# A Survey of Convolutional Neural Networks: Motivation, Modern Architectures, and Current Applications in the Earth and Ocean Sciences

Nima Mirzaee
Department of Electrical Engineering
Univeristy of California San Diego

Christopher Hirlinger
Department of Mechanical and Aerospace Engineering
Univeristy of California San Diego

**Convolutional neural networks have recently gained traction as a deep learning method for a variety of multidimensional, spatial processing problems. A myriad of architectures and applications exist and it can be daunting for the uninitiated to approach the subject. This survey paper seeks to provide a primer on Convolutional neural networks particularly within the fields of Earth and Ocean Sciences. We begin with a brief discussion motivating the development and structure of Convolutional Networks, followed by a presentation of several prominent modern architectures. We then present several canonical and novel applications within these fields to provide the reader with a better understanding of how these networks are being used to address complex processing problems.**

## I. Motivation and Basic Structure

We begin our discussion by considering why convolutional neural networks (CNNs) exist. One might ask: “why not plug image data into standard neural networks (NNs)”? NNs fail to address two key idiosyncrasies of image data. These shortcomings makes analysis of image data with traditional networks an ill-posed problem. CNNs were developed as adaptations to standard NNs specifically for image analysis.

The first characteristic of images that NNs fail to address is that the pixels of an image are not i.i.d. Image pixels have a correlation with neighboring and nearby pixels as a function of distance (really number of pixels away as distance is discrete here). It is a must for NNs that the data they are supplied with be i.i.d. CNNs address this issue by reading in blocks of pixels instead of one pixel at a time and assume that these blocks of pixels are i.i.d. This block of pixels that are read in is called a receptive field and neurons in the convolution layers are associated with these instead of individual pixels.

The second characteristic of images that needed to be addressed by NNs is that image data is naturally three-dimensional. Image data has a width and a height that correspond to spatial dimensions and a depth that corresponds to the spectral bands read in by each pixel (for standard images each pixel has red, green, and blue input channels).

Each neuron of a feature layer can then be seen as a discrete three dimensional filter over the pixels in a receptive field and the colors for each pixel. This filter is then applied to the entire image by running it over a patch of the image (of size equal to receptive field size) and then moving the filter a predetermined amount and applying it over a new patch until the entire image has been covered. One can see why these special networks have been dubbed “convolutional neural networks,” as each feature layer created is a convolution of a filter with the image (or previous convolution layer if at higher layers). Each neuron of a layer then creates a two-dimensional feature layer which is stacked up on top of feature layers formed by the other neurons in the layer to form a convolution layer. Neurons in subsequent layers then have filter dimensions of depth corresponding to the depth (number of feature layers) of the previous convolution layer.

The filters in each layer (i.e. the weights) are learned through backpropagation just like standard neural networks. In this fashion the networks learn to identify the most salient features of the image for classification.

CNNs are set up with a number of subsequent convolution layers that image data is processed through followed by a few fully connected layers akin to the layers of a standard NN in which neurons in a given layer are connected to every neuron of neighboring layers. One can think of the convolution layers as the layers that identify objects in an image, with the first layer identifying simple abstract features such as edges with different orientations or blotches of colors and subsequent layers identify more complex, higher order patterns that are composed of the features of previous layers. The fully connected layers then take the patterns deemed important by the convolution layers and classify them.

An immediate challenge to using CNNs is that their sheer size makes them very burdensome to train. They contain more layers than most typical NNs not to mention that each convolution layer is a three-dimensional structure. It should be noted here that a corollary of establishing receptive field sizes for each layer is that each neuron has only as many weights assigned to it as pixels in the receptive field times the depth of the previous layer. This cuts down on the number of weights that need to be adjusted during training significantly. There are a couple of other tricks that are typically implemented in CNNs in order to avoid prohibitively long training times. Another challenge is that the size of CNNs needed to pick out distinguishing features from images will easily lead to overfitting of the data without massive amounts of training images. Large amounts of training images are usually not

