Towards Spatial Variability Aware Deep Neural Networks (SVANN): A Summary of Results

Jayant Gupta
gupta423@umn.edu
University of Minnesota
Twin Cities, USA

Yiqun Xie
xiexx347@umn.edu
University of Minnesota
Twin Cities, USA

Shashi Shekhar
shekhar@umn.edu
University of Minnesota
Twin Cities, USA

ABSTRACT

Spatial variability has been observed in many geo-phenomena including climatic zones, USDA plant hardiness zones, and terrestrial habitat types (e.g., forest, grasslands, wetlands, and deserts). However, current deep learning methods follow a spatial-one-size-fits-all (OSFA) approach to train single deep neural network models that do not account for spatial variability. In this work, we propose and investigate a spatial-variability aware deep neural network (SVANN) approach, where distinct deep neural network models are built for each geographic area. We evaluate this approach using aerial imagery from two geographic areas for the task of mapping urban gardens. The experimental results show that SVANN provides better performance than OSFA in terms of precision, recall, and F1-score to identify urban gardens.

CCS CONCEPTS

• **Information systems** → *Data mining*; • **Computing methodologies** → **Neural networks**.

KEYWORDS

Spatial variability, Deep Neural Network, Aerial Imagery

ACM Reference Format:

Jayant Gupta, Yiqun Xie, and Shashi Shekhar. 2018. Towards Spatial Variability Aware Deep Neural Networks (SVANN): A Summary of Results. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Deep learning techniques have resulted in significant accuracy improvements in image based object recognition tasks [12, 29]. They use multiple layers that allow approximate modeling of all continuous functions [3]. Unlike traditional machine learning, which requires manual feature engineering, deep learning models interpret the data and automatically generate features [4]. The current deep learning literature [8, 18, 38] follows a spatial one size fits all approach in which deep neural networks are trained without consideration of spatial variability.

Woodstock '18, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

Geographic properties differ across different areas giving rise to varied geophysical and cultural phenomena. This spatial variability results in the lack of consistent object detection models across geographic areas. Knowledge of spatial variability is necessary to understand the spatial patterns of events and objects over an area that vary spatially [30]. The models can be affected by two types of spatial variability: the variability in the object of interest itself, which may differ in shape, size, or both; and the variability in the background of the object of interest. For example, a computational model that is trained to find residential housing in the the US may have difficulty finding houses in other places where housing construction is adapted to different local climates or other conditions (e.g., cave houses in Petra, igloos in polar regions, etc.). Here the neighbor surroundings differ as well. Figure 1 shows the spatial variability in houses and their background across the globe.

(a) Igloo in Iiyama City, Japan. (b) Apartments in Shanghai, China. (c) Neighborhood in the US. (d) Ancient cave houses in Petra, Jordan

Figure 1: Spatial variability in houses and background.

Spatial variability has been observed in many geo-phenomena including climatic zones, USDA plant hardiness zones [20], and terrestrial habitat types (e.g., forest, grasslands, wetlands, and deserts). The difference in climatic zones affect the plant and animal life of the region. Similarly, knowledge of plant hardiness zones helps gardeners and growers to assess appropriate plants for a region. Further laws, policies and culture differ across countries and even states within some countries. Spatial variability is considered as the second law of geography [14] and has been adopted in regression models (e.g., Geographically Weighted Regression [5]) to quantify spatial variability, the relationship among variables across study area. In this work, we assess the effect of spatial variability on object detection models built using deep learning techniques.

Specifically, we investigate a spatial variability aware deep neural network (SVANN) approach where distinct deep neural network models are built for separate geographic areas. The paper describes alternative ways to model spatial variability, including zones and distance-weighting. It also provides descriptions of alternative ways for training and make predictions using SVANN (Section 3.2). The proposed SVANN approach was evaluated experimentally as well as via a case study for detecting urban gardens in geographically diverse high-resolution aerial imagery. The experimental results show that SVANN provides better performance in terms of precision, recall, and F1-score to identify urban gardens.

Application domains and example use cases where spatial variability is relevant and needs to be considered include wetland mapping, cancer detection, and many others. A few examples are listed in Table 1.

Table 1: Application domain and use-case of spatial variability.

Application Domain	Example use cases
Wetland Mapping	Wetlands in Florida (e.g., mangrove forest) are different from those in Minnesota (e.g., marsh)
Cancer cell identification	Cancer in pathology tissue samples is known to be spatially heterogeneous [10]
Vehicle detection	vehicle types differ across India (e.g., autorickshaw) and USA (SUVs)
Residence detection	House types and design (e.g., igloos, huts, flatroof, ...) differs across geographic areas
Urban Agriculture	Detection of urban gardens:Urban gardens designs may vary across rural (large ones), suburban (small backyard gardens) and urban (e.g., container gardens, community gardens) areas due to differing space availability and risks (e.g., deer, rabbit, ...)

Contributions:

(1) We propose a spatial variability aware deep neural network (SVANN) approach and illustrate various training and prediction procedures.
(2) We use SVANN to evaluate the effect of spatial variability on deep learning models during the learning processe.

Scope: This paper focuses on geographic and other low-dimensional space. Generalization of the proposed approaches to model variability in high dimensional spaces is outside the scope of this paper. We use a convolutional neural network (CNN) for the experimental evaluation and case studies. Evaluation of SVANN with other types of neural networks is outside the scope of this paper.

Organization: The paper is organized as follows: Section 2 describes the details of SVANN along with different training and prediction procedures. Section 3 describes the evaluation framework giving details on the evaluation task, evaluation metric, dataset, and experiment design. In Section 4, we present the results and a discussion of the effects of spatial variability. Section 5 briefly discusses the relevant related work. Finally, Section 6 concludes with future directions. In Appendix A, we compare and contrast different types of aerial imagery. Then, we give details of object detection using YOLO framework in Appendix B. We also give details on dataset development in Appendix C.

