Exploring Transferability in Deep Neural Networks with Functional Data Analysis and Spatial Statistics

Richard McAllister
Gianforte School of Computing
Montana State University
Bozeman, MT, USA
richard.mcallister@msu.montana.edu

John Sheppard
Gianforte School of Computing
Montana State University
Bozeman, MT, USA
john.sheppard@montana.edu

***Abstract*—Recent advances in machine learning have brought with them considerable attention in applying such methods to complex prediction problems. However, in extremely large dataspaces, a single neural network covering that space may not be effective, and generating large numbers of deep neural networks is not feasible. In this paper, we analyze deep networks trained from stacked autoencoders in a spatio-temporal application area to determine the extent to which knowledge can be transferred to similar regions. Our analysis applies methods from functional data analysis and spatial statistics to identify such correlation. We apply this work in the context of numerical weather prediction in analyzing large-scale data from Hurricane Sandy. Results of our analysis indicate high likelihood that spatial correlation can be exploited if it can be identified prior to training.**

I. Introduction

It has long been known that the black-box nature of neural networks introduces challenges to wide-spread adoption of this technology, especially in safety-critical domains. The DARPA Broad Agency Announcement (BAA) for Explainable Artificial Intelligence [1] solicited research proposals for techniques for creating artificial intelligence models such as artificial neural networks (ANN) that, upon training, would enable users to understand why such models make the decisions that they make. In particular, a major factor that hinders the adoption of ANNs in many domains is that it is very difficult to understand why they produce the answers that they produce [2]. What has been learned in a trained, functioning, reliable ANN remains opaque; therefore, the model is limited in ways it can enhance the understanding of the governing processes of the system under examination. Thus one of the key focus areas in modern neural network research is in developing approaches to improve insight into what has been learned by these models, essentially opening the black box so that adopters can see inside.

The promise offered by the effectiveness of modern deep learning methods also suggests potential wide applicability in solving many of these critical problems. Unfortunately, the computational complexity of training such models, combined with the large data requirements, further limit adoption of this technology. This motivates research in transfer learning whereby trained models can be re-used as starting points in other problem areas, thus significantly reducing the computational burden in their training. Intuitively, efforts to apply transfer learning assume that it is possible for a model to have learned something fundamental to a more abstract universe that encompasses both the domain within which the model was trained and the domain within which it will be applied.

These two problem areas motivate the current work. For our approach, we focus on a single type of deep network and apply techniques from functional data analysis (FDA) and spatial statistics to develop insight into what the network has learned. We then apply that insight to select portions of the model to be transferred and test the effectiveness of the transferred knowledge in a new setting. More specifically, we defined a highly controlled environment in which we started by creating a single stacked autoencoder and initializing the weights randomly. Then for each area of interest (AOI) in the dataspace we cloned this single network and pre-trained the clones on the data from their respective AOIs in exactly the same fashion, effectively removing any stochasticity in the training process. This allowed us to remove uncontrollable sources of variation and uncertainty across the entire dataspace and concentrate on how different areas of this dataspace affect training. We utilized this structure as the foundation for our experiments.

For this analysis, we chose a problem from meteorology as our test case. Weather modeling and prediction have been in the domain of deterministic methods for many years [3]. We assert that an opportunity exists for deep learning to supplement the state of the art in weather modeling and prediction, particularly by informing traditional computational models. To do this in a way that instills confidence in users of traditional models, we need to gain an understanding of what ANNs learn when they are trained on this data.

The main contributions coming from this work are as follows. First, we provide a highly structured, highly controlled approach to evaluate learnability and transferability in deep ANNs. To support this, we draw on the disciplines of functional data analysis and spatial statistics in a novel way. We then develop an approach to apply the results of spatial analysis to determine what components of a deep network are transferable. Finally we test the transferability of deep networks that have been trained based on this approach.

This paper is organized as follows. Section II discusses relevant literature to provide both background information and

results of similar research. In Section III, we describe the data collection process for our domain of study. We then discuss how we apply techniques from FDA to prepare the data for analysis in Section IV. In Section V, we provide a detailed explanation of our experimental and analysis approach. The results of our analysis are given in Section VI, and we provide conclusions and areas of future work in Section VII.

II. Related Work

The idea behind stacked autoencoders is to use layers of autoencoders to represent lower-dimensional encodings of a dataset under study [4]. The layers are stacked together in a fashion that facilitates constructing an abstraction hierarchy of features by having learned features derived directly on the response of the lower-level feature detectors. These feature detectors are developed through an iterative process of unsupervised pre-training, where resulting autoencoders migrate towards important basins of attraction expressed in data [5]. In this work, we control the training process to gain a better understanding of these basins of attraction in the context of a problem exhibiting high spatio-temporal correlation. This permits us to apply tools to support analysis of the learned features as a function of both space and time.

Transfer learning in ANNs is an area of very active research [6], [7]. It exploits knowledge gained from auxilliary domains in order to facilitate predictive modeling in the new domains [8]. There have been studies applying transfer learning in meteorology, which is the domain of interest here. In one example, Hu, Zhang, and Zhou applied transfer learning with stacked autoencoders to improve wind speed prediction in areas lacking sufficient data to appropriately train models from scratch [9].

While praising the success that many AI models have had in recent years, Samek, Wiegand, and Muller [2] state that "it is not clear what information in the input data makes them arrive at their decisions." In this paper we take steps to determine this information, although we do not make the claim that this information will be human-understandable.

While there does appear to be considerable literature in recent years on transfer learning and explainable AI individually, there does not appear to be any significant work combining the two. The novelty of the work performed in this paper is using analysis for explanability approaches to facilitate the transfer process.

The process of analyzing data that is generated as a result of some underlying process is referred to as "functional data analysis," where the data can be modeled and represented as a function, often in time or space. Silverman and Ramsay provide one of the first texts on the subject of FDA where they describe several analysis methods directly applicable to the type of data analyzed here (namely, weather data) [10]. In recognizing that weather occurs as a function of both space and time, we use methods from FDA to assemble features expressed in meteorological data as a hurricane moves over an area of the Earth. FDA is not without precedent in Earth science. King [11] explored functional analytic methods in analyzing climate change. In that work the author fit spline functions to temperature time series data in order to track temperature changes in US cities over the last few decades. She did not, however, apply FDA in a machine learning context.