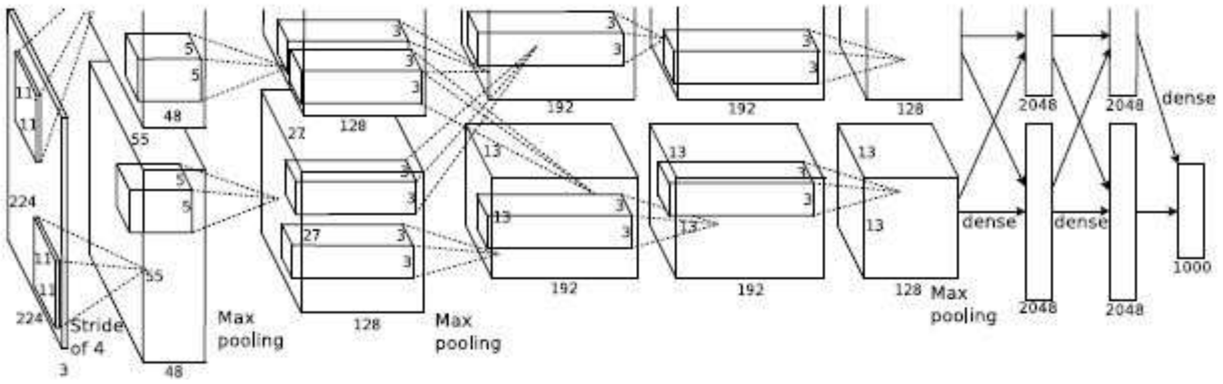


**Figure 1**: AlexNet network architecture. The architecture is shown as two parallel CNNs that share information between certain layers (but between all fully connected layers) because the network was trained in parallel on two different GPUs

available for most applications as they require experts to label data manually which is both time consuming and costly.

Instead of using neurons with saturating, nonlinear activation functions typically found in standard NNs, such as the tanh or sigmoid functions, CNNs commonly employ ReLUs (Rectified Linear Units). ReLUs refer to neurons with the activation function: f(x) = max(0,x). These neurons will be active if their convolution with a patch of the previous layer results in a value of x greater than zero or completely inactive otherwise. ReLUs exhibit behavior similar to real neurons in this sense. CNNs with ReLUs train several times faster than their equivalent networks with tanh activation functions [1]. This also prevents many of the neurons from being activated at all, which reduces the effective size of the network and reduces the potential for overfitting.

Another trick commonly employed is the use of pooling layers in between the convolution layers. These layers will take small patches of the previous convolution layer and apply a max function to them. The neuron with the largest activation will be kept will the rest will be discarded. This significantly cuts down on the size of the network in terms of connections to the subsequent layer, thus reducing training time and helping to prevent overfitting [2].

## II. A DISCUSSION OF MODERN ARCHITECTURES

The first modern CNN architecture discussed in this paper is AlexNet. AlexNet was first introduced to the world in 2012. Its architecture, as shown in fig 1, consists of five convolution layers, three pooling layers, and three fully connected layers. When AlexNet was first introduced to the CNN community is was one of the deeper networks out there. Additionally, it was one of the first networks to do away with pooling layers after each of its convolution layers and instead implemented only a few. These two modifications to previous CNN architectures radically increased the number of tunable parameters in the network and reintroduced the problems of long training times and overfitting that were previously ameliorated by ReLUs and pooling layers.

In the interest of reducing long training times a dual GPU architecture had to be designed to speed up training. In order to mitigate the problem of overfitting data the creators of AlexNet had to use a technique called dropout. Dropout is the major innovation of Alexnet and it was motivated by noticing that combining the predictions of multiple different network models resulted in lower test errors. The dropout method aims to replicate these results using a single network. It achieves this by randomly setting the activations of half the neurons in the first two fully connected layers to zero for each training image. Each image is then trained with different sets of neurons, and thus slightly different networks (using the qualifier “slightly” here because neuron weights are shared between these random subsets). During testing all the neurons in these two layers are present with neuron activations averaged. The creators of AlexNet speculate that this method prevents complex co-adaptations of neurons and forces the network to learn more robust features. By training each image on a slightly different network of significantly reduced size from the original architecture they managed to reduce the number of parameters in their system during training and solve the problem of overfitting and reduce the test error by averaging out all of the (quasi-independent) reduced size models.

Next we consider ResNet, a newer architecture that is considered state of the art. This architecture was introduced in 2015 and uses 34 convolution layers, two pooling layers, and only one fully connected layer [3]. ResNet introduced two new innovations to CNNs: the first being shortcut connections between convolution layers and the second being batch normalization.

