Graduate Theses, Dissertations, and Problem Reports

2021

Ensemble Encoder-Decoder Models for Predicting Land Transformation

Pariya pourmohammadi
West Virginia University

Follow this and additional works at: https://researchrepository.wvu.edu/etd

Part of the Artificial Intelligence and Robotics Commons, Data Science Commons, Geographic Information Sciences Commons, Spatial Science Commons, and the Sustainability Commons

Recommended Citation
https://researchrepository.wvu.edu/etd/8251

This Thesis is protected by copyright and/or related rights. It has been brought to you by the The Research Repository @ WVU with permission from the rights-holder(s). You are free to use this Thesis in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you must obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/or on the work itself. This Thesis has been accepted for inclusion in WVU Graduate Theses, Dissertations, and Problem Reports collection by an authorized administrator of The Research Repository @ WVU. For more information, please contact researchrepository@mail.wvu.edu.
2021

**Ensemble Encoder-Decoder Models for Predicting Land Transformation**

Pariya pourmohammadi

Follow this and additional works at: [https://researchrepository.wvu.edu/etd](https://researchrepository.wvu.edu/etd)

Part of the Artificial Intelligence and Robotics Commons, Data Science Commons, Geographic Information Sciences Commons, Spatial Science Commons, and the Sustainability Commons
Ensemble Encoder-Decoder Models for Predicting Land Transformation

Pariya Pourmohammadi

Thesis Submitted to
The Benjamin M. Statler College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Computer Science

Donald Adjeroh, Ph.D., Chair
Gianfranco Doretto, Ph.D.
Michael Strager, Ph.D.

Morgantown, West Virginia, USA
2021

Keywords: Convolutional Neural Networks, Evidence Fusion, Ensemble Models, Encoder-Decoder, Land Transformation Prediction
Copyright 2021 Pariya Pourmohammadi
Abstract

Ensemble Encoder-Decoder Ensemble Encoder-Decoder Models for Models for Predicting Land Transformations

Pariya Pourmohammadi

In studying dynamic and complex processes which are influenced by a system of inter-connected driving variables, it is crucial to apply models that can learn the complexity of the interactions. Land transformation is one of such complex processes, prediction of which can help to mitigate severe climate situations and improve the resiliency of communities. In this study, a multi-spectral set of data cubes is used to capture various characteristics of a geographic region. Based on the data cube, a feature space is constructed using socio-economic attributes, terrain characteristics, and landscape traits of the study region. Two-dimensional and three-dimensional convolutional neural networks (CNNs) and ensemble methods are then applied on this feature space to investigate model performance and improve the robustness of the models. This research demonstrates an application of evidence fusion at feature, decision and score levels, and applies ensemble random forests for post processing. Performance is assessed using the Dice coefficient, Receiver Operating Characteristic (ROC) curves, data visualization, and running time. This study shows that the implementation of ensemble models improved the accuracy of the models in terms of model stability, precision and recall, and dice coefficient.
Acknowledgements

I would like to thank my committee members: Dr. Michael Strager and Dr. Gianfranco Doretto, and my adviser Dr. Donald Adjeroh for his precious comments and support.
# Table of Contents

List of Tables v

List of Figures vi

1 Introduction 1
   1.1 Background and Motivation .................................. 1
   1.2 Research Objectives ........................................... 2
   1.3 Major Contributions ............................................ 2
   1.4 Organization of the Thesis .................................... 3

2 Related Work 5

3 Methods 9
   3.1 Database and Data Representation .............................. 9
   3.2 Model Structure .................................................. 15
       3.2.1 Architecture ................................................ 16
       3.2.2 Model Parameters .......................................... 20
   3.3 Evidence Fusion .................................................. 21
       3.3.1 Decision-level Fusion (DF) ................................ 22
       3.3.2 Score-level Fusion (SF) ..................................... 22
       3.3.3 Feature-level Fusion (FF) .................................. 23
   3.4 Performance Measurement ...................................... 24
       3.4.1 Area Under Receiver Operating Curve (AUROC) .......... 24
       3.4.2 Dice Coefficient ............................................ 25
       3.4.3 Time Complexity ............................................ 26

4 Results 27
   4.1 Classification Performance .................................... 27
   4.2 Computational Time Requirements ............................... 33

5 Conclusion and Future Work 35
   5.1 Conclusion ....................................................... 35
   5.2 Discussion and Future Work ................................... 36
List of Tables

3.1 Variables Used in the Analysis ........................................... 13
3.2 Summary Statistics of the Variables (The mode of the categories is
identified based on the number of the cells that are labeled per category) 14

4.1 Comparison of the model accuracy results ............................. 29
4.2 Comparison of the time required for the model runs ................. 34

5.1 Fusion results with and without 3D Models ............................ 36
List of Figures

3.1 Proposed framework for data representation, spatio-spectral data analysis, and evidence fusion for prediction of impervious/developed land expansion. .......................................................... 10
3.2 Study area (showing the region in the Appalachian Mountain range in the USA), and data representation using the spatio-temporal datacube (shown in the cut-out). .................................................. 11
3.3 2D versus 3D convolutions. Here 3D convolution assumes a spectral dimension of 3 for the kernel. The method of (2+1)D first applies 2D convolutions on each band, then 1D convolution is applied separately, along the spectral dimension in the spatio-spectral datacube. Lines with different colors show the contribution of different spatio-spectral bands in the 3D convolution. the figure is motivated by the approach used in Ji et al. (2012). .......................................................... 18
3.4 Proposed model structure and deep learning architecture for predicting impervious land expansion. Top illustration is the 3D model and bottom one is the 2D model with its layers as introduced in Pourmohammadi et al. (2019). The same layers are used in 3D model. .... 19
3.5 Sensitivity and specificity diagram ........................................... 26
4.1 Training Dice Coefficient ...................................................... 28
4.2 Validation Dice Coefficient .................................................. 29
4.4 Location of the patches ......................................................... 31
4.5 Model prediction in four patches; each patch column includes the labeled results. See Fig. 4.4 for patch locations. ................. 32
Chapter 1

Introduction

1.1 Background and Motivation

Land development has substantial impacts on different aspects of both the local and global environmental conditions. Land development can place a significant burden on the resources of a given region (Lin (2010)). The impact of developed and urbanized lands on the global ecosystem and on climate change has been widely discussed (Wilson & Chakraborty (2013)). In the past few years, the influence of urban footprint in the creation of urban heat island and the global temperature has been a ubiquitous discussion among environmental scientists. Furthermore, because of their population density, developed regions are vulnerable in the event of natural catastrophes and this is a matter of concern for risk management modelers (Ntelekos et al. (2010)). Accordingly, accurate models for projecting the complex process of land development (Pijanowski et al. (2009); Jin et al. (2013); Kamusoko & Gamba (2015)) are utilized in natural catastrophe risk modeling, and in management and climate change modeling (Ntelekos et al. (2010); Rizeei et al. (2018); Pijanowski et al. (2001)).

