neural network. Figure 2 illustrates hybrid CNN architecture with support vector machine and random forest.

### A. *CNN Components*

As mentioned above, each convolutional neural network layer consists of convolution, nonlinearity, and pooling. Convolution is used to convolve the input features [15]. A group of filters processes local sections of the input [16]. Those small filters are replicated and applied to the entire input. Three parameters are used to describe the convolution [14] [17]. They are the number of filters, the kernel size, and the stride. Each convolutional layer has a specific number of filters and in general increasing this number means increasing the learning capacity of the system. The kernel size is fixed for each filter in a specific convolutional layer. Also, the kernel size has to be within the limits of the size of the input. Finally, stride specifies the number of pixels in x and y dimension, which is skipped by the filter while it is replicated.

The second part of a CNN is a fully connected neural network. Equation (1) and (2) represent the typical activation functions of neurons, namely the tanh() and sigmoid() functions [12][14]. These two functions are saturating nonlinearities and they are proven to be slower to converge in training with gradient descent [13]. The use of non-saturating functions for activation such as the one shown in (3) is much faster than tanh() and sigmoid(). This function is called Rectified Linear Unit (ReLU) [18]. ReLU is a simple nonlinear function that thresholds the input [19] in such a way that guarantees the output will always be positive [15]. We utilized ReLU as the only activation function across all layers.

$$\tanh(x) = (e^{2x} - 1) / (e^{2x} + 1) \tag{1}$$

$$\mathrm{sig}(x) = 1 / (1 + e^{-x}) \tag{2}$$

$$f(x) = \max(0, x) \text{ where } x = \textstyle\sum_i x_i w_i \tag{3}$$

The pooling layer is a dimensionality reduction approach. The input to this layer is the non-linearity layer's output. Its output is a reduced version of the input [20] [21]. The goal of pooling is to make the system immune to transformations in the input. It minimizes the neural network's sensitivity to the pixel locations [22]. The pooling layer gives a compact representation that is more robust to noise. Two common ways to build the pooling layer are average-pooling and max-pooling. Pooling is applied to the input either in overlapped or separate mode [20]. Size of the output depends on the pooling dimension and mode.

Overfitting is one of the challenges of training a neural network. Several techniques have been suggested to overcome overfitting. More recently, dropout is one of the successful techniques developed by Srivastava et al. [23]. This approach works by turning off randomly chosen neurons in each layer during the learning stage. The turned off neurons are not used in the forward and backward computation. The turning off is performed during the training of the neural network. During validation and testing, all neurons are used to compute the output. The use of dropout layer was shown to accomplish high gains in several benchmark tests [24]. Another widely used approach to minimize overfitting is data augmentation [13]. This technique expands the training data set by using different transformations with regards to preserving labels [25] [26]. Common transformations include rotating the images using different angles, image translation by few pixels, and cropping the image from random points.

### *B. Learning Algorithm*

We train our model using stochastic gradient descent with momentum. The learning rate is set to 0.1, the momentum is set to 0.9, and batch size is fixed at 128.

## IV. Experimental Setup and Results

The plankton image dataset we chose has seven classes with different distributions for each class. Acantharia has the minimum number of 131 images. Trichodesmium has the maximum number of 789 images. The types and distributions are shown in Table I. There are a total of 3,119 images in the dataset. We divided the images randomly over training, validation, and testing sets. The size of the testing set is 312 images, which is 10% of the total images. The training and validation are 2,524 and 283 images respectively. Each of the aforementioned set is chosen proportional to the distribution in each plankton class.

Convolutional neural networks require the input to be of a fixed dimension. Images in the plankton dataset are very different in terms of their dimension. As shown in Table I, the minimum height is 43 pixels while the maximum height is 319 pixels and the average is 138 pixels. Also, the minimum and maximum width is 61 and 346 pixels, respectively. The average width is 142 pixels. There are two ways to resize the data to a specific dimension so they could be fed to the CNN. One way is to resize with regards to the aspect ratio while the other option does not consider the aspect ratio. We decided to resize each image without regards to the aspect ratio. The new dimensions were set to 32 by 32 for height and width for the sake of computational efficiency.