2 APPROACH

In this section, we provide the details of our SVANN approach and differentiate it from the spatial one size fits all (OSFA) approach. Figure 2 shows spatial One Size Fits All (OSFA) approach using a CNN with 3 layers: a convolution layer, a spatial pooling layer, and a fully connected layer. The initial 2 layers perform feature engineering and selection, whereas, the fully connected layer is responsible for output prediction. As shown, the approach does not account for the geographic location of training samples.

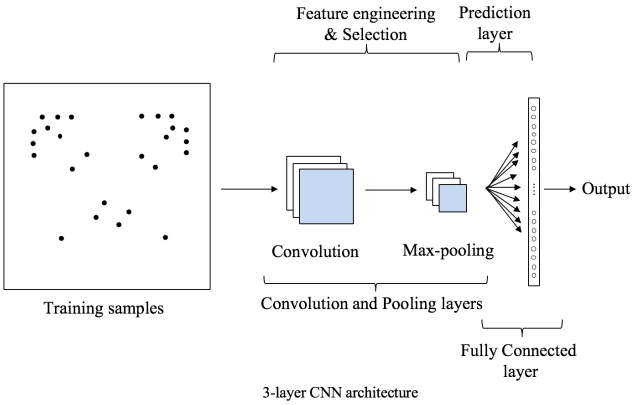


Figure 2: Spatial One Size Fits All (OSFA) approach using a CNN with 3-layers: convolution, spatial pooling, and a fully connected layer.

2.1 SVANN

SVANN is a spatially explicit model where each neural network parameter (e.g., weight) is a function of model location loc. The model f is composed of a sequence of K weight functions or layers $(w^1(loc), ..., w^K(loc))$ mapping a geographic location based training sample $x(loc)$ to a geographic location dependent output $y(loc)$ as follows,

$$y(loc) = f(x(loc); w^1(loc), ..., w^K(loc)), \quad (1)$$

where $w^i(loc)$ is the weight vector for the i_{th} layer. Figure 3 shows the SVANN approach where the geographical space has 4 zones and deep learning models are trained for each zone separately. For prediction, each zonal model predicts the test sample in its zone.

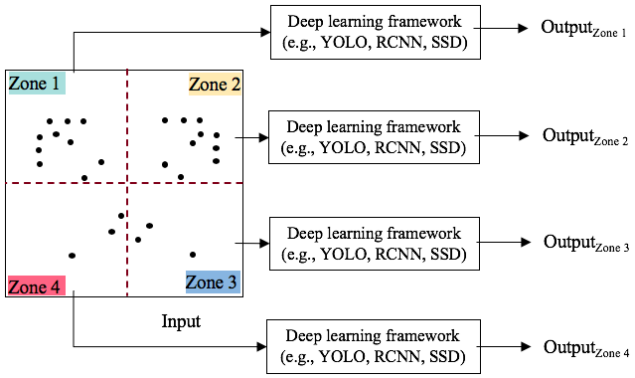


Figure 3: SVANN using fixed-partition based neighbors. Four distinct models are trained using training samples from each zone.

SVANN can be further classified by the choice of training and prediction procedures. Here, we describe some of these procedures.

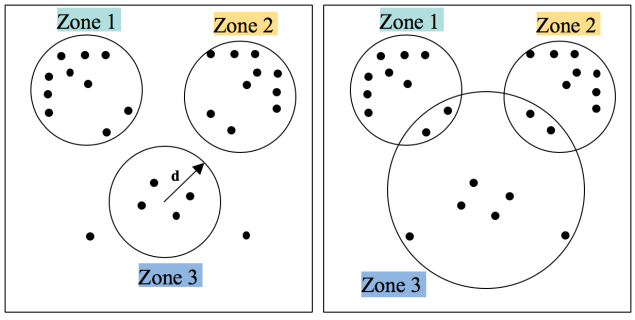
2.1.1 ***Training:*** There are at least two possible training procedures, namely, model-location dependent sampling for learning and distance weighted model-location dependent sampling for learning.

1. Model-location dependent sampling for learning: Model parameters for a location are derived by training the model using labeled samples from nearby locations. There are three types of nearest neighbor techniques that can be considered:

(a) Fixed partition based neighbors: Partitions (also known as zones) are used when policies and laws vary by jurisdictions such as countries, US states, counties, cities, climatic zones. We use administrative, zonal partitions of geographical space to build individual models. This approach is simple but relatively rigid as partitions are usually disjoint and seldom change. Figure 3 shows training SVANN models using zone based neighbors, where a sample from each zone is used to train a model for that particular zone. Partitioning the data based on zones can break up natural partitions (e.g., Zone-3 and Zone-4 in Figure 3.).

(b) Distance bound nearest neighbors: In this training regime, a model at location (loc_M) is trained using nearby training samples within distance d. This model assumes that there are sufficient training samples in the vicinity of model locations. This approach maybe more flexible than fixed partition based approach as the training samples can overlap across models and the model locations can adapt to the spatial distribution (e.g., hotspots) of learning samples. Figure 4(a) shows training of different models using training samples within distance d.

(c) K-nearest neighbors: In this training regime, a model at location (loc_M) is trained using k-nearest training samples in the geographic space. This model does not assume that there are sufficient training samples in the geographic vicinity of model locations. Thus, this approach may be more flexible than distance bound nearest neighbors. Figure 4(b) shows training of different models using k-nearest training samples.



(a) Distance bound nearest neighbors (b) K-nearest neighbor

Figure 4: Model-location dependent sampling for learning.

In Model-location dependent sampling for learning (1a-1c), selected learning samples are treated equally in the training phase.