Improving prediction outcomes for such complex and dynamic processes requires models to learn the complexities of transformation process. Advancements in machine learning and deep learning have led to improvements in predictive mod-
els in predicting land transformation (Kamusoko & Gamba (2015); Pijanowski et al. (2002, 2014); Pourmohammadi et al. (2019)). Neural networks, as universal function approximators, have been applied in land change detection and prediction, land cover classification, and object detection (Pijanowski et al. (2001); Hu et al. (2015); Castelluccio et al. (2015)).

1.2 Research Objectives

This research aims to integrate multi-sources datasets and deploy a variation of techniques to conduct spatial analysis. In this work we use multi-level investigations on data fusion in different model combinations. The variables of this research are spatially explicit map layers representing different interpretations from the geographic and socio-economic features in a geographic region. Application of deep-neural networks, inter-channel interactions, evidence fusion, classification of model outputs are major issues that we aim to examine in this research. Different parts of this study are designed to incorporate complexities and dynamicity of land transformation, and bring in model robustness by generalization.

1.3 Major Contributions

In this research we use multi-spectral data cubes to model land transformation using both 2D and 3D convolutional neural networks (CNNs), then we incorporate evidence fusion and ensemble classification methods to establish stable and generalized models. Our results show how these techniques enhance the model performance
in predicting the land cover transformation. The results of previously proposed models of \textit{deepLandU}, \textit{deepLandS} and \textit{U-Net} (Pourmohammadi et al. (2019, 2020)) are used as the baseline models. The major contribution of this work lies in the significant improvement in model performance in terms of model robustness and accuracy using the proposed 3D convolutions, ensemble learning, and multi-level evidence fusion.

More specifically, the first contribution is the incorporation of a new dimension in CNN models for analyzing land transformations. The formulated models do not necessarily result in higher model performance, however, they provide a compelling ground for incorporating the inter-variable relationships in the models. One more implication of this contribution is that these models create a schema for the datasets with higher temporal dimensions which opens up new perspectives for further improvements in predictive models for land cover. Further, we show how structuring ensemble models based on evidence fusion at both the score and label levels improve the model results. This serves to balance the trade-off between model bias and variance. Thus, the second contribution of this work is the introduction of a method for improved robustness in land change prediction.

1.4 Organization of the Thesis

In Chapter 2 we discuss related work in the area of application of machine learning and deep convolutional neural networks (CNN) in study of land transformation and other methods that are used in improving model robustness. In Chapter
the data layers used in this study, model structure, evidence fusion, and applied
methods for model evaluation are presented. We further discuss the model results
in Chapter 4, followed by conclusion, discussion, and further works (Chapter 5)
Chapter 2

Related Work

In predicting land transformation and in land scene analysis there are a vast number of machine learning models developed. One of the popular neural network models used for land change prediction is Land Transformation Model (LTM) (Pijanowski et al. (2002)). LTM is a multi-layer perceptron (MLP) network which predicts a change likelihood value for each cell of the raster map. The neural network gets several inputs including the number of transformed cells called "principle index driver" (PID). To make predictions with LTM, first, PIDs should be computed based on population and then these indices should be passed as one of the inputs to the network (Pijanowski et al. (2002)).

In remote sensing, various architectures of neural networks models have been deployed to perform object detection, land cover classification, and semantic segmentation of the land (Kampffmeyer et al. (2016); Sherrah (2016)). Hu et al. (2015) and Castelluccio et al. (2015) applied several architectures such as AlexNet, VGG (from Visual Geometry Group), Caffe, and PlaceNet to label patches of land images and compare the accuracy of models. These efforts are good examples of the use of advanced, well-known deep learning designs in analyzing geographic data of the land. Kampffmeyer et al. (2016) and Sherrah (2016) used the Convolutional Neural Network (CNN) and the Fully Connected Network (FCN) to perform patch
based and pixel to pixel segmentation. They applied deep learning for segmentation of True Ortho Photos (TOP) images into 6 classes, namely, Impervious surfaces, Building, Low vegetation, Tree, Car, and Clutter/background Kampffmeyer et al. (2016).

CNNs are a group of deep models in which an order of increasingly complex features is generated. These groups of features are basically the outcome of sequences of trainable filters and poolings in a defined window. In CNNs pre-defined kernels are used for convolving the features; pre-defined windows are also used for poolings (Ji et al. (2012)). The past few years have seen rapid developments in deep CNNs, with various applications in different disciplines (Badrinarayanan et al. (2017); Ji et al. (2012)). These methods have become very popular in image analysis and classification. One of the major advances in this area was a large, deep convolutional neural network (AlexNet) proposed by Krizhevsky et. al. which was able to classify large dataset of high resolution images using Graphics Processing Unit (GPU) and supervised learning (Krizhevsky et al. (2012)). These models were capable of conducting classifications using large training datasets and computational facilities of GPU. The dominance of accuracy of these models in classification problems have led to the growing application of CNNs.

Deep CNNs have also been widely applied in many areas other than computer vision, for instance, in genomics sequence studies, speech recognition, geoscience, and earth science (Hinton et al. (2012); Pourmohammadi et al. (2020); Ronneberger et al. (2015); Yu et al. (2020)). In remote sensing, various architectures of neural networks models have been deployed to perform object detection, land cover classifi-
cation, and semantic segmentation of the land (Kampffmeyer et al. (2016); Sherrah (2016)). U-Net, ResNet, Alexnet are some of the popular models that have been used for land classification (Zhou et al. (2019); Huang et al. (2018)).

Application of CNN models in predicting the land transformation is studied in (Pourmohammadi et al. (2019, 2020)), where 2D CNN models of deepLandU and deepLandS are proposed to predict land development. deepLandU and deepLandS (Pourmohammadi et al. (2020, 2019)) are shallow 2D networks, the first two convolutional layers of these models include 64 filters followed by average pooling, and convolutional layers in the second and third part include 128 and 256 filters, respectively. The convolving filters that were used in the first two convolutional layers were $5 \times 5$ kernels. In the first pooling, these networks applied average-pooling, while the remaining pooling layers were based on max pooling. These models used a feature space that included policy type, distance and density, socio-economic, and landscape related variables.