Figure 2 shows general architecture of the hybrid system we used for our experiments. A resized image is given as an input to the convolutional layer. For the convolutional layers we used typical configuration, i.e., each convolutional layer is followed by a max pooling layer as a subsampling layer. To keep the computational complexity low, we considered a maximum of 3 convolutional layers. Those layers learn the invariant features of the input. During this stage, we limit the number of hyper parameters to the number of filters and reception field. Output from the max pooling layer for the third convolutional layer is flattened so that it could be used as input to the fully connected neural network. The neural network has three fully connected layers. We set the number of neurons in the three hidden layers to 512, 256, and 128, respectively. A dropout percentage of 50% is used after each fully connected layer to decrease the issue of overfitting. We limit the activation function in both convolutional and fully connected layers to ReLU, the primary reason for this choice is to take advantage of the high speed of convergence of this activation function [13]. The output of the third hidden layer is fed into the output layer. The size of the output layer is set to seven, which is the number of plankton classes. Softmax activation function is used for the output layer to obtain the probability values for each class. The loss function is multiclass cross entropy.

The hybrid system we propose combines CNN with other classification algorithms. The first phase is to train CNN with training data until it converges and gives the best results. After that, the features generated from the hidden layers are used as an input to train the other classification algorithms. Essentially, CNN is employed as a feature extractor for the classification algorithm. Since there are three hidden layers, we evaluate the classification performance from features taken from each layer, which are sent as an input to a separate classifier. We use two classifiers: SVM and Random Forest. We obtain results from CNN algorithm along with additional six results from using SVM with hidden layers 1-3, and Random Forest with hidden layers 1-3.

We conduct two sets of experiments. In the first set we do not use augmentation while in the second set we augment the data. The purpose is to measure the effects of data augmentation on CNN and on the features generated from the hidden layers.

### *A. Without Augmentation*

For this set of experiments we have two hyper

TABLE II. Classificaiton Accuracy without Augmentation Using 10-Fold Cross Validation

| **No. Filter & Reception Field** | **3-Layers CNN (%)** | **Random Forest (%)** | | | **SVM (%)** | | |
|---|---|---|---|---|---|---|---|
| | | **HL-1** | **HL-2** | **HL-3** | **HL-1** | **HL-2** | **HL-3** |
| 16, 3 | 91.52 | 92.85 | 92.92 | 92.98 | 92.56 | 92.50 | 92.56 |
| 16, 5 | 90.51 | 91.83 | 91.83 | 91.86 | 90.96 | 91.12 | 91.25 |
| 32, 3 | **93.28** | 94.74 | **95.32** | 94.90 | 94.87 | 95.03 | **95.22** |
| 32, 5 | 93.01 | 93.91 | 93.91 | 93.49 | 93.72 | 93.94 | 93.81 |

HL – hidden layer

TABLE III. CLASSIFICATION ACCURACY WITH AUGMENTATION USING 10-FOLD CROSS VALIDATION

| No. Filter & Reception Field | 3-Layers CNN | Random Forest (%) | | | SVM (%) | | |
|---|---|---|---|---|---|---|---|
| | | HL-1 | HL-2 | HL-3 | HL-1 | HL-2 | HL-3 |
| 16, 3 | 93.95 | 94.84 | 94.68 | 94.68 | 94.62 | 94.65 | 94.68 |
| 16, 5 | 92.07 | 93.04 | 92.88 | 92.66 | 92.82 | 92.92 | 93.04 |
| 32, 3 | **95.59** | 96.51 | **96.44** | 96.38 | **96.70** | 96.60 | 96.57 |
| 32, 5 | 93.98 | 94.74 | 94.71 | 94.84 | 94.90 | 94.94 | 94.78 |

HL – hidden layer

parameters to tune, namely the number of filters in the convolutional layers and the size of the reception field. We set the number of filters to either 16 or 32, while the reception field is either three or five. For the results to be statistically stable we perform 10-fold cross validation for each setup of hyper parameters. We average the resulting accuracies. Table II gives the average accuracies.

As shown in Table II, the accuracy improves with the use of higher number of filters. Also, accuracy is highly related to the size of the reception field. In both cases of 16 or 32 filters the accuracy comes out to be higher when using a smaller reception field of 3. The highest accuracy by using CNN is 93.28%, which is better than the results obtained in [5-7]. The second highest accuracy of 95.22% is obtained by using SVM with the features from hidden layer 3. This result is better than state of the art obtained in [8]. When CNN was combined with Random Forest we obtain the highest with the features from hidden layer 2. The above results show that the use of CNN combined with other classifiers gives better results than the results obtained by using CNN alone.

Fig. 3 gives filter visualization for the convolutional layers. All images in the figure are for one particular randomly selected image. It shows learnt features by the network. For the sake of visual comprehension we limit the figure to show results from 16 filters.