Table I: Features for the Hurricane Sandy Dataset

Reading Source	**Reading Name**
Radiometry Measurement	Temperature
	Pressure
	Cloud Density
	Rain Density
	Ice Density
	Snow Density
	Graupel Density
Wind Speed Indicator	Wind u (East/West)
	Wind v (North/South)
	Wind w (Up/Down)

III. Data

The type of data we consider here is highly multidimensional, spatial, temporal, and functional. It is *multidimensional* in that for each AOI, we have temperature, pressure, precipitation, humidity, and wind data. It is *spatial* because we are examining these same dimensions across a three-dimensional physical space, and these spatial relationships are a significant factor in the behavior of the system. It is *temporal* because data at each location are represented as a time series as it tracks a storm over a 24-hour period. It is *functional* because changes in one area propagate through the space according to atmospheric forces and dynamics as a function of (among other things) space and time.

The data that we used for this investigation were generated by Zhang and Gasiewski in [12]. The data comes from a Weather Research and Forecasting (WRF) model of one day in the life of Hurricane Sandy from 2012 with data collected every 15 minutes. It is an aggregation of two datasets: one consisting of radiometric readings from space and one consisting of spatially and temporally located wind vectors. Since radiometers cannot measure wind speed, we considered it useful to use deep learning to predict wind vector components from such inputs.

Table I shows the data that were available for each point in our study. The radiometric and wind datasets are from different sources, but all of their respective measurements have been aligned with one another with respect to space and time.

IV. The Role of Functional Data Analysis

In this paper we examine weather data as being functional in nature. We assume that the behavior of the data from each of the points of interest is influence by common factors that influence all of these points together. Because of this, we hypothesize that the encodings that result from training networks on the data from each point of interest contain transferrable knowledge. We want the use the understanding gained through this examination to broaden the predictive capability of similarly trained neural networks, and we further hypothesize that spatio-temporally correlated feature detectors

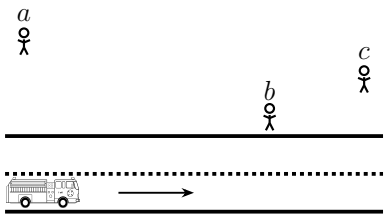


Figure 1: Pedestrians Along Road with Passing Firetruck

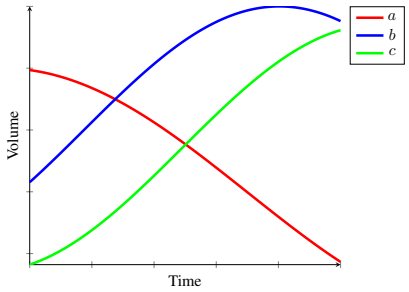


Figure 2: Volume Levels by Time for Each Pedestrian

in a trained network can be extracted and used to train networks for other parts of the dataspace more efficiently and more accurately.

To illustrate the role that FDA plays in our analysis, we use the following toy example. Suppose three people are standing along the side of a road as a firetruck passes by with its siren blaring. When the three pedestrians are in the same relative position to the street as the firetruck passes, each of their experiences will be exactly the same in terms of the volume of the siren they perceive. However, when they are at different distances from the edge of the street and at inconsistent intervals along the length of the street, the power of the functional treatment of this data is much more readily apparent. Figure 1 shows three pedestrians positioned in this way, and Figure 2 shows a notional plot of the corresponding volume perceived by each pedestrian. Notice how added distance causes the overall volume to be lower and the volume change to be flatter, in contrast with the experience of the pedestrian closest to the street. Also, the time of experiencing the change in volume of the siren varies based on the lateral position of the pedestrian.

In analyzing this situation, functional data registration would cause the peaks of these curves to be aligned (shift registration) and the amplitudes to be modified (amplitude registration) as much as possible to bring the overall shapes of the phenomena into alignment, while maintaining the individual differences of the functions [10]. Our example dataset characterizes the behavior of a hurricane, which like the firetruck in the above example, is a spatio-temporal phenomenon that is moving through our area of interest. As the phenomenon makes its way through every location on the map its "locations" in the phenomenon affect "locations" in the space in a related way, like the siren on the firetruck.

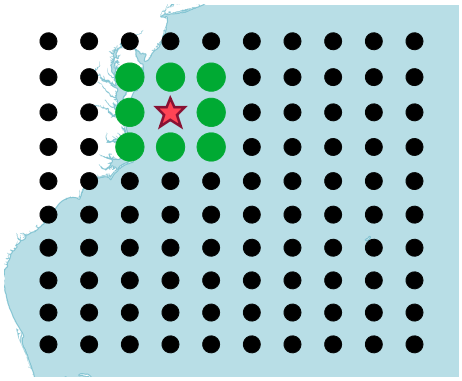


Figure 3: Analysis Locations for Training the Neural Networks

V. Approach

A. Overview

To extract information that we can use to generalize across the dataspace, we perform a two-stage training of a set of stacked autoencoders. The layers of these stacked autoencoders are trained using unsupervised pre-training. Except for the primary random initialization of the prototype network, we remove all sources of randomness in the pre-training procedure to facilitate a controlled analysis of the learned features. The initial weights of each of the autoencoders are cloned from a single random initialization and replicated across the spatial area in our dataspace so that they are exactly the same. The data used to do the pre-training are fed into each autoencoder in the same order, so there is time-correspondence across the dataspace. Maintaining this numerical consistency allowed us to trace the effects of the data using a consistent model, ensuring our focus on the dataspace rather than the model itself. The resulting weights are then analyzed with respect to how they vary across the dataspace. We use the results of applying semi-variograms from spatial statistics to determine information that may be shared across networks that correspond to each area of interest.

B. Data Preparation

1) Area of Interest Instance Data: Figure 3 shows the distribution of areas of interest where our data was collected. The left side of the figure shows the Eastern seaboard of the United States from Long Island on the upper left to Florida and Grand Bahama in the lower left. This square region is the section over which all of the data was collected, and the dots show the locations of each geographical area we analyzed. Each dot represents the center of a grid cell in a geodesic Discrete Global Gridding System (DGGS) [13] superimposed over the entire planet.