2D CNN is the prevalent model in most image-based applications of CNN. Yu et al. (2020) used 2D convolutional layers to extract features in the spatial domain and constructed a non-linear architecture where inter-spectral relationships are fed into the model. They used their proposed model to classify hyperspectral satellite image datasets. They further broadened the scope of their work by introducing 3D neural network where the 3rd dimension was deployed to ingrate inter-channel relationships. Aside from the data manipulation, for some applications, 3D CNNs have shown improvements in model output. For instance, Ji et al. (Ji et al. (2012)) proposed an application of 3D CNNs to extract spatio-temporal features from multiple
frames. They expanded the structure of their proposed network by exploring the model performance on variations of 3D networks. Tran et al. (2018) trained their so-called C3D CNNs for the purpose of action recognition. They studied variation of convolutional kernel cubes and showed that a CNN of $3 \times 3 \times 3$ kernels outperformed the other models in their experiments.
Chapter 3

Methods

The methods of this research are developed so the proposed models intake the data from historical state of land in small patches of datacube, and the output of the model is the binary class of land cover in each patch. The class of interest is developed land. Figure 3.1 presents a general schematic diagram for the research methods. This framework first captures how various types of information about the land cover is represented in a spatio-temporal data cube. Then, the data cube is passed to a deep learning framework for classification at the pixel level where the classification result implies whether each cell will go through land development. The results are further combined using various fusion techniques. In this section we described the data arrangement method in Section 3.1, in Section 3.2 model structure of this study are explained, we further discuss the evidence fusion and performance measurement metrics in Sections 3.3 and 3.4.

3.1 Database and Data Representation

In this study, we conducted data manipulation process to resemble hyperspectral images using Convolutional Neural Networks (CNNs). CNN has proven to perform well in the computer vision field and suits tasks such as classification, object detection and object localization (Badrinarayanan et al. (2017)). The explanatory
Figure 3.1: Proposed framework for data representation, spatio-spectral data analysis, and evidence fusion for prediction of impervious/developed land expansion.

variables are depicted and ordered to assimilate hyperspectral images.

The study area for this research is the Monongahela watershed (HUC6-050200) that is located along the ridge of the Appalachian Mountains in the United States (See Fig. 3.2). The data in this research is structured in such a way that each land characteristic is considered as a different image channel. We used multiple representations of landscape characteristics, geopolitical boundaries and demographic traits in the feature class. To make a proper representation of these data, the raster maps are formatted, geo-referenced, co-registered as a multi-spectral data cube. Accordingly, a cross-section of data cubes, each with a dimension of $32 \times 32 \times 71^1$, are generated from the study area (see Fig. 3.2), and any of these patches is then considered as one data sample. In total there were 21,785 patches located in the boundary of the study area. We partitioned them as 66.6% as training and the rest as test.

$^1$The patch size for this study is $32 \times 32$, and after changing the categorical variables to binary dummy variables, the dimensions reaches to 71
and validation data. The model was trained on binary data of developed versus non-developed regions using the National Land Cover Dataset (NLCD) of 2001 and on a set of variables; the test data is the National Land Cover Dataset of 2011. The class of developed land in this study is defined based on low, medium and high intensity classes of Anderson classes Anderson (1976), which are used in classifications of NLCD. These classes of land have more than 20% of imperviousness.

The target variable is a binary variable indicating whether land development has occurred between 2001 to 2011 in each cell. For training the models, we use the National Land Cover Dataset of 2001 (Jin et al. (2013)) and 2011 (Homer et al. (2015)). We randomly partitioned the data into training and test; 66% of the patches are used for training and the remaining 33% are used as test data. GIS
raster data and Python scripts are used for the data processing and building the models. Keras library, which is a high-level neural networks API that is running on top of TensorFlow is used for building the models.

We identify the explanatory variables by considering the landscape characteristics of the study area and the key influential factors noted in other related literature (Verburg (2006); Pijanowski et al. (2009); Wu et al. (2009)). After geo-referencing and co-registering raster layers were created at a spatial resolution of $30m \times 30m$. Table 3.1 provides a description of all the variables we considered in this study.

In Table 3.2 we include summary statistics of the variables and information on the nature and data type of the variables. This table indicates that the values of each variable vary according to the nature of the variable. Hence, we standardize the continuous variables in the range of 0 and 1 to improve neural network stability and modeling performance. For the categorical variables, we introduce dummy variables for each category. We also rasterized the vector variables to $30m \times 30m$ raster data and geo-referenced and co-registered them.

Spectral cubes of map layers for data manipulation are adopted. In hyperspectral imaging and remote sensing, each image can represent different bands each of which captures specific wavelength ranges and all of them point to the same geographic area. The different wavelength representations can be viewed as spectral cubes for each object. If different portrayals of a geographic region are considered as different variables, stacking the variables on top of each other resembles the idea of hyperspectral images. This creates a data cube of each patch of data.

The advantage of using such data structure is its convenience for applying
Table 3.1: Variables Used in the Analysis