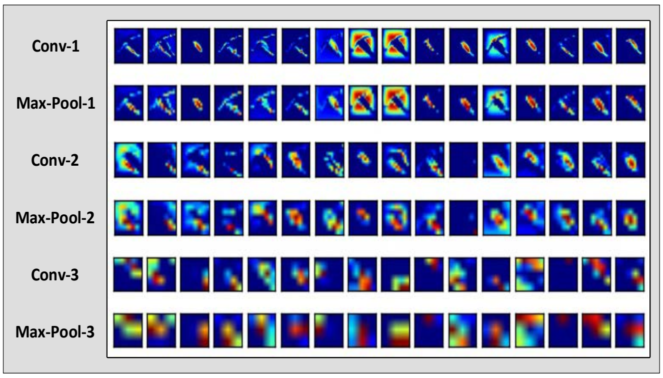


Fig. 3. Filter visualization for convolutional layers.

TABLE IV. COMPARISION OF CLASSIFICATION PERFORMANCE WITH OTHER METHODS ON THE SIPPER DATASET

| Method | Accuracy (%) |
|---|---|
| Normalized Multilevel Dominant Eigenvector Estimation [5] | 91.70 |
| Bagging Based [6] | 93.04 |
| Random Subspace [7] | 93.27 |
| Pairwise Nonparametric Discriminant Analysis [8] | 95.00 |
| †3-layers CNN | 93.28 |
| **†3-layers CNN with SVM** | **95.22** |
| **†3-layers CNN with Random Forest** | **95.32** |
| ***3-layers CNN** | **95.59** |
| ***3-layers CNN with Random Forest** | **96.44** |
| ***3-layers CNN with SVM** | **96.70** |

†without data augmentation
*with data augmentation

### B. *With Augmentation*

For this set of experiments, we apply data augmentation to the training set. We limit the augmentation transformation to rotation. We divide the angles between 0 and 360 by 22.5 and the angle is chosen randomly. We double the training set by adding the augmented version. Similar to the first set of experiments, we had two hyper parameters to tune, i.e., the number of filters and the size of the reception field. We validate results by performing 10-fold cross validation for each setup of hyper parameters. Table III gives the average accuracies.

We observe that the accuracy improves with the use of more filters and smaller reception field. The highest accuracy we obtain by using CNN is 95.59%, which is better than the best accuracy we get with the non-augmented set of experiments. The second highest accuracy of 96.44% is obtained by using Random Forest with the features from hidden layer 2. The highest accuracy of 96.70% comes from training SVM with the features from hidden layer 1.

### C. *Comparative Analysis of Results*

In this research we classified SIPPER plankton images using CNN. Also, we tested the performance of CNN by combining it with other classifiers, namely SVM and Random Forest. The results we got are comparable with previous studies that use hybrid CNN. Another problem we tried to address is which hidden layer provided the best results to be used with SVM or Random Forest. For the Random Forest, the results clearly show that the second hidden layer was the best feature layer. While for SVM the results need further study.

A comparison between our experimental results with other state of the art approaches using plankton dataset is presented in Table IV. These results point to the superiority of hybrid CNN over other methods.

## V. Conclusions and Future Work

Efficient analysis and classification of huge amounts of plankton data requires robust algorithms. In this paper, we conducted an in depth comparison of the performance results of CNN with support vector machine and random forest. We performed experiments over three layers, with and without data augmentation. Results of our experiments using the SIPPER dataset show improvement in classification accuracy in comparison to the previous approaches from other research groups. Furthermore, we found that the combination of CNN and SVM, with data augmentation gives best result amongst all approaches that we considered. One major advantage of the various approaches that we evaluated is scalability for classification of new classes without the need for features engineering.

In the future we plan to extend our work by including several other classifiers in combination with CNN. Another area of potential research is the multi-column CNN feature fusion methodology. Additionally, we plan to perform comparative assessment on multiple large scale color image datasets.

## Acknowledgment

We would like to thank Dr. Kendra L. Daly, Andrew Remsen, and Kurt Kramer (USF) for the validated SIPPER image dataset. The SIPPER imaging work was supported by a National Science Foundation grant OCE-0526545, a University of South Florida Sponsored Research Foundation grant, a Florida Institute of Oceanography/BP grant, and a Gulf of Mexico Research Initiative (GOMRI) grant to K.L. Daly.

## References

[1] S. Davis, S. M. Gallager, and A. R. Solow, "Microaggregations of oceanic plankton observed by towed video microscopy," Science, vol. 257, pp. 230–232, Jul. 1992.

