Applied Deep Learning: Case Studies in Computer Vision and Natural Language Processing

Md Reshad Ul Hoque

Old Dominion University, mhoqu001@odu.edu

Follow this and additional works at: https://digitalcommons.odu.edu/ece_etds

Part of the Artificial Intelligence and Robotics Commons, Computer Engineering Commons, and the Electrical and Computer Engineering Commons

Recommended Citation


https://digitalcommons.odu.edu/ece_etds/242

This Dissertation is brought to you for free and open access by the Electrical & Computer Engineering at ODU Digital Commons. It has been accepted for inclusion in Electrical & Computer Engineering Theses & Dissertations by an authorized administrator of ODU Digital Commons. For more information, please contact digitalcommons@odu.edu.
APPLIED DEEP LEARNING: CASE STUDIES IN COMPUTER VISION
AND NATURAL LANGUAGE PROCESSING

by

Md Reshad Ul Hoque
B.Sc. in Electronics & Communication Engineering (ECE), 2010, Khulna University Of Engineering Technology (KUET), Khulna, Bangladesh
M.Sc. in Electrical Engineering (EE), 2016, Lamar University, Texas

A Dissertation Proposal Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

DEPARTMENT OF ELECTRICAL  COMPUTER ENGINEERING

OLD DOMINION UNIVERSITY
August 2022

Approved by:

Jiang Li, Ph.D. (Director)
Dimitrie Popescu, Ph.D. (Member)
Chung Hao Chen, Ph.D. (Member)
Jian Wu, Ph.D. (Member)
Deep learning has proved to be successful for many computer vision and natural language processing applications. In this dissertation, three studies have been conducted to show the efficacy of deep learning models for computer vision and natural language processing. In the first study, an efficient deep learning model was proposed for seagrass scar detection in multispectral images which produced robust, accurate scars mappings. In the second study, an arithmetic deep learning model was developed to fuse multi-spectral images collected at different times with different resolutions to generate high-resolution images for downstream tasks including change detection, object detection, and land cover classification. In addition, a super-resolution deep model was implemented to further enhance remote sensing images. In the third study, a deep learning-based framework was proposed for fact-checking on social media to spot fake scientific news. The framework leveraged deep learning, information retrieval, and natural language processing techniques to retrieve pertinent scholarly papers for given scientific news and evaluate the credibility of the news.
Dedicated to my family and to all my mentors through the past few years, all of whom made possible to get me where I stand today.
ACKNOWLEDGEMENTS

Thanks to everyone who contributed to this dissertation. I extend my thanks and gratitude to my advisors and committee members for their guidance and help through this process. A special recognition is given to my academic advisors Dr. Jiang Li and Dr. Jian Wu, who have immensely helped me during these years and have always supported me. I would also want to thank all my lab partners during the past few years, who have given me constant feedback and support during my studies.
TABLE OF CONTENTS

| LIST OF TABLES                           | vi   |
| LIST OF FIGURES                         | ix   |

Chapter

1. INTRODUCTION                           | 1   |
   1.1 BACKGROUND                         | 1   |
   1.2 PROPOSED WORK                      | 3   |
   1.3 ORGANIZATION OF THE DISSERTATION   | 4   |

2. PRELIMINARIES                         | 5   |
   2.1 DEEP LEARNING MODELS FOR COMPUTER VISION | 5   |
   2.2 DEEP LEARNING MODELS IN NATURAL LANGUAGE PROCESSING | 8   |

3. SEAGRASS SCARS DETECTION USING DEEP LEARNING | 15  |
   3.1 SEAGRASS PROPELLER SCARS DETECTION | 15  |
   3.2 DISCUSSION AND CONCLUSION          | 33  |
   3.3 RELATED PUBLICATIONS               | 35  |

4. REMOTE SENSING IMAGE SUPER-RESOLUTION AND AN ARITHMETIC DEEP MODEL FOR TEMPORAL REMOTE SENSING IMAGE FUSION | 36  |
   4.1 REMOTE SENSING IMAGE SUPER-RESOLUTION | 36  |
   4.2 TEMPORAL REMOTE SENSING IMAGE FUSION | 49  |
   4.3 RELATED PUBLICATIONS               | 74  |

5. FAKE SCIENTIFIC NEWS DETECTION        | 75  |
   5.1 BACKGROUNDS                       | 75  |
   5.2 DOMAIN KNOWLEDGE ENTITIES (DKES) EXTRACTION MODELS | 77  |
   5.3 SCIPEP: RECOMMENDING PERTINENT SCIENTIFIC PAPERS | 87  |
   5.4 CLAIMS EXTRACTION                 | 102 |
   5.5 EVIDENCE RETRIEVAL AND CLAIMS JUSTIFICATION | 112 |
   5.6 RELATED PUBLICATIONS              | 115 |

6. CONCLUSION AND FUTURE WORK           | 117 |

BIBLIOGRAPHY                             | 119 |
LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Image size of the eight testing images.</td>
<td>20</td>
</tr>
<tr>
<td>2. Quantitative evaluation: Deckle beach dataset (threshold .6).</td>
<td>29</td>
</tr>
<tr>
<td>3. Quantitative evaluation: Virginia coast dataset (threshold .6).</td>
<td>31</td>
</tr>
<tr>
<td>4. Time complexity comparison.</td>
<td>33</td>
</tr>
<tr>
<td>5. Dimension of testing data.</td>
<td>40</td>
</tr>
<tr>
<td>6. Evaluation metrics (average)</td>
<td>47</td>
</tr>
<tr>
<td>7. CPU time for deep learning based super resolution models.</td>
<td>47</td>
</tr>
<tr>
<td>8. Specifications of datasets used in the study.</td>
<td>59</td>
</tr>
<tr>
<td>9. Details of training and testing data.</td>
<td>60</td>
</tr>
<tr>
<td>10. Results of Experiment 1. ATPRK, ESRCNN and GANs are competing methods. All others are the proposed model with different backbones or different loss functions. Results are averages on 20 testing images and shown in the format of mean(std). Best results are shown in bold.</td>
<td>63</td>
</tr>
<tr>
<td>11. Results of Experiment 2 (Landsat-8 and Sentinel fusion) and Experiment 3 (Sentinel and NAIP fusion). Results are averages on 20 testing images.</td>
<td>65</td>
</tr>
<tr>
<td>12. Results of Experiment 4 (Transfer learning). All models are trained with images collected from Norfolk, VA and tested on images collected from Palm Jumeirah, Dubai. Results are averages on 45 testing images.</td>
<td>65</td>
</tr>
<tr>
<td>13. Statistical tests between UMSEh (best overall) and GAN (best competing model) in all experiments, where h = 1 indicates the difference is significant at the 95% significance interval. Experiment 1 is for the 4x resolution case.</td>
<td>68</td>
</tr>
<tr>
<td>14. A comparison of models. All results are before the rule-based filters. Triplets are (Precision, Recall, F1).</td>
<td>83</td>
</tr>
<tr>
<td>15. A comparison of models performance on searching for pertinent scientific news evidence papers.</td>
<td>100</td>
</tr>
<tr>
<td>16. Model Evaluation on claim dataset. We report the performance of the baseline rule-based model and deep LSTM-CRF model with our proposed deep learning models with and without contrastive learning.</td>
<td>109</td>
</tr>
<tr>
<td>17. Performance evaluation of the transformer based pre-trained models. We use different transformer based pre-trained models trained on Sci-fact dataset and evaluate them on Sci-fact validation dataset.</td>
<td>115</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Image representation in the matrix format.</td>
<td>6</td>
</tr>
<tr>
<td>2.</td>
<td>Convolution operation.</td>
<td>6</td>
</tr>
<tr>
<td>3.</td>
<td>Ensemble model.</td>
<td>7</td>
</tr>
<tr>
<td>4.</td>
<td>Seagrass are shown in (a), (b) shows the seagrass scars which are created by the boats propeller.</td>
<td>16</td>
</tr>
<tr>
<td>5.</td>
<td>Training image: (a) panchromatic image, (b) ground-truth image with scars mark.</td>
<td>19</td>
</tr>
<tr>
<td>6.</td>
<td>Testing images captured by the WV-2 imaging system.</td>
<td>19</td>
</tr>
<tr>
<td>7.</td>
<td>Testing images captured from WV-3 imaging system. 1st row: Panchromatic band. 2nd row: Ground truth.</td>
<td>20</td>
</tr>
<tr>
<td>8.</td>
<td>Flowchart of deep learning-based scar detection model.</td>
<td>21</td>
</tr>
<tr>
<td>9.</td>
<td>Block diagram for pan-sharpening.</td>
<td>22</td>
</tr>
<tr>
<td>10.</td>
<td>Multispectral image bands before and after pan-sharpening procedure. 1st row for Green, Yellow, Red and Red Edge bands before pan-sharpening. 2nd row for the corresponding bands after pan-sharpened by the GS method.</td>
<td>23</td>
</tr>
<tr>
<td>11.</td>
<td>(a) Pan-sharpened image, (b) Image after application of the laplacian filter, (c) Image after application of the laplacian and sobel filters.</td>
<td>23</td>
</tr>
<tr>
<td>12.</td>
<td>Patches extraction from pre-processed pan-sharpened image. A window across 4 bands of size 17 × 17 × 4 was applied to the pan-sharpened image to extract patches.</td>
<td>24</td>
</tr>
<tr>
<td>13.</td>
<td>CNN model for scars classification.</td>
<td>25</td>
</tr>
<tr>
<td>14.</td>
<td>Diagram of the proposed committee machine for seagrass scar detection.</td>
<td>26</td>
</tr>
<tr>
<td>15.</td>
<td>Seagrass scar detection results on Virginia Cost dataset by direct application of the model trained with the Deckle Beach dataset.</td>
<td>27</td>
</tr>
<tr>
<td>16.</td>
<td>Model classification maps after applying different thresholds. 1st, 2nd and 3rd column are the result of threshold .5, .6 and .7 respectively. 1st, 2nd and 3rd, 4th row showing the model classification map for the deckle beach testing image 1, 2, 3 and 4.</td>
<td>30</td>
</tr>
<tr>
<td>17.</td>
<td>Model classification maps after applying different thresholds. 1st, 2nd and 3rd column are the result of Threshold .5, .6 and .7 respectively. 1st, 2nd and 3rd, 4th row showing the model classification map for the deckle beach testing image 1, 2, 3 and 4.</td>
<td>32</td>
</tr>
<tr>
<td>18.</td>
<td>Committee machine algorithm efficiency plots. a. accuracy vs combined output b. F1 score vs number of models.</td>
<td>34</td>
</tr>
<tr>
<td>19.</td>
<td>Training image for CNN.</td>
<td>39</td>
</tr>
<tr>
<td>20.</td>
<td>GAN training image.</td>
<td>40</td>
</tr>
<tr>
<td>21.</td>
<td>CNN-1 based image super resolution</td>
<td>42</td>
</tr>
<tr>
<td>22.</td>
<td>CNN-2 based image super resolution</td>
<td>42</td>
</tr>
<tr>
<td>23.</td>
<td>CNN architecture for image super-resolution</td>
<td>42</td>
</tr>
<tr>
<td>24.</td>
<td>GAN architecture for image super-resolution</td>
<td>43</td>
</tr>
<tr>
<td>25.</td>
<td>GAN architecture</td>
<td>44</td>
</tr>
<tr>
<td>26.</td>
<td>Models performance when applied to the testing images. From left to right, the first rows (a-e) shows the high resolution images, 2nd row (f-j) shows bicubic output, 3rd (k-o) and 4th row (p-t) for CNN-1 and CNN-2 based super resolution and 5th, 6th rows are the generated images from GAN-1 and GAN-2 respectively.</td>
<td>46</td>
</tr>
</tbody>
</table>
27. Research objective of remote sensing image fusion. ........................................ 51
28. Diagram of the proposed deep learning fusion model. .................................... 53
29. Proposed fusion model with U-Net [1] backbone. In each of the convolutional feature maps, we subtract the low-resolution image’s features at \( t_1 \) from these extracted from the high-resolution image at \( t_1 \), and add these features computed from the low-resolution image at \( t_2 \) back into the feature maps to reconstruct the high-resolution image at \( t_2 \). ................................................................. 54
30. Proposed fusion model with HRNet [2] backbone. In the high-resolution stage, we perform the same subtraction and addition arithmetic operations as that in the feature maps of the U-Net backbone model. Adding the arithmetic operations to other resolution stages did not significantly improve performance. ......................................................... 56
31. Training image examples. Resolution: Landsat-8 (30m), Sentinel-2 (10m), NAIP (1m) and WV-2 (.46m). Large red boxes display zoomed in regions in the corresponding small boxes. Significant temporal changes observed. Image contrast enhanced for better display. ................................................................. 61
32. Image fusion results in Experiment 4 (Transfer learning) by different methods, where H2 is ground truth high-resolution image at the second time point. Image generated by ESRCNN contains noise and histograms of images by HMSE, HMSEh and GAN do not match that of the ground-truth image. ................................................................. 66
33. Visual inspection of fused images by different models. For each of the experiment results, the first row shows input images and fused results by different models. “Low-\( t_1 \)” and “Low-\( t_2 \)” are input images. “Ground truth” is the high-resolution image at \( t_2 \) to approximate. The second row shows the zoomed in region in the red box above. Results of the proposed model are from the best combination of backbone and loss function in each of the Experiments. Image contrast enhanced for better display. ................................................................. 66
34. Image fusion results with temporal changes. In the ground truth image (a), the zoomed in regions at \( t_1 \) and \( t_2 \) show changes captured by the Sentinel-2 satellite image where a cargo ship docked in the Norfolk port at \( t_1 \) and left at \( t_2 \). The ESRCNN failed to reflect the change in the fused image (c). ATPRK (b), GAN (d) and our models successfully captured this change in the fused images (d,e and f). UMSE: “U-Net + MSE”. HMSE: “HRNet + MSE”. ................................................................. 69
35. Visual comparison for high-frequency details of images fused by the U-Net and HRNet backbones (Experiment 1 with 6\( \times \)). The large red box contains the zoomed in region in the small box. ................................................................. 70
36. Image registration effect. With imperfect registration in experiment 3, the resolution difference of 10\( \times \) between Sentinel-2 and NAIP images are much more difficult to bridge (image in (a) is blurry). With the perfect registration in experiment 1, even larger resolution differences resulted in much sharper fused images (b-d). ................................................................. 72
37. Workflows of the fake scientific news detection. ........................................ 76
38. The general architecture of the entity extractors, including a non-sequential classifier and a sequential tagger. The results are merged and a rule-based filter is applied. 79
39. Architecture of Attn-Res-BiLSTM-ELMo model. ........................................ 81
40. The proposed architecture of the SciPEP system. ........................................ 92
41. Transformer based DKEs extractor. .......................................................... 95
42. A comparison of performances of DKE extraction models. Categories along x-axis are below. 
   Arg: agriculture; Astr: astronomy; Bio: biology; Chem: chemistry; CS: computer science; 
   Eng: engineering; ES: environmental science; Math: mathematics; Med: medical science; 
   MS: materials science.

43. Supervised contrastive learning on the claim extraction dataset. Given a claim (anchor), 
   positive samples (claim), and negative samples (non-claim), at first we covert text to 
   its vector representation (embeddings) using the pre-trained model. In embedding space, 
   contrastive loss push samples from the same class close and samples from different 
   classes further apart in embedding space. In supervised contrastive learning, class label 
   information is used to align the same class in the embedding space.

44. CNN-1D Working flows.

45. WC-LSTM Working flows.

46. Bert-Dense Working flows.

47. Effects of contrastive learning in embedding space.

48. Rationale model for evidence sentence identification from an abstract.

49. Claim justification model.

50. Fake scientific news detection interface.
Chapter 1

INTRODUCTION

This chapter provides a brief overview of the dissertation proposal along with background of deep learning and the proposed work in the dissertation. The basic outline of this Chapter is presented as follows. Background related to deep learning is offered in Section 1.1. Section 1.2 describes the proposed work and Section 1.3 summarizes the structure of the dissertation.

1.1 BACKGROUND

1.1.1 DEEP LEARNING EVOLUTION

Deep Learning is a branch of machine learning which imitates the human thinking process for prediction. Machine learning has a long evolution history consisting of three major waves of advancements:

- Deep learning — since 2006.

Algorithms developed during the first two waves were unpopular because of the hardware limitations of the computing systems. However, these algorithms undoubtedly helped advance the field of deep learning and some are still widely used today.

Cybernetics The earliest machine learning researches were based on the idea of biological learning — how a human brain learns. This era was started by Warrens and Walter in 1943. They created a computer model with a combination of algorithms and mathematics to mimic the thought process and called the model “Threshold logic” [3], which was the start of the modern neural networks.
However, this model was not learnable. In the late 1950s, Frank Rosenblatt, an American psychologist, developed the Perceptron model, which could learn its weights automatically [4]. His idea planted the seeds of bottom-up learning and is still widely recognized as the foundation of deep neural networks (DNN). Later, Bernard Widrow developed Adaptive Linear Element (ADALINE), which could adapt its weights based on weighted inputs during training [5].

**Connectionism**  The idea of artificial neural networks (ANNs) was introduced during the second wave. Programmable intelligent computational models, hidden layers, distributed representation, and parallel processing were developed during this period. The most notable discoveries were the back-propagation for optimization and multi-layer perceptron (MLP) networks [6]. Sepp Hochreiter and Jürgen Schmidhuber introduced long short-term memory (LSTM) model which is one of the key deep learning algorithms for natural language processing (NLP) and time-series [7]. During this wave of connectionism, models like LSTM, distributed representation, parallel processing, and back-propagation were key components of various advanced applications of deep learning. However, these models could not provide the expected results because lack of computational resources.

**Deep Learning**  The third wave emerged in 2006 with a breakthrough by Geoffrey Hinton. He developed deep belief networks (DBN) with a greedy layer-wise training strategy [8]. Later, other researchers adopted Hinton’s advancements to train different deep networks. The popularity of deep learning became skyrocketed after Krizhevsky et al. [9] won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012. They extracted meaningful features from images using convolutional neural networks (CNN) for classification. They reduced the error rate of classification by nearly half as compared to the winner of 2011. In 2014, Facebook developed DeepFace which uses deep learning to classify faces with 97.35% accuracy \(^1\). Another powerful deep learning model called generative adversarial networks (GAN) was introduced by Ian Goodfellow [10] in 2014. GAN model has two adversarial networks: the first one tries to create indistinguishable samples as compared to ground-truth, while the second network tried to distinguish the created samples from ground-truth data. With advancements of computing hardware including

---

\(^1\)https://www.jitbit.com/alexblog/260-facebook-is-terrifying/
Graphical Processing Unit (GPU) and Tensor Processing Unit (TPU), more complex deep learning algorithms are developed, which created many interesting and complex applications. For example, transformer models for vision applications [11], self-driving cars, 3D mapping of the brain (Connectome), molecular reaction for drug discovery, etc.

1.1.2 DEEP LEARNING IN COMPUTER VISION

Deep learning has become a useful tool in solving a variety of computer vision problems, such as object detection [12, 13], motion tracking ([14, 15]), action recognition ([16, 17]), human pose estimation ([18, 19]), semantic segmentation ([20, 21]), medical imaging [1], and remote sensing [22]. In traditional machine learning, we need to perform feature engineering to extract features from input images while in deep learning, a trained deep learning model extracts image features. This is an enormous advantage of using deep learning over traditional machine learning. Convolutional neural network (CNN) is the heart of modern computer vision models. In computer vision models, image is represented as a matrix and condensed as a 3D tensor to feed to CNN. CNN then performs a series of convolutional and pooling operations to extract the most relevant features and condense them into a concise representation for classification or regression by a few fully connected layers.

1.1.3 DEEP LEARNING IN NATURAL LANGUAGE PROCESSING

Natural language processing (NLP) is a machine learning tool that enables computing systems to understand and respond to texts or language in the way as human do. The study of NLP has been around for over 50 years. Among NLP algorithms, deep learning models show a lot of promises and are achieving state-of-the-art results in many challenging problems such as named-entity recognition (NER) [23], sentence classification [24], and machine translation [25]. Deep learning is revolutionizing NLP domain and the most successful achievement of deep learning in NLP is the dense vector representation of text in which each word is trained to be embedded as a dense numerical vector, and the semantic relationships among texts can be accurately measured by the similarities among these representation vectors [26]. Recently, a new embedding model named Transformers set new state of the arts and produced superb results for many applications that were unimaginable before [27].
1.2 PROPOSED WORK

This dissertation proposes several deep learning models for computer vision and natural language processing applications including

- An efficient deep learning model for seagrass scar detection in multispectral images where we demonstrated that our proposed model is computationally more efficient than our previous sparse coding model and can accurately detect seagrass scars [28].

- Remote sensing super resolution models and an arithmetic deep model operated in feature space to fuse multimodal, multitemporal remote sensing images as high-resolution images for downstream tasks including change detection, land cover classification applications etc [29, 30].

- A deep learning framework for fact-checking on social media to spot fake scientific news where a 2-stage retrieval system was proposed to rank candidate scientific articles for fact-checking, and the model was integrated as SciPEP to recommend pertinent scientific papers for a given scientific news [31, 32].

Among the above three major applications, other assisting deep learning modules were developed to enhance system’s performances. A benchmark dataset containing 100 pairs of scientific news articles selected from ScienceAlert.com and their associated research papers was created. We also developed fusion dataset for image enhancement.

1.3 ORGANIZATION OF THE DISSERTATION

The rest of this dissertation is organized as follows. In Chapter 2, we review background of the models used in the dissertation. Chapter 3 presents seagrass scar detection algorithm. Chapter 4 introduces the proposed deep learning model for multimodal, multi-temporary remote sensing image fusion. Chapter 5 presents deep learning-based algorithms for spotting scientific fake news. Finally, Chapter 6 concludes this dissertation and introduces future work.
Chapter 2

PRELIMINARIES

This chapter discusses the concepts of different models that have been utilized throughout the dissertation. Section 2.1 describes the computer vision deep learning models and their backgrounds. Background for fake scientific news detection are discussed in Section 2.2.

2.1 DEEP LEARNING MODELS FOR COMPUTER VISION

2.1.1 CONVOLUTIONAL NEURAL NETWORK

Convolutional Neural Networks (CNNs) are very popular and widely used in computer vision and have achieved many state-of-the-art results in visual applications, including image classification and image reconstruction. CNN models work well in extracting both simple patterns such as lines, gradients, circles, and complex ones such as eyes and faces from images. This property makes CNN a powerful model for computer vision applications. CNN is a feed-forward neural network with multiple convolutional layers. The key components of CNN are convolutional layers, pooling layers, and fully connected layers (FCL). The power of CNN comes from convolutional layers where learned kernels in these layers can extract meaningful features to perform accurate classification or regression [33]. This convolution operation preserves the spatial relationship between pixels during feature extraction and reduces the size of the image without losing essential information, which are critical for computer vision tasks.

Fig. 1 shows an image in the matrix format and Fig. 2 presents the convolution operation. After the convolutional operation, a non-linear activation function is typically applied to the resulted feature maps so that the network can establish non-linear relations between inputs and outputs. Commonly used activation functions include sigmoid, relu, and tanh. Among of these activation functions, relu activation function is the most popular one because of its accuracy [34], and it is defined in Eq. (15).
Fig. 1. Image representation in the matrix format.

Fig. 2. Convolution operation.
ReLU($x$) = \( \text{max}(0,x) \)  

The second important component of CNN is the pooling layer. This layer reduces the spatial size of the extracted feature maps, decreases computational complexity and redundancy by keeping only dominant features that are translation invariant. There are two types of pooling layers: max pooling and average pooling. Max pooling returns the maximum value from the pooling field, whereas average pooling returns the average of all the values inside the pooling field. Finally, there are typically a couple of FCL layers attached to convolutional layers in CNN for classification or regression.

### 2.1.2 ENSEMBLE MODEL

An ensemble model is a machine learning model that combines multiple deep learning or machine learning predictions into a single prediction which is more robust and accurate as compared to a single prediction. This combination through aggregating output from each model (Fig.3) offers
two advantages: reducing the model error and maintaining its generalization. Suppose there are \( M \) trained models where each of the trained models generates a prediction result \( P_{dm} \), the ensemble model \( P_{en} \) can be defined as:

\[
P_{en} = \frac{1}{M} \sum_{m=1}^{M} P_{dm}
\]

(2)

### 2.1.3 GENERATIVE ADVERSARIAL NETWORKS

Generative Adversarial Networks (GANs) have already made a tremendous success, which was introduced in 2014 by Ian J. Goodfellow [10]. GAN belongs to generative family because of its ability to produce or generate new content. GAN model has two parts: generator and discriminator. The generator produces synthetic examples (e.g., images) from random noise, whereas the discriminator is trained on the real dataset whose task is to distinguish the real data or synthetic data. The goal of the training is to make the generator capable of generating the synthetic data looks like the real data and make the generated data distribution close to the real data distribution. During the training, both the generator and discriminator try to improve their respective abilities, and the training process continues until the discriminator cannot tell the authentic examples from the synthesized examples with better than the 50% accuracy expected of chance.