<table>
<thead>
<tr>
<th>ID</th>
<th>Data Layers</th>
<th>Data Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Distance to crop lands</td>
<td>National Land Cover Dataset (NLCD) 2001</td>
<td>Distance of each cell from crop lands</td>
</tr>
<tr>
<td>2</td>
<td>Distance to streams</td>
<td>Flow lines of the National Hydrograph Database (NHD)</td>
<td>Distance of each cell from stream</td>
</tr>
<tr>
<td>3</td>
<td>Distance to recreational sites</td>
<td>Boundaries of National Recreation Area from the United States Department of Agriculture datasets</td>
<td>Distance of each cell from recreational sites</td>
</tr>
<tr>
<td>4</td>
<td>Distance to developed lands</td>
<td>NLCD 2001</td>
<td>Distance of each cell from developed lands</td>
</tr>
<tr>
<td>5</td>
<td>Distance to main road and highway</td>
<td>U.S. Major Roads from ESRI Data</td>
<td>Distance of each cell from main road and highway</td>
</tr>
<tr>
<td>6</td>
<td>Distance to pasture/hay</td>
<td>NLCD 2001</td>
<td>Distance of each cell from pasture/hay</td>
</tr>
<tr>
<td>7</td>
<td>Distance to coal mines</td>
<td>U.S. Geological Survey Dataset</td>
<td>Distance of each cell from mining points</td>
</tr>
<tr>
<td>8</td>
<td>Distance to financial district of cities</td>
<td>Place Data of Open Street Map</td>
<td>Distance of each cell from point data of the important buildings</td>
</tr>
<tr>
<td>9</td>
<td>Distance to urban areas</td>
<td>Boundary of Urbanized Areas in US Census data (2000)</td>
<td>Distance of each cell from census urban areas</td>
</tr>
<tr>
<td>10</td>
<td>Density of oil and gas wells</td>
<td>National Oil and Gas Assessment (NOGA)</td>
<td>Density of gas within 10 Km of each cell</td>
</tr>
<tr>
<td>11</td>
<td>Density of developed lands</td>
<td>NLCD 2001</td>
<td>Density of developed lands within 10 Km of each cell</td>
</tr>
<tr>
<td>12</td>
<td>Density of crop land</td>
<td>NLCD 2001</td>
<td>Density of crop lands within 10 Km of each cell</td>
</tr>
<tr>
<td>13</td>
<td>Federal lands</td>
<td>Federal lands shape file provided by data.gov</td>
<td>Whether the cell is in a federal land or not</td>
</tr>
<tr>
<td>14</td>
<td>Elevation</td>
<td>Digital Elevation Model (DEM)</td>
<td>Elevation of the center of the cell</td>
</tr>
<tr>
<td>15</td>
<td>Slope</td>
<td>Extracted from Digital Elevation Model (DEM)</td>
<td>Slope of the cell</td>
</tr>
<tr>
<td>16</td>
<td>Population in 2000</td>
<td>Census data of the US</td>
<td>Population in year 2000 (closest to the base year)</td>
</tr>
<tr>
<td>17</td>
<td>Aspect</td>
<td>Extracted from Digital Elevation Model (DEM)</td>
<td>The aspect of each cell</td>
</tr>
<tr>
<td>18</td>
<td>Majority of land cover within 100m</td>
<td>NLCD 2001</td>
<td>Major class of land cover within 100m of each cell</td>
</tr>
<tr>
<td>19</td>
<td>Majority of land cover within 1000m</td>
<td>NLCD 2001</td>
<td>Major class of land cover within 1000 m of each cell</td>
</tr>
<tr>
<td>20</td>
<td>Have development plan or not</td>
<td>Data collected from the website of the municipality of each urban area in the region</td>
<td>Does the metropolitan area have developed plan or not</td>
</tr>
<tr>
<td>21</td>
<td>Urban Clustered</td>
<td>Census data and TIGER shapefile of Census data of the US</td>
<td>Is the cell in Urban Clusters (UCs) of at least 2,500 and less than 50,000 people (Census 2000) or not</td>
</tr>
<tr>
<td>22</td>
<td>Urbanized Area</td>
<td>Census data and TIGER shapefile of Census data of the US</td>
<td>Is the cell in Urbanized Areas (UAs) of 50,000 or more people (Census 2000) or not</td>
</tr>
<tr>
<td>23</td>
<td>County</td>
<td>TIGER shapefile of Census data of the US</td>
<td>Which county is the cell located in</td>
</tr>
<tr>
<td>24</td>
<td>State</td>
<td>TIGER shapefile of Census data of the US</td>
<td>Which state is the cell located in</td>
</tr>
</tbody>
</table>
Table 3.2: Summary Statistics of the Variables (The mode of the categories is identified based on the number of the cells that are labeled per category)

<table>
<thead>
<tr>
<th>ID</th>
<th>Continuous Variable</th>
<th>Data Type</th>
<th>Min.</th>
<th>Max</th>
<th>Mean</th>
<th>Std. Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Distance to crop lands (m)</td>
<td>Raster</td>
<td>0</td>
<td>8438.32</td>
<td>606.665</td>
<td>960.673</td>
</tr>
<tr>
<td>2</td>
<td>Distance to streams (m)</td>
<td>Raster</td>
<td>0</td>
<td>3195.95</td>
<td>240.890</td>
<td>197.565</td>
</tr>
<tr>
<td>3</td>
<td>Distance to recreational sites (m)</td>
<td>Raster</td>
<td>0</td>
<td>157531</td>
<td>63558.0</td>
<td>41038.9</td>
</tr>
<tr>
<td>4</td>
<td>Distance to developed lands (m)</td>
<td>Raster</td>
<td>0</td>
<td>4085.4</td>
<td>318.153</td>
<td>390.936</td>
</tr>
<tr>
<td>5</td>
<td>Distance to main road and highway (m)</td>
<td>Raster</td>
<td>0</td>
<td>3392.12</td>
<td>242.825</td>
<td>272.647</td>
</tr>
<tr>
<td>6</td>
<td>Distance to pasture/hay (m)</td>
<td>Raster</td>
<td>0</td>
<td>8438.32</td>
<td>606.665</td>
<td>960.674</td>
</tr>
<tr>
<td>7</td>
<td>Distance to coal mines (m)</td>
<td>Raster</td>
<td>0</td>
<td>38820.5</td>
<td>9744.021</td>
<td>7015.692</td>
</tr>
<tr>
<td>8</td>
<td>Distance to financial district of cities (m)</td>
<td>Raster</td>
<td>0</td>
<td>8012.054</td>
<td>4215.213</td>
<td>1215.548</td>
</tr>
<tr>
<td>9</td>
<td>Distance to urban areas (m)</td>
<td>Raster</td>
<td>0</td>
<td>7352.261</td>
<td>1701.160</td>
<td>809.51</td>
</tr>
<tr>
<td>10</td>
<td>Kernel Density of oil and gas wells in 1km²</td>
<td>Raster</td>
<td>0</td>
<td>21.0955</td>
<td>1.874</td>
<td>2.794</td>
</tr>
<tr>
<td>11</td>
<td>Density of developed lands in 1km²</td>
<td>Raster</td>
<td>0</td>
<td>1</td>
<td>0.041</td>
<td>0.114</td>
</tr>
<tr>
<td>12</td>
<td>Density of crop land in 1km²</td>
<td>Raster</td>
<td>0</td>
<td>1</td>
<td>0.023</td>
<td>0.041</td>
</tr>
<tr>
<td>13</td>
<td>Elevation (m)</td>
<td>Raster</td>
<td>211.96</td>
<td>1482.32</td>
<td>569.248</td>
<td>246.879</td>
</tr>
<tr>
<td>14</td>
<td>Slope (degrees)</td>
<td>Raster</td>
<td>0</td>
<td>72.5415</td>
<td>11.789</td>
<td>7.669</td>
</tr>
<tr>
<td>15</td>
<td>Population Density in 2000</td>
<td>Vector</td>
<td>0</td>
<td>8.591</td>
<td>0.038</td>
<td>0.155</td>
</tr>
<tr>
<td>16</td>
<td>Aspect (degrees)</td>
<td>Raster</td>
<td>-1</td>
<td>359.793</td>
<td>181.660</td>
<td>102.522</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>Categorical Variable</th>
<th>Data Type</th>
<th>Num. of Categories</th>
<th>Mode</th>
<th>Num. of Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>Majority of land cover within 100m</td>
<td>Raster</td>
<td>7</td>
<td>Forested</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>Majority of land cover within 1000m</td>
<td>Raster</td>
<td>8</td>
<td>Forested</td>
<td>1</td>
</tr>
<tr>
<td>19</td>
<td>Have development plan or not</td>
<td>Vector(Rasterized)</td>
<td>2</td>
<td>Yes</td>
<td>1</td>
</tr>
<tr>
<td>20</td>
<td>Urban Clustered</td>
<td>Vector(Rasterized)</td>
<td>2</td>
<td>Non-Urbanized</td>
<td>1</td>
</tr>
<tr>
<td>21</td>
<td>Urbanized Area</td>
<td>Vector(Rasterized)</td>
<td>2</td>
<td>Non-Urbanized</td>
<td>1</td>
</tr>
<tr>
<td>22</td>
<td>Federal lands</td>
<td>Vector(Rasterized)</td>
<td>2</td>
<td>Non-Federal</td>
<td>1</td>
</tr>
<tr>
<td>23</td>
<td>State</td>
<td>Vector(Rasterized)</td>
<td>4</td>
<td>WV</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>County</td>
<td>Vector(Rasterized)</td>
<td>43</td>
<td>Randolph(WV)</td>
<td>1</td>
</tr>
<tr>
<td>25</td>
<td>NLCD 2001</td>
<td>Raster</td>
<td>15</td>
<td>Forested</td>
<td>1</td>
</tr>
<tr>
<td>26</td>
<td>NLCD 2011</td>
<td>Raster</td>
<td>15</td>
<td>Forested</td>
<td>1</td>
</tr>
</tbody>
</table>
convolutional kernels in deep learning models. These kernels are capable of leveraging the contextual information from adjacent layers of variables. Therefore, representation of data as data cubes facilitates dealing with the complexity and correcting the inter-variable correlation.