ResNet was born from the question that had started to consume the CNN community a few years ago: Is creating better networks as easy as stacking more layers? When additional layers were piled on to existing networks the network’s classification error would eventually plateau with additional layers and then begin to degrade. The obvious explanation for this was that overfitting was causing the accuracy degradation , however to the surprise of everyone this was not found to be the case. This was tested by constructing a reasonably sized network with known accuracy and stacking on layers that were simply identity mappings. This deep network should in theory perform with as much accuracy as the shallow network it was built from, but it ended up performing worse. It is conjectured by the authors that very deep networks have exponentially low convergence rates which cause the training error (i.e. the solvers at hand cannot handle the optimization of these networks). The inventors of ResNet then

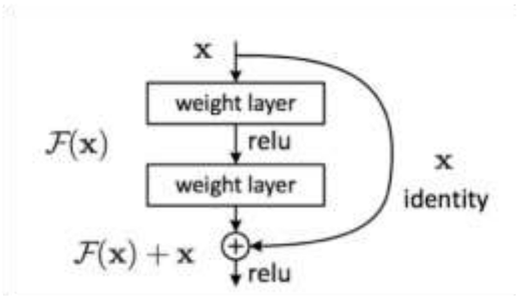

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

**Figure 2:** A shortcut mapping over a couple layers in ResNet

came up with idea of creating shortcut identity mappings of a given layer to a couple layers above it as seen in Fig 2. The idea behind this being that this would provide a reference for the solver when attempting to tune the parameters of the higher layer. This simple, but elegant solution to the new problem of running into the limits of current optimization methods allowed the creators of residual networks to build networks with over a hundred layers capable of superhuman object identification.

The creators of ResNet didn't exactly come up with batch normalization, nor were they really the first to use it. ResNet was the first highly successful CNN architecture to implement batch normalization however, and it's worth mentioning in this survey as it has since been widely adopted by CNNs in place of, or in conjunction with ReLUs. The motivation for batch normalization is that it has been known for a long time that training a network is much faster when the inputs to the network are decorrelated and whitened (i.e. linearly transformed to have zero means and unit variances) (Ioffe, 2015). This is a preprocessing trick that aims to decorrelate the data with PCA and subsequently rescale each dimension of the data matrix to a uniform scaling, giving the data a proper gaussian distribution. This works to speed up training times by preventing disparate scaling of inputs which during the forward pass can saturate activations (or in the case of ReLUs cause divergence). During backpropagation the network can then get stuck dealing with what is referred to as vanishing gradients. This is when it becomes exceedingly difficult to tune parameters due to the small gradients provided by the saturated activations. The creators of the batch normalization method posit that in the new age of deep networks this trick can be expanded upon. Each layer of the network receives as inputs the activations from the previous layer. One can view each layer as its own subnetwork and use the whitening trick on the activations of the previous layer, thus providing each layer with nicely behaved inputs. This method introduces two new tunable parameters per neuron that can linearly transform the whitened input back (i.e. provide an identity mapping from the previous activation) so that the network can tune activations to whatever extent the normalization is optimal in subsequent training steps. Covariances and mean values for activation normalization are obtained from small batches of data (hence the name) and are later used to form unbiased statistics for the training set. Finally, after the long setup, the punchline is that batch normalization is able to reduce training times by over an order of magnitude. Amazingly, it is at the same time able to mitigate overfitting in the network such that it can make the use of dropout unnecessary. The authors of the paper state that

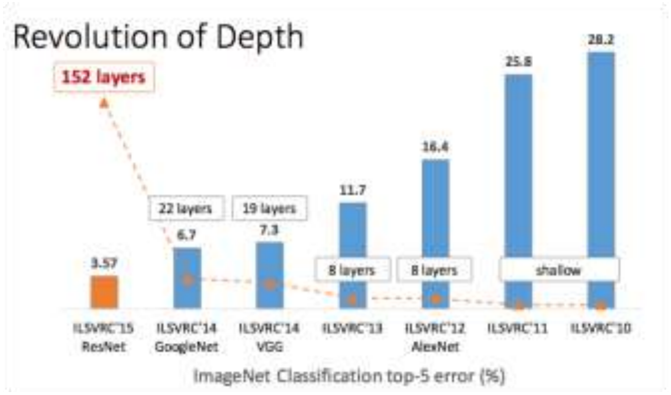


**Figure 3:** The winning architectures of the ImageNet competition by year, depicting the staggering rate improvement of CNNs through the years (Vieira, 2016)

the regularization properties emerge because each image is presented to the network together with other images in the batch so the network no longer produces deterministic values for a given training image. It is thus an incredibly versatile and useful tool that helped Resnet achieve the success it has.