Arthur Samuel developed the idea of two adversarial algorithms fighting each other. Based on this idea, Goodfellow developed the GAN algorithm. After that, many GAN models have been proposed for various applications. Nvidia developed StyleGAN that can generate high-resolution fictional head images by learning the style of the face like facial pose, freckles, and hair [35]. Microsoft developed a novel GAN that generates images and storyboards based on captions [36]. Researchers at Seoul-based developed a GAN-based tool that can map a person’s facial movements to any target headshot [37]. It was impossible to predict future events using only a few video frames, but now is possible due to GANs. DeepMind developed a novel GAN model (DVD-GAN) which can generate realistic videos by watching YouTube clips [38]. GANs are capable of more than generating images and video footage such as artworks [39], music [40], speech, medicine, etc. For a comprehensive survey relating to GAN, readers can refer to [41].

---

2.2 DEEP LEARNING MODELS IN NATURAL LANGUAGE PROCESSING

2.2.1 EMBEDDING

Unstructured text data contains information that is very important to make a decision. Machine learning models can use this unstructured text information as inputs to make a decision. But before that, we need to convert the text data from documents in a decimal form to create structured data. Embedding is a way to create structured data from text. This section offers popular embedding methods to represent the text.

**TF-IDF**  
TF-IDF (term frequency-inverse document frequency) is a statistical measure that evaluates the quality of a word. Normally, it is used to measure how relevant a word is to a document in a collection of documents. This is done by multiplying two metrics: term-frequency: how many times a word appears in a document, and inverse-term-frequency: the inverse document frequency of the word across a set of documents. Mathematically it can be defined as,

\[
\text{tfidf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)
\]

where

\[
\text{tf}(t, d) = \log(1 + \text{freq}(t, d)) \tag{4}
\]

\[
\text{idf}(t, D) = \log\left(\frac{N}{\text{count}(d \in D : t \in D)}\right) \tag{5}
\]

Here \(N\) indicates total number of documents in the corpus, \(tf\) is the relative term-frequency of term \(t\) within document \(d\), \(idf\) is the inverse-term-frequency. TF-IDF is a simple and popular word representation method which is often used for document searching and information retrieval problems [42]. It gives credit to a word that appears frequently in a document but gives offset by the number of documents that contain the word. Using this technique, we can exclude the common words such as “this”, “what”, “is”, “if”, etc. However, if a word appears many times in a document, while not appearing many times in others, it probably means that it’s very important. For example, style-GAN, is a very popular deep learning model for artwork. The document related
to style-GAN may contain the “style-GAN” word several times, but most other documents may not contain this as they are not related to style-GAN. So for that document, TF-IDF gives high weight to the “style-GAN” word.

**Word2Vec** Word2Vec is a popular pre-trained word embedding model developed by Google [43]. This model offers learned representations of words within a set of documents. This pre-trained model converts each word as a real-valued vector within a vector space. Therefore, the words that reside closer to each other within that vector space are expected to share similar meanings. The word embedding model captures the meaning of each word from surrounding words in the corpus. There are two types of Word2Vec configuration models: Continuous Bag of Words (CBOW) and Skip-gram. Both models use sliding windows to get a predetermined number of terms before and after the target word to define that target word. In CBOW, context words are used as the inputs to get the target word representation, whereas the skip-gram model is the reverse of CBOW. The word itself is input to the neural network, and the context is the output.

**GloVe** Global Vectors for Word Representation (GloVe) is another popular word embedding method developed by the researchers at Stanford [44]. GloVe is an unsupervised approach that uses a co-occurrence matrix, which assumes that similar words tend to occur together and have a similar context. GloVe uses a large and sparse symmetric term-by-term matrix, where each row and column represent a term vector and their matrix value represents their co-occurrence. Glove uses the dimensionality reduction technique to create smaller information-dense matrices and also a sliding window of terms to calculate co-occurrence.

**Elmo** Before transformer, Elmo is the most popular embedding model developed by AllenNLP [45]. Elmo is a deep contextualized word representation model trained on a large text corpus. This pre-trained model can easily be added to existing models resulting in a significant improvement of the performance in many challenging NLP problems, including question answering, textual entailment, and sentiment analysis. The salient features of the Elmo representations are:

- Word representation is contextual. Each word representation depends on the entire context in which it is used.
• Word representations are extracted by combining all layers of a deep pre-trained neural network.

• Elmo representations are purely character-based which indicates that the network uses morphological clues for robust representations even for out-of-vocabulary tokens unseen in training.

Adding Elmo to existing NLP systems has already proven to be successful for providing significantly improves the state-of-the-art for every considered task. Common examples are name entity recognition (NER), question answering [45].

**Bert**  
Bidirectional encoder representations from transformers (BERT) is a transformer-based embedding model developed by Google AI. It has already created a storm in the machine learning community due to its state-of-the-art results in NLP tasks including Question Answering (SQuAD v1.1), Natural Language Inference (MNLI), and others [46]. The core ideas of Bert are:

• Bidirectionally trained model: this is one of the key technical innovations. This technique offers the model a deeper sense of language context and flow compared to the single-direction language models.

• Masked LM (MLM): It is a novel technique used during training. It randomly masks words (15% out of the total words) in the sentence and then it tries to predict them. Because of this technique, the model looks in both directions, resulting in the full utilization of the full context of the sentence, both left and right surroundings, in order to predict the masked word.

• Next Sentence Prediction (NSP): To understand the sentence level relationship between two sentences, the BERT training process also uses next sentence prediction. During training, the model gets two input sentence pairs, and the model tries to the correct next sentence.

Till now, four types of pre-trained BERT versions exist based on the model depth. We can easily use any of these models as an embedding model to get the rich text representation and by fine-tuning, we easily get a state of art performance.
2.2.2 LONG SHORT-TERM MEMORY (LSTM)

LSTM is a family of recurrent neural network (RNN). Valia RNN often suffers from vanishing gradient [47] resulting in short-term memory. If a sequence is long enough, then RNN cannot carry information from earlier time steps to later ones. For example: if the input is a paragraph, then RNN may leave out important information from the beginning. The solution to this problem is LSTM. Therefore, LSTM is the core deep learning model for NLP. Almost all state of art models are based on this model. LSTM models are explicitly designed to solve the long-term dependency problem. Due to its inherent architecture, the model can remember information for long periods. LSTM model has a chain of repeating neural network modules as RNN. However, this model has gates that control the flow of information, which makes this model unique compare to RNN [7].

There are gates (input gates, output gates, forget gate) and cell states in the LSTM architecture which control the flow of information. These are learnable gates that learn during training and these gates and states decide which data in a sequence is important to keep or throw away. This technique eventually allows passing the relevant information down the long chain of sequences for better predictions.

2.2.3 TRANSFORMER

The transformer is the most impact full deep learning model developed by google AI, which created breakthroughs in NLP [27]. The parallelism, simplicity, and efficiency of this model bound the other researchers to create enormous and very impressive language models like BERT, Megatron [46, 48]. Due to its parallelism ability during training and the state-of-the-art performance in many NLP domains, the transformer is an obvious choice for NLP tasks, and recently transformer has even been used in computer vision tasks [11]. Transformer has two blocks: encoder and decoder. The encoder encodes the input data to get the rich feature representation and decode perform downstream tasks from the encoder output.

For a given input, tokenization is performed to split the words and then each word is converted to its vector representation by the embedding method. Transformer process words in parallel but the model does not have any sense of position/order for each word. To cope with this issue, the position information of each word is added to its embedding. There are different ways to get this
position information but the authors [27] used a fixed and non-trainable function to get the position embedding.

\[
PE(pos, 2i) = \sin \left( \frac{pos}{10000^{\frac{2i}{d_{model}}} \right)
\]

\[
PE(pos, 2i + 1) = \cos \left( \frac{pos}{10000^{\frac{2i}{d_{model}}} \right)
\]

Here \( pos \) is the position, \( i \) is the dimension and \( d_{model} \) is the dimension of key, query and values. After adding position information with the input embedding, input embedding are processed through encoder.

In the encoder layer, the multi-head self-attention layer plays a vital role to produce rich representation. Using this multi-head attention mechanism, the model can view the input from different angles. For an input embedding, the transformer model extract key, query, and value from each of the word representation. Key-value-query concepts come from information retrieval systems. For example, someone search (query) for a particular video (say: machine learning), the search engine will map the query (machine learning) against a set of keys (video title, description, etc.) associated with possible stored videos. The search engine will rank the search results and only show the best-matched videos (values). Query, key, and value are the abstractions that are useful for calculating and thinking about attention. Using these elements (key, query, and value) attention score for each word is calculated using the following formula.

\[
Attention(Q, K, V) = \text{Softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right)V
\]

Here \( \sqrt{d_k} \) is a scaling factor to make sure that the vectors won’t explode. As the authors used multi-head attention, all the head scores are concatenated and once again projected to get the final value.

\[
\text{Multi-headAttention}(Q, K, V) = \text{Concat}(head1, ..., headh)WO
\]

Here each head is calculated using the equation 8 and \( W0 \) is the multi-head projected weight. Projected multi-head attention scores are normalized followed by the feed-forward network. In the
transformer encoder block, there may exist multi-encoder layer. The output of the encoder block is fed into the decoder block which performs the final tasks.
Chapter 3

SEAGRASS SCARS DETECTION USING DEEP LEARNING

Seagrass is a vulnerable habitat in marine ecosystem and can be harmed by many human activities including mechanical damages from propellers of recreational boats to commercial ships. Extinction of seagrass at seafloor can create imbalance in marine ecosystem and lead to extinction of marine animals. Manual on-site monitoring of seagrass status is labor intensive and impossible given the large area of seagrass coverage. Therefore, there is a need of automatic tools for seagrass health surveillance. In this study, we developed a committee machine for seagrass scar detection in remote sensed multispectral images. First, we used a Gram-Schmidt (GS) pan-sharpening technique to obtain high-resolution multispectral images. Second, we utilized preprocessing and data augmentation steps to improve model training. Finally, we trained a committee machine consisting of a set of deep learning models to detect seagrass scar. The developed model improved upon our previous single deep learning and sparse coding models and was applicable across different senses and different satellite systems.

3.1 SEAGRASS PROPELLER SCARS DETECTION

Seagrass is a submerged vegetation plant usually found in tropical sea areas. As an ecological engineer, seagrass has a profound influence on the biological, chemical, and physical coastal waters environment [49], performing critical functionalities to balance marine echo-system [50]. Seagrass provides habitats and food sources to many marine animals such as sea turtles, dugongs, manatees, and fishes [51]. Even a minute quantity of seagrass (a single acre of seagrass) can provide nutrition sources to about 4,000 fishes and 50 million small invertebrates 1. Additionally, seagrass offers further vital ecological functions including sediment stabilization, water quality improvement, organic carbon production, and preservation, nutrient cycling, biodiversity, etc. [51, 52].

Recent studies found that seagrass meadows have been decreasing at a rate of 110 km² yr⁻¹ since 1980. Moreover, this rapid decline has accelerated to 7% since 1990 [53] due to natural

1http://www.sms.si.edu/irlspec/seagrasshabitat.htm
changes, human intrusions, physical disturbance, invasive species, disease, aquaculture, overgrazing, algal blooms, and pollution in the coastal areas. Human inhabitants within 50 km of the sea directly threaten seagrass survival and will eventually create a global crisis in terms of long-term sustainability to seagrass habitats [51]. One of the most critical threats to seagrass is mechanical damage from propellers, which are usually caused by recreational and commercial boats operating in shallow water. For example, seagrass habitats near the Chesapeake Bay, Florida Keys, and the Gulf of Mexico are negatively affected by disturbances to seagrass distributions and ecosystem functions from recreational boats [54–56].

There are several challenges in detecting seagrass scars. First, seagrass scars vary drastically in geometry, including different shapes like straight lines, smooth curves, circles, and ellipses. Second, these geometries do not remain static, making seagrass scar detection a very challenging task. Third, seagrass scars are typically bright in color but can appear as dark lines. In addition, bright scars are often confounded as sand sediments [57]. These variations make seagrass propeller scar detection a significantly challenging task. Fourth, there is very little research on automatic time-efficient seagrass scars detection methods.

Remote sensing imagery is a useful tool for tracking the seagrass scar distributions in coastal areas [58–60]. The technique relies on visual inspection of aerial photography used for seagrass quantification [61, 62], followed by outlining scarred habitat polygons using interactive GIS software. However, this method required expert rating of scar polygons as light, moderate, or severe.
Green et al. first proposed an automatic technique for benthic habitat mapping in the Gulf Coast of Texas using images captured from Leica ADS40 airborne digital camera in 2004 [59] and later on applied for seagrass scar detection [60]. However, this method required manual editing. In 2018, we developed a more sophisticated sparse coding-based method was developed for automatic seagrass scars detection [57]. We utilized a principal component analysis (PCA) based pan sharpening technique to enhance image quality before detecting scars. Although this method works well for scars detection, the computational complexity is high [28].

Deep learning achieved superb performances in numerous fields such as image classification [9], cybersecurity [63, 64], speech recognition [65] natural language processing [66] and medical diagnostic systems [67–69]. In recently years, deep learning also attracted significant attentions in remote sensing community [29, 70–72]. Since deep learning can extract high-level abstract features, deep learning-based hyper-spectral image classification methods can produce more robust results. Deep learning is already being used for seagrass related research, including seagrass detection [73], and seagrass quantification [74, 75].

In this study, we developed an automatic deep learning algorithm to overcome seagrass scar detection challenges. We offer CNN based classifier followed by further improvement using a committee machine algorithm that has robust, and accurate results. Our model is fully automated and applicable to different scenes, even those captured from different satellite systems.

3.1.1 MATERIALS AND METHODS

In this section, we first describe the datasets used in this study and then we present our proposed seagrass scar detection models.

Datasets

Two datasets were collected for this study, one was at the Deckle Beach, Florida in May 2010 and another at Virginia coast in 2019. We trained our model with the Deckle Beach Dataset and tested the trained model on both datasets.

Deckle Beach Dataset Our deckle beach dataset contains panchromatic and multi-spectral images captured by the WorldView-2 (WV-2) satellite (Digital Globe LLC) near Deckle Beach,
Florida, UA in May 2010. The images were captured with off-nadir view of 20 – 37 degrees and azimuth angle range at 307 degrees (true solar zenith angle of 127 + 180 degrees) plus/minus 45 degrees (or between 262 – 352 degrees). In the imaging system, the panchromatic image has 0.46m spatial resolution, and multi-spectral images have 1.84 m spatial resolution. The multi-spectral image has in total eight bands including traditional blue, green, red and near-infrared bands along with the four additional bands including costal band (a shorter wavelength blue band, centered at approximately 427 nm; yellow band centered at approximately 608 nm; a red edge band (centered strategically at approximately 724 nm; and near-infrared band, centered at approximately 908 nm, which is sensitive to atmospheric water vapor.

The WV-2 remote sensing system placed six separate georeferenced images in a mosaic to yield a single image consists of eight spectral bands (multi-spectral image) plus the pan band (panchromatic image). Atmospheric correction was performed using the empirical line method in which the spectral return values captured in the image was matched to in situ measurements made on the same day at 22 stations across the image by a survey boat. Two floating spectroradiometer systems were used at each station to measure the downwelling spectral irradiance [Es(0+)] +), upwelling spectral radiance, upwelling irradiance and radiance. More details can be found from our previous literature [57].

In this study, we identify seagrass scars in the 3rd to 6th bands (green, yellow, red, red-edge bands) since other bands do not provide much useful information for seagrass scar. To train our deep learning model, we used an 800 by 900 image collected near Hagen’s Cove Park, FL, which contains numerous prop scars, as shown in Fig. 5 where Fig.5.1a is the panchromatic image, Fig.5.1b is the corresponding ground truth image with scar labels. In the ground truth image, seagrass scars were marked by an experienced operator based on i) long narrow slice and circular like appearance, ii) color difference between the seagrass bed and the scar, often characterized by a brighter reflectance as sand is exposed and iii) large holes near straight-line scars created by the vessel operator attempting to free the vessel [57].

We used four images taken from Hagens Cove Park, Florida, USA by the WV-2 satellite (Fig.5) and four images captured from the Virginia coast by the WV-3 satellite (Fig.6) to test our model. The first three images in Fig.6 have at least two scars and the last image does not have any scars. Ground truth for scares were marked by the same experienced operator and Table 1 shows more
Fig. 5. Training image: (a) panchromatic image, (b) ground-truth image with scars mark.

Fig. 6. Testing images captured by the WV-2 imaging system.

detailed information of the images.

Virginia Coast Dataset This dataset was captured by the WV-3 satellite from Virginia coast area (Fig. 7, first row). We directly apply the trained model without re-training to this dataset test the generalization capability of our model. These images are more challenging than Deckle beach dataset as they contain very low visible scars with variety of scar shapes. Images went through the same atmospheric correction procedure and ground truth for seagrass scars are shown in the second row of Fig. 7, which were labelled by the same experienced operator. More information about the images are listed in Table.1.
Fig. 7. Testing images captured from WV-3 imaging system. 1st row: Panchromatic band. 2nd row: Ground truth

Table 1: Image size of the eight testing images.

<table>
<thead>
<tr>
<th>Images</th>
<th>Deckle Beach</th>
<th>Images</th>
<th>Virginia Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Testing-image-1</td>
<td>716x796</td>
<td>Testing-image-5</td>
<td>500 x 650 x 4</td>
</tr>
<tr>
<td>Testing-image-2</td>
<td>1024x848</td>
<td>Testing-image-6</td>
<td>750 x 950 x 4</td>
</tr>
<tr>
<td>Testing-image-3</td>
<td>788x1008</td>
<td>Testing-image-7</td>
<td>850 x 650 x 4</td>
</tr>
<tr>
<td>Testing-image-4</td>
<td>748x976</td>
<td>Testing-image-8</td>
<td>550 x 650 x 4</td>
</tr>
</tbody>
</table>

3.1.2 ALGORITHM DEVELOPMENT

Our algorithm consists of three steps (Fig.8). First, we created a high-resolution multi-bands pan-sharpened image through the fusion of multispectral and panchromatic images using the Gram-Schmidt (GS) pan-sharpening method [76]. Second, a set of preprocessing steps were applied to the pan-sharpened images for image contrast enhancement. To get more image patches,
we augmented data by rotating from 0-180 degrees. After that, we extracted scars and non-scars patches from the rotated preprocessed images. Finally, the augmented image patches were fed to a deep learning-based committee machine model for training.

### 3.1.3 PAN-SHARPENING

Pan-sharpening is a pixel-wise image fusion between a panchromatic image (PAN) and a multispectral image (MS). Using pan-sharpening methods, we can obtain high-spatial resolution as well as high-spectral resolution images. In this study, we implement the GS method [76] and deep learning models for pan-sharpening satellite images. In the GS method, panchromatic image and multispectral image are fused using the GS transformation. GS method can achieve the maximum image sharpness and less color distortion and it is computationally efficient.

Fig. 10 depicts the deep model structure for pan-sharpening [77, 78]. First, we combine PAN and MS bands in a single 5-channel image that can be fed to the deep model shown in the blue box. To do this, we down-sample the PAN band so that its dimension matches to the MS bands and we denote this low-resolution version of the PAN band as PANL. To account for noises that down-sampling causes to the PAN band, we down-sample the MS bands by the same factor as the PAN band, then up-sample back so that the size is matched. We denote this low-resolution version of the MS bands as MSL. We combine PANL and MSL bands to create an 5-band image as PANL+MSL. Finally, we extract 32x32x5 patches from PANL+MSL and use the patches to train a deep model to approximate the corresponding high-resolution patches from MS. This model can learn a mapping from low-resolution PANL+MSL to high-resolution MS and output of the

---

**Fig. 8.** Flowchart of deep learning-based scar detection model.
model will be high-resolution pan-sharpened images. Fig. 10 shows the pan-sharpened image bands where the first row lists original MS image bands and the second row are corresponding GS pan-sharpened image bands. However, from our previous study [28], we found that seagrass scars detection model performed well when we used the GS based pan-sharpening method instead of deep learning based pan-sharpening method. For the rest of this study, we use GS pan-sharpening method.

3.1.4 IMAGE CONTRAST ENHANCEMENT

We followed the same pre-processing techniques used in our previous work [57] to enhance image contrast since classification models may fail to detect scars if images have low visible scars [79]. We used the Laplacian filter to enhance the rapid intensity change regions and used Sobel (gradient) filter to finalize the enhancement as shown in Fig. 11. During the testing phase, we matched the histogram of the testing images to the training images for better performance.
Fig. 10. Multispectral image bands before and after pan-sharpening procedure. 1st row for Green, Yellow, Red and Red Edge bands before pan-sharpening. 2nd row for the corresponding bands after pan-sharpened by the GS method.

Fig. 11. (a) Pan-sharpened image, (b) Image after application of the laplacian filter, (c) Image after application of the laplacian and sobel filters.
3.1.5 PATCHES EXTRACTION

From the pre-processed pan-sharpened image, we extracted overlapping scar and non-scar patches of a size of $w \times w \times 4$ where $w$ is called “receptive field size” and 4 is the number of bands (Fig. 12). A patch is called as a scar patch if a scar going through its center. We selected “w” as 17 in this study based on trials and errors.

3.1.6 DEEP LEARNING COMMITTEE MODEL FOR SEAGRASS SCAR DETECTION

3.1.7 HYPER-PARAMETERS OPTIMIZATION

Patches dimensions are very important hyper-parameter to get good results. We tried with
different patch dimensions (5x5, 9x9, 12x12, 15x15, 17x17, and 19x19) and found that 17x17 is the optimal patch dimension to get the model’s best performances. We designed 16 CNN models to detect scars and non-scars patches for our dataset. Models are varied based on the architectural variations in layers (2 layers to 8 layers), drop out, batch normalization, skip connections, and activation function. We set the drop-out value to 0.2, epochs to 30, and optimizer to adam with the default value (Keras framework) to train all deep learning models.

Convolutional Neural Networks (CNN)

Fig. 13 shows one of the developed CNN models for seagrass scars detection. This model consists of the three convolutional layers with convolutional skip connection and fully connected (FC) layer. To avoid the gradient-explode, we used a 1x1 convolutional skip connection [80]. We also used batch normalization, Leaky-relu as activation function, stride, and dropout at each convolutional layer. Each of the models was trained with the 17 x 17 x 4 image patches extracted from the pre-processed pan-sharpened image.

Committee Machine

The common practice in machine learning is to train deep learning models while keeping the best candidate model and discarding the rest and all the efforts for the training of the discarded
deep learning models are wasted. Moreover, a single best model may perform poorly on different testing datasets. These shortcomings can be overcome by combing all the trained deep learning models to form a committee machine. The committee machine is a branch of neural network where multiple trained model’s responses are combined into a single response. Supposed we have \( M \) trained models, and each of the trained models generates a prediction result \( P_m \), the committee machine \( P_c \) can be defined as:

\[
P_c = \frac{1}{M} \sum_{m=1}^{M} P_m
\]  

Output of the committee machine has a reduced variance than that of any committee member [81]. In this study, we developed sixteen convolutional neural network models with different structures and trained them using the same training data for seagrass scars detection. We combined models’ outputs by averaging outputs from all member models to get a single response with value between 0 to 1. We then applied threshold to the averaged output to obtain the final scar detection result. Fig. 14 shows structure of the committee machine.
Transfer Learning and Fine-Tuning

Transfer learning is a prevalent deep learning technique where a model is trained with a large dataset and the trained model is applied to solve a similar task [82]. This technique is beneficial if there are not enough training data for training. However, if data distribution shift are presented between source domain and target domain, a direct transfer of the trained model from source domain to target domain is subject to significant performance degradation and fine-tuning is required. In our study, we apply the model trained for seagrass scar detection with Deckle Beach dataset to identify seagrass scar with Virginia Cost dataset and the results are not satisfactory as shown in Fig. 15, where a large amount of false positives are presented at the bottom right of the image. Multispectral images collected at Virginia Cost were captured by WV-3 satellite, which have different optical properties compared to the Deckle Beach dataset (captured by WV-2 satellite). To mitigate this problem, we fine-tune the committee model trained with Deckle Beach data by image patches extracted from the Virginia Cost dataset. To keep the efforts minimal, we utilize 600 scar patches and 600 non-scars patches from Virginia Cost dataset for fine-tuning.
Performance metrics

We computed the total bare patches area, the total bare patches with respect to the total area, total scars throughout the scene (per m²). We also computed the sensitivity, precision, recall, and F1 score as a quantitative metrics for model evaluation. These metrics are defined as follow:

\[ \text{Bare patches area} = \text{false positives area (per m}^2) \]  \hspace{1cm} (11)

\[ \text{Sensitivity} = \frac{\text{no. of correctly detected scar}}{\text{total no. of labeled scar}} \] \hspace{1cm} (12)

\[ \text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \] \hspace{1cm} (13)

\[ \text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \] \hspace{1cm} (14)

\[ \text{F1} = \frac{2 \ast \text{Precision} \ast \text{Recall}}{\text{Precision} + \text{Recall}} \] \hspace{1cm} (15)

3.1.8 EXPERIMENTAL SETUP AND RESULTS

Experiment Setup

We extract about 10k scar and non-scar patches of size 17x17x4 from the training image. We augment the training data by rotating the training image by 0 – 180 degrees with a step size of 20 degrees resulted in ten training images and repeat the patch extraction step ten times. Equivalently, we obtain 100k scar and non-scar patches in total for training. To fine-tune the deep committee model for seagrass scar detection at Virginia Coast, we extract 600 scar and 600 non-scar patches from the Virginia Coast dataset and fine-tune the model with 15 training epochs. We evaluate the trained committee machine on the eight testing images. Before evaluation, we match the histogram of each of the testing images to the histogram of the training image.
Results of Deckle Beach dataset

We train the deep committee model with the extracted patches from the training image collected at the Deckle Beach as shown in Fig. 5. After training, we apply the trained model to the four images collected at Deckle Beach and results with different thresholding values are shown in Fig.16. We gave the full credit to the models even for the partial detections of scars.