This region is a combination of rural and urban areas, Pittsburgh, PA and Morgantown, WV are metropolitan areas and are two major cities in this watershed. In a region with various forms of land development, the heterogeneity of land development processes is influenced by this variation, hence the complexity of the problem increases. In such situation the major driving factors of change act locally, the models that consider neighboring cells’ conditions would have more power over the local conditions. Besides the urban-rural interactions and configurations in the study area, dominance of non-developed regions (95.9% of region comprises non-developed lands in 2001 the base year of the study) creates a highly imbalanced data. In assessment of the model performance on class imbalanced data it is crucial to use metrics that do not lead to misleading conclusions. Considering this, we used different metrics to measure the performance, so the measures get less sensitive to the skewed domains.

3.2 Model Structure

We use the spatio-spectral datacubes, we also investigate the importance of 3D networks in fusing the output of the evidence fusion. Trainable image filters, local neighborhood operations, and subsampling operations are applied in an alternating
manner on the original raw input data, resulting in an increasingly complex hierarchy of feature maps. We use patches of size $32 \times 32$ because the model best responds to this patch size. The values at a cell (pixel location) in the output image denotes the probability value for the developed land at the given cell. The probability of 0.50 or higher represents class value of 1 (developed land), and lower values denote a class value of 0 (non-developed land).

### 3.2.1 Architecture

In 2D convolutions, a small rectangular or square window with weights is moved around the image. The window (also called convolution kernel) is centered around each point in the image, and the weights are applied to the corresponding values in the original image. The weighted values from each position within the window are now added to obtain the final value for the center position. Essentially, the value of each unit at position $(x, y)$ in the $j^{th}$ feature map (variable) in the $i^{th}$ layer, denoted as $v_{ij}^{xy}$ is computed based on Equation 3.1.

$$v_{ij}^{xy} = f(b_{ij} + \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} w_{ijm}^{pq} v_{(i-1)m}^{(x+p)(y+q)})$$

(3.1)

where function $f$ is the activation function that is applied at each convolutional layer, as explained in Section 3.2.2 this function is either $ReLU$ or $LeakyReLU$, $b_{ij}$ is the bias for this feature map, $m$ indexes over the set of feature maps in the $(i-1)^{th}$ layer at the position $(p, q)$ of the filter connected to the $k^{th}$ feature map, and $P_i$ and $Q_i$ are the height and width of the convolving filter. In 2D convolutions the filter
of weights moves around on each channel and adds up the values at each layer. Thus, 2D convolutions only account for spatial information in the image. Same 2D subsampling layers are also applied in 2D architectures, where we used poolings for reducing the resolution of feature maps in the encoder by average and max pooling over local neighborhoods on the feature maps, which increases invariance to distortion in the input. Our models are constructed by stacking layers of convolution, subsampling, and up sampling. Our models used training data in multiple iterations to learn the bias and filter weights, denoted as $b_{ij}$ and $w_{ijm}^{pq}$ in Equation 3.1.

For 3D convolutions, the window is now extended to three dimensions (typically, using time for the third dimension). This small 3D window is now moved around positions in the images, and the weights are applied as before, with final results as the sum of the weighted values within the 3D window. The 3D convolution captures the cross channel relationships so the convolution layers connect multiple cells from neighboring geographic areas and different feature spaces from previous layer. To compute the value at position $(x, y, z)$ in the $i^{th}$ layer and $j^{th}$ we used Equation 3.2, in which $R_i$ is the size of the 3D filter. $w_{ijm}^{pqr}$ is the $(p, q, r)^{th}$ value of the weights filter at the $m^{th}$ feature map in the previous layer.

$$v_{ij}^{xyz} = f(b_{ij} + \sum_m \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} \sum_{r=0}^{R_i-1} w_{ijm}^{pqr} v_{(i-1)m}^{(x+p)(y+q)(z+r)})$$ (3.2)

Fig. 3.3 shows the significant differences between 2D and 3D convolutions. In either case, different results can be obtained by simply changing the weights, window size, window shape, or how much overlap is allowed when the window is
Figure 3.3: 2D versus 3D convolutions. Here 3D convolution assumes a spectral dimension of 3 for the kernel. The method of (2+1)D first applies 2D convolutions on each band, then 1D convolution is applied separately, along the spectral dimension in the spatio-spectral datacube. Lines with different colors show the contribution of different spatio-spectral bands in the 3D convolution. The figure is motivated by the approach used in Ji et al. (2012).

Tran et al. (2018) used an intermediate approach between 2D and full 3D convolutions. They first performed 2D convolutions on each frame. Then, for each position in the resulting 2D image, they formed a temporal sequence, and then applied 1D convolution on this temporal sequence. They showed that this approach (called (2+1)D) resulted in a better performance than using full 3D convolutions. In this work, the temporal dimension used in video analysis is replaced with the spectral dimension in the data cube. Thus, the single “spectral” bands in the data cube will be analogous to individual frames in a video sequence. The 3D convolutions will thus capture the spectral changes between bands in the spatio-spectral data cube.