## III. Applications

Having introduced the theory behind CNNs and common architectures the discussion now turns towards major applications of CNNs in atmospheric and ocean science. It should come as no surprise that many of these applications are image processing problems. The remainder of this section will address three major application fields for CNNs: Specifies Identification, Sonar Imaging, and Meteorological prediction. A brief discussion of trends and future research directions concludes the section.

### A. *Organism Identification*

Perhaps the most common and familiar application of CNNs is the species identification problem. In ecology it is often useful to identify the types of species in a given region and quantify their population. Within marine ecology plankton and corals often form the backbone of an ecosystem and the presence or absence of certain species can provide telling information about the health of the ecosystem. However, it is costly to have trained personnel manually sort through samples and image files to generate these metrics. Thus there is a large interest in developing classification algorithms that can perform these tasks and free up man power for other tasks.

Plantkon are particularly challenging to classify due to their small size and huge numbers. Further images must be captured using specialized equipment either in field or collected samples which add to the overhead cost of classification. Most if not all methods are standard CNN architectures that employ several tricks to improve accuracy. For example in [6] Ornstein discusses the use an eight layer Alex net architectures which are either fine-tuned once or twice using radically different plankton image sets. Single set fine tuning produced increased accuracy from approximately 78% to 86% while double fine tuning data resulted in modest improvements of less than 1%. Ornstein concludes that fine tuning using appropriately selected data sets could greatly improve existing classification algorithms.

In a parallel evaluation of Alex net, [7] Dai et al. develop a CNN architecture titled ZooplanktoNet for use in classifying Zooplankton Images. Similar to Ornstein, this architecture is developed from Alex net, and a variety of configurations were tested. An optimal configuration was found with an 11 layer system with layer dependent 13x13 and 7x7 convolution being found optimal. Data preprocessing was purposefully limited to detrending grayscale images by subtracting the mean intensity and rescaling images to a 256x256 pixel size. Dai et al. reports that this Alex net configuration was 92.8% accurate. The run time of this method was found to be comparable to that of smaller 8 and 9 layer CNNs trained on the same data, while significantly faster than deeper layered next works with almost the same accuracy.

In juxtaposition to the aforementioned plankton classifiers, while corals are not as small their classification from image data can be similarly difficult and taxing on resources. Most identification efforts are currently focused not on individual species of coral but instead on their function in the ecosystem. Of particular interest is a CNN architecture proposed by Elawady which is focused on developing coral classification algorithms for eventual on-line learning applications [8].

Rather than using a predeveloped architecture such as Alex net or ResNet the CNN is developed from scratch using Matlab's Deep Learning Tool box. Like most image classification problems the proposed CNN architecture in [8] begins with a preprocessing phase where images are de-noised and normalized. Feature extraction is then performed on the processed images. Lastly the CNN algorithm is trained using the extracted features. Unlike most of the discussed research in plankton identification the proposed algorithm for Coral classification relies purely on the feature mappings generated by the CNN.

The architecture of the CNN Elawady proposed architecture uses six inputs, with nine outputs for each of the classes. The authors did not identify an optimal number of convolutional layers and instead presented data for several different configurations typically ranging from 2 to 3 layers. Sigmoids were used as activation functions. The input of the CNN consisted of three color channel inputs and three descriptor channels developed from the preprocessing phase. These descriptor channels are as follows:

1) *Zero Component Analysis (ZCA) Whitening:* ZCA whitening removes correlation between adjacent pixels. The returned image signal is near white.
2) *Phase Congruency:* edge deteching techique based on the fourier components of an image singal. As a result phase congruency is robust against changes in illumination and image contrast.
3) *Weber Local Descriptor (WLD):* Specialized edge detection based on pixel to pixel changes in intensity and image gradient orientation. Effective, for high-tecture images such as those used in coral identification.

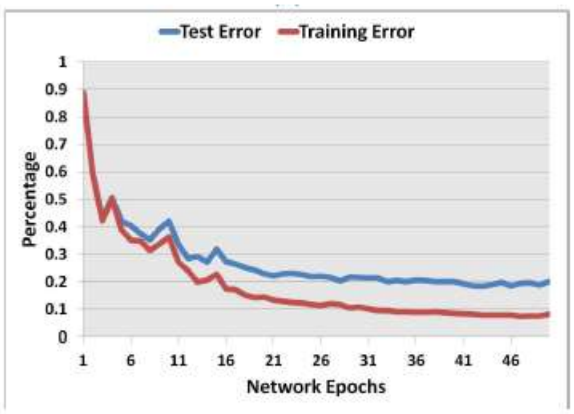


**Figure 4**: Performance of Elawady's algorithm on UCSD Moorea Labeled Corals data sets

Using these inputs, networks were trained over 50 epochs. The final results of classifier tests are shown in fig 4. Error rates by the 50th epoch were below 20 and 10 percent respectively for the test and training data using the optimized architecture.