By visual inspection, it is observed that the deep committee machine model detected all labeled scars except image-1, where one labeled scar is missed. Number of false positives decreases along with the increase of the threshold value while a large thresholding value also reduces true positives. To make a trade-off between false positives and false negatives, we choose 0.6 as threshold to produce the final seagrass scar detection, and quantitative performance metrics are listed in Table. 2. Sensitives are 100% for all testing images except that image-1 has a sensitivity of 87.5%. Those sensitivities indicate that most of the scars can be detected by the deep committee model. At pixel level, the deep committee model has much lower efficiency with Recall less than .35 and produces lot of false positives with Precision less than 0.08.

<table>
<thead>
<tr>
<th>Table 2: Quantitative evaluation: Deckle beach dataset (threshold .6).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model scars Map</td>
</tr>
<tr>
<td>Detected bare patch area (false positive, m²)</td>
</tr>
<tr>
<td>Percentage of bare patch area to total area (%)</td>
</tr>
<tr>
<td>Detected scar area (m²)</td>
</tr>
<tr>
<td>Percentage of detected scar area to total area (%)</td>
</tr>
<tr>
<td>Sensitivity (%)</td>
</tr>
<tr>
<td>Number of scars detected</td>
</tr>
<tr>
<td>Number of scars labeled</td>
</tr>
<tr>
<td>Precision</td>
</tr>
<tr>
<td>Recall</td>
</tr>
<tr>
<td>F1 score</td>
</tr>
</tbody>
</table>
Fig. 16. Model classification maps after applying different thresholds. 1st, 2nd and 3rd column are the result of threshold .5,.6 and .7 respectively. 1st, 2nd and 3rd, 4th row showing the model classification map for the deckle beach testing image 1, 2,3 and 4.
Results of Virginia Coast dataset

The deep committee model trained with the training image collected at Deckle Beach did not perform well on Virginia Cost dataset (Fig. 15) due to differences in the image acquisition systems, locations, scars visibility, and shapes between the two datasets. We extract 600 scar and non-scar patches, respectively, from the Virginia Coast dataset and use them to fine-tune the deep committee model. This fine-tuning step drastically improves the performances as shown in Fig. 17. The threshold of 0.5 produces thicker scars with false positives and 0.7 threshold misses some scars. We choose 0.6 as the final threshold value and compute quantitative performances metrics as listed in Table.3.

Sensitives are 100% for all the testing images except that image-6 has a sensitivity of 87.5%. Those sensitivities indicate that most of the scars can be detected by the deep committee model. At pixel level, the deep committee model has achieved very high efficiency with Recall larger than .65 and produced very few false positives with Precision larger than .61.

Table 3: Quantitative evaluation: Virginia coast dataset (threshold .6).

<table>
<thead>
<tr>
<th>Model scars map</th>
<th>Testing image 5</th>
<th>Testing image 6</th>
<th>Testing image 7</th>
<th>Testing image 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detected bare patch area (false positive, m²)</td>
<td>250</td>
<td>1305</td>
<td>4696</td>
<td>17</td>
</tr>
<tr>
<td>Percentage of bare patch area to total area (%)</td>
<td>.80</td>
<td>1.91</td>
<td>8.85</td>
<td>.05</td>
</tr>
<tr>
<td>Detected scar area (m²)</td>
<td>2548</td>
<td>3389</td>
<td>18084</td>
<td>3569</td>
</tr>
<tr>
<td>Percentage of detected scar area to total area (%)</td>
<td>8.16</td>
<td>4.95</td>
<td>34.06</td>
<td>10.39</td>
</tr>
<tr>
<td>Sensitivity (%)</td>
<td>100</td>
<td>87.5</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Number of scars detected</td>
<td>3</td>
<td>7</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>Number of scars labeled</td>
<td>3</td>
<td>8</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>Precision</td>
<td>.90</td>
<td>.61</td>
<td>.74</td>
<td>.99</td>
</tr>
<tr>
<td>Recall</td>
<td>.89</td>
<td>.65</td>
<td>.99</td>
<td>.66</td>
</tr>
<tr>
<td>F1 score</td>
<td>.90</td>
<td>.63</td>
<td>.85</td>
<td>.80</td>
</tr>
</tbody>
</table>
Fig. 17. Model classification maps after applying different thresholds. 1st, 2nd and 3rd column are the result of Threshold .5, .6 and .7 respectively. 1st, 2nd and 3rd, 4th row showing the model classification map for the deckle beach testing image 1, 2, 3 and 4.
Computational complexity and performance analysis

All of our experiments were performed on a desktop computer equipped with an Intel Core i7-4790, 3.60GHz (16 cores), and 32 GB Ram. On average, it took about 2 hours for training for each of the CNN models and took on average less than 2 minutes for testing an image. Therefore, the committee machine took approximately 20 minutes to produce the classification map for a testing image. The sparse coding method, on the other hand, took on average 4 hours for training, and took 4 hours for testing an image, which is much more computational expensive in testing.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Testing time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sparse coding</td>
<td>4hrs</td>
</tr>
<tr>
<td>Single CNN</td>
<td>2 mins</td>
</tr>
<tr>
<td>Committee machine</td>
<td>20 mins</td>
</tr>
</tbody>
</table>

Table 4: Time complexity comparison.

Though a single CNN model was faster than the committee machine, the committee machine model produced a more robust and accurate classification map than a single CNN model. Fig.18.a and Fig.18.b, shows Accuracy and F1 Score vs Number of CNN models in the committee and it is observed that both accuracy and F1 score increase with the number of the CNN models in the committee. There was a sharp increase in accuracy and F1 score when the committee machine had three CNN models. Eventually, the model produced consistent outputs (accuracy and F1 score) when the committee model contained all CNN models.

3.2 DISCUSSION AND CONCLUSION

The proposed deep learning-based committee machine can perform automatic propeller scars detection across the submarine landscape. It could be a useful tool for managing and protecting shallow water resources, including seagrass meadows. The proposed committee machine outperformed both individual single CNN scar detection models and sparse coding model in terms of accuracy and F1 score. In addition, testing time of the committee machine model was far less than
the sparse coding model.

It is tough to find the right pan-sharpening methods for our problem. For better performance, time efficiency, we used the GS-based pan-sharpening method rather than PCA and deep learning. Preprocessing steps were used to enhance the image contrast, and it is crucial for better scars detection. We used different thresholds (.5, .6, and .7), which filters out the false positives resulting in obsolete the necessity of the post-processing steps.

Our initial classification maps from the committee machine algorithm produced better scars detection maps than the initial maps of sparse coding and the single CNN model. Our developed model detected most of the scars resulting in highly sensitive (100%) with lower bare patches. However, the model trained on the Deckle beach dataset could not perform well on the Virginia coast dataset as the images were captured by different satellites. Here transfer learning was not enough as the optical properties, shapes, and visibility of scars are different between these datasets. The fine-tuning process with very few samples of data from the new location boosted the performance of the model. After the fine-tuning, it is clearly illustrated that that the model can detect round shape scars (testing image 6).

In this study, we proposed a committee machine model for sea grass scar detection and the proposed model produced robust and accurate scars maps which outperformed than our previous sparse coding model. Our proposed method showed better scar detection sensitivities with lower
bare patches, better precision and F1 score. Moreover, the proposed model was more computation-
ally efficient than sparse coding in the testing phase. We concluded that our proposed algorithm
could be used as a coastal management tool to monitor seagrass scar changes in habitat character-
istics in coastal waters on a regular basis.

3.3 RELATED PUBLICATIONS

1. Hoque, Md Reshad Ul, Kazi Aminul Islam, Daniel Perez, Victoria Hill, Blake Schaeffer,
Richard Zimmerman, and Jiang Li. “Seagrass propeller scar detection using deep convolu-
tional neural network.” In 2018 9th IEEE Annual Ubiquitous Computing, Electronics

2. Seagrass Propeller scars detection using deep committee machine (Arxiv preprint, 2022)
Chapter 4

REMOTE SENSING IMAGE SUPER-RESOLUTION AND AN ARITHMETIC DEEP MODEL FOR TEMPORAL REMOTE SENSING IMAGE FUSION

This Chapter presents two different studies using deep learning models for remote sensing image super-resolution and multi-temporal images fusion. Specifically, Section 4.1 explores deep learning models for remote sensing image super-resolution and makes a constructive comparison of the GAN-based models with the CNN models. Additionally, Section 4.2 presents a deep learning-based fusion where our proposed models perform an arithmetic operation applied to the low-resolution images at the two-time points in feature space to take care of temporal changes. We evaluated the proposed models on three modality pairs for multi-modal temporal image fusion, including downsampled WorldView-2/original WorldView-2, Landsat-8/Sentinel-2, and Sentinel-2/NAIP.

4.1 REMOTE SENSING IMAGE SUPER-RESOLUTION

Remote sensing imagery is an effective tool for obtaining information related to the earth’s resources and environment which can be used in numerous applications such as changes detection, clouds tracking, biomass quantification, classifications, ocean-bottom mapping, urban traffic analysis, and many more applications [83]. However, satellite imagery systems largely suffer from poor image details resulting in low spatial resolution due to the large distance between space sensors and the sensed objects. Moreover, upgrading the space-borne systems are very expensive and difficult [84]. So, the better way is to use software solutions which are less complex and cost-effective.

Image super-resolution research is a part of image and signal processing study where the purpose of image super-resolution is to increase spatial resolution and preserve the integrity of spectral resolution. Image super-resolution methods can be classified as multi-image super-resolution and
single image super resolutions [85, 86]. It is difficult to get multiple images of the same scenes because scene may changes too fast due to shadow, clouds, moving objects, etc. making multi-image super-resolution approach uncommon [87]. On the other hands, single image super solution uses a single image which is easy to obtain.

Bicubic interpolation methods are very straightforward and common for single image super-resolution (SISR). Due to the recent development of computational power, machine learning, learning-based algorithms become popular for SISR. Deep learning is one of the examples of learning-based algorithms. In the learning-based algorithm, the model learns an end to end mapping from a vast amount of training data. Trained models are then eventually used to enhance spatial resolution. Deep learning most prominently convolutional neural networks (CNNs) has already proved as successful to show state-of-the-art performance for SISR [88] but CNN based super-resolution can not recover the finer texture details for large up-scaling factors. Generative adversarial network (GAN) can recover the photo-realistic sharp high-resolution image from the low-resolution image even for large upscaling factors [89].

Remote sensing satellite images are affected by weathers, topographic conditions, and sensors that may have different characteristics when compared to natural images [90]. Moreover, due to extensive coverage, large spectral bands of remote sensing (RS) images, generic deep learning model trained on natural images can not perform well on RS images. Furthermore, there are very few researches on RS image super-resolution. In this section, we develop a deep learning-based algorithm for image super-resolution. The contribution of this study are:

1. We developed two GAN (GAN-1, GAN-2) and two CNN (CNN-1, CNN-2) based super-resolution methods for remote sensing imagery.

2. We fine-tuned Vgg model on satellite images to improve the performances.

3. We critically compare them in terms of visual and quantitative perspective.

The remainder of this section are organized in following ways. Section 4.2.1 provides related research works of super resolution. Section 4.2 describes the proposed deep learning methods. Sections 4.1.3 and Section 4.2.6 present our experimental results and discussions, respectively, and Section 4.2.7 represent the conclusion of this paper.
4.1.1 RELATED WORK

Traditional Super Resolution Methods

The idea of super-resolution is not new. Duchon et al. first proposed bicubic interpolation and later, Lanczos interpolation [91] which are the first steps towards image super-resolution. But these filtering methods provide overly smooth texture details of the image. Interpolation method, especially the bicubic interpolation method is very fast, and implementation is very simple. Therefore, it is used as a baseline for most of today’s super-resolution research.

Later, the more powerful and complex mapping between low to high-resolution images were proposed. Yang et al. developed sparse representations of high and low-resolution patterns. They converted image patches to sparse representation and used the coefficient of these representation to generate a high-resolution image. They proved that by joint training with two dictionaries of low and high-resolution image patch, it is possible to generate high-resolution image [92, 93]. Later, Timofte et al. improved the performance of sparse coding based super-resolution in terms of image quality and time complexity [94].

Deep Learning Based Methods

Deep learning has already become a buzzword in today’s tech world and has obtained state-of-the-art performances on medical diagnostic systems [68, 95], image classification [96], cybersecurity [97], and many more other fields. Dong et al. first proposed deep learning-based image super-resolution technique where the model learns an end-to-end mapping from low and high-resolution image patches [88]. Ledig et al. first proposed the GAN based photo-realistic super-resolution (SRGAN) method. They proposed perceptual loss function, and through this loss function with adversarial training between generator and discriminator, generator generated high-resolution images from heavily down-sampled images [98]. Wang et al. developed progressive super-resolution GAN (PRGAN) and proposed model could successfully upsample an image to 8x with good image quality [89].

Although remote sensing community already benefited from deep learning [28, 73], there is very few research on deep learning-based image super-resolution. Liebel developed CNN based
remote sensing image super-resolution on SENTINEL-2 satellite images [99]. Tuna et al. also experimented on SPOT satellite images using CNN [90]. Jian et al. developed a GAN based edge-enhancement network (EEGAN) for remote sensing image super-resolution [100].

4.1.2 METHODOLOGY

Datasets

We use Worldview-2 satellite images for CNN and GAN based image super-resolution model training. The size of the CNN training image is 500x900x3 pixel (Fig. 19) and the dimension of each image patch is 32x32x3 with a total number of 400k.

For GAN training, we use about 17000 images with a dimension of 256x256x3 collected from Worldview-2 satellite images. Fig. 20 shows samples of training images for GAN training. To finetune the VGG network which is used for GAN training, we use SpaceNet. SpaceNet is satellite imagery corpus collected from Worldview-2, Worldview-3 satellite. For our case, we use 3 bands (R,G,B) 50 cm images collected from Worldview-2 corpus [101]. For testing, we use five remote sensing images collected from WorldView-2 for model evaluation. Table 5 shows the details of
Preprocessing

Preprocessing steps are crucial for super-resolution methods. For our experiments, we only consider RGB channels. Therefore, we extract RGB channels from WorldView-2 multiband imagery \(^1\). Though satellite system has made immense advancement, yet high-resolution images are limited. Therefore, it is impractical to use the original image as a low-resolution image. To cope

\(^1\)https://www.satimagingcorp.com/satellite-sensors/worldview-2/
up with this problem, we simulate the low-resolution image from the original image. We down-sample the original image by 2 and then upsample it by 2 to make a low-resolution image. The simulated low-resolution image is used for CNN-1 training where the original image is used as ground-truth.

We develop two CNN super-resolution models where one model (CNN-2) was trained only on the brightness (luminance) channel. For this purpose, we convert RGB WordView-2 training image to YIQ color space. I and Q channels mostly contain spectral information, whereas Y channel contains luminance (brightness) information \(^2\) [102]. We extract the Y channel from YIQ channels and simulate the low-resolution image from it.

**Convolutional neural network (CNN) based super-resolution**

We put a low-resolution RGB image at the input and the original RGB high-resolution image as output to be mapped. Fig. 21 shows the training and testing phase for CNN-1 model. Once the deep CNN model (CNN-1) was trained, we applied the simulated low-resolution images to the trained model and model generate a higher resolution image (2x) from it. We upsample the generated image by 2, then feed it again to the model and model generate the high-resolution image from it. In this way, we applied the trained model recursively to the output image to achieve four times (4x) higher resolution image.

As human eyes are more sensitive to brightness [103], so we perform CNN based image super-resolution only on the brightness channel instead of RGB channels. Fig. 23 shows the details of CNN-2 method. In this method, we convert the training RGB image to YIQ color domain. After the conversion of RGB to YIQ, we separate the Y channel from IQ bands. We then recursively apply trained CNN model on brightness channel in a similar manner as the CNN-1 model. To retrain spectral information, we apply the bicubic interpolation (4x) on IQ bands. Then we merge all bands to get high-resolution YIQ image and apply the inverse transform to get high-resolution RGB image.

\(^2\)https://www.mat.univie.ac.at/~kriegl/Skripten/CG/node14.html
Fig. 21. CNN-1 based image super resolution

Fig. 22. CNN-2 based image super resolution

Fig. 23. CNN architecture for image super-resolution
Fig. 24. GAN architecture for image super-resolution

**CNN Architecture**

We use shallow CNN architecture in our experiments. Fig. 23 shows CNN based image super-resolution architecture (CNN-1). We use three convolutional layers where each layer followed by batch normalization, drop out and rectified linear unit (ReLu) as an activation function. In order to avoid vanishing gradient and efficient training [104], we use skip-connection. The trained model (CNN-1) take low-resolution image patches (32x32x3) as input and generate high-resolution image patches as output. For CNN-2, input patches dimension is 32 x 32 x 1 instead of 32x32x3 and in 3rd convolutional layer of CNN architecture, the filter dimension change to 5 x 5 x 1.

**GAN based super resolution (GSR)**

Fig. 24 shows the block diagram for GAN based image super-resolution. Here the inputs are low-resolution images, and the output of the Generator is a high-resolution image. GAN consists of a Generator and a Discriminator. It is like playing a game where the Generator tries to win over the Discriminator. The Generator wants to produce high-resolution images similar to the original high-resolution images in the training , and the Discriminator acts like a judge to decide whether the input is a generated fake high-resolution image or from the original training datasets.

Fig. 25 shows the training process where the Discriminator decides whether its input is from the Generator or from the real data (Generated HR image/Fake HR images in Fig. 24). The error
message of the Discriminator is then used to optimize the Generator to make its outputs look more like real. The error message is also used to update the Discriminator to make it can better judge if its input is real. The Discriminator error messages are guiding the Generator to produce realistic data. After training, the Generator can produce fake images that can match the real image distribution so that they look real (like real high-resolution images in our study). In GSR, MSE based content loss is replaced with features based loss computed from pre-trained VGG19 network which is trained on ImageNet [105].

**GAN based super-resolution with fine-turned VGG network**

We develop two GAN models with (GAN-2) and without (GAN-1) fine-tuning the VGG network. We fine-tune VGG network with SpaceNet remote sensing and train the GAN network with the same data, same GAN model and also identical experiment setups. Our training data is Worldview-2 satellite image data, but the pre-trained VGG network is trained on ImageNet. As SpaceNet is collected from Worldview-2 satellite images [101], we hope that fine-tuning VGG network with SpaceNet data may improve the generated image from the GAN model (GAN-2).
**Evaluation Metrics**

We use five testing images collected from WorldView-2 satellite image and evaluate our models with peak signal and noise ratio (PSNR) and structural similarity (SSIM) metrics. We also use Spectral Angle Mapper (SAM) \(^3\), Universal Image Quality Index (UIQI) or Q-index, and quality without reference (QNR) as image quality metrics [78] for critical comparison between original high-resolution images and generated images. The higher value of the $PSNR$, $SSIM$, $Q$, $QNR$, and lower value of $SAM$ indicate the better the quality of the generated high-resolution images from deep learning models.

**4.1.3 RESULTS**

For remote sensing image super-resolution, we develop CNN (Fig. 23) and GAN (Fig 24) based image super-resolution models to generate high resolution images. We train CNN-1 models using 32x32x3 image patches extracted from WorldView-2 RGB training image. For CNN-2 model training, we use only the brightness channel. For GAN based image super-resolution, we also use WorldView-2 images for training. We use about 17000 images for training and evaluate the GAN model with and without fine-tuning VGG19 network.

We evaluate our developed models on five testing images in term of visual, quantitative perspective. Table 15, show the overall deep learning super-resolution models performance in terms of image quality metrics. From Table 15, it is explicitly visible that CNN based super-resolution methods outperform other methods. Between two CNN methods, CNN-1 performs better.

We also evaluate our models from visual points of view. Fig. 26 shows high-resolution images generated from our developed deep learning models. Visually bicubic based super-resolution images are over-smooth and blurry. CNN based generated high-resolution images are visually better than bicubic based generated images. High-resolution images generated from GAN has sharp outputs, and visually GAN-2 outperform to the other deep learning models.

Our all experiments are conducted through a desktop computer which has 16 GB Ram, an Intel Core i7-4790, 3.60GHz (8 cores) processor. We use CPU time of this desktop to evaluate our developed deep learning models. Table 7 shows the time complexity of our models. CNN-2 model

---

\(^3\)https://www.harrisgeospatial.com/docs/SpectralAngleMapper.html
Fig. 26. Models performance when applied to the testing images. From left to right, the first rows (a-e) shows the high resolution images, 2nd row (f-j) shows bicubic output, 3rd (k-o) and 4th row (p-t) for CNN-1 and CNN-2 based super resolution and 5th, 6th rows are the generated images from GAN-1 and GAN-2 respectively.
faster than CNN-1 in training and testing. Though GAN model took much higher time for training, but it’s testing time is lower than CNN models.

Table 6: Evaluation metrics (average)

<table>
<thead>
<tr>
<th>Matrics</th>
<th>Bicubic</th>
<th>CNN-1</th>
<th>CNN-2</th>
<th>GAN-1</th>
<th>GAN-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSIM</td>
<td>.73</td>
<td>.78</td>
<td>.77</td>
<td>.68</td>
<td>.69</td>
</tr>
<tr>
<td>PSNR</td>
<td>27.04</td>
<td>28.63</td>
<td>28.10</td>
<td>20.39</td>
<td>20.65</td>
</tr>
<tr>
<td>SAM</td>
<td>1.70</td>
<td>1.93</td>
<td>1.81</td>
<td>19.65</td>
<td>19.40</td>
</tr>
<tr>
<td>Q</td>
<td>0.98</td>
<td>0.99</td>
<td>0.99</td>
<td>0.88</td>
<td>0.89</td>
</tr>
<tr>
<td>QNR</td>
<td>.86</td>
<td>0.84</td>
<td>.85</td>
<td>0.81</td>
<td>0.80</td>
</tr>
</tbody>
</table>

Table 7: CPU time for deep learning based super resolution models.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Training</th>
<th>Testing</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN-1</td>
<td>23 h</td>
<td>11.78 m</td>
<td>23.20 h</td>
</tr>
<tr>
<td>CNN-2</td>
<td>9 h</td>
<td>6.1 m</td>
<td>9.12 h</td>
</tr>
<tr>
<td>GAN-1/GAN-2</td>
<td>55 h</td>
<td>9 s</td>
<td>55.00 h</td>
</tr>
</tbody>
</table>

4.1.4 DISCUSSION

We developed CNN and GAN based image super-resolution models which can generate the high-resolution images at factor 4. CNN based image super-resolution methods produce the best results in terms of quantitative and CPU time perspective. Visually bicubic based generated images
are inferior to other methods as they are blurry though the generated images have higher metrics in some cases. On the other hand, GAN generated high-resolution images are visually sharp but fail in terms of quality metrics. Though GAN generated images are visually sharp, but models also produce artifacts which ultimately degrade its image quality metrics value.