2. Distance weighted model-location dependent sampling for learning: In this training approach, all learning samples can be used to train models at different locations. To address spatial variability, nearby samples are considered more important than further away samples by adapting the learning rate. To update the neural network weights, the learning rate is multiplied by a back propagation error and a function of the distance between the selected learning sample and the location of the model. Equivalently, the learning rate depends on the distance between the labeled sample and the location for which the model is being trained. The distance function can be thought of as the inverse of distance squared as follows,

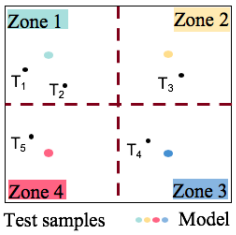
$$w^i(loc_M) = w^i(loc_M) + \frac{\eta}{d^2(loc_M, loc_S)} * x^i(loc_S) * \Delta y^i(loc_S) \quad (2)$$

where, η is the learning rate, d is the distance between the location of learning sample (loc_S) and the location of model (loc_M), $x^i(loc_S)$ is the input to the i^{th} layer, and $\Delta y^i(loc_S)$ is the backpropagated error at layer i. This approach is similar to boosting techniques [6] where weak learners or hypotheses are assigned weights based on their accuracy. It is also similar to geographically weighted regression (GWR) [5] where regression coefficients and error are location dependent.

In the context of object detection in imagery via CNN, we note that CNN may favor nearby pixels over distant pixels (by using convolutional and pooling layers) within a single labeled sample (e.g., an 512x512 image), whereas the proposed method further favors nearby labeled samples over distant labeled samples.

2.1.2 ***Prediction:*** Since multiple models are trained at different locations and a new sample may not be at those locations we discuss two prediction methods (i.e., zonal and distance weighted voting) to combine the predictions from multiple models for the new sample.

1. Zonal: Given a fixed partitioning of the geographic space (e.g., counties) prediction results from the model within the same partition will be used for prediction. If, there are multiple models within a partition, voting (e.g., majority, mean) can be used for prediction. Here the votes from all models within the partition are treated equally. Also, samples located at zone boundaries are disjoint and are assigned to a single zone. Zonal prediction is suitable for models trained on model-location dependent learning samples. Figure 5 shows an example with 5 test samples (T_1 - T_5) and 4 partitions, where each model in a partition is a binary classifier representing classes as (0, 1). The Zone 1 model is used to make prediction for test sample T1 and T2. The Zone 2 model is use to make prediction for T3 and so on.

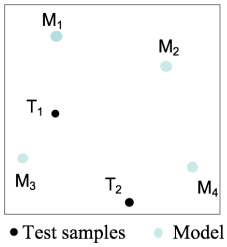


Partition	Sample	Prediction
Zone-1	T_1	1
	T_2	1
Zone-2	T_3	1
Zone-3	T_4	0
Zone-4	T_5	0

Figure 5: Zonal prediction using 5 test samples and 4 partitions.

2. Distance weighted prediction: Given a test sample and distances from all models, we weigh the predictions from each model as an inverse function of the distance. The highest weighted prediction is assigned as the class of the test sample. Distance weighted prediction is suitable for models trained using distance weighted model-location dependent learning samples. Figure 6 shows an example with 2 test samples and 4 models where each model predicts sample class (0 or 1). Assume that the adjacent (top right) table shows the predictions and distance ($D(M_i, T_i)$)) of each model from

test samples that are used to calculate class weights and assign class. All models are used to make a prediction for each test sample. For T_1, the nearest models (M_1, M_3) predict its class as 1, whereas for T_2, the nearest models (M_3, M_4) predict its class as 0 which results in final the assigned classes (shown in botton right table) for the two test samples of 1 and 0 respectively.



Model	Class T_1	$D(M_i, T_1)$	Class T_2	$D(M_i, T_2)$
M_1	1	2	1	5
M_2	0	3.5	1	4
M_3	1	1.5	0	2.5
M_4	1	5	0	1.5

Assigned Class T_1	Assigned Class T_2
1	0

Figure 6: Distance weighted prediction using 2 test samples and 4 models. For illustration, approximate distances are used.

2.2 Discussion

One Size Fit All model vs SVANN: Given adequate learning samples and computational resources, SVANN can provide better accuracy over spatial one-size-fits-all models. Indeed, extreme cases of training a singular model may exhibit Simpson's paradox [31], where global behavior may differ from local behavior.

SVANN and number of learning samples: SVANN need more learning samples for training to capture location specific features. However, spatial big data [24] provides a wealth of data with opportunities to develop SVANN. Furthermore, citizen science [25] provides ways where broader participation from scientists and volunteers can help generate relevant training data.

Computational challenges: In SVANN, the number of weights depends on the size of the network, number of locations, and the number of samples. This adds to the existing high computational cost of deep learning frameworks.

Parametric vs Nonparametric: A learning model that summarizes data with a set of parameters of fixed size (independent of the number of training examples) is called a parametric model. In contrast, the number of parameters in non-parametric models is dependent on the dataset [23]. In general, SVANN can be a non-parametric model if the number of locations is not constrained. However, in special cases locations may be constrained to a fixed number of zones (e.g., US states, countries) to create parametric SVANN models.

Using SVANN to assess spatial variability in a phenomenon: If OSFA and SVANN have similar performance on a task then, it will not support the existence of spatial variability in a phenomenon. However, if SVANN outperforms OSFA then the results support spatial variability hypothesis in the phenomenon.

Spatial partitioning: The proposed training procedures do not need partitioning of input training samples. In Fixed partition based neighbors training approach (Section 2.1.1 1a.), partitions are given as input or are part of application domain. For example, COVID-19 models are built based on political boundaries (e.g., countries). In other situations, application domain may be willing to explore data-driven (e.g., spatial characteristics) partitioning, or may depend on the underlying task, which can be explored in future work. In addition, hierarchy may add further benefits and is relevant only when partitioning is needed.

3 EVALUATION FRAMEWORK

This section details the evaluation framework for the SVANN approach. We explain the evaluation task and metric. We then describe the dataset categorized by object characteristics, including conversion from satellite imagery and manual annotation. Finally, we describe the experiment design including the computing resources used for experiments. Further details are given in the Appendix.

3.1 Evaluation Task Definition

An urban garden is defined as a specific piece of land that is used to grow fruits or vegetables. Area-based knowledge of urban gardens aids the development of urban food policies, which currently place a strong focus on local food production. Urban gardens can be divided into many types based on ownership and structure. Figure 7(a) shows gardens based on two types of ownership, private backyard urban garden and community urban farm. Structurally, urban gardens fall into three categories, raised beds, open fields, and rooftop gardens [2]. Figure 7(b) shows two types of gardens based on the physical structure of their beds. As seen, raised beds are highly distinctive due to chracteristics such as surrounding stone walls. In open fields, the distinction in boundaries is relatively low, resulting in lower visual variation of garden and the surrounding area.