## B. *Sonar Imaging*

Another promising application of CNN architectures is sonar image processing. The use of Neural Networks in sonar processing is not a novel concept, with research spanning back at least 30 years into the 1980s. However, there has been reticence, particularly within military research communities, to apply neural network architectures to maritime surveillance and target detection problems. This is understandable as accurate target identification is critical to mission success, and error rates of 5-10% seen in the plankton classifiers become infeasible when human lives are involved. Further because of the uncertain nature of deep learned features and their contribution to the classifier dynamics, CNN's are sometime seen as black boxes. This has led to a stigma which has resulted in sparse application to a problem that it is well suited to address [9]. For this reason the driving platform for CNN use in sonar image processing is the growing interest in Autonomous Underwater Vehicles (AUVs). These applications are typically framed as navigation or coordination problems where one or more vehicles.

In [10] Valdenegro-Toro addressed the use CNNs in the autonomous navigation problem. Forward looking sonar (FLS) data is classified by CNN in a two stage approach which first detects if/where an object is in the FLS data, and then classifies the object into one of several classes. The CNN was developed independent of an existing architecture due to its dual classification and is summarized in fig 5. After initial training, fine tuning was performed on convolutional layers initialized with the pre-trained weights. In detecting marine debris, the classifier performed with 93% detection recall and 75% classification accuracy. While detection recall rates are high the classification accuracy is lower than expected. However, the network was shown to accurately detect untrained/unlabeled objects and further research is being done to improve the results of the classification algorithm.

Kim et al. address the autonomous coordination problem in [11]. Of particular interest to this paper is tracking a

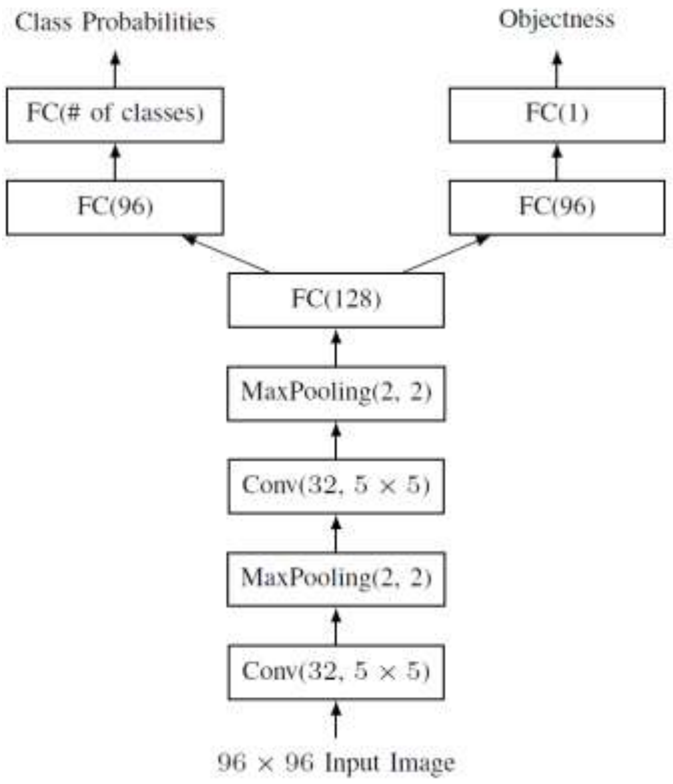


**Figure 5**: graphic of Valdenegro-Toro's class and object search classifier. Note that until the second FC layer processing is the same across both searches.

manipulator or small AUV/ROV and differentiating if from obstacles captured in the sonar data. To this end the You Only Look Once (YOLO) architecture was used. YOLO is known for being able to analyze a large number of frames per-second (FPS) and is thus perfect for a coordination problem involving streamed sonar data. Though quantitative metrics were sparse, it was shown that the trained YOLO architecture could classify position with sufficient resolution and speed for a standard control problem.