Figure 4 depicts one area of interest, corresponding to one of the points from Figure 3. Each numbered cell is a 15km resolution DGGS cell. We train networks to predict the wind vector conditions in cell 0 at each location for each succeeding time step in the dataset. We assume, as was assumed in [14], [15], that the wind vector values in cell 0 for the current time

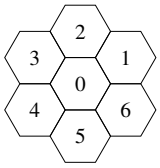


Figure 4: One DGGS Area of Interest

slice (excluding those forces not represented in the data) can be determined by the radiometric readings of that same cell for the current time slice and the radiometric and wind readings of all cells in the previous time slice.

2) Time Shift Instance Data: The task before us is to take data from radiometric readings at a particular point in time t, which we denote $r_t^0, \ldots, r_t^n$. We use this data to predict the wind vectors at time $t+1$, which we denote u_{t+1} (zonal, or east-west), v_{t+1} (meridional, or north-south), and w_{t+1} (vertical, or up-down) respectively. Thus we set this up as a one-step time series prediction problem.

3) Scale Data: Each input and output data variable varied in magnitude greatly. Each of the data variables was also captured using differing units, for example: kilometers per hour for wind and degrees Celsius for temperature. To minimize the impact such variability, we scaled the values for each of the variables to a range between 0 and 1. Each of the data variables was scaled individually so that the functional character of the data represented was preserved.

4) Separation of Input and Output Data: The radiometry measurements were the main features used for prediction. In the future we would like to use the wind vector predictions from the prior time step as inputs for the next time step, creating a more comprehensive system with enhanced predictive capabilities. But for now we wanted to provide the networks with as little information as possible about the state of the wind vector, save for the ground truth outputs in training.

5) Random Data Padding: To overcome an underflow issue with the functional data registration procedure, we padded each input dimension with a random value between 0.0001 and 0.001. This small value was adequate to prevent underflow while not being large enough to affect the overall patterns within the data. This was not necessary for the output data dimensions.

6) Functional Data Registration: To bring the shapes of each of the input dimensions into greater relief we performed functional data registration over the dataset. This included shifting the functional form of the data to bring them into alignment with regard to time and intensity. This is referred to as shift and amplitude registration—shift registration describing the adjustment of the function's time dimension to align features along the abscissa, and amplitude registration describing the increase or reduction in intensity to align features along the ordinate.

To perform registration on the data, it must be converted into a functional form. This means that we represented the data using a set of basis functions and a corresponding set of coefficients. The two choices for the basis functions explained in [10] are the Fourier basis and the spline basis. Since the Fourier basis function is primarily used in periodic data, and our data only spanned a single day, we chose the spline basis as being better suited for our non-periodic dataset.

7) Train, Validate, and Test Separation: Since this is a study exploring the dataspace rather than an exhaustive validation of a training methodology, it was important that we kept the treatment of the data consistent among training, validation, and testing sets. For this reason, the time indices for the training, validation, and testing segments of the data were pre-selected before any of these processes proceeded. After completing the aforementioned procedures, there were 93 data instances for each area of interest. For training we reserved 73 of these, selected at random, and 10 each for validation and testing.

8) Sequential Data Training: The data from all of the points of interest on the map are bound together by time. This means that the first instance for each location happens at the same time as the first instance in all other locations, and so on. Normally, during the training of neural networks, the examples are fed to the procedure randomly. This would have the effect of scrambling the temporal sequence across the dataspace, rendering the instances incomparable. Since we wanted total consistency in the training of these networks, and so that there was no randomization during the training process, exactly one ordering of data was used This was done to ensure that the same time indices were used for all areas of interest and all of the training epochs corresponded with each other. This ordering was pre-selected and was applied system-wide.

C. Layer Pre-Training

1) Prototype Network: In the interest of removing all sources of variation in the pre-training procedure, the networks trained on the data from each area of interest were cloned from a single, randomly initialized, autoencoder. Therefore, all pre-training had the same random starting point. The *prototype autoencoder* was a single layer of 150 nodes and its layer weights were each initialized to random values between -1 and $+1$.

2) Node Profiles Pairwise Dot Products: After pre-training, each network was the result of the original, cloned autoencoder having been pre-trained on data from its respective area of interest. To compare what was learned by each node across the dataspace, we collected each node's incoming weight vector. Specifically, for each autoencoder and for each area of interest we collected the weight vector of each corresponding node and arranged them into a similarity matrix. Because of the way the networks were trained, we assert that the feature learned by a particular node at one location corresponds to the same feature learned by that node in another location. More formally, suppose we have two autoencoders A_1 and A_2. Suppose we order the hidden nodes of each autoencoder as $h_1, \ldots, h_n$. Because of the controlled pre-training procedure and the fact each autoencoder started from the same state, we assert that node $h_i^{A_1}$ and $h_i^{A_2}$ are examining the same

feature of the underlying data and are, therefore, comparable. We computed the pairwise dot products of the node weight vectors for each node as the measure of similarity between each of the node profiles for each location.

3) Semivariograms: Having the matrix of pairwise dot products based on location allowed us to analyze the variance as a function of distance between each node. For this we used semi-variograms [16], [17], which depict the differences in the dot products for all of the nodes for each location. This geostatistical tool enabled us to examine the extent to which the results of unsupervised pre-training were spatially dependent [18]. A geostatistical analysis is appropriate here because we have endeavored to remove all other randomness from the model, and are instead analyzing the systems that remain: those of the pre-trained neural model and the mixture of random and functional dynamics that are endemic to the storm system [18]. The mathematical definition of a semivariogram is as follows [17].

$$\gamma(\vec{h}) = \frac{1}{2} Var\left[Z(\vec{x}+\vec{h}) - Z(\vec{x})\right]$$

where γ, is derived from spatially distributed random variables $Z(\vec{x})$ and $Z(\vec{x}+\vec{h})$, and $\vec{x}$ and $\vec{x}+\vec{h}$ are the spatial positions separated by $\vec{h}$ [17].