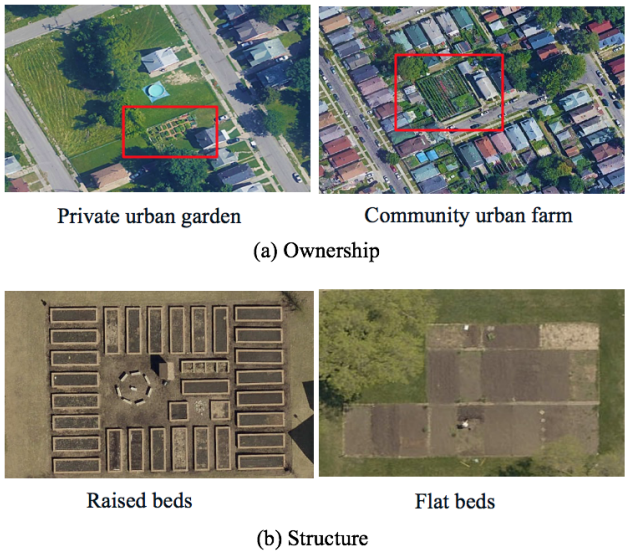


Figure 7: Type of urban gardens. (Best in color)

Given aerial images from different places and an object definition (e.g., urban garden), we build a computational model to detect the object having high precision and recall. There are four key constraints to the task. First, spatial variability can make a spatial one size fits all approach unusable and may require training different models at multiple locations. Second, the imagery encompasses a large geographical area (order of 1000 km^2) that is hard to observe

manually and computationally. Third, the object in this work has no specific features and is only defined by its function to the application domain. Thus, to find and mark these objects for training is hard and can result in ambiguous annotations. Fourth, the objects of interest are in close proximity to taller neighboring objects (e.g., buildings, houses, etc). These neighboring objects cast their shadow depending on the direction of the sun, which results in partial or complete occlusion of the objects. Figure 8 illustrates the task, where the red dotted boxes are the annotations.

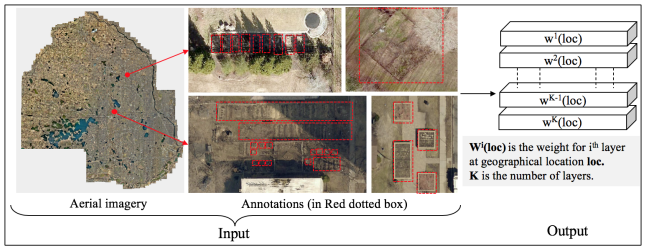


Figure 8: Example Input and output for the SVANN Evaluation Task. (Best in color)

3.2 Evaluation metric

We use the F-1 metric [16] to evaluate the results. The metric is defined as a function of precision and recall where precision is the ratio of true objects detected to the total number of objects predicted by the classifier, and recall is the ratio of true objects detected to the total number of objects in the data set. Precision and recall can be written as the function of True Positives (TP), False Positives (FP), and False Negatives (FN). Figure 9 illustrates TP, FP, and FN in image based detection results, where Figure 9(a) is the input image and Figure 9(b) is the detection result. The detection results are color coded (as shown in the legend) to mark the TP, FP, and FN that are used to calculate the precision, recall, and F-1 score. True negatives i.e. objects other than urban gardens (e.g., Roads, Houses, Trees) were not defined. Hence, metrics using true negatives (e.g., accuracy) were not used for evaluating the results.

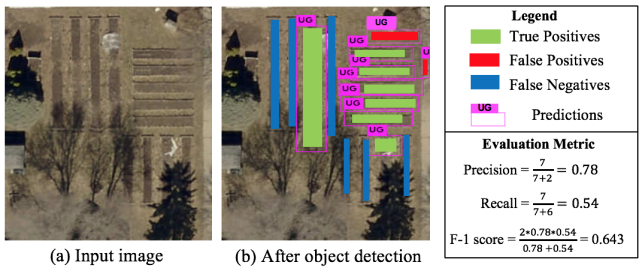


Figure 9: Illustration of true positives, false positives, and false negatives (Best in color).

3.3 Dataset

The initial dataset [1] had object samples from the Minneapolis and St. Paul, MN region that were created using 2015 Google Earth imagery. Due to lower visual variation of garden and the surrounding area higher resolution imagery is preferred (Figure 10). Hence, the samples were converted to high spatial resolution using high resolution aerial imagery [28]. The details on conversion can be found in Appendix C.1. Our team also travelled to hundreds of urban gardens in Minneapolis for ground truth verification.

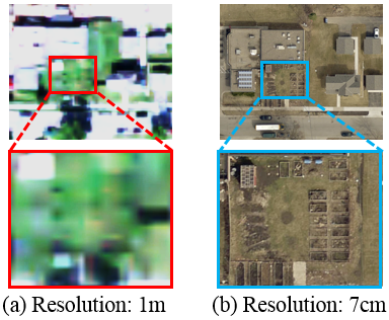


Figure 10: Urban garden in 1m and 7cm resolution imagery. (Best in color) [36].

We used ArcGIS to browse the Fulton county high resolution imagery [27] and annotate the objects. The annotated aerial imagery was then used to train the models. The details on annotation sequence can be found in Appendix C.2.

Object characteristics: Besides annotations, the following generic and application specific characteristics were used for finer analysis of the results (Figure 11).

– **Axis parallel (Yes/No):** Axis parallel objects allow rectangular annotations with reduced background (i.e., false positives) and have better recognition ability. This feature was recorded for every object to assess the model's efficiency on non-axis parallel objects.

– **Rectangular (Yes/No):** Objects that have a clear rectangular shape can be distinctly observed as man-made, which allows better recognition. However, urban gardens can have distinctive shapes that can be geometrical or have curvy boundaries. Hence, we marked the gardens that were (approximately) rectangular from those that were clearly non-rectangular.