Due to this progress in sonar processing CNNs are gradually being introduced for maritime military operations. One example application is in mine countermeasure operations where [9,12]. In [12] Williams develops a 10 layer CNN with carefully tune layers and window sizes. The CNN takes in synthetic aperture sonar (SAS) data and returns a binary classification of whether or not the image is of a valid "target" Three experiments are then run on three different target classes, each simulating a variety of typical mine shapes, and seeking differentiate them from rocks and other marine debris that may be captured in the sonar imaging. In the most general test where a variety standard mine shapes (cones, cylinders, etc) from all other non-targets, the CNN methods developed in [12] significantly outperformed existing neural network methods in both probability of detection and false positive rates. Further the CNN proved extremely effective in differentiating a specific type of mine known as the manta (fig 6) from other truncated conic objects with some data sets approaching 100% detection rate with 0 false alarms.

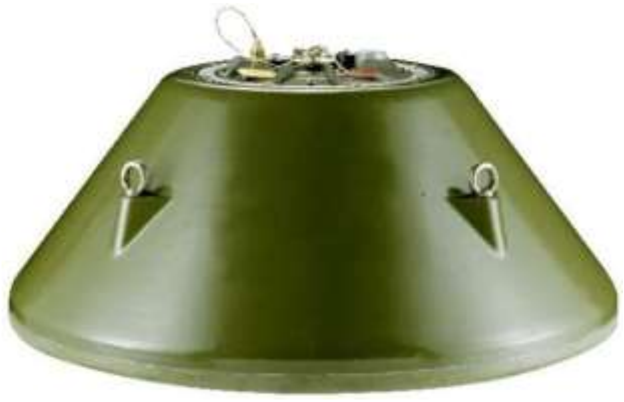


**Figure6**: Image of standard manta mine. Note that the manta is a truncated conic. The fact that William's Machine learning algorithm can accurately classify mantas from other truncated conics purely from SAS imaging demonstrates the efficacy of CNN's in SAS processing.

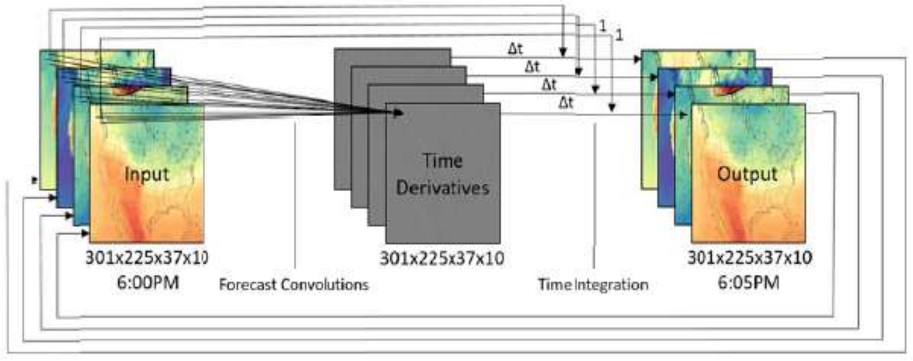
### C. *Meteorological Prediction and Modeling*

Lastly CNNs have seen wide application in atmospheric and oceanographic prediction. The canonical example is extreme weather identification, where a human expert is relied upon to identify and classify extreme weather events from satellite imagery and forecasts.

Liu et al. present a promising method for extreme weather classification based on CNNs [13]. Leveraging the Alex Net architecture a 4 layer system is developed using dimensionally large kernels in the convolution layers to reflect the relative size and simplicity of weather patterns in imaging data. A three channel input of RGB intensities is used to produce a binary classification of the image. Three classification problems were tested, tropical cyclone identification, weather front identification, and atmospheric river identification. Image patching and layer parameters were modified for each classification problem, but were not optimized or otherwise tuned.

Data used for the training and testing was from a combination of simulation and reanalysis products. Ground truth was established by having human experts label the events individually. The reported results using binary classifiers promising, showing that accuracy on test data ranged from 89% to 99%. The worst performance corresponded to classification of weather fronts and the best performance on classification of Tropical cyclones. While these results are promising, Liu et al. do not address localization of atmospheric phenomena, and rely on binary classification rather than a signal classifying network, leading to immediate directions for future work.