Figure 5 shows three examples of semi-variograms that were produced during this procedure. Each of these charts uses a different scale on the ordinate, since the scale of each of the pairwise differences differed significantly. To remove the influence of the scale in the expression of the patterns in the data we individually scaled each of the semi-variograms to a range between 0 and 1. This allowed us to pairwise compare the patterns that were in the semivariograms, rather than the data that generated them.

4) Hierarchical Agglomerative Clustering: Since all of the semi-variograms were now on the same scale, we used hierarchical agglomerative clustering (HAC) to determine interrelatedness among semi-variograms. Since the number of clusters is an input parameter to HAC, we assessed each clustering from 2 clusters through 9 clusters. To determine what we regarded as an optimal clustering we used the Calinski-Harabaz score (also known as the pseudo-F score) [19] as follows.

$$F_{NC} = \frac{\sum_{i=1}^{NC} n_i d^2(c_i, c)/(NC-1)}{\sum_{i=1}^{NC}\sum_{x\in C_i} d^2(x, c_i)/(N-NC)}$$

where NC is the number of clusters, N is the number of objects in the dataset, n_i is the number of examples in the ith cluster, $d(x, y)$ is the Euclidean distance between x and y, c_i is the center of the ith cluster, and c is the center of the dataset. The Calinski-Harabaz measure is an evaluation of cluster validity based on intra-cluster distance and inter-cluster distance. An example Calinski-Harabaz score profile is shown in Figure 6. The chart shows that, for this configuration, the optimal HAC clustering to use is three clusters.

We identify the clusters inside the clusterings by their silhouette scores, which is a quality measure based on pairwise differences of between and within-cluster distances. After using the Calinski-Harabaz score to select the number of clusters as an input parameter, the clusters that were produced using HAC had different silhouette scores, indicating varying cluster quality. Since we know that a maximal silhouette score is to be preferred, we wanted to see if cluster quality in this respect had an impact on the resulting convergence and prediction performance. The results depicted in Figure 9 are divided by average silhouette score for each cluster. The silhouette score is defined as:

$$Sil_{NC} = \frac{1}{NC}\sum_{i=1}^{NC}\left[\frac{1}{n_i}\sum_{x\in C_i}\frac{b(x)-a(x)}{\max\{b(x),a(x)\}}\right]$$

where

$$a(x) = \frac{1}{n-1}\sum_{x\in C_i, y\neq x} d(x, y)$$

and

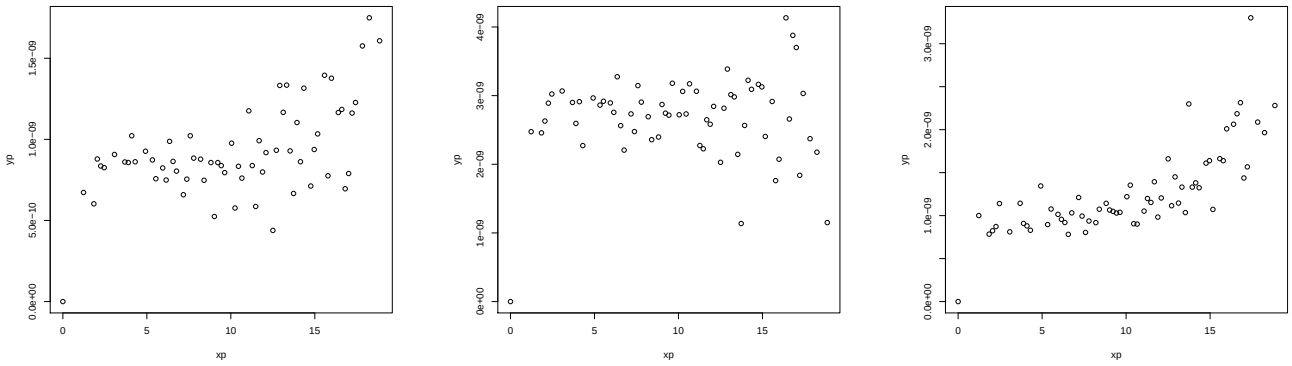
$$b(x) = \min_{i,j}\left[\frac{1}{n_j}\sum_{y\in C_j} d(x, y)\right].$$

Again, NC is the number of clusters, n is the number of objects in the dataset, $d(x, y)$ is the Euclidean distance between x and y, and c_i and c are the centers of an individual cluster and entire dataset respectively.

The three charts in Figure 5 are semi-variograms of nodes that were examples of three clusters, each with different average Silhouette coefficients (ASC). When we examined the semi-variograms across the space of all nodes, we observed this variety of patterns. It is this data that allows us to separate nodes that we fix in the next step, as opposed to nodes that we allow to vary.

5) Fixed Pre-Training: What we obtained from each cluster is the *fixed-set*, which is a list of nodes to transfer to other locations in the dataspace. This forms the basis for the transfer learning experiment. In this procedure, the original autoencoder layers were once again copied from the single prototype autoencoder. The node weights for the fixed-set of nodes were then transferred into the copy of this prototype. Holding the weights of the fixed-set constant throughout the pre-training procedure, the resulting autoencoder layers were pre-trained in this configuration. Figure 7 shows this situation for one layer of the network. In this figure, the shaded (red) nodes are copies from another network identified from a cluster of spatially correlated feature detector nodes. In this paper we only used a one-layer stacked autoencoder; however, we intend to extend this to multiple layers in future work.

For the "Surrounding POI Experimnts" described in section VI-A we refer back to Figure 3. The location in the upper left indicated by the star shows an example area of interest whose node weights were copied. The surrounding dots represent the areas of interest to which these node weights were copied.