– **Occlusion (Yes/No):** Objects that have restricted view due to neighboring buildings and trees are harder to detect and may result in lower recall values. Thus, to assess the recall values occlusion was recorded.

– **Object Type (Flat/Raised):** Urban gardens can have raised beds or flat beds. Raised beds are usually accompanied by distinctive stone boundaries that improve the recognition. We marked the garden type to analyse the model performance based on their type.

For this work we annotated 1314 urban gardens from Hennepin County, Minnesota, US and 419 urban gardens from Fulton County, Georgia, US. The annotated images were divided into train (80%) and test (20%) sets for training and testing the assessment of spatial variability. The data was categorized based on the object characteristics. Table 2 and Table 3 show the dataset details for the two regions. For reproducibility, the dataset used in this work and relevant code are provided [9].

To train well, CNNs require a large number of training data. However, creation of labeled data is expensive, leading to the use of transfer learning and data augmentation. We used Microsoft COCO [15] for transfer learning. It is an extensive dataset that has around $200K$ labeled images with around 1.5 million object

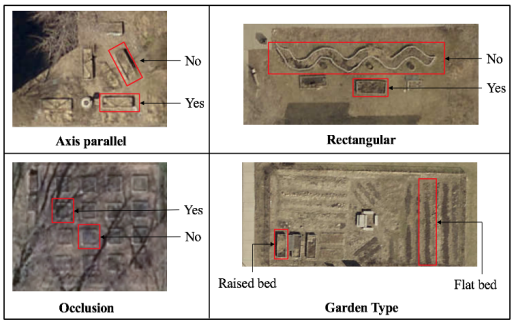


Figure 11: Object characteristics.

Table 2: Train and test data for Hennepin County, MN categorized by object characteristics.

	Axis parallel		Rectangular	
	Yes	No	Yes	No
Train	611	217	761	67
Test	381	105	450	36
Total	992	322	1211	103
	Occlusion		**Garden Type**	
	Yes	No	Flat	Raised
Train	170	658	353	475
Test	116	370	213	273
Total	286	1028	566	748

Table 3: Train and test data for Fulton County, GA categorized by object characteristics.

	Axis parallel		Rectangular	
	Yes	No	Yes	No
Train	89	199	279	9
Test	77	54	125	6
Total	166	253	404	15
	Occlusion		**Garden Type**	
	Yes	No	Flat	Raised
Train	88	200	12	276
Test	27	104	9	122
Total	115	304	21	398

instances divided into 80 object categories. Further, the framework used to evaluate SVANN in this work uses random crops, color shifting, etc for data augmentation [22].

3.4 Experiment design

Since our goal here is a proof of concept, we limited our experiments to the special case of two types of training approaches. We trained individual models for two disjoint and distant geographic regions (i.e., Hennepin county, MN and Fulton County, GA). Due to rigid boundaries it is the base case of fixed partition based neighbors where number of partitions is 2. Furthermore, due to the large distance between the two regions, the samples are not neighbors. Overall, we trained and compared three models where, Model-1 (Hennepin County, MN) and Model-2 (Fulton County, GA) were trained separately on imagery data from different geographical areas; Model-3 was based on a spatial One Size Fits All (OSFA) approach that was trained on imagery data from both areas together. Figure 12 shows the experiment design composed of 4 key parts, Data, Modeling, Parameter tuning, and model evaluation measures.

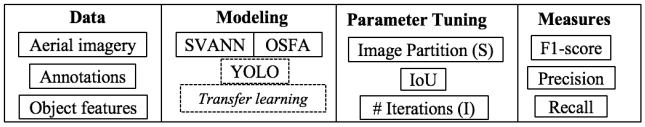


Figure 12: Experiment design

Model 1 and Model 2 were SVANN-baed approach and Model 3 was OSFA-based approach. The evaluation on test data allow an "apples to apples" comparison of the three models. Figure 13 shows the design to assess spatial variability. There were three sets of comparison: SVANN model is compared to OSFA using Hennepin County test data (OSFA_HC); SVANN Model 2 is compared to OSFA (OSFA_FC) using Fulton County test data; Combined SVANN models 1 and 2 compared to OSFA Model 3 on the complete test dataset. Object characteristics are further used for finer level analysis. Framework parameters such as the number of iterations (I), image partitions (S), and IoU were assessed for the model performance. These results were used for tuning the model. The effect of transfer learning was also assessed on the model efficiency.

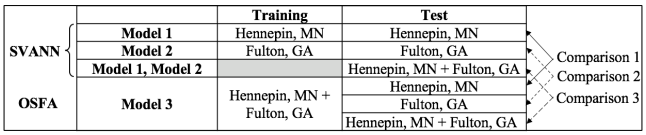


Figure 13: Assessment of spatial variability.

Resources: The experiments were conducted using backpropagation algorithm using a python based Google Tensorflow variant of Darknet [22]. We used K40 GPU composed of 40 Haswell Xeon E5-2680 v3 nodes. Each node has 128 GB of RAM and 2 NVidia Tesla K40m GPUs. Each K40m GPU has 11 GB of RAM and 2880 CUDA cores.

4 EXPERIMENTAL RESULTS

This section presents our spatial variability assessment results and feature based interpretation.

What is the effect of spatial variability? To assess the effect of spatial variability we considered three comparisons (Figure 13). Table 4 shows the results. As shown, both SVANN Model 1 and SVANN Model 2 perform better than OSFA on all the measures (precision, recall, and F1-score) for all three comparisons. The results clearly demonstrate the effectiveness of SVANN over OSFA approach. The latter had learning samples from both regions, which may have resulted in a relatively larger generalization and low recognition of area specific objects. The effect can increase as spatial variation increases across the regions. This results in the dilution of regional differences that may otherwise be useful to identify the objects more accurately. Due to limited training data for the application, we rely on external weights to build effective models. The results might have showed higher variability if we had not used external weights. In addition, spatial variability results

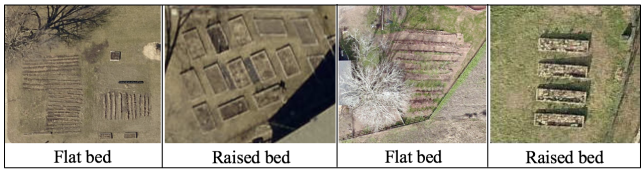
Table 4: Comparison results between SVANN and OSFA.