According to Table 15, CNN based generated images have high SSIM, PSNR, Q and low SAM value whereas GAN generated images have lowest metrics value and highest SAM value although GAN models generate sharp and photo-realistic outputs (Fig. 26). For example, higher SAM value of GAN generated images clearly indicate that there is much spectral mismatch between the generated image and the original high-resolution image. Between GAN-1 and GAN-2, GAN-2 generate images are visually and quantitatively better. As our training and testing data are remote sensing data and the GAN model want to generate images close to original ground truth distribution, so fine-tuning VGG19 network with remote sensing data boost image quality.

CPU time is also an important for model evolution. The testing time is an important factor after training. From CPU time analysis (Table 7), CNN-2 is faster than CNN-1 model. It is obvious that CNN-2 model is trained on one channel image whereas CNN-1 is on three channels (RGB). Though GAN models took much longer in training, the testing time is much lower than other models.
4.1.5 SUMMARY

In this section, we proposed deep-learning image super-resolution methods for remote sensing imagery and compared them from the visual, quantitative perspective. We develop CNN and GAN based image super-resolution models which can generate high-resolution images from low-resolution images. Both developed methods have certain advantages and disadvantages. CNN based models outperform to other models in terms of image quality metrics, whereas GAN generated image has better visual quality but the image may contain artifacts.

Super-resolution on remote sensing imagery has numerous applications on bio-monitoring tasks. But unfortunately, super-resolution research is still in developing stage and deep learning models trained on natural images can not apply on remote sensing images. Our developed deep learning models for remote sensing image super-resolution can be applied on bio-monitoring tasks. In the future, we will optimize our deep learning models, specially GAN models, and try to develop deep CNN models for high-resolution images generation.

4.2 TEMPORAL REMOTE SENSING IMAGE FUSION

With rapid global environmental changes in recent years, remote sensing imaging systems become effective tools for vegetation monitoring [106–109], land cover detection [110], and human-nature interaction [111]. Availability of high temporal and spatial resolution remote sensed images plays a critical role in the success of these systems [112, 113]. However, it is challenging to obtain high resolutions in both spatial and temporal domains by current satellite platforms due to the technical and budget limitations. Different commercial remote sensing imagery providers produce different resolutions with various revisit frequencies and costs. For example, WorldView-2 (WV-2) satellite and National Agriculture Imagery Program (NAIP) aerial images have very high spatial resolution but with very high prices. Landsat satellite images are free of charge with medium spatial resolutions [114].

One possible solution is to fuse multi-temporal spatially coarse images with multi-temporal high-resolution images to achieve adequate resolutions in both temporal and spatial domains [115].

http://www.landinfo.com/satellite-imagery-pricing.html
http://worldview3.digitalglobe.com/
The fusion algorithm integrates high spatial, low temporal resolution (HSLT) images with low spatial, high temporal (LSHT) images. Through fusion, high temporal resolution images at medium spatial scale with nominal revisit interval of few days can be achieved. Though significant advances have been achieved, development of algorithms that can obtain sharp fused images and carry temporal changes in the image series remain as a challenging task [116].

In this section, we propose a deep learning model that takes full advantages of available temporal and spatial information to enhance spatial and temporal resolutions of remote sensing images as shown in Fig. 27. We propose an arithmetic fusion module to carry temporal changes in image series into high-resolution fused images. A single satellite usually has a low revisit frequency for the same area, and different satellites asynchronously sample the same area. For the time stamps shown in Fig. 27 from $t_1$ to $t_n$, different resolution images from different systems may be available, our ultimate goal is to produce a set of high-resolution images that can achieve a dense sampling of the Earth for change detection. Our main contributions in this study are:

1. We propose a deep model that performs arithmetic operations in feature space to fuse multimodal temporal remote sensing images. The arithmetic operation can effectively carry temporal changes and obtain high-resolution fused images, making it suitable for change detection applications.

2. We successfully apply the proposed model to fuse historical satellite image pairs including Sentinel-2 satellite images (10m spatial resolution) and NAIP aerial images (1m spatial resolution) to predict future NAIP images. To the best of our knowledge, this is the first attempt to bridge the 10× resolution gap in remote sensing images.

This section is organized as follows. Section 4.2.1 reviews related work. Section 4.2.2 describes our proposed model. Section 4.2.4 introduces our experimental setups and s. Section 4.2.5 presents experimental results and discussions, and Section 4.2.7 summarizes this section.

**4.2.1 RELATED WORK**

Traditional multitemporal fusion algorithms for remote sensing images can be grouped into three categories including 1) filter-based, 2) unmixing-based and 3) learning-based methods. In
recent years, Geostatistics and deep learning have been introduced into this application and deep learning based fusing methods achieved superb results.

In filter-based methods, image pixels in the fused image are calculated by selecting and weighting similar neighboring pixels from input images. The most popular classic spatial and temporal adaptive reflectance fusion model (STARFM) builds a simple approximating relationship between HSLT and LSHT pixels and searches similar neighboring pixels, based on spectral, temporal, and location distance to generate the fused image [117]. STARFM was improved by Zhu et al. [118] by assigning different coefficients for homogeneous and heterogeneous pixels. Shen et al. [119] made further development by considering sensor observation differences. Filter-based methods require paired fine and course images from same day for training, which is not always possible in practice.

Zurita-Milla et al. [120] introduced an unmixing-based fusion method where the synthetic images were generated using the spatial information of Landsat/TM data and the spectral information of medium-resolution imaging spectrometer (MERIS) data. This method was later improved by
the same research group [121]. The unmixing-based methods outperformed filter-based methods. However, the methods assumed that there are no significant changes between the images to be fused that is unrealistic in most cases.

Learning-based methods such as sparse representation learning were proposed in recent years [122, 123], where a dictionary was first learned from different image modalities and a fused image was then generated by selecting and weighting elements in the learned dictionary. In those algorithms, feature extraction, dictionary learning, sparse coding, and image reconstruction were carried out separately which ultimately increases complexity of the algorithms. In addition, changes between different images were not well addressed.

Recently, Area-To-Point Regression Kriging (ATPRK) based on Geostatistics technique was first introduced by Wang et al. for image fusion [124]. Later on, they applied ATPRK to fuse Landsat-8 Operational Land Imager (OLI) and Sentinel-2 Multispectral Imager (MSI) data and achieved better performances than STARFM [125], and it can address temporal changes. However, ATPRK is computationally expensive and fused images are usually not sharp.

In the past few years, deep learning has made numerous contributions in computer vision [126], natural language processing [46, 127], speech recognition [128, 129], and remote sensing [70, 71]. For example, Dong et al. developed a convolutional neural network (CNN) for image super-resolution [88]. Motivated by the CNN model, Song et al. [130] applied CNN to fuse MODIS and Landsat images. Li et al. improved the model by introducing the sensor bias-driven fusion method [131]. Shao et al. developed an extended super-resolution CNN (ESRCNN) model to blend Landsat-8 OLI and Sentinel-2 MSI data [116]. Chen et al. proposed a generative adversarial network (GAN) for feature level image fusion for Landsat/Sentinel-2 images [132] during their over-lapping period. Their model achieved better results than those by non-deep learning methods.

Though deep learning based models achieved excellent resolution enhancement in image fusion, it is still a challenging task for most of the deep models to handle temporal changes in image series. In change detection or change monitoring applications, those changes are the most important attributes to focus on. While the ATPRK model can take care of temporal changes, its fusion results usually have inferior resolutions than those produced by deep models. In this paper,
we propose a simple yet effective arithmetic fusion approach that can not only achieve resolution enhancement but also capture temporal changes. We evaluate our model on four s, including Landsat-8, Sentinel-2, NAIP and WV-2 for fusion. Experimental results show that temporal changes in these s can be accurately captured by our proposed model in the fused high-resolution images, potentially facilitating subsequent change detection tasks.

4.2.2 METHODOLOGY

Proposed Model

Fig. 28. Diagram of the proposed deep learning fusion model.

Fig. 28 shows the diagram of the proposed approach. We put low- and high-resolution images collected at $t_1$ and low-resolution image at $t_2$ as inputs. We then perform an arithmetic operation to approximate high-resolution image at $t_2$. The arithmetic operation is performed in feature space of the deep learning model. At convolutional layers in the deep model, we subtract features extracted from low-resolution image at $t_1$ from the features extracted from high-resolution image at $t_1$, and add features extracted from low-resolution image at $t_2$. During training, we provide the model low- and high-resolution image pairs from $t_1$ and $t_2$. The arithmetic operated features from these pairs
are then used to reconstruct high-resolution images at $t_2$.

Our research goal is to combine low- and high-resolution images at $t_1$ with low-resolution images at $t_2$ to predict high-resolution images at $t_2$. If there are changes from $t_1$ to $t_2$ that are captured in the low-resolution images at $t_2$, the arithmetic operation will add these changes to high-resolution images at $t_1$ to produce high-resolution images at $t_2$. Low-resolution images have contours for the changes but with less details. We use the arithmetic operation to carry the contours from low-resolution images at $t_1$ and $t_2$ and add details from high-resolution images at $t_1$ to reconstruct high-resolution images at $t_2$. In this study, we utilize the popular U-Net [1] and the recent HRNet [2] architectures as backbone for the deep learning model.

**U-Net Architecture**

Fig. 29. Proposed fusion model with U-Net [1] backbone. In each of the convolutional feature maps, we subtract the low-resolution image’s features at $t_1$ from these extracted from the high-resolution image at $t_1$, and add these features computed from the low-resolution image at $t_2$ back into the feature maps to reconstruct the high-resolution image at $t_2$.

Fig. 29 shows the U-Net architecture backbone for image fusion. First, we use a shared U-Net structure to encode the three input images. The encoder structure has five convolutional layers, and each convolutional layer is followed by a batch normalization layer and a ReLu activation layer. At each convolutional layer in the feature space, we subtract low-resolution image at $t_1$ from high-resolution image at $t_1$ and add low-resolution image at $t_2$. We hope that the subtraction and addition
will capture the changes over time. Then, outputs of the encoder are fed into the decoder to produce high-resolution image at $t_2$. There are also skip connections to copy the operated features from the encoder to the decoder at the same level of the convolutional layers. During training, histograms of all images are matched to the same reference image (low-resolution image at $t_2$ in our study).

**HRNet Architecture**

In the U-Net architecture, image feature maps are downsampled through polling to lower resolution in the encoder and up-sampled in the decoder to match the input image size. High-resolution information is not kept in this downsampling and upsampling process, instead, high-resolution information is copied to the decoder through the skip connections. On the contrary, the HRNet architecture maintains all level resolution channels throughout the whole process during learning such that detail information are better preserved in the reconstructed images. HRNet is now becoming the mainstream in many computer vision applications [2].

Fig. 30 shows our proposed HRNet architecture. It also has five convolutional layers where each layer is followed by a normalization layer and a ReLu layer. It has three resolution stages including high-, medium- and low-resolution as shown Fig. 30. In each convolutional layer at the high-resolution stage, we perform the same feature level arithmetic operations as that in the U-Net architecture. We also conducted experiments by applying the arithmetic operations at all resolution stages but only obtained marginal performance improvement with significant increase of model complexity. Therefore, we choose this relative simple structure. A similar histogram matching process is applied to all the images before training.

**4.2.3 LOSS FUNCTIONS**

To train the proposed fusion model, we use the popular mean square error (MSE) and the well-recognized metric named high-frequency error norm normalized (HFENN) [133] as cost functions. The MSE metric is defined as,

$$MSE = \frac{1}{WHC} \sum_{i=1}^{W} \sum_{j=1}^{H} \sum_{k=1}^{C} (I_{i,j,k} - \hat{I}_{i,j,k})^2$$

(16)
Fig. 30. Proposed fusion model with HRNet [2] backbone. In the high-resolution stage, we perform the same subtraction and addition arithmetic operations as that in the feature maps of the U-Net backbone model. Adding the arithmetic operations to other resolution stages did not significantly improve performance.

where \( I \) and \( \hat{I} \) denote ground truth and predicted high-resolution image at \( t_2 \), and \( W, H \) and \( C \) represent width, height and number of channels in the images, respectively. The HFENN cost function is defined as,

\[
HFENN = \frac{||\text{LoG}(I) - \text{LoG}(\hat{I})||_2}{||\text{LoG}(\hat{I})||_2}
\]  

(17)

where \( \text{LoG} \) denotes the Laplacian of Gaussian operator, which captures high frequency information in the fused images [134]. The total cost function used in our study is then defined as the combination of the two,

\[
L = MSE + 10 \times HFENN
\]  

(18)

We put a larger weight (10) on \( HFENN \) such that the model can focus on high frequency details. The total loss function \( L \) not only tells the pixel-wise difference between two images in general,
but also put more emphasis on high frequency component differences between the two images.

Performance Metrics

We use five performance metrics to evaluate different models in our study including peak signal to noise ratio (PSNR), structural similarity (SSIM), spectral angle mapper (SAM), relative dimensionless global error (ERGAS), and root mean square error (RMSE) [78]. These metrics are defined as follows.

$$PSNR = 10 \log (\frac{R^2}{MSE})$$  \hspace{1cm} (19)

where $R$ is the maximum range of input image data type. A higher PSNR value indicates a better image quality of the reconstructed high-resolution image.

$$SSIM = \frac{(2\mu_I \mu_J + L_1)(2\sigma_{IJ} + L_2)}{(\mu_I^2 + \mu_J^2 + L_1)(\sigma_I^2 + \sigma_J^2 + L_2)}$$  \hspace{1cm} (20)

where $\mu_I$ and $\mu_J$ represent mean pixel values of ground truth and its predicted image, $\sigma_{IJ}$ denotes covariance between ground truth and the predicted image, and $L_1$ and $L_2$ are predefined constants. A higher value of SSIM also indicates a better image quality.

$$SAM = \frac{1}{N} \sum_{i=1}^{N} \arccos \left( \frac{\sum_{j=1}^{C} (I_{ij} \hat{I}_{ij})}{\sqrt{\sum_{j=1}^{C} I_{ij}^2} \sqrt{\sum_{j=1}^{C} \hat{I}_{ij}^2}} \right)$$  \hspace{1cm} (21)

where $N$ indicates the total number of pixels in the fused image and $C$ is the number of bands in the image. The SAM metric is used to measure spectral distortion of an image. A small value indicates a better image quality.

$$RMSE = \sqrt{\frac{1}{HWC} \sum_{i=1}^{H} \sum_{j=1}^{W} \sum_{k=1}^{C} (I_{i,j,k} - \hat{I}_{i,j,k})^2}$$  \hspace{1cm} (22)

$$ERGAS = \frac{100 \times l}{h} \sqrt{\frac{1}{C} \sum_{i=1}^{C} \frac{RMSE(I_i)^2}{u_i}}$$  \hspace{1cm} (23)

where $h$ and $l$ denote spatial resolutions of high- and low-resolution images, respectively. RMSE is used to calculate global radiometric difference between ground truth and the fused image. ERGAS
is used to evaluate the quality of fused image based on normalized average error of each band in the processed image. Smaller values of RMSE and ERGAS represent better image qualities.

4.2.4 EXPERIMENTAL SETUP

Datasets

We utilize WV-2, Sentinel-2, Landsat-8 and NAIP images to evaluate the proposed fusion model. Details of these imagery systems and data for both training and testing are listed in Table 8 and Table 9. For WV-2 images, the study area covers the Millerovo airport which is located in Russia and the images contain airplanes, buildings, and vegetation etc. We collect cloud free images at two time points (04/2014 and 07/2015) with image resolution of 0.46m for training. For NAIP, Sentinel-2 and Landsat-8 images, we choose Norfolk, VA, as our study area due to the rapid urban development over times in this region. Fig. 31 shows some training image pairs in this study. Image resolutions of NAIP, Sentinel-2, and Landsat-8 images are 1m, 10m and, 30m, respectively. For evaluation, we choose twenty smaller images with different dimensions in the adjacent area for each of the experiments. Time difference between the two time points is about two years. Significant temporal changes can be observed between the image pairs. In addition, we collected 45 testing images from Palm Jumeirah, Dubai, which is located in another country for generalization capability testing. This consists of low-resolution Landsat-8 images and paired high-resolution Sentinel-2 images. The time difference between the two time points is about four years and significant temporal changes present between image pairs.

Preprocessing

Different satellite images contain different number of channels. In this study, we only consider to fuse RGB channels. As we apply arithmetic operations to different images collected at different times, these images need to be registered. We utilize the projection distortion based on control points method provided by Matlab R2020a (MathWorks Inc.) for registration. We also perform data normalization and histogram matching for better performances.
### Table 8: Specifications of datasets used in the study.

<table>
<thead>
<tr>
<th>System</th>
<th>Modality</th>
<th>Bands</th>
<th>Resolution</th>
<th>Revisit frequency</th>
<th>Charge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Landsat-8</td>
<td>Satellite</td>
<td>8</td>
<td>30 m</td>
<td>16 days</td>
<td>Free</td>
</tr>
<tr>
<td>Sentinel-2</td>
<td>Satellite</td>
<td>12</td>
<td>10 m</td>
<td>10 days</td>
<td>Free</td>
</tr>
<tr>
<td>NAIP</td>
<td>Aerial</td>
<td>4</td>
<td>1 m</td>
<td>3 years</td>
<td>Variable</td>
</tr>
<tr>
<td>WV-2</td>
<td>Satellite</td>
<td>8</td>
<td>.46 m</td>
<td>≥3 days</td>
<td>Expensive</td>
</tr>
</tbody>
</table>

### Experiments

**Experiment 1:** We use the WV-2 dataset collected at the two time points for image fusion. We downsample the images by averaging pixels with different factors ($2 \times, 4 \times, 6 \times, 10 \times$) to simulate low-resolution images at both time points, making the low- and high-resolution images at the same time point perfectly registered. We expect better performances than other experiments where the images registration is not perfect. We will test the upper limit on the number of times the high-resolution images can be downsampled where good fused high-resolution images are still achievable.

**Experiment 2:** Landsat-8 images have a resolution of 30$m$ and Sentinel-2 images have a resolution of 10$m$. The images collected from Norfolk, VA, are around two years apart. We fuse Landsat-8 and Sentinel-2 images at $t_1$ with Landsat-8 images at $t_2$ to generate Sentinel-2 images at $t_2$. There is a factor of 3× difference in resolutions between the two image datasets, and we expect good performances though the registration is not perfect.

**Experiment 3:** Sentinel-2 images have a resolution of 10$m$ and NAIP images have a resolution of 1$m$. The images collected from Norfolk, VA, are around two years apart. We fuse Sentinel-2 images and NAIP images at $t_1$ with Sentinel-2 images at $t_2$ to generate NAIP images at $t_2$. This is the most challenging task in our study with a resolution difference of 10× between the two modalities. We will test the proposed fusion model on this dataset with an imperfect registration.
**Table 9**: Details of training and testing data.

<table>
<thead>
<tr>
<th>Experiments</th>
<th>Type</th>
<th>Dimension</th>
<th>No. of Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>WV-2 (<em>Experiment 1</em>)</td>
<td>Training</td>
<td>1024x1024x3</td>
<td>1</td>
</tr>
<tr>
<td>WV-2 (<em>Experiment 1</em>)</td>
<td>Testing</td>
<td>200x300x3-700x800x3</td>
<td>20</td>
</tr>
<tr>
<td>Landsat-8 &amp; Sentinel-2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(<em>Experiment 2</em>)</td>
<td>Training</td>
<td>406x766x3</td>
<td>1</td>
</tr>
<tr>
<td>Landsat-8 &amp; Sentinel-2</td>
<td>Testing</td>
<td>200x200x3-300x300x3</td>
<td>20</td>
</tr>
<tr>
<td>Landsat-8 &amp; NAIP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(<em>Experiment 3</em>)</td>
<td>Training</td>
<td>1500x1500x3</td>
<td>1</td>
</tr>
<tr>
<td>Landsat-8 &amp; NAIP</td>
<td>Testing</td>
<td>100x200x3-500x800x3</td>
<td>20</td>
</tr>
<tr>
<td>Landsat-8 &amp; Sentinel-2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(<em>Experiment 4</em>)</td>
<td>Testing</td>
<td>100x200x3-400x600x3</td>
<td>45</td>
</tr>
</tbody>
</table>

**Experiment 4**: To evaluate generalization capability of the proposed models, we apply models trained with data collected in Norfolk, VA, USA to data collected in Palm Jumeirah in Dubai without model fine-tuning. Palm Jumeirah has been experiencing rapid growth of urbanization over the past years. The time gap between the image pairs we collected is about 4 years and significant temporal changes present in the image pairs. Low-resolution images captured by Landsat-8
Fig. 31. Training image examples. Resolution: Landsat-8 (30m), Sentinel-2 (10m), NAIP (1m) and WV-2 (.46m). Large red boxes display zoomed in regions in the corresponding small boxes. Significant temporal changes observed. Image contrast enhanced for better display.

have 30m resolution and high-resolution images are from Sentinel-2 with a resolution of 10m.

**Statistical Tests:**

We perform two statistical (parametric and non-parametric) tests to show if performance differences between the proposed model (best) and the best baseline model are significant. Parametric tests are performed when the data size is large and the data are normally distributed whereas non-parametric tests are conducted if data size is small and the data doesn’t assume normality [135]. For parametric test, we apply the Two sample t-test and for non-parametric test, we perform the Wilcoxon rank sum test. Both tests assume the involved samples are independent [136].

**Competing Methods and Abbreviations**

Our proposed model can combine different backbones and loss functions, resulting in four methods: “U-Net + MSE” (UMSE), “U-Net + MSE + HFENN” (UMSEh), “HRNet + MSE” (HMSE) and “HRNet + MSE + HFENN” (HMSEh). We also compare our model to the
ATPRK method [124], the deep models ESRCNN [116], GAN [132] and STFGAN [137]. For the implementations of ESRCNN, GAN and STFGAN, we follow the same settings utilized in the original papers and combine all temporal images to form inputs (similar to our proposed models) for fair comparisons.

4.2.5 RESULTS AND DISCUSSIONS

Hyper-parameter Determination

In all of the three experiments, we set input image patch dimension to $32 \times 32 \times 3$, convolutional kernel size to $3 \times 3$, batch size to 32, and number of training epochs to 35. We include five convolutional layers in both the U-Net and HRNet backbones where each convolutional layer is followed by a batch normalization layer and a ReLu layer. We fix the dropout rate at 0.1 for all dropout layers and use the Adam optimizer during training with default learning rates. All experiments are trained on a high performance computing cluster. The model is implemented using the Keras platform.

Results

Results of Experiment 1

Table 10 shows performance metrics by different models where we downsampled the high-resolution WV-2 images as perfectly registered low-resolution images for fusion. We tested four down-sampling factors including $2 \times, 4 \times, 6 \times, \text{and } 10 \times$ and compared the proposed model with the ATPRK, ESRCNN and GAN models. The ATPRK model shows consistently poor performances for all cases. ESRCNN produced competitive results for the $2 \times$ case and achieved the best PSNR of 40.50dB. However, for most of the other cases, the proposed model with the U-Net backbone and MSE loss achieved better results. The HRNet backbone also performed well and achieved slightly worse results than the U-Net backbone.

7https://keras.io/api/
8https://github.com/mldeeplearning123/Image-Fusion-
Table 10: Results of Experiment 1. ATPRK, ESRCNN and GANs are competing methods. All others are the proposed model with different backbones or different loss functions. Results are averages on 20 testing images and shown in the format of mean(\textit{std}). Best results are shown in bold.