Node 19 (ASC = 0.11) Node 1 (ASC = 0.18) Node 18 (ASC = 0.43)

Figure 5: Example Semivariograms from Each Cluster In Selected Clustering. ASC = Average Silhouette Coefficient

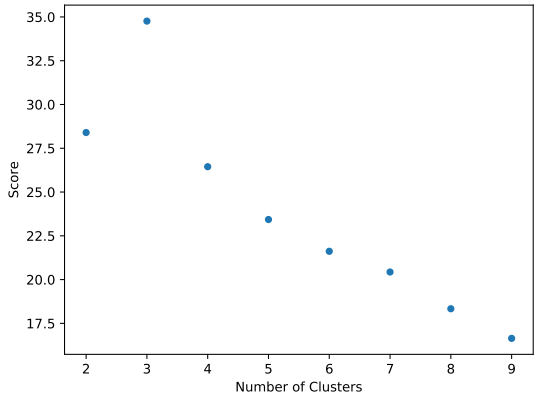


Figure 6: Calinski-Harabaz Score for Clusterings: 2–9 Clusters

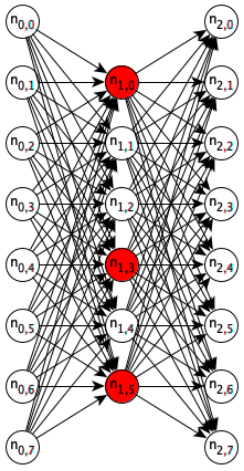


Figure 7: Autoencoder Fixed Nodes

For the "Linear Cross Transfer Experiment," whose results are described in section VI-B we refer to Figure 11, where the weights were copied from each corner location (locations 12, 19, 82, and 89) into the networks corresponding to the line of AOI's leading to the respective diagonal opposite corners. For example, for location 12 in the Figure, the fixed-set was copied to the networks to be trained on trained on data from locations 23, 34, 45, 56, 67, 78, and 89.

D. Fine Tuning and Testing

After transfer and during fine-tuning, the training data are fed into the network in the same order as they were fed in for the unsupervised pre-training procedure. This, again, is to remove as many sources of variation as we could during the entire procedure. The number of iterations and associated mean squared error of the networks were tracked and compared to the original training to determine if the transfer learning process was more efficient and effective as hypothesized.

VI. Results

A. Surrounding POI Experiments

Figure 8 shows the difference in convergence time that is typical for each of the locations surrounding the location from which we transferred the fixed-set. As can be seen, fixing the nodes from the pre-training of the center cell had the effect of substantially reducing convergence times. It also shows consistently lower overall mean squared error with respect to autoencoder reconstruction during the pre-training procedure.

Figure 9 shows the test performance when predicting the wind vectors for each of four configurations that we used. We tested configurations that were both regularized (L1 regularization) and unregularized, and we used both the ReLU and hyperbolic tangent (tanh) activation functions in the networks that were assembled from the autoencoder labels. Of particular interest is the configuration that used regularization and the ReLU activation function. In this configuration, using fixed

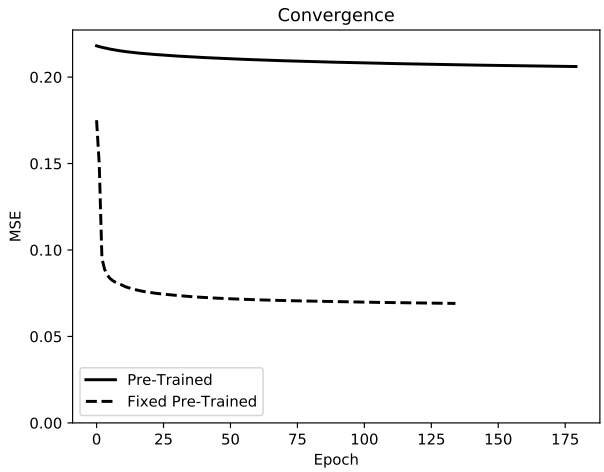


Figure 8: Typical Convergence Comparison Plot

nodes from the cluster corresponding to the higher average silhouette coefficient produced better predictive results, in general, than either those of the fine-tuned pre-trained networks or the fixed pre-trained configuration using the lower average silhouette coefficient. In general, the results from the fixed pre-trained configurations corresponding to the lower silhouette coefficients produced more erratic results.

The heatmaps in Figure 10 are visual representations of the average difference in predictive accuracy for the 8 cells surrounding the cell from which the fixed-set was copied. Again, in reference to Figure 3 the fixed-set was copied from the location represented by the star. We created the heatmaps by averaging the differences of the MSE's of each location surrounding the starred location. The darker cells represent lower differences and the lighter cells represent higher differences.

The first row of heatmaps represents the cluster with the lowest silhouette coefficient, and the second row represents the cluster with the highest silhouette coefficient. In these heatmaps, we observe a greater mean squared error variability in the cluster with the lowest silhouette coefficient. This may suggest that greater transferability is achieved using the fixed-set from the cluster with the greater silhouette coefficient. For the u component, these differences are shown numerically in Tables II and III.

B. Linear Cross Transfer Experiment

In this experiment we wanted to see how the convergence properties and prediction accuracy changed as we moved far across the data space. To achieve this the fixed-sets were taken from the networks corresponding to the corners and copied to the networks corresponding to the line crossing the data space diagonally, as shown in Figure 11. For example, nodes were copied from location 12 to each network along the path to location 89, etc.

The convergence behavior resembled that from the previous experiment, whose results are shown in Figure 8, with minor variation. The convergence time was substantially lower and the MSE converged to a similarly lower value.

Figure 12 shows the prediction performance of the resulting networks for the configuration where no regularization was used and ReLU was used as the activation function. For the first two rows the locations moving to the right across the x axis indicate a movement in the dataspace farther away from the network from which the trained nodes were copied. In the last two rows, movement to the left along the x axis indicates this movement.

VII. Conclusions and Future Work

We observe that selecting a clustering from a HAC clustering with the highest Calinski-Harabaz score, and using this clustering with the information about the silhouette scores of the clusterings can provide information about what random initializations allowed better generalization across the dataspace, given the context of the other initializations. We note, however, that this procedure is not practical in the sense of providing a general training and transfer process. But then, coming up with such a procedure was not our goal. Rather, our goal was to analyze spatial correlation in learned feature detectors to determine whether or not such spatial correlation might be exploited in transfer learning.