Approach	Model	Test data	Precision	Recall	F1-score
SVANN	Model 1	Hennepin	0.794	0.419	0.549
OSFA	Model 3	Hennepin	0.713	0.341	0.461
SVANN	Model 2	Fulton	0.924	0.674	0.779
OSFA	Model 3	Fulton	0.886	0.618	0.728
SVANN	Model 1, Model 2	Hennepin + Fulton	0.836	0.485	0.614
OSFA	Model 3	Hennepin + Fulton	0.771	0.412	0.537

that use minimal or no transfer learning can be used to further assess the effects and observe the variability in the results.

Before these experiments our urban planning collaborators assume that urban gardens in Minnesota and Georgia are similar. They were surprised by our results that SVANN outperform OSFA and asked for detailed interpretation of our results. In this paragraph we summarize the findings from the detailed interpretation of results in context of urban garden detection.

Characteristic based interpretation: As shown in Table 2 and Table 3, Fulton County has a significantly higher proportion of raised beds to flat beds. This may suggest different gardening practices in the two regions. Further, this difference may have resulted in higher measure values for SVANN model 2 compared to SVANN model 1; because detection of raised beds is less challenging due to distinct boundaries. In terms of spatial variability, we found that gardens differed in their texture across the two regions. In particular, gardens in Fulton County, GA had a higher green cover as compared to Hennepin Couty, MN. The difference is highlighted in Figure 14, which depicts both raised and flat bed gardens from the two regions.



(a) Sample urban gardens in Hennepin County, MN

(b) Sample urban gardens in Fulton County, GA

Figure 14: Spatial variability in the dataset. As shown, the backyard urban gardens in Fulton county, GA have greener surroundings compared to the backyard urban gardens in Hennepin county, MN. (Best in color).

5 DISCUSSION

Our approach (Section 2.1.1 2. distance weighted model-location dependent sampling for learning) is similar to Geographically Weighted Regression (GWR) [5] where regression coefficients and error are location dependent. However, GWR is a regression based technique that rely on manual features to calculate model weights. In contrast, we use multi-layer CNN [13], where initial layers perform feature engineering and later layers are responsible for prediction.

The approach is also related to common practice in data mining where we first partition the data, and then develop prediction model separately for each partition. The partitions are formed in a high dimensional space which may mute geographic variability. In contrast, we use the partitions in low dimension geographic space in the proposed technique (Section 2.1.1 1a. Fixed partition based neighbors). Similar approach was followed in [11], where a spatial ensemble framework was proposed that explicitly partitions input data in geographic space and use neighborhood effect to build models within each zone. Further, spatial variability has been discussed as a challenge to detect other geospatial objects such as trees [33, 37] and buildings [34, 35] using remote sensing datasets.

6 CONCLUSION AND FUTURE WORK

In this work, we investigated a spatial-variability aware deep neural network (SVANN) approach where distinct deep neural network models are built for each geographic area. We also describe some of the training and prediction procedures for SVANN and list key points relevant to the approach. We chose high spatial resolution imagery for better object detection performance and built deep learning models using a state-of-the-art single stage object detection technique. We evaluated this approach using aerial imagery from two geographic areas for the task of mapping urban gardens. The experimental results show that SVANN provides better performance in terms of precision, recall, and F1-score to identify urban gardens. We also provide a case study that interprets spatial variability in terms of the relative frequency of urban garden characteristics.

In the future, we plan to extend our evaluation of SVANN using other training and prediction approaches. We plan to generalize the proposed approach to model variability in high dimensional spaces. We also plan to evaluate SVANN with other types of neural networks, where we will evaluate the choice of models and network structure in terms of number of layers, neurons per layer, etc. Finally, we will evaluate the trade-off between spatial variability awareness and transfer learning.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grants No. 1029711, 1737633. We would like to thank Dr. Dana Boyer (Princeton University) and Dr. Anu Ramaswami (Princeton University) for providing the Hennepin county dataset and for useful guidance on Urban Agriculture. We would like to thank Minnesota Supercomputing Institute (MSI) for GPU resources. We would also like to thank Kim Koffolt and the spatial computing research group for their helpful comments and refinements.

REFERENCES

[1] Dana Boyer, Rachel Kosse, Graham Ambrose, Peter Nixon, and Anu Ramaswami. 2020. A hybrid transect & remote sensing approach for mapping urban agriculture: informing food action plans & metrics. — under review. *Landscape and Urban Planning* (2020).

[2] ME Brown and JL McCarty. 2017. Is remote sensing useful for finding and monitoring urban farms? *Applied geography* 80 (2017), 23–33.

[3] George Cybenko. 1989. Approximations by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems* 2 (1989), 183–192.

[4] Bradley J Erickson, Panagiotis Korfiatis, Zeynettin Akkus, and Timothy L Kline. 2017. Machine learning for medical imaging. *Radiographics* 37, 2 (2017), 505–515.

[5] A Stewart Fotheringham, Chris Brunsdon, and Martin Charlton. 2003. *Geographically weighted regression: the analysis of spatially varying relationships.* John Wiley & Sons.

[6] Yoav Freund, Robert E Schapire, et al. 1996. Experiments with a new boosting algorithm. In *icml*, Vol. 96. Citeseer, 148–156.

[7] Noel Gorelick, Matt Hancher, Mike Dixon, Simon Ilyushchenko, David Thau, and Rebecca Moore. 2017. Google Earth Engine: Planetary-scale geospatial analysis for everyone. *Remote sensing of Environment* 202 (2017), 18–27.