The dimension of the convolutional layers and poolings is expanded along the third dimension. By applying cubic convolutional filters the cross-variable relation-
ships can be captured. Figure 3.4 shows the network architecture for the proposed deep learning models using 3D convolutions. The order of the layers for the 3D CNN were similar to deepLandU, however, instead of 2D convolutions and poolings I used 3D ones. The kernel size in the full 3D model was $3 \times 3 \times 3$ with stride of one. Except for the first convolutional layer that had ReLU activation, in other convolutional layers we used LeakyReLU activation and the activation function of last layer was softmax. After encoding the input cubes we transposed and upsampled the feature map back to $32 \times 32$. The kernel size of the convolutional layer in this model convolves across three of the channels. A variation of 3D CNN kernels was also applied, where we broke the 3D kernels into a (2+1)D kernels. In (2+1)D CNN instead of applying full 3D kernels with weights we used a combination of 2D and 1D kernels. This breakdown considerably reduces the number of weights that need to be fine-tuned, and hence the computations required.

![Figure 3.4: Proposed model structure and deep learning architecture for predicting impervious land expansion. Top illustration is the 3D model and bottom one is the 2D model with its layers as introduced in Pourmohammadi et al. (2019). The same layers are used in 3D model.](image)
3.2.2 Model Parameters

In the proposed 3D models, just like 2D models of deepLandU and deepLandS (see Pourmohammadi et al. (2019)), we use dropout at the final convolutional layer of the encoder, which accelerates the training and prevents over-fitting. The decoder has more convolutional layers with more upsamplings. In all the layers except the first two layers, LeakyReLU, which is a version of rectified linear unit (ReLU) function, is used. The first two layers use ReLU as activation function.

**Activation functions:** In the applied models we use two activation functions of LeakyReLU which is applied after each convolutional layer, and softmax that is used at the last layer of each model for computing the class probability. LeakyReLU minimizes the effect of the class-imbalance data (See Equation 3.3).

\[
    f(x) = \begin{cases} 
    x, & \text{if } x \geq 0 \\
    \alpha x, & \text{if } x < 0 
    \end{cases} 
\]  
(3.3)

where \( x \) is the value of the input for the function from the convolutional layer and \( \alpha \) is a coefficient for negative values of \( x \) (we use a value of 0.05 for \( \alpha \)). \( f(x) \) is the output of the activation function which is passed to the next layer. At the last layer, softmax function (Equation 3.4) is used as the activation function. Since the output in binary softmax returns a probability value in the range of \((0,1)\) per cell.

\[
    \sigma(\tilde{z}_i) = \frac{e^{\tilde{z}_i}}{\sum_{j=1}^{K} e^{\tilde{z}_j}} 
\]  
(3.4)

where \( \sigma(\tilde{z}_i) \) is the probability of the \( i^{th} \) cell to be a hit, and cells with values of
higher than 0.5 are labeled as developed lands, \( \vec{z}_i \) is the vector of model output from the last convolutional layer, and \( e^{\vec{z}_i} \) is the exponential function of this vector, and \( e^{\vec{z}_j} \) is the exponential function of the output vector for \( K \) classes, where in the study we have only two classes, namely, developed land versus non developed land.

**Optimizer:** We use the adaptive moment estimation (Adam) optimizer, which calculates the adaptive learning rates for each parameter (Ruder (2016)).

**Loss function:** The loss function for this model was binary cross entropy which computes the loss-value based on equation (3.5).

\[
L(y, \hat{y}) = -\frac{1}{N} \sum_{i=1}^{N} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]
\]  

where \( \hat{y}_i \) is the estimated value per example, \( \hat{y} \) is the true value, and \( N \) is the number of samples.

3.3 Evidence Fusion

Training deep networks requires investment of time and resources, yet there is no guarantee that the generalized error is low for the test samples Hinton et al. (2015). Ensemble models use the outcomes coming from different networks and select the best. Training multiple networks carries higher computational and time expenses, nevertheless, the combination of results add to the bias which balances the variance of only one trained network Hinton et al. (2015). Through combining the predictions from multiple models, ensembles reduce the variance and generalization error of predictions. In this study we used evidence fusion methods at decision,
feature and score levels to avoid over-fitting drawbacks of applying single networks and improve the model performance.

3.3.1 Decision-level Fusion (DF)

We utilized decision level fusion using majority vote. This method is applied to identify the label of cell \( c \) at patch \( p \). The label of \( c_{ij} \) in row \( i \) and column \( j \) is the most frequent label from \( l = \{l_{ij}^1, l_{ij}^2, l_{ij}^3, ..., l_{ij}^m\} \), where \( l_{ij}^t \) is the label from the \( t \)-th model in row \( i \) and column \( j \), and \( l_{ij}^t \in \{0, 1\} \).

3.3.2 Score-level Fusion (SF)

We used two methods of averaging and classification for conducting SF.

**Averaging the Scores:** For cell \( x \) at patch \( k \) we implemented Eq. 3.6 to compute the score of \( x \).

\[
s_{ij} = \frac{\sum_{t=1}^{m} s_{ij}^t}{m}
\]

where, \( s_{ij} \) is the score of cell \( c \) at row \( i \) and column \( j \) of patch \( k \), \( s_{ij}^t \) is the output score of the \( t \)-th model at row \( i \) and column \( j \), \( m \) is the total number of models whose results are considered in this method.

**Learning-based Score-level fusion:** We used a random forest classifier to classify the output scores of \( m \) models. Random forest is capable of dealing with class imbalanced data, which is the case for the dataset. Random forest, deploys a random subset of features and looks for the most discriminative threshold, where these
thresholds are randomly drawn for each feature and the splitting rule sets the best of these randomly-generated thresholds. This causes a variance reduction for the model at the expense of higher bias. We used extremely randomized trees, which goes one step further than the regular random forest method and allows a high level of randomness in splitting the trees. This method requires adjustments in two parameters of the number of trees (estimators) in the forest and maximum depth of the trees. Large number of trees will be computationally expensive and at some threshold of number of trees and does not improve the results significantly. The number of trees represents the size of the random subsets for the features that are considered when splitting a node. The smaller this value the greater the variance reduction and bias increase. To find the best values for the number of estimators and trees we cross validated the results. Such that k-fold cross-validation is performed on sets of \( s = \{s^1, s^2, s^3, ..., s^m\} \), where \( s^i \) is a set the score results acquired from \( i \)-th model. And for set \( s \in \{s^1, s^2, s^3, ..., s^m\} \) the value of \( |s| \) is equal.

3.3.3 Feature-level Fusion (FF)

At the feature level data fusion a pairwise merge of weights is conducted before applying \textit{softmax} function at each epoch. A linear combination of fine-tuned weights is passed to the activation function at the last layer, we used a \textit{sum} operation of weights, such that for models \( m_1, m_2, m_3, ..., m_n \) the stacked weights is computed as suggested in Eq. 3.7.
\[ W = \sum_{i=1}^{n} w_i \]  

(3.7)

where, \( w_i \) is the output weights of model \( i \), \( n \) is the number of feature-wise fused models and \( W \) is the total weights feature that is passed to the softmax function. In this analysis the value of \( n \) is set to two. This procedure helps the models to essentially get the weights fine-tuned over a linear combination of weights.