For future work, we will shift our focus from analysis to developing a practical training procedure. Specifically, our intent is to apply model reduction strategies (e.g., weight pruning) to determine those feature detectors in the network that are likely to exhibit the desired spatial correlation. An alternative approach might be to apply the ideas from the lottery ticket hypothesis [20], [21] as a way of identifying transferable subnetworks. These features might then be transferable and tunable in the other areas of the dataspace.

An interesting component to this problem is in determining whether or not spatial differences in this type of data cause the behavior of the attendant phenomena to be different enough to warrant an original network for each specific area to be modeled. We do not think this is so, but if we are right then there must be a way to divide the dataspace intelligently into areas appropriate for application of models that were trained on "compatible" areas. For this, we also need to determine what we mean when we call these areas "compatible." There also may be a need to allow overlap, in which case we may introduce fuzzy clustering or a type of mixture model.

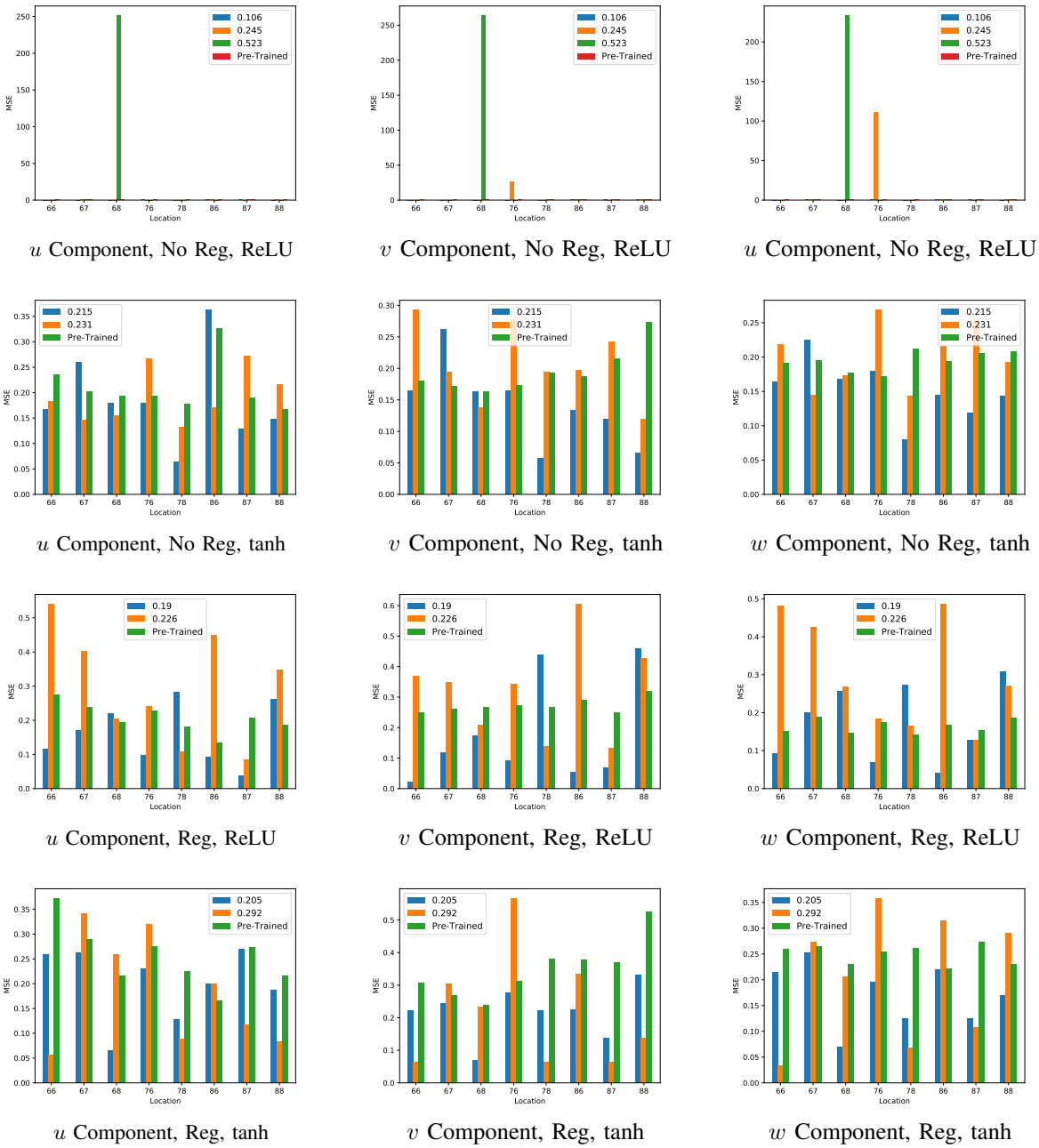
References

[1] D. Advanced, "DARPA XAI BAA," pp. 1–52, 2016. https://www.darpa.mil/attachments/DARPA-BAA-16-53.pdf

[2] W. Samek, T. Wiegand, and K.-R. Muller, "Explainable Artificial Intelligence: Understanding, Visualizing, and Interpreting Deep Learning Models," *ITU Journal: ICT Discoveries*, no. Special Issue No. 1, 2017.

[3] T. T. Warner, *Numerical Weather and Climate Prediction*. Cambridge University Press, 2011.

[4] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A Learning Algorithm for Boltzmann Machines*," *Cognitive Science*, vol. 9, pp. 147–169, 1985.



u Component, No Reg, ReLU | v Component, No Reg, ReLU | u Component, No Reg, ReLU

u Component, No Reg, tanh | v Component, No Reg, tanh | w Component, No Reg, tanh

u Component, Reg, ReLU | v Component, Reg, ReLU | w Component, Reg, ReLU

u Component, Reg, tanh | v Component, Reg, tanh | w Component, Reg, tanh

Figure 9: Prediction Results for location 77 (37.07°Lat, -73.79°Lon): Silhouette Coefficients Given in Legends

[5] D. Erhan, Y. Bengio, A. Courville, P. Vincent, and S. Bengio, "Why Does Unsupervised Pre-training Help Deep Learning?" *Journal of Machine Learning Research*, vol. 11, pp. 625–660, 2010.

[6] S. J. Pan and Q. Yang, "a Survey on Transfer Learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 1, no. 10, pp. 1345–1359, 2010.