Another interesting application of CNNs is the sparse data set problem. In in such problems one uses a machine learning algorithm to predict missing data points of a system based on a small number of measurements. Ducournau et al. introduce the use of CNNs for analysis and prediction of sparse 2D data sets, in particular sea surface temperature [14]. The problem was chosen for its practically and ease of available data. Satellite infrared measurements such as those available are readily



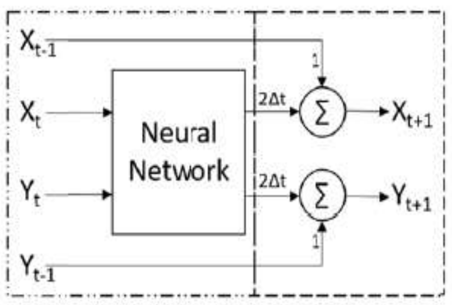


Figure 7: DITS network architecture. Left: standard recursive prediction method between forecast convolution and time integration layers. Right: Motivating block diagram representing use of CNN to predict PDE time derivatives.

available, but often have low spatial resolution (on the order of 5 km). Data from OSTIA was used as both low resolution and high resolution data were readily available for training and testing. Higher resolution data sets were generated through a combination of temperature measurements from buoys and other remote measurement platforms. To this end Ducournau et al. apply a caffe derivative SRCNN to a sparse satellite data set from OSTIA and attempt to reconstruct a high resolution prediction of sea surface temperature. Traditional methods use a variety of interpolation methods. In this case the results of SRCNN are compared against traditional interpolations such as bi-cubic and EOF-linear interpolation. To quantify results the mean PSNR was calculated for each approximation, and of the methods tested SRCNN produce the best PSNR across the tests. Further a comparison of gradients showed that PSNR better captured the dynamics of the ground truth data where other interpolations tended to smooth dynamics.

Finally, CNN's have been shown to be capable of short term weather forecasting. However, such predictors typically have little basis in physical models and are primarily based on feature extraction leading to similar stigma to that found in CNN applications to military sonar. In [_] Firth proposes Differentiation-Integration Time Step Network (DITS) a novel CNN application that incorporates a CNN into the iterative calculations of any time marched PDE. Firth's architecture uses a CNN to estimate the time derivatives of the variables of a PDE rather than the overall system behavior. This is done by feeding in a two dimensional data for the system variables such as temperature, humidity, and pressure. DITS then estimates time derivatives from the data which is then integrated using any of the common integration methods (Leap-frog, RK methods, etc). This results in a modified CNN architecture of alternating convolution and integration layers which replace standard pooling layers. This is summarized in fig 7.

In this architecture the feature map effectively becomes a map of time derivatives, and use of the CNN immediately provides several benefits. Error calculated at each step can be handled via backpropagation into the neural network. Combined with the recursive formulation this allows models augmented with DITS to perform multistep predictions rather than short term forecasts. Further, the use of the CNN when combined with appropriate time marching schemes lends itself to parallelization through spatially decomposing the 2D data set and known acceleration techniques for time integration methods. Lastly DITS assumes that the same physics apply everywhere within a 2D data set and that there is no spatial variations within it. This drastically reduces the number of parameters which DITS must learn and thus reduces the complexity relative to other forecast algorithms. This in turn allows DITS to be run on smaller mesh sizes, refining the result of simulations.

Firth applies his DITS to two case studies. The first compares DITS with NCOM which is capable of predicting temperature, salinity, and other metrics in addition to water current forecasting. In the second case study Firth applies DITS to RAP an atmospheric forecasting model. Both NCOM and RAP represent state of the art algorithms in oceanographic and atmospheric forecasting. In both cases training data for DITS was generated by capturing the time derivatives generated from NCOM and RAP. DITS was trained on this data and then run in parallel with NCOM and DITS on a new set of input data. It was found that the trained DITS forecasters were comparably accurate to their counterparts. However, DITS methods were significantly faster, and found to be 3.6 times faster than NCOM and 24 times faster than RAP! It is worth noting that the results presented by Firth used leap-frog integration and could perhaps be further improved by using a method from the RK family.

## D. Summary

Examining the presented applications it is clear the CNNs have found niches in two types of application. First image processing, and second spatially correlated data analysis and filtering.

Image processing problems were primarily addressed in the species classification and sonar algorithms. Comparison across these algorithms is a nebulous task as they are developed and targeted at two wholly different data sets. However, one can draw some interesting conclusions from their methodologies that may be useful for formulating future experiments. First, and foremost the optimized architecture of ZooplanktoNet should be subjected to the same fine tuning used by Orenstein to see if similar improvements in classification can be achieved with an architecture that is approaching 100% accuracy. Further, it is interesting to note that the coral classifier developed by Elwady with its modest depth and more generalized classification categories performed with similar levels of accuracy to Ornstein's double fine-tuned classifiers. Though comparison across problems is moot, the coral classifier's use of image preprocessing metrics may be worth exploring in the plankton and sonar data sets. More generally the performance of sonar applications lags behind CNNs used in species identification. Whether this is due to stigma or lack of interest as discussed by Williams in [15], or the nature of information available through sonar imaging and waveforms is unclear. The latter point might be addressed by adopting Elwady's incorporation of signal processing metrics as network inputs, or incorporating relevant measures of ambient conditions.