[8] Yanming Guo, Yu Liu, Ard Oerlemans, Songyang Lao, Song Wu, and Michael S Lew. 2016. Deep learning for visual understanding: A review. *Neurocomputing* 187 (2016), 27–48.

[9] Jayant Gupta. 2020. *Urban Garden Dataset and SVANN code.* https://tinyurl.com/yczutkjw

[10] Andreas Heindl, Sidra Nawaz, and Yinyin Yuan. 2015. Mapping spatial heterogeneity in the tumor microenvironment: a new era for digital pathology. *Laboratory investigation* 95, 4 (2015), 377–384.

[11] Zhe Jiang, Arpan Man Sainju, Yan Li, Shashi Shekhar, and Joseph Knight. 2019. Spatial ensemble learning for heterogeneous geographic data with class ambiguity. *ACM Transactions on Intelligent Systems and Technology (TIST)* 10, 4 (2019), 1–25.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[13] Y LeCun, Y Bengio, and G Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436.

[14] Michael Leitner, Philip Glasner, and Ourania Kounadi. 2018. Laws of geography. In *Oxford Research Encyclopedia of Criminology and Criminal Justice*.

[15] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*. Springer, 740–755.

[16] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*. Cambridge university press.

[17] Renaud Mathieu, Claire Freeman, and Jagannath Aryal. 2007. Mapping private gardens in urban areas using object-oriented techniques and very high-resolution satellite imagery. *Landscape and Urban Planning* 81, 3 (2007), 179–192.

[18] Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel T Dudley. 2018. Deep learning for healthcare: review, opportunities and challenges. *Briefings in bioinformatics* 19, 6 (2018), 1236–1246.

[19] US Department of Agriculture. 2019. Geospatial data gateway. Retrieved February 27, 2020 from https://datagateway.nrcs.usda.gov

[20] United States Department of Agriculture. 2012. *USDA Plant Hardiness Zone Map*. USDA. Retrieved 2019-12-23 from https://planthardiness.ars.usda.gov/PHZMWeb/

[21] Shi Qiu. 2019. *Bbox label tool.* Retrieved 2019-12-23 from https://github.com/puzzledqs

[22] Joseph Redmon and Ali Farhadi. 2017. YOLO9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7263–7271.

[23] Stuart Russell and Peter Norvig. 2002. Artificial intelligence: a modern approach. (2002).

[24] Shashi Shekhar, Viswanath Gunturi, Michael R Evans, and KwangSoo Yang. 2012. Spatial big-data challenges intersecting mobility and cloud computing. In *Proceedings of the Eleventh ACM International Workshop on Data Engineering for Wireless and Mobile Access*. 1–6.

[25] Jonathan Silvertown. 2009. A new dawn for citizen science. *Trends in ecology & evolution* 24, 9 (2009), 467–471.

[26] James D Spinhirne. 1993. Micro pulse lidar. *IEEE Transactions on Geoscience and Remote Sensing* 31, 1 (1993), 48–55.

[27] Fulton County Geographic Information Systems. 2019. The Aerial Imagery Download Tool. Retrieved February 05, 2020 from https://gis.fultoncountyga.gov/apps/AerialDownloadMapViewer/

[28] Hennepin County Geographic Information Systems. 2015. Hennepin County Aerial Imagery. Retrieved December 27, 2020 from https://gis.fultoncountyga.gov/apps/AerialDownloadMapViewer/

[29] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1–9.

[30] Monica G Turner and F Stuart Chapin. 2005. Causes and consequences of spatial heterogeneity in ecosystem function. In *Ecosystem function in heterogeneous landscapes*. Springer, 9–30.

[31] Clifford H Wagner. 1982. Simpson's paradox in real life. *The American Statistician* 36, 1 (1982), 46–48.

[32] Curtis E Woodcock, Richard Allen, Martha Anderson, Alan Belward, Robert Bindschadler, Warren Cohen, Feng Gao, Samuel N Goward, Dennis Helder, Eileen Helmer, et al. 2008. Free access to Landsat imagery. *Science* 320, 5879 (2008), 1011–1011.

[33] Yiqun Xie, Han Bao, Shashi Shekhar, and Joseph Knight. 2018. A TIMBER Framework for Mining Urban Tree Inventories Using Remote Sensing Datasets. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1344–1349.

[34] Y Xie, J Cai, R Bhojwani, S Shekhar, and J Knight. 2019. A locally-constrained YOLO framework for detecting small and densely-distributed building footprints. *International Journal of Geographical Information Science* (2019), 1–25.

[35] Yiqun Xie, Jiannan Cai, Rahul Bhojwani, Shashi Shekhar, and Joseph Knight. 2020. A locally-constrained yolo framework for detecting small and densely-distributed building footprints. *International Journal of Geographical Information Science* 34, 4 (2020), 777–801.

[36] Yiqun Xie, Jayant Gupta, Yan Li, and Shashi Shekhar. 2018. Transforming Smart Cities with Spatial Computing. In *2018 IEEE International Smart Cities Conference (ISC2)*. IEEE, 1–9.

[37] Yiqun Xie, Shashi Shekhar, Richard Feiock, and Joseph Knight. 2019. Revolutionizing tree management via intelligent spatial techniques. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. 71–74.

[38] Xiao Xiang Zhu, Devis Tuia, Lichao Mou, Gui-Song Xia, Liangpei Zhang, Feng Xu, and Friedrich Fraundorfer. 2017. Deep learning in remote sensing: A comprehensive review and list of resources. *IEEE Geoscience and Remote Sensing Magazine* 5, 4 (2017), 8–36.

A AERIAL IMAGERY

Unlike the related work [17, 38], which is based on satellite imagery with one-meter order resolution, we use higher resolution (7.5cm) aerial imagery taken at the start of spring season. Figure 15 shows the types of imagery that we considered, their resolution, season, time of the day, spectral bands and a visual example. It shows 5 types of imagery namely, Landsat [32], National Agriculture Imagery Program (NAIP) [19], areal imagery [27, 28], LIDAR point cloud [26], and Google Earth imagery [7]. As shown, areal imagery taken in the beginning of the spring season is better to detect urban objects due to lower occlusion from leaves and snow cover. In addition, higher resolution results in sharper images that allow better detection of distinctive features. For this work, we have not used LIDAR data for training, as it may add sensor based variability in addition to the geographic variability. Further, satellite based imagery and the available Google Earth were not used due to seasonal cover and relatively low resolution respectively.