3.4 Performance Measurement

To evaluate the model performance we used the methods that facilitate avoiding downsides of class imbalanced data. These methods moderate or remove the impact of True Negative (TN in Fig. 3.5) in the assessment. Moreover, the time complexity of CNNs model run, encouraged us to use time efficiency as a performance measurement metric.

3.4.1 Area Under Receiver Operating Curve (AUROC)

The AUROC shows the trade-off between true positive rate (TPR or recall) and false positive rate (FPR) across different decision thresholds (See Eq. 3.8 and Eq. 3.9).

\[ TPR = \frac{TP}{TP + FN} \]  

(3.8)

\[ FPR = \frac{FP}{FP + TN} \]  

(3.9)
AUROC is the probability that a classifier will rank a random positive instance higher than a negative one Fawcett (2006). This metric is used for assessing the performance of binary classification models. We used AUROC in Pourmohammadi et al. (2019) to measure the model performance, however, we noticed that the AUROC value can be too optimistic about the performance of the models. The reason is the highly imbalanced datasets used in this study, in which an excessive improvement in the number of false positives changes the false positive rate negligibly. This happens because small $TN$ value will produce very high $FPR$, which will reduce the impact of $FP$ on the $FPR$ Fawcett (2006). Still we are interested in considering AUROC to compare models, we also used precision and recall($TPR$) metrics in model evaluation (eq. 3.8, 3.10)

$$\text{precision} = \frac{TP}{TP + FP}$$  \hspace{1cm} (3.10)

### 3.4.2 Dice Coefficient

In semantic segmentation Dice similarity coefficient approximates the similarity of positive labeled cells to the positive ground truth 3.11. This similarity metric has been widely used in evaluation of image segmentation results. Since $TN$ regions do not impact the magnitude of Dice coefficient, it is robust to class imbalance (Fig. 3.5).

$$\text{Dice} = \frac{2 \times TP}{(TP + FN) + (TP + FP)}$$  \hspace{1cm} (3.11)
3.4.3 Time Complexity

A major drawback in application of deep learning is the time complexity for training the features. Run time of these models depends on the number of trainable parameters. In this research, we investigated the running time of the models as a performance measurement. To verify that the reported times are standardized we used NVIDIA Tesla K80 GPU nodes on Pittsburgh Supercomputing Center Bridges for all the reported times. Access to the GPUs was provided through the Extreme Science and Engineering Discovery Environment (XSEDE) virtual system Towns et al. (2014). However, due limitations in the access time to Bridges GPUs, used EC2 g3.16xlarge instances of Amazon Web Services (AWS) to run the models with high total time.
Chapter 4

Results

We used different methods to study the performance of different evidence fusion and ensemble models. These performance measurements can be presented as classification and computation time performance which are described in this chapter.

4.1 Classification Performance

The results of this study are based on 160 epochs per model. We used AUROC and dice coefficient in performance measurement of training. Figures 4.1 and 4.2 illustrate the dice coefficient value of training and validation. This figure shows that the training process of 2D fusion models have similar learning trend, and the training dice coefficient value of these models is lower than other models. These figures also show similarity in the dice values in 3D models. We tested the results of this similarity in learning and fine-tuning trends by creating a dice similarity confusion matrix of the model outputs.

Pairwise dice-coefficient of model outputs (excluding the score and feature fusion results) (Fig 4.3) shows that the results of DeepLand-U, deepLand-S, and 2D FF models confirm each other. On the other hand, the results of 3D models, U-net and 3D FF models have a low level of agreement with the other models. This confusion matrix brings a legitimate description on the bias-variance balance
Figure 4.1: Training Dice Coefficient

of the fusion models. It shows that since 2D models of deepLand-U and deepLand-S generate similar results, fusing these models at the feature level would still lead to similar results. On the other hand, U-Net and deepLand-U have lower similarity, and the fusion of these two models represents much higher accuracy. But, this is not the case for 3D and (2+1)D models which represent 73.59% dice similarity. Feature fusion of these two models in 160 epochs represents a lower dice coefficient value compared to independent implementation of these models. Table 4.1 indicates that the precision value has increased in 3D FF results but recall has decreased, which means that the false negative labels are higher in the fusion of 3D and (2+1)D models.

The results of SF indicate the highest dice similarity and precision values among all the results. After conducting 5-fold cross validation for the random forest models the RF classifier has been trained with 1000 estimators and 41 trees. This
model outperformed all other models in the study. The model results of SF-RF show that this model is capable of predicting both positive and negative classes at high accuracy and it has the lowest difference between precision and recall values.

Running time of the models show that the time expenses correlate to the

number of trainable parameters (see Table 4.2). Also, adding a dimension to the convolutional kernel window and poolings has considerable impacts on the required time for training the models. As mentioned, variation of 3D network requires less time for training, still this run time is much higher than time expenses of 2D CNNs model training.

An illustration of the results in four different patches (Fig. 4.4) is represented in Figure 4.5. These four patches show model prediction on the test dataset in four areas. We selected these patches such that they represent model prediction in geographic regions of different developed land density and urbanization level. The first Patch is located in Pittsburgh, PA metropolitan urban area, which according
Figure 4.4: Location of the patches

to US census data in 2010 has population of more than 2,300,000. Patch 2 is in Morgantown, WV metropolitan area, a college town with population of 137,251. Two other patches are selected from rural regions.

Figure 4.5 shows a visualization of model performance in predicting land development in both rural and urban areas. This figure shows that the performance of models for different forms of development in rural and urban areas varies. Visualized results of first five models shows that these models predict denser land development area than the ground truth and they were not capable of teasing out the state of adjacent cells. So, their prediction, mostly in urban areas, creates aggregated new developments. Visualization of 2D FF(1&2) shows that in patch 1 (P1)- more urbanized regions- the results are akin to the ground truth. However in patches 2, 3,
Figure 4.5: Model prediction in four patches; each patch column includes the labeled results. See Fig. 4.4 for patch locations.

and 4 the results of this model are not as promising as SF-RF model. SF-RF demonstrate promising results at different levels of development density. Visualization of
SF-RF in all patches exhibits similar results to the ground truth. Since the whole study area is a combination of different levels of land development with dominance of rural areas, the generalization of the model results in different types of regions becomes a critical consideration in overall model performance.