<table>
<thead>
<tr>
<th>Scale</th>
<th>Method</th>
<th>PSNR</th>
<th>SSIM</th>
<th>SAM</th>
<th>RMSE</th>
<th>ERGAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>10x</td>
<td>ATPRK [124]</td>
<td>17.85 (5.75)</td>
<td>0.48 (0.12)</td>
<td>2.51 (0.40)</td>
<td>34.27 (11.49)</td>
<td>26.18 (3.45)</td>
</tr>
<tr>
<td></td>
<td>ESRCNN [116]</td>
<td>39.43 (3.49)</td>
<td>0.95 (0.01)</td>
<td>2.70 (0.77)</td>
<td>2.91 (1.01)</td>
<td>12.56 (2.88)</td>
</tr>
<tr>
<td></td>
<td>GAN [132]</td>
<td>36.87 (2.41)</td>
<td>0.92 (0.01)</td>
<td>4.01 (1.39)</td>
<td>3.76 (0.77)</td>
<td>14.26 (2.14)</td>
</tr>
<tr>
<td></td>
<td>STFGAN [137]</td>
<td>36.51 (2.52)</td>
<td>0.93 (0.01)</td>
<td>4.17 (1.28)</td>
<td>3.96 (1.18)</td>
<td>14.97 (3.72)</td>
</tr>
<tr>
<td></td>
<td>UMSE</td>
<td>40.30 (2.91)</td>
<td>0.96 (0.01)</td>
<td>2.47 (0.55)</td>
<td>2.58 (0.77)</td>
<td>10.52 (2.22)</td>
</tr>
<tr>
<td></td>
<td>UMS(\text{e}h)</td>
<td>40.05 (2.65)</td>
<td>0.96 (0.01)</td>
<td>2.71 (0.98)</td>
<td>2.64 (0.74)</td>
<td>10.66 (2.25)</td>
</tr>
<tr>
<td></td>
<td>HMSE</td>
<td>38.80 (2.47)</td>
<td>0.94 (0.01)</td>
<td>2.94 (0.68)</td>
<td>3.03 (0.78)</td>
<td>14.77 (2.49)</td>
</tr>
<tr>
<td></td>
<td>HMSE(\text{e}h)</td>
<td>39.50 (2.77)</td>
<td>0.95 (0.01)</td>
<td>2.74 (0.64)</td>
<td>2.82 (0.80)</td>
<td>12.89 (2.30)</td>
</tr>
<tr>
<td>6x</td>
<td>ATPRK</td>
<td>19.32 (2.93)</td>
<td>0.54 (0.13)</td>
<td>2.31 (1.01)</td>
<td>29.26 (11.43)</td>
<td>24.69 (4.02)</td>
</tr>
<tr>
<td></td>
<td>ESRCNN</td>
<td>40.93 (2.67)</td>
<td>0.95 (0.01)</td>
<td>4.97 (1.07)</td>
<td>2.38 (0.64)</td>
<td>10.29 (1.83)</td>
</tr>
<tr>
<td></td>
<td>GAN</td>
<td>39.34 (1.85)</td>
<td>0.93 (0.01)</td>
<td>5.83 (1.27)</td>
<td>2.80 (0.56)</td>
<td>12.43 (1.90)</td>
</tr>
<tr>
<td></td>
<td>STFGAN</td>
<td>40.54 (2.53)</td>
<td>0.95 (0.01)</td>
<td>4.17 (0.92)</td>
<td>2.50 (0.88)</td>
<td>10.43 (2.94)</td>
</tr>
<tr>
<td></td>
<td>UMSE</td>
<td>40.65 (2.05)</td>
<td>0.95 (0.01)</td>
<td>5.16 (1.15)</td>
<td>2.42 (0.53)</td>
<td>10.24 (1.61)</td>
</tr>
<tr>
<td></td>
<td>UMS(\text{e}h)</td>
<td>40.72 (2.62)</td>
<td>0.95 (0.01)</td>
<td>4.87 (1.08)</td>
<td>2.43 (0.64)</td>
<td>11.44 (1.76)</td>
</tr>
<tr>
<td></td>
<td>HMSE</td>
<td>40.97 (2.17)</td>
<td>0.95 (0.01)</td>
<td>4.87 (1.15)</td>
<td>2.35 (0.55)</td>
<td>9.92 (1.72)</td>
</tr>
<tr>
<td></td>
<td>HMSE(\text{e}h)</td>
<td>41.14 (2.47)</td>
<td>0.95 (0.01)</td>
<td>4.99 (1.10)</td>
<td>2.33 (0.59)</td>
<td>9.98 (1.70)</td>
</tr>
<tr>
<td>4x</td>
<td>ATPRK</td>
<td>20.43 (2.92)</td>
<td>0.58 (0.14)</td>
<td>5.37 (1.06)</td>
<td>26.32 (11.14)</td>
<td>23.24 (4.61)</td>
</tr>
<tr>
<td></td>
<td>ESRCNN</td>
<td>41.63 (2.56)</td>
<td>0.95 (0.01)</td>
<td>4.76 (0.99)</td>
<td>2.19 (0.59)</td>
<td>9.51 (2.19)</td>
</tr>
<tr>
<td></td>
<td>GAN</td>
<td>39.73 (1.08)</td>
<td>0.94 (0.01)</td>
<td>5.67 (1.28)</td>
<td>2.64 (0.34)</td>
<td>11.43 (2.01)</td>
</tr>
<tr>
<td></td>
<td>STFGAN</td>
<td>39.63 (5.35)</td>
<td>0.94 (0.06)</td>
<td>4.45 (1.41)</td>
<td>3.32 (3.08)</td>
<td>13.29 (10.01)</td>
</tr>
<tr>
<td></td>
<td>UMSE</td>
<td>41.87 (2.32)</td>
<td>0.95 (0.01)</td>
<td>5.02 (1.21)</td>
<td>2.12 (0.55)</td>
<td>10.31 (2.36)</td>
</tr>
<tr>
<td></td>
<td>UMS(\text{e}h)</td>
<td>42.16 (2.54)</td>
<td>0.96 (0.01)</td>
<td>4.76 (1.14)</td>
<td>2.06 (0.57)</td>
<td>9.02 (2.04)</td>
</tr>
<tr>
<td></td>
<td>HMSE</td>
<td>41.82 (2.38)</td>
<td>0.96 (0.01)</td>
<td>4.91 (1.16)</td>
<td>2.13 (0.55)</td>
<td>9.70 (2.09)</td>
</tr>
<tr>
<td></td>
<td>HMSE(\text{e}h)</td>
<td>40.32 (1.66)</td>
<td>0.95 (0.01)</td>
<td>4.91 (1.07)</td>
<td>2.49 (0.45)</td>
<td>13.04 (2.84)</td>
</tr>
<tr>
<td>2x</td>
<td>ATPRK</td>
<td>22.98 (3.42)</td>
<td>0.68 (0.17)</td>
<td>2.48 (0.90)</td>
<td>19.71 (9.99)</td>
<td>20.95 (6.19)</td>
</tr>
<tr>
<td></td>
<td>ESRCNN</td>
<td>42.19 (2.53)</td>
<td>0.97 (0.01)</td>
<td>3.32 (0.71)</td>
<td>2.06 (0.58)</td>
<td>9.31 (2.22)</td>
</tr>
<tr>
<td></td>
<td>GAN</td>
<td>39.90 (3.82)</td>
<td>0.96 (0.01)</td>
<td>4.33 (0.58)</td>
<td>2.62 (1.11)</td>
<td>13.33 (3.58)</td>
</tr>
<tr>
<td></td>
<td>STFGAN</td>
<td>38.81 (4.34)</td>
<td>0.96 (0.02)</td>
<td>4.53 (1.55)</td>
<td>3.24 (4.94)</td>
<td>14.62 (4.81)</td>
</tr>
<tr>
<td></td>
<td>UMSE</td>
<td>42.19 (2.53)</td>
<td>0.97 (0.01)</td>
<td>3.32 (0.71)</td>
<td>2.02 (0.58)</td>
<td>9.31 (2.22)</td>
</tr>
<tr>
<td></td>
<td>UMS(\text{e}h)</td>
<td>42.16 (2.19)</td>
<td>0.97 (0.01)</td>
<td>3.29 (0.58)</td>
<td>2.05 (0.53)</td>
<td>8.71 (2.38)</td>
</tr>
<tr>
<td></td>
<td>HMSE</td>
<td>41.41 (1.76)</td>
<td>0.97 (0.01)</td>
<td>3.77 (1.05)</td>
<td>2.21 (0.46)</td>
<td>10.95 (3.42)</td>
</tr>
<tr>
<td></td>
<td>HMSE(\text{e}h)</td>
<td>42.40 (2.54)</td>
<td>0.98 (0.01)</td>
<td>3.72 (1.00)</td>
<td>2.01 (0.58)</td>
<td>9.20 (2.39)</td>
</tr>
</tbody>
</table>
Results of Experiment 2

Table 11 shows results of fusing Landsat-8 images to generate high-resolution Sentinel-2 images, where the resolution difference is $3 \times$. The proposed models achieved similar (SSIM by GAN and ATPRK) or better results (all other cases) than the competing methods (ATPRK, ESRCNN and GAN) in terms of all the five metrics. The HRNet backbone with the high frequency loss of the proposed model (HMSEh) outperformed the U-Net backbone in terms of all the metrics except PSNR.

Results of Experiment 3

Image fusion results from Sentinel-2 to NAIP are listed in Table 11, where the resolution difference is $10 \times$ and image registration between the two modalities are imperfect. The proposed method with U-Net as backbone and MSE + HFENN as loss functions (UMSEh) won four times out of the five performance metrics. The GAN model did not perform well. Surprisingly, the ATPRK model also achieved the same best SSIM (0.48) though its other metrics are not close to that of UMSEh. In summary, from Tables 10 and 11, it is clear that the proposed model achieved the best overall results.

Results of Experiment 4

Table 12 shows transfer learning performances of the proposed models trained with data collected in Norfolk, VA and applied to data collected in Palm Jumeirah. The objective of the fusion is to generate high-resolution Sentinel images from low-resolution Landsat-8 images across two-time points with temporal changes. Our proposed models achieved the best performances in all of the five metrics as compared to ATPRK, ESRCNN, and GANs. In particular, the U-Net backbone with high-frequency loss (UMSEh) achieved the best PSNR, SSIM, and RMSE. The HRNet bone with regular loss obtained the best SAM and ERGAS. Fig. 32 shows some of the fused images in the Palm Jumeirah area. Quantitatively, STFGAN is similar to GAN and GAN is slightly better, we only show results by GAN for all the experiments to save space. Visual inspection shows that GAN and U-NET back-boned models performed well. The color contrast in images produced by other computing methods (ESCNN, HMSE, HMSEh) is not matched to that in the ground truth
Table 11: Results of Experiment 2 (Landsat-8 and Sentinel fusion) and Experiment 3 (Sentinel and NAIP fusion). Results are averages on 20 testing images.

<table>
<thead>
<tr>
<th>Method</th>
<th>Experiment 2</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR</td>
<td>SSIM</td>
<td>SAM</td>
<td>RMSE</td>
<td>ERGAS</td>
</tr>
<tr>
<td>ATPRK</td>
<td>23.40 (1.86)</td>
<td>0.70 (0.07)</td>
<td>6.81 (1.23)</td>
<td>18.87 (4.00)</td>
<td>19.57 (1.65)</td>
</tr>
<tr>
<td>ESRCNN</td>
<td>27.34 (1.99)</td>
<td>0.80 (0.04)</td>
<td>5.91 (1.05)</td>
<td>10.15 (2.61)</td>
<td>15.83 (2.27)</td>
</tr>
<tr>
<td>GAN</td>
<td>28.05 (1.74)</td>
<td>0.82 (0.03)</td>
<td>5.89 (1.03)</td>
<td>9.36 (2.02)</td>
<td>14.10 (0.89)</td>
</tr>
<tr>
<td>STFGAN</td>
<td>28.28 (1.69)</td>
<td>0.81 (0.04)</td>
<td>5.90 (0.59)</td>
<td>9.09 (1.88)</td>
<td>14.02 (0.88)</td>
</tr>
<tr>
<td>UMSE</td>
<td>29.11 (1.95)</td>
<td>0.83 (0.04)</td>
<td>5.50 (0.79)</td>
<td>9.12 (1.98)</td>
<td>13.57 (1.02)</td>
</tr>
<tr>
<td>UMSEh</td>
<td>29.55 (1.84)</td>
<td>0.85 (0.03)</td>
<td>5.95 (0.95)</td>
<td>7.80 (1.74)</td>
<td>12.84 (0.89)</td>
</tr>
<tr>
<td>HMSE</td>
<td>28.64 (1.87)</td>
<td>0.82 (0.04)</td>
<td>5.39 (0.85)</td>
<td>8.87 (2.06)</td>
<td>13.95 (1.20)</td>
</tr>
<tr>
<td>HMSEh</td>
<td>28.57 (1.91)</td>
<td>0.82 (0.04)</td>
<td>5.57 (0.64)</td>
<td>9.41 (2.10)</td>
<td>13.76 (1.25)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Experiment 3</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR</td>
<td>SSIM</td>
<td>SAM</td>
<td>RMSE</td>
<td>ERGAS</td>
</tr>
<tr>
<td>ATPRK</td>
<td>10.07 (3.90)</td>
<td>0.22 (0.12)</td>
<td>15.33 (6.63)</td>
<td>84.81 (36.33)</td>
<td>20.19 (7.85)</td>
</tr>
<tr>
<td>ESRCNN</td>
<td>9.16 (1.68)</td>
<td>0.14 (0.04)</td>
<td>7.30 (2.54)</td>
<td>96.25 (19.80)</td>
<td>27.97 (9.20)</td>
</tr>
<tr>
<td>GAN</td>
<td>10.72 (4.03)</td>
<td>0.30 (0.11)</td>
<td>5.55 (0.80)</td>
<td>83.93 (37.27)</td>
<td>17.79 (8.07)</td>
</tr>
<tr>
<td>STFGAN</td>
<td>10.45 (2.70)</td>
<td>0.30 (0.09)</td>
<td>6.83 (1.13)</td>
<td>80.05 (24.91)</td>
<td>12.35 (3.86)</td>
</tr>
<tr>
<td>UMSE</td>
<td>10.98 (2.52)</td>
<td>0.22 (0.09)</td>
<td>6.72 (2.17)</td>
<td>79.68 (27.92)</td>
<td>14.51 (5.53)</td>
</tr>
<tr>
<td>UMSEh</td>
<td>12.37 (3.80)</td>
<td>0.33 (0.11)</td>
<td>5.95 (1.23)</td>
<td>70.63 (35.97)</td>
<td>10.62 (7.79)</td>
</tr>
<tr>
<td>HMSE</td>
<td>11.72 (3.42)</td>
<td>0.30 (0.10)</td>
<td>5.54 (0.92)</td>
<td>75.72 (33.64)</td>
<td>13.90 (7.07)</td>
</tr>
<tr>
<td>HMSEh</td>
<td>11.45 (2.99)</td>
<td>0.23 (0.10)</td>
<td>4.67 (0.65)</td>
<td>78.62 (31.08)</td>
<td>20.76 (6.21)</td>
</tr>
</tbody>
</table>

Table 12: Results of Experiment 4 (Transfer learning). All models are trained with images collected from Norfolk, VA and tested on images collected from Palm Jumeirah, Dubai. Results are averages on 45 testing images.

<table>
<thead>
<tr>
<th>Method</th>
<th>Experiment 4: Transfer learning</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR</td>
<td>SSIM</td>
<td>SAM</td>
<td>RMSE</td>
<td>ERGAS</td>
</tr>
<tr>
<td>ATPRK</td>
<td>9.88 (1.37)</td>
<td>0.14 (0.03)</td>
<td>5.35 (1.63)</td>
<td>82.71 (12.96)</td>
<td>30.69 (1.47)</td>
</tr>
<tr>
<td>ESRCNN</td>
<td>13.73 (0.75)</td>
<td>0.31 (0.02)</td>
<td>2.26 (0.50)</td>
<td>52.62 (4.41)</td>
<td>27.68 (0.51)</td>
</tr>
<tr>
<td>GAN</td>
<td>13.74 (0.73)</td>
<td>0.32 (0.02)</td>
<td>2.45 (0.57)</td>
<td>52.56 (4.28)</td>
<td>27.66 (0.50)</td>
</tr>
<tr>
<td>STFGAN</td>
<td>14.08 (0.93)</td>
<td>0.31 (0.04)</td>
<td>2.87 (1.25)</td>
<td>50.66 (5.19)</td>
<td>27.73 (0.69)</td>
</tr>
<tr>
<td>UMSE</td>
<td>14.13 (0.63)</td>
<td>0.33 (0.02)</td>
<td>2.68 (0.41)</td>
<td>50.24 (3.54)</td>
<td>27.44 (0.54)</td>
</tr>
<tr>
<td>UMSEh</td>
<td>14.50 (0.71)</td>
<td>0.34 (0.02)</td>
<td>2.55 (0.46)</td>
<td>48.16 (3.82)</td>
<td>27.44 (0.53)</td>
</tr>
<tr>
<td>HMSE</td>
<td>14.17 (0.64)</td>
<td>0.33 (0.03)</td>
<td>2.23 (0.55)</td>
<td>49.99 (3.57)</td>
<td>27.42 (0.57)</td>
</tr>
<tr>
<td>HMSEh</td>
<td>14.23 (0.63)</td>
<td>0.33 (0.02)</td>
<td>2.76 (0.48)</td>
<td>49.63 (3.55)</td>
<td>27.38 (0.55)</td>
</tr>
</tbody>
</table>
image (H2).

Fig. 32. Image fusion results in Experiment 4 (Transfer learning) by different methods, where H2 is ground truth high-resolution image at the second time point. Image generated by ESRCNN contains noise and histograms of images by HMSE, HMSEh and GAN do not match that of the ground-truth image.

Results of Statistical Tests

We performed statistical tests between the proposed model UMSEh vs GAN, where UMSEH achieved the overall best results and GAN is the best competing baseline model for all of our Experiments as shown in Table 13. For Experiment 1, we report the statistical test for 4x resolution case. Table 13 shows that UMSEh is statistically better than GAN in terms of almost all of the cases except Experiment 3.

Visual Inspection

Fig. 33 shows one of the fused testing images resulted from each of the experiments. The fused images by the ATRPK model (fifth column) are blurry, and color contrast of some images does not match that of the ground truth (fourth column). The ESRCNN model generated images containing more details than these by ATPRK. The GAN model produced sharp outputs with visible details in
### Results of Experiment 1

Low- and high-resolution image pairs are the downsampled by $6 \times$ WV-2 image and its original version.

### Results of Experiment 2

Low- and high-resolution image pairs are Landsat-8 and Sentinel-2 images.

### Results of Experiment 3

Low- and high-resolution image pairs are Sentinel-2 and NAIP images.

**Fig. 33.** Visual inspection of fused images by different models. For each of the experiment results, the first row shows input images and fused results by different models. “Low-$t_1$”, “High-$t_1$” and “Low-$t_2$” are inputs images. “Ground truth” is the high-resolution image at $t_2$ to approximate. The second row shows the zoomed in region in the red box above. Results of the proposed model are from the best combination of backbone and loss function in each of the Experiments. Image contrast enhanced for better display.
Table 13: Statistical tests between UMSEh (best overall) and GAN (best competing model) in all experiments, where $h = 1$ indicates the difference is significant at the 95% significance interval. Experiment 1 is for the 4x resolution case.

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Experiment 1</th>
<th></th>
<th>Experiment 2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Two sample t-test</td>
<td>Wilcoxon rank sum test</td>
<td>Two sample t-test</td>
<td>Wilcoxon rank sum test</td>
</tr>
<tr>
<td></td>
<td>$p$-value</td>
<td>$h$</td>
<td>$p$-value</td>
<td>$h$</td>
</tr>
<tr>
<td>PSNR</td>
<td>$4.0 \times 10^{-4}$</td>
<td>1</td>
<td>$1.50 \times 10^{-3}$</td>
<td>1</td>
</tr>
<tr>
<td>SSIM</td>
<td>$1.0 \times 10^{-4}$</td>
<td>1</td>
<td>$1.79 \times 10^{-4}$</td>
<td>1</td>
</tr>
<tr>
<td>SAM</td>
<td>$2.2 \times 10^{-2}$</td>
<td>1</td>
<td>$1.03 \times 10^{-3}$</td>
<td>1</td>
</tr>
<tr>
<td>RMSE</td>
<td>$4.0 \times 10^{-4}$</td>
<td>1</td>
<td>$1.50 \times 10^{-3}$</td>
<td>1</td>
</tr>
<tr>
<td>ERGAS</td>
<td>$6.0 \times 10^{-4}$</td>
<td>1</td>
<td>$6.86 \times 10^{-4}$</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Experiment 3</th>
<th></th>
<th>Experiment 4</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Two sample t-test</td>
<td>Wilcoxon rank sum test</td>
<td>Two sample t-test</td>
<td>Wilcoxon rank sum test</td>
</tr>
<tr>
<td></td>
<td>$p$-value</td>
<td>$h$</td>
<td>$p$-value</td>
<td>$h$</td>
</tr>
<tr>
<td>PSNR</td>
<td>$2.60 \times 10^{-1}$</td>
<td>0</td>
<td>$1.13 \times 10^{-1}$</td>
<td>0</td>
</tr>
<tr>
<td>SSIM</td>
<td>$4.10 \times 10^{-1}$</td>
<td>0</td>
<td>$3.89 \times 10^{-1}$</td>
<td>0</td>
</tr>
<tr>
<td>SAM</td>
<td>$4.80 \times 10^{-1}$</td>
<td>0</td>
<td>$6.29 \times 10^{-1}$</td>
<td>0</td>
</tr>
<tr>
<td>RMSE</td>
<td>$3.40 \times 10^{-1}$</td>
<td>0</td>
<td>$1.13 \times 10^{-1}$</td>
<td>0</td>
</tr>
<tr>
<td>ERGAS</td>
<td>$1.06 \times 10^{-1}$</td>
<td>0</td>
<td>$6.50 \times 10^{-2}$</td>
<td>1</td>
</tr>
</tbody>
</table>

Experiment 1 (Fig. 9a) but performed the worst in Experiment 3 (Fig. 9c). The proposed model produced much better results, while images generated by the HRNet backbone (HMSEh) are the sharpest by visual inspection.

Images with Temporal Changes

Fig. 34 shows image fusion results when temporal changes are presented in the Landsat-8 and Sentinel-2 dataset. The region is located inside the Norfolk port in Virginia where a fleet was presented at $t_1$ and the fleet left at $t_2$ when the image was collected. The contour of the fleet is still visible in the fused image by the ESRCNN model (Fig. 34c) while our proposed method with different loss functions and backbones (Fig. 34e-f) clearly reflected the change. The ATPRK and GAN algorithms also successfully captured the temporal changes in the fused image.
Fig. 34. Image fusion results with temporal changes. In the ground truth image (a), the zoomed in regions at $t_1$ and $t_2$ show changes captured by the Sentinel-2 satellite image where a cargo ship docked in the Norfolk port at $t_1$ and left at $t_2$. The ESRCNN failed to reflect the change in the fused image (c). ATPRK (b), GAN (d) and our models successfully captured this change in the fused images (d,e and f). UMSE: “U-Net + MSE”. HMSE: “HRNet + MSE”.

Images with High-frequency Details

Fig. 35 shows one fused image in Experiment 1 of $6 \times$ with high-frequency details inside the red square area. It is observed that the HRNet backbone (Fig. 35b, HMSEh) can better keep these high-frequency details as compared to the U-Net backbone (Fig. 35a, UMSEh), producing sharper details.

Upper Limit of Downsampling Factor

We performed experiment on the WorldView-2 dataset to find out to what extend we can downsample the image and the proposed model still can achieve good fused images given that the image registration is perfect. We continued to downsample the image with factors of $12 \times$ and $16 \times$ and applied the proposed model for image fusion. Fig. 36b-d) show the fused images with factors $10 \times$, $12 \times$ and $16 \times$, respectively. It is observed that the fused image with the factor of $16 \times$ is blurry and we conclude that the upper limit may be $12 \times$ with which our proposed model can still perform
Fig. 35. Visual comparison for high-frequency details of images fused by the U-Net and HRNet backbones (Experiment 1 with $6\times$). The large red box contains the zoomed in region in the small box.

well if the registration is perfect.

4.2.6 DISCUSSIONS

For environmental monitoring or land surface change detection, a remote sensing imagery system that can densely sample a particular region with high spatial resolution is desired. The proposed model is an attempt to fuse multiple satellite image modalities to generate high-resolution in both temporal and spatial domains.

Our proposed methods are simple yet fast and effective to capture temporal changes. We utilized arithmetic operations in feature space to achieve this goal. In the Encoder parts of U-Net or HRNet, we subtracted features of low-resolution images from features of high-resolution images at $t_1$, and added features of low-resolution images at $t_2$. These low-resolution features represent contours, and high-resolution features represent both contours and detail texture information in the images. We utilized these arithmetic operations to explicitly reflect low-resolution temporal contour changes. Though high-resolution information was not provided at input, we assume that high-resolution texture information are correlated with shapes of contours and can be learned from similar texture patches in training data so that the changed contours at $t_2$ can be correctly filled with details. We believe direct deep learning models such as ESRCNN will eventually figure out the arithmetic relationship between the three inputs and output and correctly reflect temporal changes, if more data are provided. The explicit arithmetic feature operations guided the training and made
the learning much easier, which will be further investigated in our future work.

If images are perfectly registered (Experiment 1), our proposed models can handle up to 12× spatial resolution differences (PSNR of UMSE reached 40.30\text{dB} at 10× down-sampling) and carry temporal changes to the fused high-resolution images. The U-Net backbone outperforms the HR-Net backbone in most of all the metrics for 10× and 4×. For 6× and 2×, HRNet backbone outperforms the U-Net backbone (Table 10). However, Fig. 35 shows that the HRNet backbone retained more details in the fused image, making it much sharper. These performance metrics may not be ideal to capture high-frequency details in image and better matrices are desired. Our proposed model shows potential to fuse different remote image modalities in practice. The competing method ARPRK failed to generate sharp image while ESRCNN failed to capture temporal changes. Neither of them could achieve the both at the same time. The GAN model produced sharp images in Experiment 1 but quantitatively did not perform well (GAN, STFGAN) and failed the fusion task in Experiment 3 as compared to our proposed models.