[7] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big Data*, vol. 3, no. 1, p. 9, 12 2016.

[8] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, and G. Zhang, "Transfer Learning using Computational Intelligence: A Survey," *Knowledge Based Systems*, vol. 80, no. 10, pp. 14–23, 2015.

[9] Q. Hu, R. Zhang, and Y. Zhou, "Transfer learning for short-term wind speed prediction with deep neural networks," *Renewable Energy*, vol. 85, pp. 83–95, 1 2016.

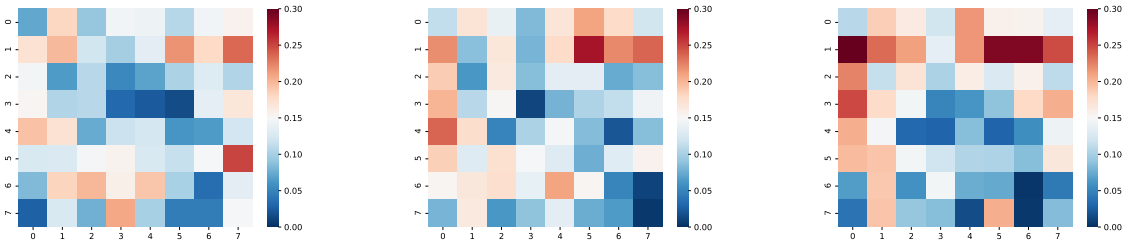
[10] B. W. Silverman and J. O. Ramsay, *Functional Data Analysis*. Springer, 2005.

[11] K. King, "Functional Data Analysis With Application to United States Weather Data," Ph.D. dissertation, 2014. https://core.ac.uk/download/pdf/51066752.pdf

[12] K. Zhang and A. J. Gasiewski, "Microwave CubeSat fleet simulation for hydrometric tracking in severe weather," in *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2016, pp. 5569–5572.

[13] K. Sahr, D. White, and A. J. Kimerling, "Geodesic Discrete Global Grid Systems Discrete Global Grid Systems: Basic Definitions," *Cartography and Geographic Information Science*, vol. 30, no. 2, pp. 121–134, 2003.

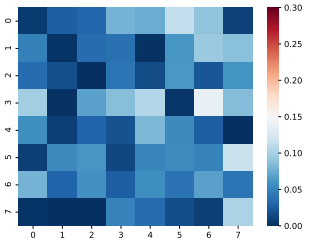
[14] R. A. McAllister and J. W. Sheppard, "Deep Learning for Wind



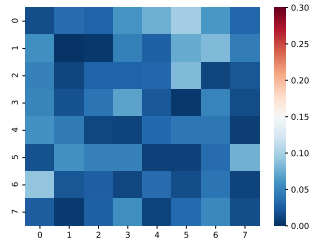
u Component, SC = 0.190
Avg RMSE: 0.1240 ± 0.0029

v Component, SC = 0.190
Avg RMSE: 0.1292 ± 0.0035

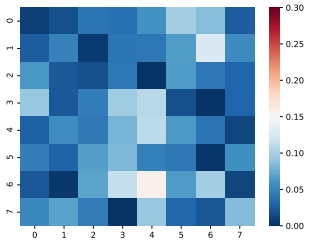
u Component, SC = 0.190
Avg RMSE: 0.1364 ± 0.0053



u Component, SC = 0.226
Avg RMSE: 0.0456 ± 0.0012



v Component, SC = 0.226
Avg RMSE: 0.0365 ± 0.0006



w Component, SC = 0.226
Avg RMSE: 0.0507 ± 0.0012

Figure 10: Heat Maps Depicting Neighborhood Mean Squared Error Differences: Regularization, ReLU

Table II: Differences in Prediction Error in Surrounding Regions for Locations in Cluster with Silhouette Coefficient 0.190

	0	1	2	3	4	5	6	7
0	0.071663	0.181648	0.091526	0.144587	0.142470	0.107549	0.144603	0.156638
1	0.173422	0.198090	0.120792	0.098738	0.134753	0.216859	0.180250	0.236516
2	0.146268	0.065116	0.107992	0.053564	0.069530	0.102851	0.129786	0.104743
3	0.152467	0.104508	0.108294	-0.031736	0.024594	0.015864	0.134914	0.167747
4	0.193729	0.172512	0.074459	0.117674	0.123894	0.061272	0.065049	0.122773
5	0.126530	0.127691	0.147620	0.155785	0.126462	0.115997	0.148730	0.251933
6	0.083317	0.183645	0.200183	0.159251	0.192128	0.101274	0.035824	0.133931
7	0.027401	0.126359	0.077443	0.208089	0.099937	0.045052	0.044794	0.148031

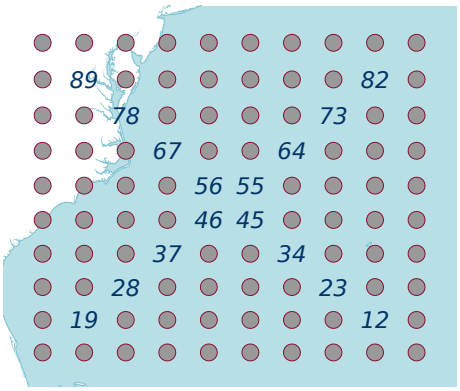


Figure 11: Linear Cross Training Locations

Vector Determination," in *IEEE Symposium Series on Computational Intelligence*, Honolulu, HI, 2017.

[15] R. McAllister and J. Sheppard, "Evaluating Spatial Generalization of Stacked Autoencoders in Wind Vector Determination," in *FLAIRS Conference*, Melbourne, FL, 2018.

[16] A. E. Gelfand, P. J. Diggle, M. Fuentes, and P. Guttorpt, *Handbook of Spatial Statistics*, 2010, vol. 20103158.

[17] M. Bachmaier and M. Backes, "Variogram or Semivariogram? Variance or Semivariance? Allan Variance or Introducing a New Term?" *Mathematical Geosciences*, vol. 43, no. 6, pp. 735–740, 8 2011.