Imagery type	Landsat	NAIP	Areal Imagery	LiDAR	Google Earth
Spatial resolution	15m / 30m / 100m	1m	7.94cm	0.5m – 1m	15cm - 15m
Season (Time of the year)	Near-global seasonal coverage	Agricultural growing seasons	Spring	Depends on study time	Near-global seasonal coverage
Time of the day	Day	Day	Day	Day/Night	Day
Spectral bands	Panchromatic / Visible (RGB), short wave infrared, near infrared / Thermal infrared	Visible (RGB), near infrared (for some states)	Visible (RGB)	Laser based 3D imaging	Visible (RGB)

Figure 15: Imagery types

B YOU ONLY LOOK ONCE (YOLO) FOR OBJECT DETECTION

SVANN uses YOLO deep learning framework to train multiple models. Here, we briefly describe its architecture, object detection procedure, and a subset of relevant parameters.

Architecture [22]: The framework has 24 convolution layers for feature engineering and selection, and two fully connected layers for prediction. The framework uses 3×3 filters to extract features and 1×1 filters to reduce output channels. The prediction layer has two fully-connected layers that perform linear regression on the final two layers to make the boundary box predictions to detect objects. The first layer flattens the 3-dimensional vector

output from the convolution layer to a single dimension 4096 vector. The final layer converts the 1D vector to a 3D vector with the detected values.

The input to the framework is a 448×448 dimension image, which is a product of a prime number (7) and multiple of 2. This allows the reduction of the dimensions by 2 across the convolution and pooling layers. The reduction is also affected by stride, that is, the step size to move the convolution matrix. The dimension reduces by half whenever pooling (e.g., Maxpool) and convolution layer with stride 2 is used. The channels increase with the dimension in the convolution filters.

Object detection in YOLO: Figure 16 shows the object detection sequence in YOLO [22] using one of our example images. The first step is to break the image into an SxS grid. Then, B bounding boxes are predicted for each grid cell. B is a parameter provided to the framework using the value suggested in the YOLO paper. Each bounding box is represented by 4 values (x, y, w, h) and a confidence score. All the 5 values (x, y, w, h, and confidence score) are predicted by the fully connected layers. The (x, y) coordinates represent the center of the bounding box, w represents the width, and h represents the height of the box relative to the complete image. The confidence score reflects the accuracy of the detected object. It is the product of the probability of the object and the Intersection over Union (IoU), where IoU is the ratio of the ground-truth and prediction intersection over their union (Figure 16(b)). The box with the highest IoU is selected.

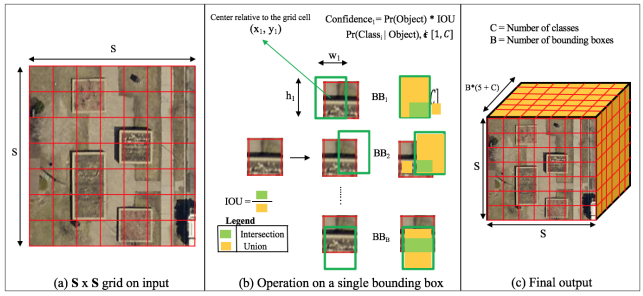


Figure 16: Object detection in YOLO (Best in color)

Framework parameters affect the training time of the model and the number of predictions from the prediction layer. Each iteration of the gradient descent updates all the weights in the layers, which is time consuming. Further, after a certain number of iterations there is no significant change in precision and recall values. Besides iteration, the number of partitions for detection and IoU values affects the number of predictions. A lower number of partitions may result in limited detection in high density partitions. Further, higher IoU values may result in limited detection of small objects, whereas lower IoU values can increase the number of False Positives in the detection process.

C DATASET DEVELOPMENT

C.1 Conversion to high spatial resolution aerial imagery

The key challenge was inconsistency in the naming, such as, use of informal names (e.g., 3437 Garden) that could not be mapped to the latitude and longitude. Thus, we visually looked up the objects in Google Earth and through the nearest road intersection identified the formal location using Google Maps (3437 S 15th Ave Minneapolis, MN 55407). The locations were Geocoded in ArcGIS and overlayed on the high spatial resolution aerial imagery to extract improved object of interest samples. Figure 17 shows the annotated input, underlying object of interest, conversion process, and the object of interest in the high spatial resolution imagery.

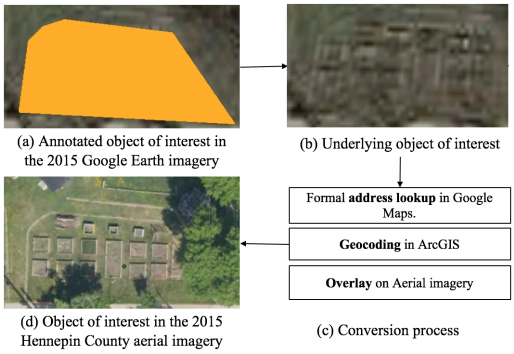


Figure 17: Conversion of existing annotations.

C.2 Manual annotations from high spatial resolution aerial imagery

Figure 18 shows the image annotation sequence where the zoom level increases from left to right. The area of the object was annotated in a 4-step process: First, geo-tagging the object; Second, creating a rectangular buffer; Third, clipping the object; and Fourth, annotating the object using a bounding box. Annotations were done using BBox-Label-Tool [21], a python based annotation tool.

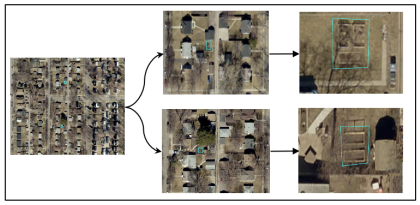


Figure 18: Image annotation sequence.