The resolution difference between Sentinel-2 and Landsat-8 is 3× (10m vs. 30m) and the fusion results visually look good. However, the best PSNR dropped to 28.36\text{dB} (by UMSEh) as compared to the best case at 4× in Experiment 1 (40.37\text{dB} by UMSEh), partially because the image registration is hardly perfect. However, the temporal changes were captured nearly perfectly (Fig. 34). The fused images ideally will have an 10m spatial resolution and are suitable for large object detection including boat, building etc. Since both of the modalities are free of charge so that it has potential for practical nonessential applications.

NAIP provides very high-resolution (1m) time-series images, which is designed for agriculture application monitoring. One limitation is that NAIP is not free of charge. As shown in Table 11, PSNR of the fused images from free Sentinel-2 images can reach 20.69\text{dB} by UMSEh. However, it is still not clear and needs further investigation if the fused images can be used for agriculture monitoring applications. As compared to Experiment 2, the resolution difference in Experiment 3 is 10× so that it is not surprising that the PSNR performance metrics dropped significantly.

Our proposed models are generalizable. In the transfer learning setting (Experiment 4), we trained the proposed models with data collected in Norfolk, USA, and applied the trained models without fine-tuning to fuse images collected in a different country (Palm Jumerirah). Results in Table 12 show that all performance metrics degraded as compared to these in Table 11 (Experiment
Fig. 36. Image registration effect. With imperfect registration in experiment 3, the resolution difference of 10× between Sentinel-2 and NAIP images are much more difficult to bridge (image in (a) is blurry). With the perfect registration in experiment 1, even larger resolution differences resulted in much sharper fused images (b-d).
However, the fused images generated by our models (U-Net backbone model) still outperformed all other competing methods as shown in Table 12.

In general, our proposed models can accurately capture temporal changes while enhancing spatial resolution, which are confirmed by both the performance metrics and visual inspection. ATRPK can also capture such changes but the generated images are blurry, and image contrast sometimes does not match that of ground truth. ESRCNN can preserve image contrast while the fused images cannot carry temporal changes. GAN can catch temporal changes but it also has the image contrast mismatch issue and it failed in Experiment 3. It is worth noting that our models heavily depend on image registration. Perfect registration can tolerate spatial resolution difference up to $12 \times$ during fusion (Fig. 36).

Though the proposed U-Net backbone quantitatively outperformed the HRNet backbone in many cases. Visually, the HRNet backbone generated sharper images, and details in the generated images are much more sharper than those in the U-Net generated images, because the HRNet backbone models maintain high-resolution throughout the whole learning process, and can capture high-frequency components better than U-Net [2]. Fused images by the ATRPK model are usually blurry. ESRCNN and GAN were also outperformed by the U-Net and HRNet backbone models in most cases.

### 4.2.7 SUMMARY

We proposed an arithmetic deep image fusing method, AriFusion, for multimodal temporal remote sensing image fusion. We applied it to Landsat-8, WorldView-2, Sentinel-2 and NAIP satellite image pairs in this study. AriFusion with both the U-Net and HRNet backbones achieved better results as compared to the traditional method (ATPRK) and the deep models (ESRCNN, GAN, STFGAN). While HRNet obtained similar performance metrics to that by U-Net, the fused images by HRNet are much sharper. GAN, ESRCNN and ATRPK either cannot catch temporal changes or the fused images are blurry. AriFusion successfully tackled these two challenges, making it a suitable candidate tool for fusing multimodal temporal remote sensing images to leverage other applications.
4.3 RELATED PUBLICATIONS


Chapter 5

FAKE SCIENTIFIC NEWS DETECTION

This chapter presents deep learning approach towards the detection of the fake scientific news.

5.1 BACKGROUND

The phenomenon of spreading scientific misinformation to the public is not new. The gap between public opinions and the scientific consensus on topics such as vaccine safety, evolution, or climate change have existed for a long time [138]. Nowadays, people are more strongly connected through social media such as Facebook, Tweeter, and LinkedIn. Statistics indicate the 62% of U.S. adults get news on the social media in 2016 as opposed to only 49% in 2012. Through the widespread social media and mobile devices, it is easier to spread fabricated and misleading news and articles, which may cause illusion, confusion, even violence. People typically have high confidence in scientific news as it is believed to be backed by scientific theory, experiments, and observations. However, most news audiences are in lack of understanding of science and their trustfulness could be taken advantage of by news editors or armatures, who may exaggerate, distort, or misinterpret assertions in the original scientific publications, making it look more attractive and shocking. It is hard for non-professionals to distinguish it from regular and factual scientific news.

Fake scientific news may contain statements, assertions, and claims that appear factual [139], especially when it is claimed to be from authoritative agencies. It often can be identified by contradictory, exaggerated, or unfeasible claims that are harmful to society. For example, a news website called newswatch33.com published an article that claimed NASA confirms earth will experience 15 days of complete darkness in November 2015. This piece news was a hoax [140] but it became viral on digital media (Facebook) and made many people get panic [141]. To prevent this type of scientific news from being pervasive and mitigate their potential detriment to the society, we desperately need a mechanism to check the credibility (truthfulness) of scientific news articles.

\[1\] http://www.journalism.org/2016/05/26/news-use-across-social-media-platforms-2016/
Fact-checking services are already available at several websites, e.g., www.snopes.com and www.factcheck.org, but they are based on laborious web browsing and cross verification by humans in order to trace provenance. There are computational models developed to automatically detect fake news [142]. The majority of these models relies on news content, including the author/publisher, the headline, the body text, images, and videos. Computational-oriented fact-checking methods try to solve two major issues: identifying check-worth claims and discriminating the credibility of fact claims. The two typical external sources include the open web and knowledge graph. However, there is a lack of automatic fact checking algorithm that leverages Artificial Intelligent (AI) techniques to evaluate the credibility of scientific news articles by **directly comparing them with the provenance scholarly papers**.

![Fig. 37. Workflows of the fake scientific news detection.](image)

Fig.37 shows the whole workflows of the fake scientific news detection. Given a news article, we first extract domain knowledge entities (DKES). We use these extracted DKESs to search pertinent scholarly papers from publicly available digital library Arxiv. In general, we may able to extract about 1000 to 5000 scholarly papers per news article. We perform cross view matching using vector similarity matching to find the most similar scholar papers (top 50) to the news article.
We then extract claims from both news and extracted pertinent scholarly papers followed by claim justification to get the final verdict (support or refute).

Our project will fill this gap by connecting news media with scholarly big data. The scientific questions identified as follows. (1) Given a vast amount of scholarly papers, how to find the most relevant ones pertaining to a given scientific news report? (2) How to evaluate the consistency of scientific report against a list of relevant publications? We are trying to answer these questions using deep learning. This chapter offers information retrieval pipeline to extract the relevant scientific papers from the news article.

This chapter describes different aspects of the NLP problems and solutions to address the fake scientific news detection problems. This chapter is organized as follows. Section 5.2 describes and proposed deep learning models for domain knowledge entities extraction and also its reviews related work. Section 5.3 discuss how to extract pertinent scholarly papers given a news using deep learning. Section 5.4 discuss our proposed models for claims extraction from unstructured text and rationale identification followed by claim justification are discussed in section 5.5.

5.2 DOMAIN KNOWLEDGE ENTITIES (DKES) EXTRACTION MODELS

Named entity recognition (NER) is a fundamental task in information extraction that seeks to extract named entities from unstructured text and classify them into predefined categories. The outcome can be utilized for many downstream applications such as constructing knowledge bases, data linking, and question answering. In the past decade, NER has been extensively studied based on models trained on general text, such as Wikipedia articles and newsletters, e.g., [143]. The most extracted entities fall into predefined categories including but not limited to people, organization, location, time expression, and monetary values. Although the general NER has achieved remarkable accuracy [143, 144], entity recognition from domain specific text, represented by scholarly papers published in research venues, is still challenging. Since the BioNLP shared task in 2004, much effort has been put on identifying DNA, RNA, cell line, cell type, and protein in biomedical papers. In this paper, we conduct a comparative study of sequential tagging models on domain knowledge entity extractions from biomedical papers on Lyme disease. Our research question is: how do different sequential tagging approaches, with recently proposed boosting mechanisms, perform in extracting domain knowledge entities?
We first define domain knowledge entities (DKEs), best described as noun phrases (NPs) representing domain knowledge of interest. DKEs are different from keyphrases [145], generally defined as important phrases or concepts in a paper. Keyphrases provide high-level description of a paper but DKEs can be at low-levels. For example, the article titled “Lyme disease: A rigorous review of diagnostic criteria and treatment” [146] has 4 keyphrases. However, the following sentence contains 6 DKEs, marked by red color.

“*Spirochetes with similar morphology, protein profile and antigenic determinants were detected in Ixodes ricinus ticks from Switzerland and Ixodes pacificus ticks from Oregon and subsequently in Ixodes persulcatus ticks in Russia.*”

DKEs have been found useful in detection of fake scientific news [147], because they represent unique knowledge in a certain domain and a combination of them can be good identifiers of text snippet for research works. DKEs in medical science were extracted to generate knowledge triples and construct knowledge graphs from Electronic Medical Record (EMR) [148].

Sequential tagging can be used for recognizing DKEs from unstructured text. In this paper, our comparison is focused on conditional random field (CRF), bidirectional long-short term memory (BiLSTM), and their variants. We do not compare Hidden Markov Model (HMM) because CRF has already shown advantages over HMM in many sequence tagging problems, e.g., [149]. Our work is based on a relatively small dataset containing 100 documents in biomedical science. Each document consists of 1–3 paragraphs manually curated from journal articles on Lyme disease. The documents are manually annotated and validated by domain experts in biomedical and health sciences. To the best of our knowledge, there is no open access dataset on this particular domain. We demonstrate that despite of the relatively small size, the best model achieves a decent $F_1 = 0.55$ on extracting entity strings. The results also shed light on the critical roles of pre-trained WE and the attention mechanism in training with relatively small samples.

5.2.1 RELATED WORK

In a 2011 paper, key aspects of scientific papers were extracted by matching language patterns in dependency trees [150]. The authors extracted *focus, technique, and domain* from titles and abstracts in the ACL Anthology corpus using handwritten patterns. A similar work in 2017 used an
unsupervised method [151] but only application and technique were extracted, and the evaluation was on computer science papers.

In 2015, SEGPHRASE was developed to extract quality phrases from text corpora using distant supervision [152]. The method rectifies the TF-IDF of segmented phrases in order to raise the rank of more informative phrases. A similar framework FACETGIST extracts application, technique, evaluation metrics, and dataset from academic papers, using POS-tagging to segment phrases. The final selection of candidate concepts is made by solving a joint optimization problem. The experiments were based on ACL and DBLP titles and abstracts. Both frameworks are best used on text corpora rather than individual documents.

The SemEval 2017 (SE17) had a task to extract “keyphrases” from scientific documents, which are essentially named entities in their context [153]. The winner of SE17 proposed a model [154] to stack CRF on top of BiLSTM. The model represents each word token using a vector $x_k$ by concatenating a vector $c_k$ (from character embedding) and a vector generated by word embedding (WE) $w_k$, in which $k$ denotes a token position. Next, the feature representations of words are learned using neural language models. The token representation $x_k$ is fed through a BiLSTM to embed the history into a fixed dimensional vector. The bi-directional embeddings are concatenated and used for sequence tagging. The BiLSTM layers are followed by a CRF layer to predict the
tag of each token. The model achieved an $F_1 = 0.54$ (the ensemble model achieved an $F_1 = 0.55$), however, the implementation was not open source.

### 5.2.2 MODELS

**CRF and BiLSTM**

Sequential tagging is a method to label individual tokens such as words in a sequence, a sentence for instance, in which order is important. One commonly used model is CRF. In CRF, the probability of tags for a token depends on its own features, and features and tags of the tokens surrounding it. CRF computes the joint probability distribution of the entire label sequence when an observation sequence intended for labeling is available. Recurrent neural network (RNN) is an nonlinear model for representation learning. The bidirectional long short-term memory (BiLSTM) has been proposed to in lieu of the vanilla RNN to overcome its vanishing gradient problem. In this model, the vector representation of the current token depends on the representations of context tokens. BiLSTM is usually followed by a fully connected layer or a CRF layer for sequence tagging, e.g., [155].

Recently, the attention mechanism was proposed to be incorporated in many BiLSTM-based models [27]. The idea is to apply attention weights of individual tokens, calculated using context vectors, when aggregating them to generate the output vector. The attention mechanism has been adopted in many NLP tasks such as machine translation [156] and question-answering [157]. In our work, we apply a special type called “self-attention”, in which the weights are computed based on the correlation between a sentence itself. Self-attention has been used for semantic role labeling [158].

The residual unit structure was designed to solve the degraded performance of very deep neural networks. In the residual unit, the output of a shallow layer is directly added to the output of a deep layer, providing a clear path for gradients to back propagate to shallow layers, making the learning process smoother and faster. Residual networks have been applied to image classification and significantly boosted the performance and training time, e.g., [159].

Our comparative study also utilize pre-trained WE, which has shown advantageous to train an model when the dataset is relatively small [160]. In this paper, we use ELMo, a language model
Attn-Res Unit
\[ X_1 + X_2 \]
BiLSTM-1
\[ X_1 \]
BiLSTM-2
\[ X_2 \]
Res Unit
\[ X_1 + X_2 \]
Attn
BIO Tags

Fig. 39. Architecture of Attn-Res-BiLSTM-ELMo model.

trained on 1 billion word benchmark [161]. The pre-trained model uses a multilayer BiLSTM and calculates the weighted sum of hidden states to represent each word.

**Baseline Models**

In a previous work [162], we proposed HESDK, a hybrid method to extract DKEs (Fig 38). The method used an NP chunker followed by an SVM classifier to classify NPs based on TF-IDF calculated using 1M abstracts in the Medline 2016 database. The method also employs a CRF model to label word-level tokens in the Inside-Outside-Beginning (IOB) scheme [163]. The results of these two methods were merged and a rule-based filter was applied before the final results are obtained. In this pilot study, we freeze the non-sequential component and change the models in the sequential tagging component. The goal is to investigate whether neural network models always achieve better performance than feature-based models and how the WE, attention, and residual unit affect the performance. The sequential models are listed below.

**CRF** The sequential model used in HESDK [162]. The model extracts 9 features from the current word and the word before and after the current word (in total 27 features).
**Enriched CRF**  Similar to (1) except that the number of features of each token increases to 16. New features include the first two characters of the POS-tags (e.g., VB from VBD), the type of the phrase the word is in (e.g., NP or VP), the first two characters of a phrase, the first two characters of the word, etc.

**Res-BiLSTM**  This model contains two BiLSTMs (so four LSTM layers) and there is a residual unit after the second layer (Fig 39 without the attention layer, ELMo not used).

**Attn-BiLSTM**  This model contains 1 BiLSTM followed by an attention layer (Fig 39 without \(X_2\) and the residual unit). A model with two BiLSTM layers was shown to underperform.

**BiLSTM-CRF**  This is basically the model implemented by [154] with one BiLSTM followed by a CRF layer.

**BiLSTM-ChE**  In this model, the BiLSTM is enhanced by character embedding. The motivation is that although WE is powerful to encode most words, rare or unseen words are usually embedded as dummy vectors. Character embedding is a solution to mitigate the out-of-vocabulary (OOV) problem. A word is modeled as a character sequence. An LSTM layer is first used to generate WE using character encoding. Another BiLSTM is used to further generate the encoding of each word. The word and character-WEs are concatenated to generate the final WE.

**Res-BiLSTM-ELMo**  This model is based on (3) except that the input to the Res-BiLSTM was initialized using the pre-trained ELMo WE [161] (Fig 39 without the attention layer).

**Attn-Res-BiLSTM-ELMo**  This is the most complicated model we applied. The input was initialized using the pre-trained ELMo, followed by two BiLSTMs, a residual unit was used to add outputs of the first BiLSTM (i.e., \(X_1\)) and the second BiLSTM (i.e., \(X_2\)). An attention layer was applied after the residual layer and a TimeDistributed layer in Keras is applied to output IOB tags for each token (Fig 39).
Table 14: A comparison of models. All results are before the rule-based filters. Triplets are (Precision, Recall, $F_1$).

<table>
<thead>
<tr>
<th>Sequential model</th>
<th>Hard</th>
<th>Soft</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sequential model only</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRF</td>
<td>0.42</td>
<td>0.26</td>
</tr>
<tr>
<td>Enriched CRF</td>
<td>0.17</td>
<td>0.11</td>
</tr>
<tr>
<td>BiLSTM</td>
<td>0.21</td>
<td>0.13</td>
</tr>
<tr>
<td>Res-BiLSTM</td>
<td>0.32</td>
<td>0.20</td>
</tr>
<tr>
<td>Attn-BiLSTM</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Attn-BiLSTM-25</td>
<td>0.17</td>
<td>0.12</td>
</tr>
<tr>
<td>BiLSTM-CRF</td>
<td>0.37</td>
<td>0.30</td>
</tr>
<tr>
<td>BiLSTM-ChE</td>
<td>0.26</td>
<td>0.15</td>
</tr>
<tr>
<td>Res-BiLSTM-ELMo</td>
<td><strong>0.42</strong></td>
<td><strong>0.47</strong></td>
</tr>
<tr>
<td>Att-Res-BiLSTM-ELMo</td>
<td>0.1304</td>
<td>0.0102</td>
</tr>
<tr>
<td>Att-Res-BiLSTM-ELMo-25</td>
<td>0.25</td>
<td>0.13</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sequential model + Non-sequential classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRF</td>
</tr>
<tr>
<td>Enriched CRF</td>
</tr>
<tr>
<td>BiLSTM</td>
</tr>
<tr>
<td>Res-BiLSTM</td>
</tr>
<tr>
<td>Attn-BiLSTM</td>
</tr>
<tr>
<td>Attn-BiLSTM-25</td>
</tr>
<tr>
<td>BiLSTM-CRF</td>
</tr>
<tr>
<td>BiLSTM-ChE</td>
</tr>
<tr>
<td>Res-BiLSTM-ELMo</td>
</tr>
<tr>
<td>Att-Res-BiLSTM-ELMo</td>
</tr>
<tr>
<td>Att-Res-BiLSTM-ELMo-25</td>
</tr>
</tbody>
</table>
5.2.3 EXPERIMENTS

Data Processing and Experimental Setups

The ground truth data are compiled by first searching a list of keywords, such as “Lyme disease”, “tick-borne disease” on Google Scholar, resulting in 140 articles, from which 41 were randomly selected ranging from 1990 to 2018. We visually inspected them and extracted 100 documents to annotate, each consisting about 1–3 passages. Each document is manually cleansed such that (1) Each passage occupies only one line; (2) All characters (e.g., \( \alpha \)) are encoded in UTF-8; (3) Superscripts and subscripts are expressed in the Latex way, e.g., “\(+\)”; (4) Citation marks are preserved at the original places and canonicalized to Arabic numbers in square brackets, e.g., “[10]”. Cleansing the data allows us to focus on information extraction, without affected by noise introduced when converting PDFs into text.

We use a web-based tool called brat for annotation. The annotator follows five general rules. (1) A DKE must be a noun or an NP; (2) Acronyms of DKEs are also DKEs (e.g., “LNB” for “Lyme Neuroborreliosis”); (3) Conjunction connected phrases are treated as a whole (e.g., “endemic and nonendemic areas”); (4) Try to label semantically meaningful superphrase when it contains a subphrase (e.g., “B. afzelii infection” instead of “B. afzelii”); (5) medicine names and body parts, even if commonly seen in daily life, are still labeled as DKEs (e.g., “antibiotics” and “brain”). The final ground truth corpus contains 1952 DKEs.

We used Keras v2.1.6, Tensorflow v1.8.0, and Tensorflow-hub v0.3.0 for implementation. For BiLSTM, we set the number of memory units to 512. The input/output dropout rate and the recurrent dropout rates are both set to 0.20. The learning rate was set to \(10^{-4}\). The sparse categorical cross entropy was used as the loss function. The models were trained up to 15 epochs. The Adaptive Moment Estimation (Adam) optimizer was adopted in the training process. The pre-trained ELMo WE has a dimension size of 1024. We randomly chose 75% documents for training and the remaining 25% for testing.

Evaluation and Discussion

We consider two types of evaluations (Table 14). Under the hard criteria, an extracted entity is taken as a true positive (TP) if both phrase strings and positions are correctly identified. Under the
soft criteria, only phrase strings are considered. It is intuitive that the performance is better under the soft criteria. For completeness, we present the performance with only the sequential tagging component and a combination with the non-sequential classifier. The following discussion pertains to the sequential tagging results unless otherwise noted.

1. The neural models do not necessarily outperform the feature-based models (i.e., CRF and Enriched-CRF). Although the Attn-Res-BiLSTM-ELMo model achieves the best performance under the soft criteria. The Enriched-CRF achieves the best performance under the hard criteria.

2. The pre-trained WE model (i.e., ELMo) plays an important role in the neural network models. The $F_1$ increases from 0.25 (Res-BiLSTM) to 0.40 (Res-BiLSTM-ELMo).

3. The attention mechanism significantly increases the recall of Res-BiLSTM-ELMo (from 0.36 to 0.45) with a marginal loss of precision (from 0.46 to 0.44).

4. The enriched features in the CRF model are helpful in predicting exact positions of entity mentions. A feature analysis indicates that the type of the phrase plays an important role.

5. The character embedding model BiLSTM-ChE underperforms compared with the plain BiLSTM model. The residual unit also decreases the $F_1$. This is likely to be caused by the relatively small training sample size. There are not many OOV characters and the advantage of the residual unit is more prominent on tasks with a large amount of training data.

6. The non-sequential classifier significantly boosts the overall performance. In the Attn-Res-BiLSTM-ELMo model, the $F_1$ increases by 9% (from 0.46 to 0.55) under the soft criteria. In the Enriched CRF model, the $F_1$ increases by 18% (from 0.21 to 0.39) under the hard criteria. This implies the benefit of combining sequential and non-sequential models.

5.2.4 SUMMARY

We conducted a comparative study of sequential labeling methods on the task to recognize DKEs from biomedical papers on Lyme disease. The results indicate that the CRF models outperforms all variants of BiLSTM models under typical settings when predicting both entity strings and
positions. The CRF underperforms BiLSTM with attention, residual unit, and ELMo when predicting only entity strings. When the training sample is relatively small, pre-trained WEs and the attention mechanism can significantly boost the performance. However, the overall performance of all sequential tagging methods on predicting the positions of DKEs still need to be improved. We will expand the ground truth size to at least 300. We will also fine-tune hyper-parameters and consider using Stochastic Gradient Descent (SGD) instead of the default Adam optimizer.
5.3 SCIPEP: RECOMMENDING PERTINENT SCIENTIFIC PAPERS

Scientific articles, containing scientists’ understanding of domain knowledge, analytical and experimental results, and new discoveries are traditionally read by researchers whose research is relevant to the particular domains the articles fall into. This is to a large extent due to the language used in scientific articles, including the appearance of jargon and terminologies as well as the complicated methods and datasets that are difficult to interpret without enough professional training. As a result, scientific research is usually believed to be a field far away from the general public. Therefore, many scientific breakthroughs remain not well-known until they are “translated” into more readable text via scientific news. This can cause a gap between people’s common beliefs, which may be prevalent but not up to date with the recent scientific research outcome. One example is the misunderstanding that mosquitoes transmit coronavirus, which was revealed wrong by a recent study [164].

In the past years, many websites emerged to fill this gap by press releasing the frontier scientific results in news articles, which usually contain hyperlinks or references to scientific papers. Examples of these websites including ScienceAlert.com and ScienceDaily.com. These references provide scientific evidence to the news content and boost the credibility of scientific news articles. However, the majority of these hyperlinks were added manually and a vast amount of scientific news articles do not contain references to source scientific articles. A fraction of these articles may contain misinterpreted, exaggerated, or distorted information that deviates from the scientific truth revealed by original scientific articles [165]. Furthermore, much faithful scientific news also lacks appropriate references, which can degrade their credibility. Therefore, it is necessary to develop computational models to automatically search for scientific articles that are pertinent to scientific news.