[18] M. A. Oliver and R. Webster, *Basic Steps in Geostatistics: The Variogram and Kriging*, ser. SpringerBriefs in Agriculture. Cham: Springer International Publishing, 2015.

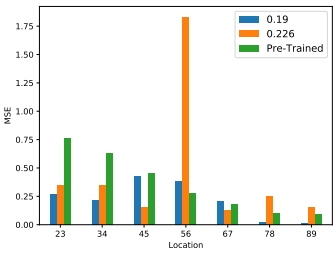
[19] Y. Liu, E. Racah, Prabhat, J. Correa, A. Khosrowshahi, D. Lavers, K. Kunkel, M. Wehner, and W. Collins, "Application of Deep Convolutional Neural Networks for Detecting Extreme Weather in Climate Datasets," in *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, 5 2016, pp. 81–88.

[20] J. Frankle and M. Carbin, "The Lottery Ticket Hypothesis: Finding Small, Trainable Neural Networks," Tech. Rep. https://arxiv.org/pdf/1803.03635.pdf

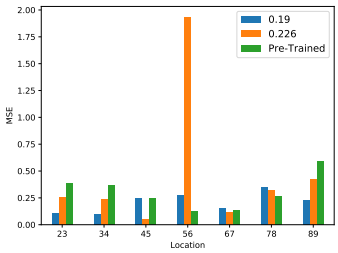
[21] R. V. Soelen and J. W. Sheppard, "Pruned Networks for Transfer Learning," in *IEEE International Joint Conference on Neural Networks*, 2019.

Table III: Differences in Prediction Error in Surrounding Regions for Locations in Cluster with Silhouette Coefficient 0.226

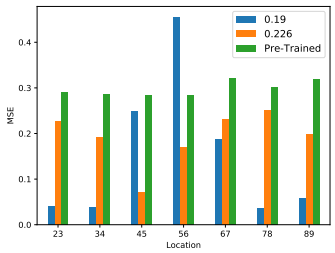
	0	1	2	3	4	5	6	7
0	-0.006103	0.026441	0.031365	0.079269	0.075486	0.113234	0.089962	0.010246
1	-0.046888	0.002377	-0.035044	-0.038244	-0.001453	0.061009	0.093929	0.087198
2	0.034286	0.017958	0.001052	-0.040874	-0.016071	0.062118	0.022195	0.059957
3	0.097574	-0.001716	-0.068291	-0.085860	-0.106535	0.004428	0.137219	0.085019
4	0.057002	0.008859	0.029313	0.019012	-0.083162	-0.054874	-0.026438	-0.000454
5	-0.009035	-0.054717	-0.061529	-0.013007	-0.050549	-0.055664	0.049241	0.116108
6	0.079068	0.029965	0.058726	0.026723	-0.056399	-0.039050	-0.068245	0.040383
7	-0.002633	0.000531	-0.000337	0.049586	-0.034041	0.017514	0.008220	0.102648



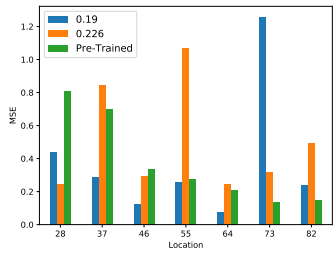
u Component, From Loc 12,



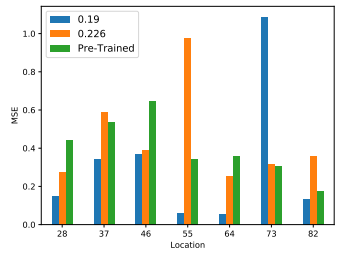
v Component, From Loc 12



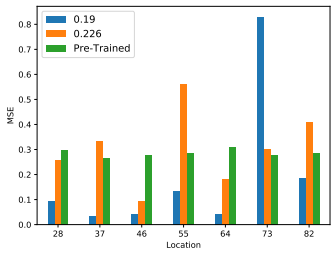
u Component, From Loc 12



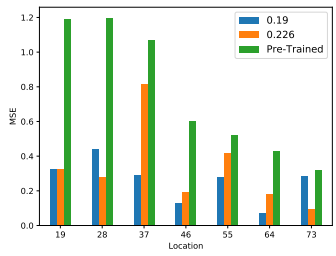
u Component, From Loc 19



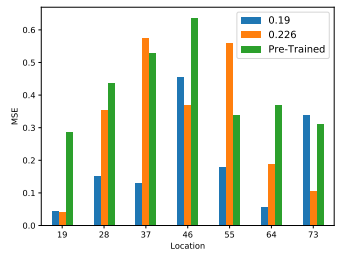
v Component, From Loc 19



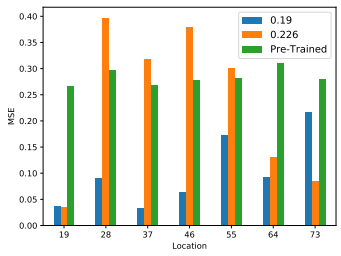
w Component, From Loc 19



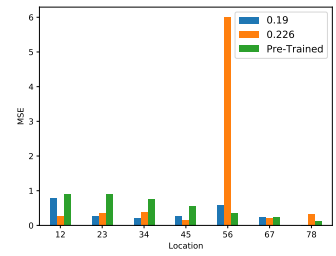
u Component, From Loc 82



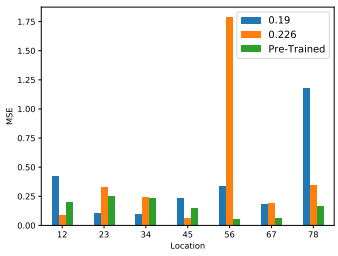
v Component, From Loc 82



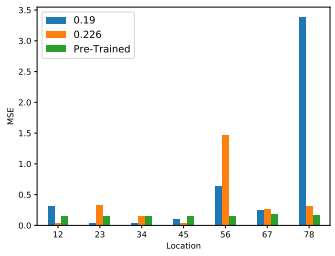
w Component, From Loc 82



u Component, From Loc 89



v Component, From Loc 89



w Component, From Loc 89

Figure 12: Cross Location Prediction Performance: Silhouette Coefficients Given in Legends