4.2 Computational Time Requirements

Computational efficiency, in terms of both memory and time, becomes a major consideration in developed land analysis which often include large datasets. To alleviate the computational challenges, we performed the experiments on high-performance computing facilities at the Pittsburgh Supercomputing Center and AWS EC2 machines, and observed how the required computational time changed with the execution of the proposed deep convolutional neural networks. We recorded the time for running each of the models on a dual processor PC with 3.47GH CPUs and 48GB RAM, and on the GPUs available on PSC. We used 4 NVIDIA Tesla K80 GPU nodes each of which holds two NVIDIA K80 GPU cards with 24GB of GPU memory (48GB/node). Table 4.2 shows a comparison of the time per step and time per each epoch, for each model.
Table 4.2: Comparison of the time required for the model runs

<table>
<thead>
<tr>
<th>Model</th>
<th>Time Per Step</th>
<th>Time Per Epoch</th>
<th>Number of Trainable Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>deepLandS</td>
<td>6ms</td>
<td>86s</td>
<td>17,630,913</td>
</tr>
<tr>
<td>deepLandU</td>
<td>4ms</td>
<td>64s</td>
<td>10,772,997</td>
</tr>
<tr>
<td>U-Net</td>
<td>5ms</td>
<td>65s</td>
<td>30,764,805</td>
</tr>
<tr>
<td>3D deepLand</td>
<td>42ms</td>
<td>606s</td>
<td>32,190,981</td>
</tr>
<tr>
<td>(2+1)D deepLand</td>
<td>28ms</td>
<td>398s</td>
<td>12,208,927</td>
</tr>
<tr>
<td>2D FF(1&amp;2)</td>
<td>9ms</td>
<td>123s</td>
<td>28,403,910</td>
</tr>
<tr>
<td>2D FF(2&amp;3)</td>
<td>11ms</td>
<td>164s</td>
<td>48,395,718</td>
</tr>
<tr>
<td>2D FF(1&amp;3)</td>
<td>12ms</td>
<td>171s</td>
<td>41,537,802</td>
</tr>
<tr>
<td>3D Fusion</td>
<td>69ms</td>
<td>988s</td>
<td>44,399,908</td>
</tr>
</tbody>
</table>
Chapter 5

Conclusion and Future Work

5.1 Conclusion

The results of this study imply that evidence fusion at decision and score levels improves the model performance. With feature-level fusion, the model output shows some improvement in the results after stacking the weights in some of the models. 3D-FF results did not show much of improvements in the output, this is in fact the result of the number of parameters that needs to be fine-tuned at this level of fusion. We suggest that more consideration on the learning rate parameter and number of epochs which could improve the output of these models. Due to time constraints for access to computational facilities we did not explore an extensive range of parameter values. This could be done as part of future work in this area. Cross channel relationships impact the output of the 3D CNN models. Hence, to some extent, the order of the channels could impact the outcome of the 3D CNN models. This question could be further investigated, for instance, by considering the different permutations of the feature space, and the accompanying data management issues this will raise. Thus, additional research is needed to better understand how the interaction between the spectral bands can affect the results. On the other hand, feature-level fusion of 2D CNNs indicate higher performance measurement on validation data compared to training. The results of this research also indicate that
Table 5.1: Fusion results with and without 3D Models

<table>
<thead>
<tr>
<th>Model</th>
<th>Dice</th>
</tr>
</thead>
<tbody>
<tr>
<td>With 3D Models</td>
<td></td>
</tr>
<tr>
<td>Averaging</td>
<td>0.7755</td>
</tr>
<tr>
<td>SF-RF</td>
<td>0.8070</td>
</tr>
<tr>
<td>Voting</td>
<td>0.7548</td>
</tr>
<tr>
<td>Without 3D Models</td>
<td></td>
</tr>
<tr>
<td>Averaging</td>
<td>0.7601</td>
</tr>
<tr>
<td>SF-RF</td>
<td>0.8013</td>
</tr>
<tr>
<td>Voting</td>
<td>0.7234</td>
</tr>
</tbody>
</table>

SF-RF model which is an ensemble model with ransom forest classifier applied to the score outputs has promising performance in different geographic regions with various forms of land development. This idea could be further elaborated by applying similar methods to other geographic regions and different classes of land cover.

5.2 Discussion and Future Work

Although 3D CNN models did not show very high accuracy on their own (see Table 4.1), still the results of these models can be investigated in terms of their role in casting good votes and balancing the scores using decision fusion or score fusion. We used the output of 2D models to conduct decision-level and score-level fusion (see Table 5.1). The results in this table imply that fusion with 3D models leads to a higher accuracy when compared to fusion results without the 3D models. This suggests that we can improve the overall results by incorporating the results from 3D models, even when using only 3D models independently may not always lead to the highest accuracy.

Numerous machine learning models have been developed to predict the complex and dynamic process of land development; however, highly complex processes of land development can barely be modeled to predict the future of land change.
In this work, we examined the applicability of pixel-wise classification-based image segmentation models for use in studying land change. Deep deconvolutional networks for predicting developed land expansion, namely deepLand, has been applied in this work.

We applied 2D and 3D CNN models and used the idea of representing different land characteristics in the form of a multidimensional multispectral data cube, where each spectrum (or data layer) corresponds to a different characteristic, and each cube captures data in different patch or location within the region under study. The results of each model has been integrated in different levels of evidence fusions. Performance of the experiments using the proposed methods, along with comparison of the prediction results show that the proposed method is able to learn the transition rules of the land change more accurately and is capable of establishing more robustness.

Overall, the findings of this research demonstrate how the structure of model along with application of computational facilities can lead to reduction in the temporal costs and enhancement in the accuracy of the model results. In this work we showed that some models might not be able to demonstrate high accuracy in the assigned tasks, however, when combined with other models they can contribute to overall accuracy of models.

The models that are developed for this research can be applied to other problems in different disciplines or in different geographic regions. We believe that this work can also be improved by integrating semantic fusion semantic fusion, i.e. attention networks models. Moreover, the learned weights of this research can be used
in transfer learning and the time complexity of such models can be studied in the future work.
Bibliography


Fawcett, T. 2006, Pattern recognition letters, 27, 861


Homer, C., Dewitz, J., Yang, L., et al. 2015, Photogrammetric Engineering & Remote Sensing, 81, 345


Ji, S., Xu, W., Yang, M., & Yu, K. 2012, IEEE transactions on pattern analysis and machine intelligence, 35, 221


Kamusoko, C., & Gamba, J. 2015, ISPRS International Journal of Geo-Information, 4, 447


Lin, G. C. 2010, Eurasian Geography and Economics, 51, 80


Verburg, P. H. 2006, Landscape Ecology, 21, 1171

Wilson, B., & Chakraborty, A. 2013, Sustainability, 5, 3302


Zhou, K., Ming, D., Lv, X., Fang, J., & Wang, M. 2019, Remote Sensing, 11, 1