The problems we are dealing with can be generally classified as a citation recommendation problem, the goal of which is to find an article that should be cited given a context of a document [166]. A specific type of this problem, which is closer to our problems, is news citation recommendation in which the cited article is another news article. In general, citation recommendations can be dealt with using supervised or unsupervised methods. A function is trained to map a citation context $z_{ij}$ and the document it belongs to ($d_i$) to a reference (aka the cited document $r_m$).
In a supervised method, a classifier is trained to incorporate global (document or cross-document) and local (context) features, e.g., [167]. In an unsupervised method, a re-ranking model is applied which assigns probabilistic scores to a list of candidate documents, e.g., [168].

Our problem can be more challenging than the problem that involves either only news articles [168] or scientific articles [167] for the following reasons. First, there is a discrepancy between vocabularies used in news articles and scientific papers. The second problem is data sparsity. Unlike dense citation networks [169], there are much fewer citation relations between news and scientific articles, making it very difficult to apply graph embedding. Lastly, there are not many open-access datasets that can be used for training and evaluation.

To overcome the challenges, we proposed an unsupervised method by querying domain knowledge entities (DKEs) extracted from news [66], based on the assumption that DKEs are used in both the news articles and scientific papers. DKEs are entities expressed as noun phrases that deliver domain knowledge. DKEs are in general different from general named entities and keyphrases [145]. In the sentence below,

\textbf{Dual-energy computed tomography (DECT)} has been widely used in many applications that need \textbf{material decomposition}. The concept of \textbf{dual-energy CT} was initially described in 1973 by \textbf{Godfrey Hounsfield}.”.

the text span in red is a regular named entity but the text span in green are DKEs.

In our previous work, we proposed a method to retrieve pertinent scientific papers of a given news article using the term frequency-inverse document frequency (TFIDF). In this section, we propose a more robust model and significantly improve the performance. Our contributes of this study are summarized below.

1. We propose a 2-stage retrieval system including a neural ranking model that re-ranks candidate scientific articles obtained by querying using a frequency-based ranker. The model is integrated into a system called SciPEP that recommends pertinent scientific papers given a scientific news article.

2. We propose a transformer-based model that improves the performance of the DKE extraction.
3. We contribute a benchmark dataset that contains around 100 pairs of scientific news articles mostly from ScienceAlert.com and their associated research papers.

5.3.1 RELATED WORK

The exponential growth of scientific papers each year poses a great challenge for news editors and researchers to find the most pertinent citations [170]. Citation recommendations can be broadly classified into three categories depending on the source articles and the articles to be cited. In the first type, a news article cites another news article (news \rightarrow news). In the second type (paper \rightarrow paper), a scientific paper cites another scientific paper. In the third type (news \rightarrow paper), a news article cites a scientific paper. Our work focuses on the third type. Citing a news article in a research paper is relatively uncommon and beyond the scope of our study.

The collaborative filtering (CF) method has been widely used for news recommender systems since it was proposed [171]. This method requires building the document and the user profiles. However, the reading history of news articles is usually not accessible, making it difficult to build the user profile and apply this method.

Recently, many citation recommender systems were proposed using different text representation models. Early work used Synset Frequency-Inverse Document Frequency (SF-IDF) to represent the news [172] and used it for citation recommendation. SF-IDF was similar to TFIDF except that it used WordNet synonym sets to expand the semantic representation of a given word. Peng et al. developed a news citation recommendation system using a word-embedding based re-ranking and grounded entities (i.e., explicit semantics) [168]. Okura et al. proposed a model called embedding-based news recommendation (EBNR) using the denoising autoencoders variant for news representations [173]. Wang et al. utilized news titles and entities to represent the news via a knowledge-aware convolutional neural network (CNN) [174]. Sasaki et al. developed a robust news recommendation system where they combined news titles and bodies using average embeddings [175].

Depending on the type of input, citation recommendation systems can be classified into local and global citation recommendations. Local citation recommendation systems are based on text snippets, such as a sentence or even several words [176]. Global citation recommendation systems make recommendations based on the full text or the abstract of a document [177].
we propose the Pertinent Scientific News Evidence (SciPEP) retrieval system, which can be classified as a news→paper system based on global text. In a traditional citation recommendation system, the recommended articles are usually used for supporting the context or the source article. Our system, instead, searches for pertinent scientific papers that could support or refute the given news article. The refute articles will play a more important role for news articles containing misinformation that is contrary to or inconsistent with scientific findings. We propose using DKEs as queries to search for highly relevant papers. Our system can be built either on a local digital library search platform or an online search API. We propose a 2-stage information retrieval model that combines a frequency-based and a neural network-based ranker. We compare recently proposed pre-trained language models that best represent news articles and scientific papers.

5.3.2 METHODS

System Overview

The architecture of the SCiPEP system is inspired by the two-stage retrieval model proposed by [178]. The architecture of the SciPEP system contains four modules (Fig 40), described below.

Preprocessing  The input of this module is an HTML page of a scientific news article, downloaded from the Web. Only textual content is retained for further analysis. The current preprocessor can parse web pages downloaded from ScienceAlert.com. The custom parser can be easily written to parse news body text from web pages downloaded from other websites. The content will be cleansed so square brackets, extra spaces, special characters (such as @,#), and numerical digits are removed. Finally, the cleansed news article is segmented into sentences. Each sentence is tokenized and each token is labeled with part-of-speech (POS) tags.

DKE Extraction  We extract DKE from the news article body text and use those DKEs to extract candidate research papers. We use DKEs rather than the keyphrases or general name entities because DKEs represent the scientific domain knowledge and certain DKEs in news articles are borrowed from the source papers. Our previous work has demonstrated the efficacy of using DKEs to query a digital library search engine for candidate papers [66].
Candidate Paper Retrieval (CPR)  This module searches for the scientific paper candidates using the extracted DKEs. Here, we assume a frequency-based ranking algorithm such as BM25 [179], which is still a fast and popular unsupervised ranking algorithm in many search platforms, such as Apache Solr and Elasticsearch. Even if BM25 is not used in this step, our method will still achieve a comparable performance if the ranker achieves a comparable P@10 (see below). In our experiment, we use arXiv.org, a digital repository for researchers to submit pre-printed papers. By the date of writing, it has indexed around 2 million scientific papers, covering major branches in Computer Science, Mathematics, Physics, Astronomy, Statistics, Materials Science, and Social Science. The website offers a free API through which we search scientific papers.

To obtain a high recall in this stage, we perform a union of multiple query results. Each query contains a single or a combination of up to 3 DKEs (connected by “AND”). The final candidate list is obtained by merging the top 10 results of all queries and remove duplicate papers by titles and authors. There are approximately 1000-3000 candidate papers found for each news article. This step is necessary because the time complexity of constructing vector representations and rank all papers by cosine similarities (next step) is much higher than the BM25 algorithm. This module reduces the candidate pool down to thousands of articles, which boosts the efficiency of the overall retrieval model.

Paper Re-ranking This module uses a neural model to re-rank the candidate papers by calculating the cosine similarities between their vector representations and news articles. The vector representation of the scientific paper is constructed by encoding the abstracts into a fix-length vector using a pre-trained language model. The vector representations of the news articles are constructed using a similar way. The candidate papers are then re-ranked by cosine similarities. The purpose of this step is to increase the precision by promoting scientific papers that are highly topically relevant to the news article. The key is to generate a high-quality vector representation.
The ground truth of our experiments is obtained using 100 scientific news articles downloaded from ScienceAlert, ScienceNews, EurekAlert, and Forbes\(^2\). The articles were manually curated so that at least one source scientific paper is provided as a hypertext link or a reference in the original news article. There are at maximum 5 papers linking to a news article. This dataset doubles our previous effort \[66\]. The average length of these news articles is approximately 900–1000 words. The news articles are from a variety of domains such as history, arts, astronomy, biology, environment, computer science, and medicine.

To train a robust DKE extractor for articles in various domains, we adopt the OA-STM \(^3\) dataset. The OA-STM dataset contains pre-processed abstracts from scientific papers in 10 domains, including agriculture, astronomy, biology, chemistry, computer science, earth science, engineering, materials science, math, and medicine. There are about 11 abstracts in each domain. For each abstract, four core scientific concepts were annotated including PROCESS, METHOD, MATERIAL, and DATA. Existing studies indicated that a classifier trained on data from all 10 domains performs


\(^3\)https://github.com/elsevierlabs/OA-STM-Corpus
better than trained on data from a single domain [180].

However, not all the labeled text spans in OA-STM conform to our definition of DKEs. For example, the text span “-55 deg C, -80 deg C, and -109 deg C” and “decreased significantly” were both labeled but intuitively they were not DKEs. Therefore, we use this dataset to fine-tune the CoNLL2003 model only to show that a fine-tuned language model can achieve superior performance on the DKE extraction task. To mitigate the effect of these noisy labels, we fine-tune the pre-trained CoNLL2003 model using the SemEval2017 competition Task 10 dataset [153] (hereafter SemEval2017 dataset), and apply this fine-tuned model to our system. This dataset contains 500 passages extracted from journal papers in Computer Science, Materials Science, and Physics. The dataset was double annotated and three types of entities were identified, namely, MATERIAL, METHOD, and PROCESS. This dataset are used in our previous work for DKE extraction [181]. Note that when using these two datasets, we collapse all categories into one category called DKE.

5.3.3 DKE EXTRACTION

Scientific news articles can be written in multiple domains. To get the robust DKEs extractor model, we compare the performance of multiple BiLSTM-based models and transformer-based models trained on the OA-STM dataset.

BiLSTM-based Models

In our previous work, we compared the performance of multiple sequence tagging models on the task of extracting DKEs from biomedical science papers [32]. Because we change the labeled dataset, we re-evaluated each model and compared their performance on this task.

BiLSTM-CRF and Res-BiLSTM-CRF  In this model, we applied the BiLSTM to obtain the hidden representation of a token, followed by a conditional random field (CRF) layer. This model has been successfully applied for NER tasks [182]. We also considered an alternative model with two BiLSTM networks with a residual connection. In the residual unit, the output of a shallow layer is directly added to the output of a deeper layer [183]. Either model uses random weights as input and learns the hidden representation of each token from the context only.
**BiLSTM-W2Vec**  In this model, the representation of each token is constructed by concatenation of the hidden representation output by a BiLSTM with the pre-trained word2vec model [184]. Then, a CRF layer is used to classify each token.

**BiLSTM-ChE**  Character embedding can be used for capturing morphological information of words [185] and mitigating the out-of-vocabulary problem [186]. In this model, we first use a BiLSTM to encode each character and combine them into a word-level vector. We use another BiLSTM to encode each word-level token into a new vector. These two vectors are concatenated to generate the final representation of each work-level token. A CRF classifier is then applied to tag each token.

**BiLSTM-ChE-Attention**  In this model, a self-attention layer is added after combining the character and word embeddings in the BiLSTM-ChE model.

**Res-BiLSTM-ELMo**  ELMo is a context-dependent language model trained on the 1 Billion Word Benchmark [45], providing word representations with rich features. In this model, we initialize the Res-BiLSTM model using the pre-trained ELMo embedding.

**Transformer Model**

Self-attention and positional encoding are the two key characteristics of the transformer model [27]. The aggregation of these two factors has made the transformer model successful for many tasks such as named entity recognition (NER). One representative language model is Bidirectional Encoder Representations from Transformers (BERT) [46], which has been successfully applied for NER [187] and other downstream tasks. For DKEs extraction, we fine-tune the pre-trained BERT language model as a backbone encoder on the OA-STM dataset. Before the classification layer, the BERT encoder extracts high-quality language features from our text data. Based on these features, the classification layer classifies these entities into DKEs and non-DKEs (Fig. 41).
5.3.4 BASELINE METHODS

We also used three baseline models for comparison. These models are unsupervised, machine learning-based, and a model that extracts regular named entities.

**Text-Rank** is an unsupervised graph-based model inspired by Google’s PageRank algorithm to extract keyphrases [188]. The algorithm builds an undirected graph for each target document, in which the nodes correspond to words in the target document, and edges are drawn between two words that occur next to each other in the text.

**HESDK** HESDK is a hybrid approach to extract DKEs [181]. In the first phase, candidate phrases are extracted by a grammar-based chunk parser which is then filtered by a linear support vector machine (SVM). In the second phase, a CRF model is used to predict the probabilities of tags for a given token based on lexical and morphological features. The results from both approaches are merged and further filtered by a rule-based filter.

**Stanford NER** To demonstrate the advantage of using DKEs, we extract regular named entities using the Stanford CoreNLP [144]. We use 7 class Stanford NER model trained on the MUC6 and MUC7 datasets. The model extracts seven NEs, including LOCATION, PERSON, ORGANIZATION,
MONEY, PERCENT, DATE, and TIME.

Depending on the length of the news article, the DKE extractors can extract 50-200 DKEs per article, resulting in 500-2000 candidate scientific papers. For all baseline methods, we use the term frequency-inverse document frequency (TFIDF, see below) to represent news articles and scientific papers.

5.3.5 DOCUMENT EMBEDDING

To calculate semantic similarities between the news article and the scientific papers, we represent both of them with fix-length feature vectors. We use paper abstracts because the full text is not always accessible. We compare the following text representation models.

TFIDF weighted Bag-of-Words (BoW)  BoW is a traditional text representation model [189]. In this model, each news article or the scientific abstract is represented as a sparse vector containing $|V|$ elements, in which $|V|$ is the vocabulary size of a retrieval corpus. Each element is the TFIDF value of the corresponding term. The retrieval corpus is defined as the combination of the news article and its candidate papers. The TFIDF for each term is calculated based on the retrieval corpus it belongs to.

d2vec  In this model, for each given document, the vector representation of each word is aggregated in a certain way to represent the whole document. We use the pre-trained word2vec model to calculate a 300 dimensional vector representation of each word. The document vector is the average of vectors of all tokens.

Doc2vec  Doc2vec is a model to create a vector representation of a document [190] of various lengths. The $d2vec$ model above does not count the word sequence information and does not incorporate the context into the embedded vectors. In doc2vec, when training the word vectors, the document vector $D$ is trained as well. When the document sequence is finished, the document vector $D$ holds a representation of the document. We use the model implemented by Python Gensim Doc2Vec.
**Weighted Doc2Vec**  After getting document representation using the Python Gensim Doc2Vec, we extract word representation for each of the words from the document. We then weighted that word using the TFIDF value. Eventually, we combine all the word representation followed by feature-wise averaging to create a new document representation.

**SciBERT**  SciBERT is a transformer-based encoder trained on a large corpus of scientific text [191]. Because this model is trained on scientific literature, it has shown advantages over BERT in scientific text classification and summarization tasks [192]. partially because its relatively larger vocabulary overlaps with the given corpus. We perform a similar aggregation to the $d2vec$ to obtain the document embedding by averaging the vector representation of each token in a document.

**SBERT**  Sentence transformer or sentence-BERT (SBERT) is a modified version of the pre-trained BERT model [193]. It uses a Siamese network with the triplet loss function to produce sentence embeddings. Each sentence in a document is encoded using SBERT. We then use the averaged embedding as the document embedding.

**SPECTER**  SPECTER is a document embedding model trained on EMNLP scientific publications ranging from 2016 to 2018 [194]. The SPECTER model was designed to overcome the limitations of SciBERT by leveraging inter-document relatedness. This model uses a pre-trained SciBERT transformer model as a backbone and incorporates inter-document context into the SciBERT model. SPECTER builds embeddings from the title and abstract of a paper.

### 5.3.6 EVALUATION AND COMPARISON

**Metrics**

We use precision, recall, and F1 score to evaluate the DKEs extractor models. Precision is calculated as the ratio of correctly extracted DKEs divided by the total number of DKEs extracted. The recall is calculated as the ratio of the correctly extracted DKEs divided by the total number of DKEs labeled. F1 score is the harmonic mean of precision and recall.

We use the following metrics to evaluate the system.
Mean-reciprocal-recall (MRR)  MRR is defined as

\[
MRR = \frac{1}{|Q|} \sum_{i} \frac{1}{\text{rank}(i)}
\]  

(24)

in which \(Q\) is the total number of queries, and \(\text{rank}(i)\) is the rank of a relevant scientific paper. MRR assumes there is only one relevant document in the search results of each query. So when evaluating queries corresponding to multiple papers, we use the top-ranked paper to calculate MRR.

Normalized Discounted Cumulative Gain (NDCG)  We use NDCG with a binary relevance. The metric was used for evaluating cases in which one query returns multiple relevant papers.

\(P@K\)  We use the precision at rank \(K\) to measure the fraction of relevant scientific papers within certain top results. It can be used when there are multiple relevant papers. We evaluate \(P@K\) when \(K = 1, 5, 10, 20,\) and \(50.\)

5.3.7 EXPERIMENTS AND RESULTS

DKE Extraction

DKE extraction is the key step to generate keyword-style queries to obtain paper candidates. Each model was trained on 80% documents from all domains and tested on 20% of documents on individual domains. 42 show the comparison of performance of DKE extraction models. The results indicate that the fine-tuned BERT model outperforms all other models, achieving a nearly perfect performance for all domains, with F1 varying from 0.92 to 1.00 depending on the domain. Specifically, the model correctly extracted all DKEs in the math domain. The superior results are attributed to the BERT transformer encode, which was pre-trained on a giant dataset (CoNLL-2003) [46].

The other models underperformed because they were mostly trained from scratch on much smaller training datasets. Among these models, the ELMo-BiLSTM performed relatively well. Specifically, the model achieved an F1=52.8%, 53.5%, and 51.1% for materials science, biology, and chemistry, respectively. The results verified the advantage of initializing the BiLSTM encoder with pre-trained language models [32]. One interesting phenomenon is that adding self-attention
Fig. 42. A comparison of performances of DKE extraction models. Categories along x-axis are below. Arg: agriculture; Astr: astronomy; Bio: biology; Chem: chemistry; CS: computer science; Eng: engineering; ES: environmental science; Math: mathematics; Med: medical science; MS: materials science.
to the BiLSTM-ChE model boosts the performance on certain domains such as agriculture, engineering, math, and biology but decreases the F1 scores of other domains.

Table 15: A comparison of models performance on searching for pertinent scientific news evidence papers.

<table>
<thead>
<tr>
<th>Model Name</th>
<th>Query Type</th>
<th>DKE Model</th>
<th>Text Embedding Method</th>
<th>P@K</th>
<th>MRR</th>
<th>Average NDCG</th>
<th>T_PRR (sec)</th>
<th>T_ALL (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline1</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>14%</td>
<td>38%</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Baseline2</td>
<td>KP²</td>
<td>TextRank</td>
<td>TFIDF</td>
<td>18%</td>
<td>23%</td>
<td>23%</td>
<td>0.19</td>
<td>0.8</td>
</tr>
<tr>
<td>Baseline3</td>
<td>NE³</td>
<td>CoreNLP</td>
<td>TFIDF</td>
<td>38%</td>
<td>44%</td>
<td>45%</td>
<td>0.40</td>
<td>0.8</td>
</tr>
<tr>
<td>Baseline4</td>
<td>DKE⁴</td>
<td>HESDK</td>
<td>TFIDF</td>
<td>39%</td>
<td>45%</td>
<td>49%</td>
<td>0.43</td>
<td>0.8</td>
</tr>
<tr>
<td>BERT-TFIDF</td>
<td>DKE</td>
<td>BERT</td>
<td>TFIDF</td>
<td>50%</td>
<td>71%</td>
<td>74%</td>
<td>0.59</td>
<td>0.70</td>
</tr>
<tr>
<td>BERT-d2vec</td>
<td>DKE</td>
<td>BERT</td>
<td>d2vec</td>
<td>20%</td>
<td>37%</td>
<td>41%</td>
<td>0.28</td>
<td>0.40</td>
</tr>
<tr>
<td>BERT-Doc2Vec</td>
<td>DKE</td>
<td>BERT</td>
<td>Doc2Vec</td>
<td>36%</td>
<td>51%</td>
<td>54%</td>
<td>0.43</td>
<td>0.52</td>
</tr>
<tr>
<td>BERT-WDoc2Vec</td>
<td>DKE</td>
<td>BERT</td>
<td>Weighted Doc2Vec</td>
<td>35%</td>
<td>55%</td>
<td>60%</td>
<td>0.43</td>
<td>0.55</td>
</tr>
<tr>
<td>BERT-SciBERT</td>
<td>DKE</td>
<td>BERT</td>
<td>SciBERT</td>
<td>19%</td>
<td>30%</td>
<td>36%</td>
<td>0.25</td>
<td>0.36</td>
</tr>
<tr>
<td>BERT-SBERT</td>
<td>DKE</td>
<td>BERT</td>
<td>SBERT</td>
<td>47%</td>
<td>69%</td>
<td>74%</td>
<td>0.57</td>
<td>0.69</td>
</tr>
<tr>
<td>BERT-SPECTER</td>
<td>DKE</td>
<td>BERT</td>
<td>SPECTER</td>
<td>47%</td>
<td>69%</td>
<td>73%</td>
<td>0.57</td>
<td>0.69</td>
</tr>
</tbody>
</table>

1 Quoted from [195]. The specific corpus used is not available, making it impossible to make a fair comparison.
2 Keyphrases extracted using TextRank [188].
3 Named entities extracted using Stanford CoreNLP. [144].
4 HESDK: A hybrid approach to extracting scientific domain knowledge entities [181]
5 Our previous model [66]

System Performance

Table 15 shows the performance of the system on retrieving pertinent scientific papers. For each model, we show the model name, query type, the model used for DKE extraction, and the method to embed paper abstracts. We report P@K, MRR, the average NDCG, the time it takes to finish ranking (the PRR module only), and the overall time for the entire system (40). The results
indicate that the BERT-TFIDF and the BERT-SPECTER are the top two models. We compare our models with four baseline models. All baseline models use TFIDF weighted BoW model to represent documents but use different query types. Among the baselines, baseline4 which is our previous scholarly paper retrieval model performs better. However, the BERT-based DKEs extractors along with different document representation methods perform better than any baseline models.

The result indicates that the transformer-based retrieval model shows a significant improvement over our previous work (baseline4), which can retrain 39% desired scientific paper at the 1st position whereas the BERT-TFIDF model achieves 50%. Our previous model can retrieve 60% scientific papers within the top 50th position whereas the BERT-SPECTER model can retrieve 91%. The reason behind these improvements is the BERT-based DKEs extractor model can extract more meaningful DKEs than our previous DKEs model.

Our results indicate that the simple TFIDF document presentation with the DKEs extracted from fine-tuned BERT model performs better in terms of the time complexity and performance metrics. Transformer-based document representation models with the same query type also performed well. For certain news articles, the BERT-TFIDF method fails to rank scientific papers within the 50th position, achieving P@K=86%. However, the BERT-SPECTER and SBERT with BERT-based DKE extractors retain more relevant scientific papers (91%) within the 50th position. From Table 15, we can see that BERT-SPECTER and BERT-SBERT have higher P@K values when $K = 20$ and 50 than BERT-TFIDF. These results suggest that (1) the transformer-based models can better represent semantics in both the news article and the scientific paper at lower ranks ($K \geq 20$), and (2) the TFIDF model is better used for retrieving relevant documents at higher ranks ($K < 20$).

Through error analysis, we found that the major reason that caused our retrieval models to fail was that the scientific news contain few DKEs. One example is a news article called “How to spot deepfakes? Look at light reflection in the eyes”\(^4\). Other types of news articles use more images, videos, and equations to convey scientific discoveries, rather than plain text. One example is a news article titled “Math Genius Has Come Up With a Wildly Simple New Way to Solve

\(^4\)http://www.buffalo.edu/news/releases/2021/03/010.html
5.3.8 SUMMARY

This work extended our previous work on recommending pertinent papers for scientific news articles [66]. In this work, we replaced our BiLSTM-based DKEs model with a transformer DKE extractor and explored different document embedding models to represent the news article and scientific paper abstracts. Our experiments showed that using DKEs was an effective and efficient way to retrieve pertinent scientific papers. In particular, all retrieval models using DKEs as queries outperformed the retrieval models using general named entities or keyphrases.

The DKEs extracted from the BERT-based model with TFIDF document embedding model has the potential to find out highly pertinent scientific papers faster and more accurately when $K < 20$. The DKEs extracted from the BERT-based model with SPECTER embedding achieve better P@$K$ when $K \geq 20$.

The ultimate goal is to build an application that is capable of predicting the credibility of scientific news, based on pertinent scientific papers. To this end, we need to find effective and efficient ways to find the most relevant ones pertaining to a given scientific news report from the vast amount of scholarly papers and to evaluate the consistency of a scientific report against a list of relevant publications. The SciPEP system we proposed here answers the first question. However, the overall runtime is still over 10 seconds. In the future, we will consider a more efficient method to further reduce the overall runtime to seconds by parallelizing queries and text embeddings.