Lastly the applications in Meteorological prediction show that CNN's are incredibly powerful tools for spatial processing. The sparse data problem presented by Ducournau is not new, but the use of CNN's to address spatially dependent data opens new avenues for analysis for spatially driven sparse processing problems, which could be generalized to other disciplines such in signal processing and distributed sensing. Further, the DITS network presented by Firth has ramifications that extend to virtually all fields of science and bears an immediate need for investigation. From a numerical standpoint the architecture might be improved further by using different integration methods, in particular the adaptive time step RK methods. Considering the performance of CNNs in numerical prediction problems it is clear that CNNs have vast potential outside of just image processing. If nothing else the small body of work presented here should impress upon the reader that CNNs provide flexible solutions to many spatial driven problems and can be restructured to address a surprising number of problems.

## IV. Conclusion

This paper seeks to have provided a brief yet comprehensive introduction to convolutional neural networks for the uninformed reader. Using motivationg examples in the fields of earth and ocean science it is clear that scientists and engineers have only begun to scratch the surface of what is possible with these networks. As our computers become faster and more scientists are clued in to their capabilities we will start to see them used to their full potential.

## References

[1] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 25 (pp. 1097–1105). Curran Associates, Inc. Retrieved from http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[2] Karpathy, A. (2015). Convolutional Neural Networks (CNNs / ConvNets). Retrieved from http://cs231n.github.io/convolutional-networks/

[3] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. CoRR, abs/1512.03385. Retrieved from http://arxiv.org/abs/1512.03385

[4] Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. CoRR, abs/1502.03167. Retrieved from http://arxiv.org/abs/1502.03167

[5] Vieira, A. (Jun 23, 2016). The Revolution of Depth. https://medium.com/@Lidinwise/the-revolution-of-depth-facf174924f5.

[6] Orenstein, E.C. and Beijbom, O., 2017, March. Transfer Learning and Deep Feature Extraction for Planktonic Image Data Sets. In *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on* (pp. 1082-1088). IEEE.

[7]J. Dai, et al., "ZooplanktoNet: Deep Convolutional Network for Zooplankton Classification", in *OCEANS 2016 - Shanghai*, Shanghai, China, 2017.

[8]M. Elawady, "Sparse Coral Classification Using Deep Convolutional Neural Networks". arXiv preprint arXiv:1511.09067, 2015.

[9] D. Williams, "Demystifying Deep Convolutional Neural Networks for Sonar Image Classification", in *Underwater Acoustics Conference*, Skiathos, Greece, 2017.

[10] Valdenegro-Toro, M., 2016, November. End-to-end object detection and recognition in forward-looking sonar images with convolutional neural networks. In *Autonomous Underwater Vehicles (AUV), 2016 IEEE/OES* (pp. 144-150). IEEE.

[11] Kim, J. and Yu, S.C., 2016, November. Convolutional neural network-based real-time ROV detection using forward-looking sonar image. In *Autonomous Underwater Vehicles (AUV), 2016 IEEE/OES* (pp. 396-400). IEEE

[12] D. Williams, "Underwater Target Classification in Synthetic Aperture Sonar Imagery Using Deep Convolutional Neural Networks," *Proceedings of the 23rd International Conference on Pattern Recognition*, Cancún, Mexico, December 2016.

[13]A. Ducournau and R. Fablet, "Deep learning for ocean remote sensing: an application of convolutional neural networks for super-resolution on satellite-derived SST data", *9th IAPR Workshop on Pattern Recognition in Remote Sensing*, Cancun, Mexico, 2016.

[14]Y. Liu, et al., "Application of deep convolutional neural networks for detecting extreme weather in climate datasets,"

22nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining, San Fransisco, USA, 2016.

[15] R. Firth, "A NOVEL RECURRENT CONVOLUTIONAL NEURAL NETWORK FOR OCEAN AND WEATHER FORECASTING". Ph.D dissertation, CSEE Department, LSU, LA, 2016