[2] J. Watson, G. Craig, V. Chalvidan, J. P. Chambard, A. Diard, G. L. Foresti, B. Forre, S. Gentili, P. R. Hobson, R. S. Lampitt, P. Maine, J. T. Malmo, H. Nareid, G. Pieroni, S. Serpico, K. Tipping, and A. Trucco, "High resolution in situ holographic recording and analysis of marine organisms and particles (Holomar)," *In Proc. IEEE Int. Conf. OCEANS*, 1998, pp. 1599–1604.

[3] S. Samson, T. Hopkins, A. Remsen, L. Langebrake, T. Sutton, and J. Patten, "A system for high-resolution zooplankton imaging," IEEE J. Ocean. Eng., vol. 26, no. 4, pp. 671–676, Oct. 2001.

[4] X. Tang, W. K. Stewart, "Plankton image classification using novel parallel-training learning vector quantization network," *OCEANS'96. MTS/IEEE. Prospects for the 21st Century. Conference Proceedings*. Vol. 3, 1996.

[5] T. Luo, K. Kramer, S. Samson, A. Remsen, D. B. Goldgof, L. O. Hall, and T. Hopkins "Active learning to recognize multiple types of plankton." In *Proceedings of the 17th International Conference on Pattern Recognition*, 2004, vol. 3, pp. 478-481.

[6] X. Tang, F. Lin, S. Samson, and A. Remsen, "Binary plankton image classification," IEEE J. Ocean. Eng., vol. 31, no. 3, pp. 728–735, Jul. 2006.

[7] F. Zhao, F. Lin, and H. Seah, "Bagging based plankton image classification," in *Proc. IEEE Int. Conf. on Image Process.*, 2009, pp. 2517–2520.

[8] L. Zhifeng, *Z.* Feng, L. Jianzhuang, *Q.* Yu, "Pairwise Nonparametric Discriminant Analysis for Binary Plankton Image Recognition," IEEE J. Ocean. Eng., vol. 39, no. 4, pp. 695–701, 2014.

[9] K. Kramer. System for Identifying Plankton from the SIPPER Instrument Platform. Ph.D. dissertation, University of South Florida, 2010.

[10] D. Hubel, and T. Wiesel, (1968). Receptive fields and functional architecture of monkey striate cortex. Journal of Physiology (London), 195, 215–243.

[11] K. Fukushima, (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics, 36, 193–202.

[12] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278–2324.

[13] A. Krizhevsky, I. Sutskever and G. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, Proc. Neural Information and Processing Systems, 2012.

[14] Y. LeCun, K. Kavukcuoglu, and C. Farabet. Convolutional networks and applications in vision. In Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium, pp. 253-256.

[15] M. D. Zeiler, and R. Fergus. Visualizing and understanding convolutional networks. In Computer Vision–ECCV, 2014, pp. 818-833. Springer International Publishing.

[16] O. Abdel-Hamid, A. R. Mohamed, H. Jiang, and G. Penn. "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition". In IEEE Acoustics, Speech and Signal Processing (ICASSP), 2012, pp. 4277-4280.

[17] D. C. Ciresan, U. Meier, J. Masci, L. Maria Gambardella, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification". *In IJCAI Proceedings of International Joint Conference on Artificial Intelligence*, July 2011, vol. 22, No. 1, p. 1237.

[18] V. Nair, and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines." *In Proceedings of the 27th International Conference on Machine Learning*, 2010, pp. 807-814.

[19] L. Kang, P. Ye, Y. Li, and D. Doermann. "Convolutional neural networks for no-reference image quality assessment." *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014, pp. 1733-1740.

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

[21] Y. L. Boureau, J. Ponce, and Y. LeCun. "A theoretical analysis of feature pooling in visual recognition." *In Proceedings of the 27th International Conference on Machine Learning*, 2010, pp. 111-118.

[22] M. D. Zeiler, and R. Fergus. (2013). Stochastic pooling for regularization of deep convolutional neural networks. arXiv preprint arXiv:1301.3557.

[23] 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, pages 1929–1958, 2014.

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

[25] D. Ciresan, U. Meier, and J. Schmidhuber. "Multi-column deep neural networks for image classification." *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2012, pp. 3642-3649.

[26] P. Y. Simard, D. Steinkraus, and J. C. Platt. (August 2003). Best practices for convolutional neural networks applied to visual document analysis. In Proceedings of IEEE (p. 958).