5.4 CLAIMS EXTRACTION

Because of the rapid increase of scientific papers [196], there is an emergent need for powerful information extraction tools from large volumes of unstructured text. It is usually challenging for novice scientists to identify key claims from scientific papers. Even for experienced scientists, efficiently identification of key claims from many scientific papers is time-consuming and error-prone. Recently, there is a growing concern about scientific reproducibility in many domains, e.g., [197, 198]. Mining claims is a crucial step to automatically predict reproducibility, which is investigated in DARPA's Systematizing Confidence in Open Research and Evidence (SCORE)

5https://www.sciencealert.com/math-genius-has-come-up-with-a-wildly-simple-new-way-to-solve-quadrati
program [199]. Recently, the task of claim extraction from scientific papers has raised the attention of natural language processing and understanding (NLPU) communities, e.g., [200]. However, effectively extracting claims from unstructured text is still an open question [201].

A claim from a scientific paper can be defined as a sentence that provides the core finding of the paper. A claim is atomic [202], independent (without external references), declarative, absolute [203]. However, annotating scientific claims usually require domain knowledge, so the existing datasets with annotated claims are scarce and not available in all domains. Therefore, it is crucial to develop deep learning models that efficiently use existing data and possibly transfer learned models across multiple domains.

In this section, we explore deep learning-based methods to extract claims from scientific papers. We propose a framework called CLAIMDISTILLER, word and character embedding model with BiLSTM (WC-LSTM) with contrastive learning. We demonstrate that the contrastive learning and transfer learning mechanisms significantly improve the model performance to distinguish between claims and non-claims. We compare our proposed model with baseline models on a relatively small annotated claim extraction dataset [200] in the biomedical science domain. The results show that contrastive learning and transfer learning significantly boost the performance of a simple WC-LSTM model, achieving F1=0.87 on the testing dataset.

5.4.1 RELATED WORK

Document summarization and argumentation mining are NLPU tasks that are closely related to claim extraction tasks. The goal of document summarization is to extract or generate text that is much shorter than the original documents and delivers the main idea of the given documents. Extractive summarization [204] and abstractive summarization [205] are the two main categories of this task. Early methods of document summarization used graph-based methods, such as TextRank [188]. Later, Latent Semantic Analysis was applied to this task and showed an improved result [206]. Recently, CNN was applied to the extractive text summarization [207]. A feed-forward neural network language model with the attention mechanism was proposed for abstractive summarization [208]. Recently, a planning-based model was proposed for abstractive summarization [209].

Argument mining, a closely related domain to claim extraction, is an automated inference
extraction, reasoning identification method to extract the structure of inference and reasoning expressed in arguments presented in natural language [210]. Using argumentation mining, premises were extracted from news [211], social media [212], scientific article [213], and Wikipedia [214]. Most existing work on argument mining in scientific papers applied heuristic methods [215, 216] and classic machine learning methods [217]. Recently, deep learning methods, including weak supervision and transfer learning mechanisms, have been proposed for this task [218].

Claim extraction is different from document summarization and augmentation mining because the claim is the absolute, independent, core findings of the document. Dataset scarcity is the major challenge for the scientific claims extraction task. Thorne et al. [214] developed the FEVER dataset which contains factual claims extracted from Wikipedia. However, this dataset is not suitable for extracting claims from scientific papers. Scientific claims annotation is a very labor extensive task and requires significant domain expertise. Dernoncourt et al. [219] developed a scientific discourse dataset (PubMed RCT) in which sentences were labeled as background, introduction, method, result, and conclusion. However, claims were not explicitly labeled in this dataset. Several methods were proposed for scientific claim extraction. Jansen et al. [215] used a rule-based method to extract claims from scientific papers. A scientific claim extraction dataset in biomedical domains was released [200]. The authors also proposed a BiLSTM-CRF model for the claim extraction tasks. They trained their model on the PubMed RCT dataset and then fine-tuned the model on their in-house claim dataset. This transfer learning method significantly improved the performance of baseline models. We adopt the training and test dataset in [200] to evaluate our proposed model.

5.4.2 METHODS

Dataset

We use the Pubmed RCT dataset [219] and claim extraction dataset [200] for our task. Our goal was to build a model to extract claims from the claim dataset. To build a more robust claim extraction model, we first train a model using the Pubmed RCT dataset and then fine-tune it using the claim extraction dataset. Pubmed RCT is relatively a large dataset consisting of 20,000 abstracts. Each sentence in an abstract is labeled one of the five classes, namely, Objective, Introduction,
Method, Result, and Conclusion. The claim extraction dataset contains only 1,500 scientific abstracts from the biomedical domain. Each sentence was annotated by domain experts into two categories, namely, claim and non-claim. Here are the samples of claim and non-claim sentences which are used for training.

Claims: We find that when Pol I moves into transcriptionally paused state, TBP/TFIIB remain at the promoter.

Non-claims: Here, we collected viable sperm from first-generation hybrid male progeny of Mus musculus castaneus and M. m. domesticus, two subspecies of rodent in the earliest stages of speciation.

Model Development Paradigms

There are three main categories of our developed models (training from scratch, transfer learning, and training with supervised Contrastive learning [220]).

Training from scratch In our first category, we train our deep learning models from scratch using the claim extraction dataset.

Transfer learning As the claim extraction dataset is very small, we at first trained the deep learning models using the Pubmed RCT dataset and then fine-tune the models using the claim extraction dataset as our 2nd category.

Training with supervised Contrastive learning There are two phases of learning. In the first phase of learning, we use the triplet loss function which forces the models to group claim class in the same embedding space and non-claim class in the different embedding space. We fix a 128-dimensional vector as our final embedding for the encoder. After getting the trained encoder, we fix the embedding layer while training the classification layers for classification in the 2nd phase of learning. The key idea of contrastive learning is to group the same class while simultaneously pushing apart different classes in the embedding space. Self-supervised contrastive learning is a division of self-supervised learning that focuses on data representation and uses both positive and negative samples from the data. Supervised contrastive loss (SCL) can be defined as:
\[ SCL = \sum_{i \in I} \frac{-1}{|C(i)|} \sum_{c \in C(i)} \log \frac{\exp(z^*_i z^p / \tau)}{\sum_{a \in A(i)} \exp(z^*_i z^q / \tau)} \]  

(25)

Here \( C(i) \) is the claim dataset class information and \( i = 1 \ldots 2N \) is the index of an arbitrary sample. \( z^*_i, z^p \) and \( z^q \) indicate anchor, positive and negative sample respectively.

**Fig. 43.** Supervised contrastive learning on the claim extraction dataset. Given a claim (anchor), positive samples (claim), and negative samples (non-claim), at first we covert text to its vector representation (embeddings) using the pre-trained model. In embedding space, contrastive loss push samples from the same class close and samples from different classes further apart in embedding space. In supervised contrastive learning, class label information is used to align the same class in the embedding space.

Fig. 43 shows the working flows of the contrastive loss. From each batch, a claim sentence is selected as an anchor and similar numbers of positive and negative samples are extracted from the same batch to train the encoder. The contrastive loss force the same classes to group together. Supervised contrastive learning makes use of label information to create more accurate embeddings and thus subsequent classification based on it can achieve better performance.
Deep learning models

**1D-CNN.** Convolutional Neural Networks (CNN) are very powerful to extract features from natural and remote-sensing image [221]. Similar to a 2D-CNN, 1-dimensional convolutions can be used for extracting patterns from local 1D patches (subsequences) from sequences. Because the same input transformation is performed on every patch, a pattern learned at a certain position in a sentence can later be recognized at a different position, making 1D-CNNs translation-invariant (for temporal translations) [222].

After converting words from claim sentences to vectors, 1D convolutional layers with the convolution kernels with size \( w \) are used to extract the patterns (Fig 44) and these layers can recognize these patterns in any context in an input sequence. We use a 2-layered 1D-CNN, which is flattened at the end before the representations are fed to a dense layer.

**USE-Dense** We use pre-trained the Universal Sentence Encoder (USE) [223] to encode the text of the claims into high dimensional vectors (512-dimensional vectors) which is eventually fined on the claim dataset. After encoding the text, we use two dense layers to get outputs.

**WC-LSTM** Models with only word embedding have some potential shortcomings. During prediction time, unseen words have to encode as unknown and have to infer their meaning from their surrounding words. Often word postfix or prefix contains a lot of information about the meaning of the word. Using these characters’ information is very important for dealing with unseen words [185]. We combined pre-trained Word2Vec embedding [43] with character embedding to
encode unseen words [224] followed by BiLSTM layers to extract patterns from the claim sentences (Fig.45). Then we use dense layers for classification.

Transfer-Bert Similar to other deep learning models, we used the pre-trained Bert-based sentence classification model [46] to encode sentences to form sentence encoding. The encoder eventually tuned on claim dataset during training. After getting sentence encoding, we use dense layer for classification (Fig.46).

5.4.3 RESULTS AND DISCUSSIONS

The goal of this task is to classify the given sentence as claim or non-claim. To accomplish this task, we have explored and developed deep learning models. We explore different word embedding and sentence embedding methods to encode the sentences and use different deep learning layers.
Table 16: Model Evaluation on claim dataset. We report the performance of the baseline rule-based model and deep LSTM-CRF model with our proposed deep learning models with and without contrastive learning.

<table>
<thead>
<tr>
<th>Models Name</th>
<th>Validation Precision</th>
<th>Validation Recall</th>
<th>Validation F1-score</th>
<th>Testing Precision</th>
<th>Testing Recall</th>
<th>Testing F1-score</th>
<th>Time complexity (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule-based-1 [216]$^1$</td>
<td>0.34</td>
<td>0.36</td>
<td>0.35</td>
<td>0.31</td>
<td>0.32</td>
<td>0.31</td>
<td>–</td>
</tr>
<tr>
<td>Transfer Learning CRF (GloVe) [200]$^1$</td>
<td>0.85</td>
<td>0.75</td>
<td>0.80</td>
<td>0.86</td>
<td>0.72</td>
<td>0.79</td>
<td>–</td>
</tr>
<tr>
<td>CNN-1D</td>
<td>0.83</td>
<td>0.84</td>
<td>0.81</td>
<td>0.83</td>
<td>0.84</td>
<td>0.82</td>
<td>6.23</td>
</tr>
<tr>
<td>Contrasive-CNN-1D</td>
<td><strong>0.87</strong></td>
<td><strong>0.87</strong></td>
<td>0.86</td>
<td>0.86</td>
<td><strong>0.87</strong></td>
<td><strong>0.87</strong></td>
<td>230.32</td>
</tr>
<tr>
<td>CNN-1D-transfer</td>
<td>0.86</td>
<td><strong>0.87</strong></td>
<td>0.86</td>
<td>0.86</td>
<td>0.87</td>
<td>0.85</td>
<td>774.57</td>
</tr>
<tr>
<td>Contrastive-CNN-1D-transfer</td>
<td>0.86</td>
<td><strong>0.87</strong></td>
<td><strong>0.87</strong></td>
<td>0.86</td>
<td>0.87</td>
<td>0.85</td>
<td>855.18</td>
</tr>
<tr>
<td>USE-Dense</td>
<td>0.83</td>
<td>0.85</td>
<td>0.82</td>
<td>0.82</td>
<td>0.84</td>
<td>0.81</td>
<td>15.72</td>
</tr>
<tr>
<td>Contrasive-USE-Dense</td>
<td>0.86</td>
<td><strong>0.87</strong></td>
<td>0.85</td>
<td><strong>0.87</strong></td>
<td>0.87</td>
<td>0.86</td>
<td>1359.17</td>
</tr>
<tr>
<td>USE-Dense-transfer</td>
<td>0.86</td>
<td><strong>0.87</strong></td>
<td>0.86</td>
<td>0.85</td>
<td>0.86</td>
<td>0.86</td>
<td>465.59</td>
</tr>
<tr>
<td>WC-LSTM</td>
<td>0.83</td>
<td>0.85</td>
<td>0.83</td>
<td>0.84</td>
<td>0.85</td>
<td>0.83</td>
<td>66.82</td>
</tr>
<tr>
<td>WC-LSTM-contrastive</td>
<td><strong>0.87</strong></td>
<td><strong>0.87</strong></td>
<td><strong>0.87</strong></td>
<td><strong>0.87</strong></td>
<td><strong>0.88</strong></td>
<td><strong>0.87</strong></td>
<td>3603.55</td>
</tr>
<tr>
<td>WC-LSTM-Transfer</td>
<td>0.86</td>
<td><strong>0.87</strong></td>
<td>0.84</td>
<td>0.86</td>
<td>0.85</td>
<td>0.84</td>
<td>2173.75</td>
</tr>
<tr>
<td>BERT-Dense</td>
<td><strong>0.87</strong></td>
<td>0.84</td>
<td>0.86</td>
<td>0.86</td>
<td>0.84</td>
<td>0.85</td>
<td>172.84</td>
</tr>
<tr>
<td>BERT-Transfer-Dense</td>
<td><strong>0.87</strong></td>
<td>0.85</td>
<td>0.86</td>
<td>0.86</td>
<td>0.85</td>
<td>0.85</td>
<td>4751.26</td>
</tr>
</tbody>
</table>

$^1$ Quoted from reference.
to extract the patterns of claims to classify them. As the claim dataset is small, we utilized the Pubmed RCT dataset along with the claim dataset to make a more robust claim extraction model. We also fuse contrastive learning with our deep learning models and found that this contrastive learning is very useful for a small and imbalanced dataset. We compare the deep learning model’s performance with the rule-based model (Rule-based-1) and previous state of art model (Transfer Learning CRF (Glove)) on this dataset and found that our developed models perform better.

From the Table.16, our deep learning models perform better compared to the competing rule-based baseline model [215] and previous state-of-art deep learning based claim model [200]. Among the baseline models, transfer Learning CRF (Glove) performs better than rule-based models. This suggests that the rule-based approaches can not extract claims properly. For all our deep learning models, training from scratch using the claim dataset can not perform well compared to the fine-tuned models. The difference between the F1-score of the models training from scratch and models with fine-tuning is 5%. However, contrastive learning plays a significant role where the dataset is small. From the table, it is clear that training from scratch with contrastive learning on the claim dataset has a similar performance to fine-tune models. For example, precision, recall, and F1 score of Contrastive-CNN-1D have a higher value (4% higher precision, 3% higher
recall, 5% higher F1 score) than CNN-1D but have similar performance compared to CNN-1D-transfer for both validation and testing dataset. In general, our all deep learning models perform better than baselines and all models with transfer learning and constructive learning have similar results. WC-LSTM-Cons perform best compared to all models and produce the highest precision, recall, and F1 value (0.87,0.88,0.87) value. The Bert-based model also performs well and has competitive results. CNN-1D model has the worst quantitative values compared to the other deep learning models. However, all models with sentence embedding have better performance compared to word embedding models. This result suggests that sentence embedding may capture more semantic meaning for a whole sentence than word embedding resulting in better performance.

We also record the training and testing time of our developed models. Contrastive-CNN-1D is the fastest model in the prediction phase and takes less than 1 sec. The WC-LSTM-contrastive model is computationally expensive for both the training and testing phase though the WC-LSTM-contrastive model is the best performing model in terms of performance metrics. Bert-based models also take a longer time for training. Among the Bert models, BERT-Transfer-Dense takes 4751.26 sec for training. All of our models are trained and tested on 16 GB memory and CPU-enabled personal laptop. Only Bert models are trained and tested on google collab with a GPU-enabled container.

The most challenging issue of this task is the scarcity of datasets and the available claim dataset
is very small. To overcome this challenge, we use the Pubmed RCT dataset. Training and then fine-tuning improve the results compared to the models trained from scratch. This indicates that the discourse information is relevant and improves performance substantially. Contrastive learning is another option to boost results for the smaller and imbalanced dataset. This technique forces the similar class data to stay close whereas repels the dissimilar data to stay away from the embedding space. We project 128-dimensional vectors into 2-dimensional vectors using the TNSE method for display. Fig.47 shows how contrastive learning helps to boost the results. From the Fig. 47, it is clearly visible that model with contrastive learning grouped same class altogether resulting better result (Table. 16).

5.4.4 SUMMARY

To get the scientific findings from the information overload, the automatic claim extracting tool is becoming increasingly important for information retrieval tasks. To accomplish this task, we develop several deep learning-based claim extraction models using CNN, LSTM, transformer, and its variants. We show that our models outperformed all baseline models and found that 1D-CNN with contrastive learning performs best. Contrastive learning and transfer learning help the models to boost their performance. We believe our developed models will help the researchers to find the core claims easily and help them up-to-date with the latest developments in their scientific field. Graph neural network (GCC) and graph transformer (GT) models are the possible models we are thinking to explore for claim extraction in the future.

5.5 EVIDENCE RETRIEVAL AND CLAIMS JUSTIFICATION

Claims justification is a fact-checking task to verify the claims extracted from scientific news against a corpus of documents that support or refute the claims. In general political claims are verified by crowd workers which are readily available on fact-checking websites. However, it is very hard to verify the scientific claims because of the requirement of annotators with extensive domain knowledge.

Data scarcity is the major challenge for this task. FEVER [214] is a popular fact-checking dataset that has 185,445 claims manually verified against the introductory sections of Wikipedia pages. However, this dataset is not purely scientific resulting in not being suitable for our task.
Scifact [225] is a scientific fact-checking dataset containing 1,409 scientific claims against a corpus of 5,183 abstracts relating to Covid-19. Each sentence of the abstract is categorized as rationale or non-rational based on the claim by domain expert annotators. A claim can be supported or refuted by the rationale sentences of abstracts. For this task, we used a pre-trained model trained on the Sci-fact dataset.

Two tasks have been identified for this claims justification and evidence retrieval task. They are:

- Given a claim $c$ and a corpus $A$, the system must predict a set of evidence abstracts $\xi(c)$.
- For each abstract, it must predict a label and a collection of rationale sentences.

**Rationale Identification** 1st task can be defined as rationale sentences identification from abstracts. Given a claim $c$ and an abstract corpus, the rationale model needs to predict the rationale sentences from each abstract. It is a binary classification problem where there are two classes (rationale sentence or non-rational sentence). To train the model, we need to encode a claim and a sentence of abstract using Bert and concatenate them both using separator (SEP). The goal of the model is to predict a score using the following formula.

$$z_i = \sigma[f(CLS(w_i))]$$ (26)

Here $\sigma$ is the sigmoid function, $f$ is a linear layer and $CLS(w_i)$ is the $CLS$ token from the encoding of $w_i$.

![Fig. 48. Rationale model for evidence sentence identification from a abstract.](image-url)
Fig. 48 shows the rationale model using Bert encoder. A claim and a sentence of abstract are encoded and concatenated followed by hidden layers. The final layer predict whether a sentence is the rationale sentence or not.

**Claims label prediction** The final task is to justify the claims. Sentences identified by the rationale selector are passed to a separate BERT-based model to make the final labeling decision. We performed a similar procedure (claim-sentence concatenation) except for the final layers where there are three classes (support, refute, and no-info).

Fig. 49 shows the claim justification model. Like rationale identification model, we encoded a claim and a rationale sentence extracted from abstract identified by rationale model followed by concatenation and feature extraction by hidden layers. The finale layer predict the verdict where the claims are supported by the abstract (scholarly papers).

To replicate the results [225], we used different per-trained Bert models trained on Sci-fact dataset and test on the Sci-fact validation dataset. From Table. 17, it is visible that pre-trained models trained on Sci-fact dataset work well based on the performance metrics. If we retrieve the desired pertinent scholarly papers and claims, pre-trained claim justification model should justify the claim with proper label.

As our goal is to justify claims extracted from the news articles, we used a pre-trained claim justification model trained on the Snopes dataset. However, our whole pipeline is only able to retrieve claims that are supported by scholarly papers. There are still some challenges that need to address to get a robust, accurate scientific news fact-checking tool.

- We need to expand the search domain to retrieve the pertinent scholarly papers. Currently,
Table 17. Performance evaluation of the transformer based pre-trained models. We use different transformer based pre-trained models trained on Sci-fact dataset and evaluate them on Sci-fact validation dataset.

<table>
<thead>
<tr>
<th>Pre-trained Models</th>
<th>Rationale Sentence Identification</th>
<th>Claim classification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Precision</td>
<td>Recall</td>
</tr>
<tr>
<td>Roberta-base</td>
<td>0.76</td>
<td>0.66</td>
</tr>
<tr>
<td>Roberta-large</td>
<td>0.72</td>
<td>0.67</td>
</tr>
<tr>
<td>Scibert</td>
<td>0.74</td>
<td>0.74</td>
</tr>
<tr>
<td>Biomed-roberta-base</td>
<td>0.75</td>
<td>0.69</td>
</tr>
<tr>
<td>Roberta-large (snopes)</td>
<td>0.26</td>
<td>0.87</td>
</tr>
</tbody>
</table>

we are using Arxiv API that retrieves scholarly papers from Arxiv which is not cover all domains (i.e. the medical domain). We should explore other search domain API (semantic API) or combined APIs (Arxiv API + Semantic API + google scholar API) to retrieve all desired pertinent papers.

- We also need to explore a graph-based model (Graph convolutional neural network or graph transformer) to extract claims.

- We need to build a dataset that will cover all domains to train robust rationale identification and claims justification model.

By addressing these challenges, we are very hopeful to get a robust scientific fact-checking tool which will spot fake scientific news from social media. We have designed our initial fact-checking web interface (Fig.50).

5.6 RELATED PUBLICATIONS

Fig. 50. Fake scientific news detection interface.


Chapter 6

CONCLUSION AND FUTURE WORK

This dissertation offers a series of studies in which deep learning models play a significant role to achieve state-of-the-art results.

Chapter 3 demonstrates the superiority of deep learning models over traditional methods in terms of accuracy and time complexity for seagrass scar detection application. Seagrass habitats are becoming extremely vulnerable due to human intrusion into seagrass meadows, which results in unbalanced marine ecosystems and extinction of marine animals. Traditional methods for seagrass scar detection require manual scarring and are time-consuming. Experimentally we show that our proposed single CNN-based deep learning model performs better than sparse coding without any human intervention. Furthermore, the deep committee machine makes seagrass scar detection more robust and accurate.

The remote sensing image super-resolution and fusion projects are presented in Chapter 4. We show that the GAN models achieve the visually sharper high-resolution images as compared to CNN models. However, quantitatively, the GAN models’ performances are worse as the synthetic high-frequency details are heavily penalized by the performance metrics if not being perfectly aligned with the target image. More specifically, we propose two CNN-based models and two GAN-based models that progressively enhance low-resolution images by 2x to get 4x high-resolution images. This Chapter also presents a fusion study where we show that feature-based arithmetic operations to fuse the multi-temporal images can produce high-resolution images without losing temporal information. We evaluate our proposed models with baselines on three historic datasets and our proposed models outperform the baselines by large margins.

Apart from the computer vision projects, we also illustrate the deep learning model’s efficacy in fake scientific news detection in Chapter 5. We develop models to extract DKEs from a given scientific news and implement a searching pipeline to retrieve pertinent scholarly papers for the news. To extract pertinent scholarly papers, DKEs play a vital role. We explore different deep learning models to extract DEKs and find out that the transformer model can extract nearly all required DKEs with very high recall, precision, and F1 score. We also explore different embedding
methods to embed the whole news articles and abstracts and find out that the transformer-based
DKEs extractor with the TF-IDF and transformer-based document embedding models have supe-
rior performances. Claim extraction is another important task where we use the extracted claims
to justify a news article against scholarly papers. We develop different deep learning models to
extract claims from unstructured texts. Experimentally, we show that contrastive learning and
transfer learning play a vital role in performance improvement if dataset is small. Our proposed
models have outperformed by at least 5% in terms of precision, recall, and F1 score as compared
with the current state of the arts. For claim justification, we explore different pre-trained models
for rationale identification and claim justification and test them on the Sci-fact dataset. We find
that Bert based model trained on the Sci-fact dataset performs well for claim justification which
ultimately proves the concept of our whole fact-checking tool pipeline.

In this dissertation, we develop a set of deep learning models to solve some interesting and
utter important computer vision and NLP problems. However, we still can improve our models
by adopting and developing more sophisticated deep learning models which is our future work.
For our seagrass scar detection project, we are planning to use the vision transformer [11] model
to improve the detection accuracy. Instead of using manual image registration, we will use polar
transform [226, 227] to register the image pairs for our fusion project and perform arithmetic op-
erations in feature space where features can be extracted by transformer. Our fake scientific news
detection tool can be improved by developing new datasets that can cover all domains to train deep
learning models for rationale identification and claim justification. Graph neural networks and
graph transformer models are alternative options for claim extraction.
BIBLIOGRAPHY


[141] “No, nasa is not predicting 15 days of complete darkness in november,” https://bit.ly/2Tfmp0I.


